Statistical Learning of Household Composition from Television Viewing Behaviour

---

Joseph Tsun Kiet Man

June 21, 2016
Version: 2.7.18

Statistical Learning of Household Composition from Television Viewing Behaviour

Joseph Tsun Kiet Man

June 21, 2016

1. Supervisor dr. A.J. Feelders,
prof. dr. A.P.J.M. Siebes
Department of Information and Computing Sciences
Utrecht University

Universiteit Utrecht

2. Supervisor Msc. Harald Hoogstrate
Director System & Development
Pointlogic

Joseph Tsun Kiet Man
Statistical Learning of Household Composition from Television Viewing Behaviour
June 21, 2016
Supervisors: dr. A.J. Feelders, prof. dr. A.P.J.M. Siebes and Msc. Harald Hoogstrate

Utrecht University
Algorithmic Data Analysis Group
Department of Information and Computing Sciences
Faculty of Science
Princetonplein 5
3584 CC Utrecht, The Netherlands

Pointlogic
Fascinatio Boulevard 290
3065 WB Rotterdam, The Netherlands

Abstract

As video consumption rises and more households use set-top boxes, an environment is created where cable operators are able to unobtrusively collect data from households. This in turn can be used by advertisers to predict household characteristics and deliver targeted advertisements. This thesis examines whether it is possible to predict household compositions using program level television viewing data. The household structures are split into eight categories, namely: family (`F`), family with children (`FC`), household (`H`), household with children (`HC`), single female (`SF`), single male (`SF`), single female parent (`SPF`) and single male parent (`SPM`). Three different machine learning algorithms are used to perform this task: regularized multinomial logistic regression, stochastic gradient boosting and support vector machines with a linear, polynomial or a radial basis kernel. For training these models, the Nielsen National People Meter (NPPM) data set is used. Households in this data set are continuously added and removed after a fixed period, therefore we preprocess the households by a process called unification. For each algorithm we examine three models selected by three different performance measures using 5-fold cross-validation. The three performance measures that we use are: accuracy, Kappa statistic (Kappa) and the area under the curve (AUC). In our research we observe that the stochastic gradient boosting model is performing the best, with an accuracy of 0.488 on the validation set. This can also be shown in the resampling distribution of the models along with statistical tests. In addition, we use a regularized multinomial logistic and stochastic gradient boosting model to perform dimension reduction prior to training the support vector machines. This improves the support vector machines slightly. We also create a cascading classification model, which consists of two models, each training a different aspect of our target variable. In the first stage, the first model classifies whether a child is present and the second model classifies the number of adults in the household. This consists of the following categories: family (`F`), household (`H`), single female (`SF`) and single male (`SM`). The resulting cascading model has a validation accuracy of 0.479, which is slightly less than the stochastic gradient boosting machine model even though both use the same machine learning algorithms.

Keywords: Household, Composition, Prediction, Television, Viewing, Behaviour, SVM, Regularization, Stochastic, Gradient, Boosting.

Acknowledgements

I would like to devote this section to the people; without whom I would not be able to finish my thesis.

First and foremost, I would like to gratefully and sincerely thank dr. A. J. Feelders for his guidance and understanding. He provided me with crucial input at Pointlogic and at his office. His patience and support helped me finish this thesis.

Second, I would also like to thank H. Hoogstrate, who provided me with the opportunity to work and do my internship at Pointlogic. It has been a great experience and it provided me with a lot of responsibility in projects.

Third, I would like to thank all my colleagues at Pointlogic, who helped me during issues regardless whether it was for my thesis or for work.

I would also like to thank Yan Yeng Tang, without whom this thesis would have probably be finished 2 months earlier and I did not have to try the patience of my supervisors. However, it would also mean a less pleasant journey in writing this thesis.

Finally, I am grateful for the unconditional support and encouragement I have received from my family and friends.

Contents

List of Figures

List of Tables

1 Introduction

> *I'm the TV. I'm the all-seeing eye and the world of the cathode ray.*
>
> — **from American Gods by Neil Gaiman**
> (English author; 2001)

As more people have access to viewing devices (e.g. televisions, smartphones, computers), more content is consumed at an increasing rate. According to Nielsens cross-platform report in 2014, the digital video consumption on mobile devices and computers more than doubled in the second quarter of 2014 compared to the same quarter in 2012 [74]. However, it was still dwarfed by the average time one spends watching TV. Research showed that people, in general, prefer advertisements on TV rather than on the web, making the platform attractive to advertisers [66].

The increase in video consumption on all viewing devices has not escaped the attention of advertisers, who are eager to follow these trends to reach more consumers. This interest was seen for example in their ad spending. In the United States (US), advertising on viewing devices accounted for more than 68% of the entire advertising market of 200 billion dollars in 2014 [15]. This is an increase of 4 percentage points compared to 2013. In addition to the increasing video consumption, the increased use of set-top boxes, gaming consoles and other smart devices accompanying the TV opens even more possibilities for advertisers, which is similar to what Neil Gaiman imagined: "as we watch on our viewing devices, it allows our viewing devices to watch us". These possibilities allow advertisers to unobtrusively collect behavioural data (e.g. watched channels, channel switching, programs watched), which then can be used for user profiling to target advertisements more effectively. In normal circumstances, television viewing data is not accompanied by data regarding household characteristics. Therefore, it would be very interesting for advertisers if a model could be constructed using labelled data to predict household characteristics for households where these characteristics are unknown. Such a model allows advertisers to target advertisements to households with specific characteristics.

Unfortunately, this approach has not yet been widely studied (see Chapter 2). Therefore, this thesis hopes to contribute to the understanding of the problem and improve the current approaches known at Pointlogic. To narrow the scope of this thesis, we are focusing on predicting household composition rather than general household characteristics (e.g. average age, income, etcetera). The household composition or household structure captures the basic properties of a household. The household composition variable contains the presence of a child and the relationship among adults if multiple adults are present. Earlier research done at Pointlogic by Klein [47] focused on household composition and this thesis hopes to

extend the current knowledge. Thus, it is sensible to keep the same characteristic of interest. Also, models predicting household composition benefit a diverse range of advertisers, for example, cleaning services, home appliances, kitchen accessories, etcetera [1]. This leads us to the research question of this thesis, which is:

> Is it possible to predict household composition using program level television viewing data?

In more concrete terms, this boils down to classifying households as singles, families or households, with or without children. The classification approach we use in this paper, also known as *statistical classification*, is the problem of identifying to which set of categories a new observation belongs, based on a model trained using observations whose category membership is known [2]. Classification algorithms are successfully used in a wide range of applications from credit scoring [25] to handwriting recognition and medical diagnosis [82]. In this thesis, we re-examine the two classification algorithms from previous work by Klein [47], *logistic regression* and *random forest*, with the addition of regularization techniques and boosting. As a new addition, we also compare random forests and logistic regression with *support vector machines* with different kernels. The data set we use is slightly different from the data set of Klein. It contains program level data (i.e. which programs were watched by whom) instead of genre level. It also contains channel data, but that is not used extensively since the focus will be on the program level data. The data is also fully labelled, which is hard to come by in a realistic setting when data is unobtrusively collected.

This thesis consists of eight chapters and is organised as follows. The next chapter reviews the relevant literature and research that has been done at Pointlogic. The third chapter discusses the methodology. The fourth chapter discusses the experimental setup and discusses some data characteristics. The fifth chapter shows the results. The sixth is devoted to the discussion and the last chapter concludes and provides new ideas for additional research.

Literature review

2

A survey of the literature on inferring household characteristics based on television viewing patterns reveals a dearth of empirical research. James and Cunningham pioneered one of the first researches that incorporated television viewing behaviour [39]. They found that direct marketing television (DMTV) shoppers watched more television between midnight and 6 : 00 a.m. Nowak did an extension of this research, where he suggested that consumer characteristics might be related to viewer reactions to DMTV [61]. He also noted that specific viewer characteristics, rather than broad segmentation categories, have the strongest relationship with viewer reactions to DMTV. As the personal video recorder became more common, more data is collected and analysed. Sadly, most of these researches focus on profiling user preferences [57], rather than inferring user characteristics. Researchers inferring household composition is almost non-existing. Therefore, we start by presenting some researches regarding predicting user identity based on behavioural data, before we continue to present a research that predicts household composition based on viewing behaviour.

Carey et. al. [7] used a formal approach and devised a mathematical framework that can be used to infer user identity based on user viewing behaviour. They proposed two methods to describe a program, one was using classes (e.g. genres), while the other used frequent words that are used in the program summary. For the latter, they also used singular value decomposition to perform dimension reduction. Their data set consisted of 33 users with 9 months of program data. This data was then split into a training and test set, where training set had 7 months of data and the test set had only the following 14 days. The model then had to predict the user based on the viewing behaviour in the 14 days. This resulted in an error of 20% for the first approach and 18% for the latter approach, failing to match 5 to 7 users. The error rates were not determined by cross-validation, but by averaging the error rates of 8 different train and test sets. A more recent research done by Lim. et. al. [53] predicts user profiles based on genre preferences and TV viewing durations of programs. There were 14 user profiles defined over different combinations of age and gender. Lim et. al. used a multi-stage classifier, which consists of a similarity measure and the weighted distance k-NN algorithm to predict user profiles. A data set of 2 522 users was used, where 70% were allocated to the training set. This resulted in a validation accuracy of 80.17%, whereas using the Euclidean distance or a vector correlation method only resulted in accuracies of 64.54% and 66.81%. Unfortunately, the Lim et. al. did not mention the majority class. Other similar researches on inferring user identity based on behaviour focus on different behavioural data. Chang et. al. used TV remote control movements to infer user identity [9]. They used 327 features, which includes press codes, key timings, dominant frequencies, energy cost (of the remote to send the signal) and 3-axis acceleration parameters. These were collected from 5 households with 12 users. The research showed that the Max-Margin Markov Network had the best performance, being able to achieve a cross-validation accuracy $70 - 90\%$ depending on the household. It is

observed that the household with the most users performs the worst, as it was only able to correctly assign three viewing behaviours to the users that created them.

Kitts et. al. researched a different way to achieve better results in targeted advertising. This method did not involve predicting household characteristics directly, but by computing the score between the demographics of a viewer and the demographics of viewers who bought the product. If a score exceeded a certain threshold, the would most likely buy the product. This improved the efficiency of advertising by 7 times (i.e. more people who watched the advertisement were more likely to buy the product).

Spranger et. al. [72] were one of the first who tried to predict household composition based on viewing behaviour. They used an ensemble method with different machine learning algorithms and incorporated confidence intervals in each of them. They also added weights in their vote-counting strategy. The result was a profiling system, which could increase the efficiency of advertising. Their paper demonstrated this by the following example, if an ad is sent to all households, 25.18% will include a female aged 18 to 34, which is the target group. But, if their profiling system is used, the ad is sent to only a select number of households, where 58.06% have a female aged 18 to 34.

Pointlogic themselves had also done some research regarding household composition inference. Klein used logistic regression (or logit model) and random forest to predict household composition of 1 364 households using 132 features. The data set had a majority class of families with children (36.5%), which gave the naive classifier an error rate of 63.5%. The random forest model using 2 000 trees performed the best, achieving an error rate of 37.1%. This was slightly better than the 40% error rate of its logistic regression benchmark model. Due to the small size of the data set, a ten-fold cross-validation was done to get the error rates. From the random forest model, Klein noted that the most important variables were children programs. The importance quickly decreases with each program. The first non-children program is live sports, which is highly indicative of a single man.

3 Methodology

Statistical classification generally consists of a set of variables or *features*[1], which can be referred to as *inputs* and a target variable. The target variable contains information to which category an observation belongs, which is also called an *output*. The input features are denoted as X and the output variable as Y. The goal of statistical classification is then to use the inputs to predict the values of the outputs, based on a model which is created using a training set.

In general, a training set consists of a subset of these inputs and outputs used to train the data. The inputs can be represented by a matrix of size $N \times p$, where each row is an observation and each column represents a feature. Matrices are denoted with a bold upper-case letter; therefore, the features matrix in training set is written as $\mathbf{X}$. Observation i from the training set is a vector of size $p + 1$, with p features and one target variable. The features vector is written with a lower-case x_i, where $1 \leq i \leq N$. All vectors are assumed to be column vectors by default and their transpose is written as x_i^T. A feature j of the training set is a scalar or a vector and is written as $\mathbf{x}_j$, where $1 \leq j \leq p$. Thus, vectors are only bold when they contain N components. Estimated values are presented with a hat (e.g. $\hat{\beta}$). As for the output, also called *responses*, they can be simply represented by a vector $\mathbf{y}$ of size N, where each of the components is from the set of labels $C = \{1, \ldots, K\}$, consisting of K classes.

In this section, an introduction is given to the three classification methods that will be used in this thesis. The first one is the *multinomial logistic model* with regularization. The second is *stochastic gradient boosting* and finally *support vector machines* with different kernels. In addition, a small section is devoted to several model evaluation techniques.

3.1 Multinomial Logistic Model

The *multinomial logistic model*, sometimes also called the *logistic* or *logit model*, analyses the relationship between multiple features, and a categorical response. The logistic model arises from the need to model the posterior probabilities of K classes using a linear function in terms of the features $\mathbf{x}_1, \ldots, \mathbf{x}_p$, while adhering to the probability axioms [29, p. 119]. The multinomial logistic model can be defined as follows:

[1] According to Hastie et al. [29] the term "*features*" is preferred over "*independent variables*" in machine learning literature.

$$\begin{aligned} \log \frac{P(Y=1|X=x_i)}{P(Y=K|X=x_i)} &= \beta_{1,0} + \beta_1^T x_i \\ &\vdots \\ \log \frac{P(Y=K-1|X=x_i)}{P(Y=K|X=x_i)} &= \beta_{K-1,0} + \beta_{K-1}^T x_i \end{aligned} \tag{3.1}$$

where $\beta_{c,0}$ is the intercept and β_c^T the weights or coefficients of equation c. The x_i is a features vector of observation i. To find the corresponding weights, the logistic regression model is usually fit by maximum likelihood method, using the conditional likelihood of Y given X. For multiclass problems, the multinomial distribution is used and the log-likelihood for N observations can then be written as:

$$\ell(\theta) = \sum_{i=1}^{N} \left(\sum_{c=1}^{K-1} y_{i,c} \theta_c^T x_i \right) - \log \left(1 + \sum_{c=1}^{K-1} \exp\left(\theta_c^T x_i\right) \right) \tag{3.2}$$

where $\theta_c = (\beta_{c,0}, \beta_c)$ and $y_{i,c}$ a value from the indicator matrix created using y_i. In other words, $y_{i,c} = 1$ if observation i belongs to class c. In all other cases, $y_{i,c} = 0$. In order to find the maximum likelihood estimates, $\hat{\theta}$ (see Equation 3.3), the derivatives of the log-likelihood function is set to zero and solved for $\hat{\theta}$ (see Equation 3.4).

$$\hat{\theta} = \arg\max_{\theta} \ell(\theta) \tag{3.3}$$

$$\frac{\partial \ell(\theta)}{\partial \theta} = \sum_{i=1}^{N} x_i (y_i - p_{y_i}(x_i; \theta)) = 0 \tag{3.4}$$

where $p_{y_i}(x_i; \theta) = P(Y = y_i | X = x_i; \theta)$. Unfortunately, it will not be possible to solve this system of equations, as there is no closed-form solution. Numerical optimisation methods such as the Newton-Raphson are used to approximate the solutions.

3.1.1 Regularization

In some cases, when the variance is sufficiently large, the maximum likelihood estimator could result in estimates that are far from their true values. In order to provide some insights into the *bias-variance trade-off*, consider an arbitrary model of the following form: $Y = f(X) + \epsilon$, where the error term ϵ is normally distributed with a mean of zero (i.e. $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon)$). The estimated model can then be written as $\hat{f}(X)$ and the expected squared prediction error at a certain point x_0 is:

$$\mathrm{Err}(x_0) = E\left[\left(Y - \hat{f}(x_0) \right)^2 \right] \tag{3.5}$$

Equation 3.5 can be decomposed into Equation 3.6.

$$\begin{aligned} \mathrm{Err}(x_0) &= \left(\mathrm{E}\left[\hat{f}(x_0)\right] - f(x_0)\right)^2 + \mathrm{E}\left[\hat{f}(x_0) - \mathrm{E}\left[\hat{f}(x_0)\right]\right]^2 + \quad \sigma_\epsilon^2 \\ &= \qquad \mathrm{Bias}^2 \qquad + \qquad \mathrm{Var}(\hat{f}(x_0)) \qquad + \mathrm{Var}(\epsilon) \end{aligned} \tag{3.6}$$

Equation 3.6 shows that the error of the estimated model consists of the sum of the bias squared, the variance and the irreducible error term. If the bias is reduced, the estimated the model should be able classify more data points correctly. Hence, the variance of the model will be larger. This is also known as the bias-variance trade-off. Consequently, to minimise the error of the model, one should consider the effects of the bias and variance terms. A different reason for why selecting a less complex model is beneficial is due to the interpretability of the model, as fewer features with strong effects are easier to interpret. Hastie et. al [40, p.204] classifies three major groups of model selection techniques: subset selection, shrinkage and dimension reduction. Klein [47] used a subset selection method called the *backward stepwise selection method*, which starts with the full model and iteratively removes the least useful predictor provided that the Akaike Information Criterion (AIC) value decreases. This approach, however, leads to biased R-squared values, p-values and overestimation of the parameters [28, 75]. Furthermore, an increase in sample size does not alleviate the problems [10].

In this thesis we use a different approach to trade-off variance for bias, namely *regularization*. Regularization belongs to the class of shrinkage methods and can be considered as an alternative approach to obtain a parsimonious model.

Ridge regression

The first regularization method, also called *Tikhonov regularization* or *ridge regression*, was first proposed by Andrey Tikhonov to address ill-posed problems [76]. Consider a basic linear model (see Equation 3.7) with standardised features; the coefficients can then be estimated by Equation 3.8.

$$\mathbf{y} = \mathbf{X}^T\beta + \epsilon \tag{3.7}$$

$$\hat{\beta} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} \tag{3.8}$$

Since the features are standardised, $\mathbf{X}^T\mathbf{X}$ equals the correlation matrix and should be non-singular or otherwise it will not have an inverse. Consequently, multicollinearity may not exist. If multicollinearity exists between features, most statistical packages will drop one or more features to obtain the coefficient estimates. Ridge regression, however, deals with it by adding a small value, λ to the diagonal elements of the correlation matrix, resulting in the following equation (see Equation 3.9).

$$\hat{\beta}^{\text{ridge}} = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y} \tag{3.9}$$

where $\mathbf{I}$ is the identity matrix. A different way to write the ridge regression is writing it as a minimisation problem. Ridge regression minimises the following loss function (see Equation 3.10) [29, p.63].

$$\hat{\beta}^{ridge} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{3.10}$$

where $\lambda \geq 0$ serves as a tuning parameter also called an *hyperparameter*, which is determined separately. The second term is also called the *shrinkage penalty* and is sometimes also written as $||\beta||_2$. Equation 3.10 trades off two different criteria between the maximum likelihood estimates and the size of the parameters. The λ serves to control the relative impact of those two terms. When $\lambda = 0$, the penalty term has no effect, and equation boils down to Equation 3.8. If $\lambda \to \infty$, the impact of the shrinkage penalty grows and the estimates approach zero. Notice that the intercept term β_0 has been left out of the penalty term. If this were to be included, it would make the procedure dependent on the origin chosen. In other words, adding a constant to the response y_i would not result in a shift in predictions by the same amount.

Ridge regression can also be done by augmenting the original data matrix with pseudo-observations as follows. Observe that it is possible to write the penalty term of Equation 3.10 as p pseudo-observations if and only if $\left(y_{n+j} - x_{n+j}^T\beta\right)^2 = \lambda\beta_j^2$, for $j = 1, \ldots, p$. This can be achieved by letting $y_{n+j} = 0$, $x_{n+j,j} = \sqrt{\lambda}$ and $x_{n+j,k} = 0, j \neq k$; then we can write the manufacturing rule as Equation 3.11 [12, p.395].

$$\left(y_{n+j} - \sum_{k=1}^{p} x_{n+j,k}\beta_k \right)^2 = \lambda\beta_j^2 \qquad \text{for } i = 1, \ldots, p \tag{3.11}$$

This means appending the response vector with a zero vector $\mathbf{0}$ of size p and appending the original data matrix with $\sqrt{\lambda}\mathbf{I}$ of size $p \times p$. This viewpoint is also referred to as the "phoney data" viewpoint and has led to some concern and controversy in its use and interpretation, as manufacturing data yields the same results as ridge regression [12, p. 394].

One of the main obstacles in using ridge regression is choosing an appropriate λ. Hoerl and Kennard proposed a graphical approach, called *ridge trace* to determine the value of λ [30]. The procedure is as follows: first, perform some ridge regressions with different values of λ. Second, plot the ridge trace, by plotting the coefficients for different values of λ. Last, pick the smallest value of λ for which the coefficients have been stabilised (i.e. are not changing rapidly). A more recent paper, however, expressed some criticisms of this method, since it uses the wrong graphical form, as ridge trace is "an essentially univariate plot of trace

lines for what is essentially a multivariate problem" [24]. Two analytical methods have been proposed by Hoerl et al. [31] to find the correct value of λ. The first one is called the *fixed point ridge regression* and is defined as follows (see Equation 3.12).

$$k = \frac{p\hat{\sigma}^2}{\hat{\beta}^T\hat{\beta}} \tag{3.12}$$

where $\hat{\sigma}^2$ and $\hat{\beta}$ are used from the *ordinary least squares* (OLS) estimates. The second method is an iterative method (see Equation 3.13) [33].

$$k_t = \frac{p\hat{\sigma}^2(t-1)}{(\hat{\beta}(t-1))^T\hat{\beta}(t-1)} \tag{3.13}$$

where t stands for the iteration number and the first values of $\hat{\sigma}^2$ and $\hat{\beta}$ are used from the OLS estimation. Due to the increase in computing power, cross-validation and bootstrapping have increased in popularity in determining the value of λ.

Least absolute shrinkage and selection operator (lasso)

Tibshirani proposed a different regularization method in 1996 [75]. Instead of minimising Equation 3.10, one can also minimise the absolute values of the parameters (see Equation 3.14). This approach is also called *least absolute shrinkage and selection operator* or *lasso*[2].

$$\hat{\beta}^{lasso} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{3.14}$$

The penalty term is sometimes also written as $||\beta||_1$. One advantage of lasso over ridge regression is that it performs feature selection (i.e. some coefficients of the features are zero). To illustrate this, note that it is possible to rewrite Equation 3.10 and 3.14 as a constrained optimisation problem (see Equation 3.15 and 3.16 respectively).

$$\begin{aligned} \min_{\beta} \quad & \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \\ \text{s. t.} \quad & \sum_{j=1}^{p} \beta_j^2 \leq s \end{aligned} \tag{3.15}$$

[2]This is also written as Lasso or LASSO.

$$\begin{aligned} \min_{\beta} \quad & \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \\ \text{s. t.} \quad & \sum_{j=1}^{p} |\beta_j| \leq s \end{aligned} \tag{3.16}$$

There is a direct connection between s and λ, as a large λ is associated with a small s since the coefficients are heavily penalised and therefore, the maximum allowed value, s of the sum of the absolute value of coefficients or the coefficients squared is lowered.

Examining the case when there are only two parameters (i.e. $p = 2$), the standardised coefficients of the OLS can be plotted at $\hat{\beta}$ (see Figure 3.1), while the feasibility regions of the constraints are in algae green. For the lasso, the corners are the first that intersect the contours of the *residual sum of squares* (RSS), corresponding to a zero coefficient. While for the ridge regression, there are no corners, thus coefficients with a zero value are rare. In addition, lasso is less sensitive to extreme values as compared to ridge regression [81].

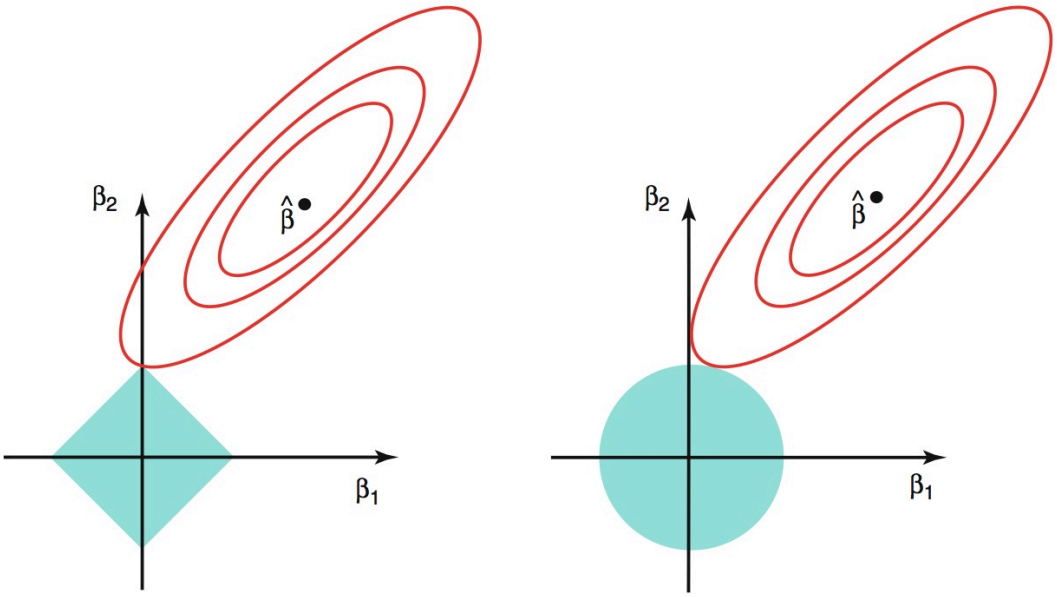


Fig. 3.1.: Estimation graph for the lasso (left) and ridge (right) regression, where the green areas are the constraint regions and the red ellipses are contours of the RSS of an OLS regression.

Source: [40, p. 222]

To solve the minimisation problem in Equation 3.14, one has to resort to convex optimisation methods as there is no closed form solution for lasso. One method to solve the lasso problem is using *least angle regression* [13]. Least angle regression is very fast as it exploits the piecewise linear structure of the solution to the lasso problem and therefore, can provide an efficient way to compute the solutions simultaneously for all values of s or λ.

Frank and Friedman generalised lasso and ridge regression using bridge regression which has a penalty term as $J(\beta) = \sum_{j=1}^{p} |\beta_j|_q$, where $q = 1$ in case of lasso and $q = 2$ in case

of ridge regression. Unfortunately, it has been proven that only lasso performs variable selection in the L_q, where $q \geq 1$ [17]. In addition, $0 < q < 1$ makes the problem non-convex and the optimisation problem more difficult.

Elastic net

Even though lasso performs feature selection, it also has its limitations [85, p.302]. In case $N > p$ and the features are highly correlated, it can be shown empirically that the performance of lasso is dominated by ridge regression. In case $p > N$, the lasso will select at most N features before it saturates, as any additional feature can only increase the loss function. A different problem arises when there is a group of features that are highly correlated. In that case, lasso will select only one arbitrary feature from that group, which can lead to highly inconsistent results [71]. It also interferes with feature ranking methods that use the estimated coefficients [77]. In our case, due to the large number of features, we are more likely to have $p > N$ than $N > p$ during our model construction phase.

Zou and Hastie suggested a new approach to deal with those deficiencies called *elastic net* [85]. Elastic net can select more than N features in the case $p > N$ and selects the group of features that are highly correlated. Due to these properties, this is also the main regularization technique that is thoroughly examined in this thesis. The elastic net is defined as a penalty term that is a linear combination of lasso and ridge regression.

$$\hat{\beta}^{elastic} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda_2 \sum_{j=1}^{p} \beta_j^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| \right\} \tag{3.17}$$

A different way to write Equation 3.17 is as follows (see Equation 3.18).

$$\hat{\beta}^{elastic} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \left((1-\alpha) \sum_{j=1}^{p} \beta_j^2 + \alpha \sum_{j=1}^{p} |\beta_j| \right) \right\} \tag{3.18}$$

where $\alpha = \frac{\lambda_1}{\lambda_1 + \lambda_2}$. If $\alpha = 0$, Equation 3.18 becomes the ridge regression. For $\alpha = 1$, Equation 3.18 is equal to the lasso.

Unfortunately, based on empirical evidence this method does not perform satisfactorily unless it is close to a ridge regression or lasso [85, p.307]. This is due to the double amount of shrinkage by the ridge and lasso penalty term. Hence, rescaling the elastic net coefficient with $(1 + \lambda_2)$ is needed to improve the results.

To solve Equation 3.18, methods optimising lasso can be used, as ridge regression can be written using a "phoney data" representation. It follows that minimising Equation 3.18

is equivalent to a lasso-type minimisation. However, more recent developments favour the use of pathwise coordinate descent to solve Equation 3.18 due to its performance [23]. The algorithm implemented in the R package **glmnet** is given below.

Data: Response Y, standardised features $\mathbf{X}$, mixing parameter α, $\epsilon = 0.001$[3]
Result: A regularization path for different values of λ
1 $\lambda_{max} = \frac{\max_l |\langle x_l, y \rangle|}{\alpha N}$
2 $\lambda_{min} = \lambda_{max}\epsilon$
3 **for** $\lambda \in \{\lambda_{max}, ..., \lambda_{min}\}$ **do**
4 **while** *iter* $< max_{iter}$ *and not converged* **do**
5 **for** *every* $l \in \{1, ..., K\}$ **do**
6 Update $\ell_{Q,l}$ using the current parameters $\{\widetilde{\beta}_{0k}, \widetilde{\beta}_k\}_1^K$
7 Solve $\min_{(\beta_{0l}, \beta_l)}\{-\ell_{Q,l}(\beta_{0l}, \beta_l) + \lambda P_a(\beta_l)\}$ using coordinate descent
8 **end**
9 **end**
10 **end**

Algorithm 1: glmnet algorithm to solve the regularized multinomial regression.

where $\{\widetilde{\beta}_{0k}, \widetilde{\beta}_k\}_1^K$ are the current[4] parameter estimates for the classes $1, \ldots, K$ and $P_a(\cdot)$ the penalty term used in elastic net (i.e. $P_a(\beta_l) = (1-\alpha)\sum_{j=1}^p \beta_j^2 + \alpha \sum_{j=1}^p |\beta_j|$). In Algorithm 1 the log likelihood function, $\ell_l(\cdot)$ is not used, but a second-order Taylor approximation of that function, $\ell_{Q,l}(\cdot)$ (see Equation 3.19). This increases the performance and simplifies the minimisation problem significantly (see Line 7).

$$\ell_{Q,l}(\beta_{0l}, \beta_l) = -\frac{1}{2N}\sum_{i=1}^N w_{il}\left(z_{il} - \beta_{0l} - x_i^T\beta_l\right)^2 + cost(\{\widetilde{\beta}_{0k}, \widetilde{\beta}_k\}_1^K) \tag{3.19}$$

$$z_{il} = \widetilde{\beta}_{0l} + x_i^T\widetilde{\beta}_l + \frac{y_{il} - \widetilde{p}_l(x_i)}{w_{il}} \tag{3.20}$$

$$w_{il} = \widetilde{p}_l(x_i)(1 - \widetilde{p}_l(x_i)) \tag{3.21}$$

$$p_l(x_i) = P(Y = l | X = x_i) \tag{3.22}$$

where $\ell_{Q,l}$ stands for the quadratic approximation (Q) for class l of the log likelihood function and *cost* is a function that depends on the current parameter estimates $\{\widetilde{\beta}_{0k}, \widetilde{\beta}_k\}_1^K$ for classes $1, ..., K$.

To summarise, Algorithm 1 can be stated as follows. The algorithm starts by computing λ_{max} (see Line 1). This is the smallest λ such that all $\beta_j = 0$. As the lambdas decrease, more β_j will have a non-zero value and thus will be included in the model. The third loop loops through all the classes and applies coordinate descent to solve the penalised weighted least-squares problem (see Line 7) until the estimated parameters converge.

[4]The tilde indicates the current status.

3.2 Stochastic Gradient Boosting

The second algorithm that we will discuss is *stochastic gradient boosting* as originally derived by Friedman [21].

In any function estimation problem, we would like to find a function, $\hat{f}(x)$ that minimises the expectation of some loss function $\Psi(y, f)$, i.e. $\hat{f}(x) = \arg\min_{f(x)} E_{y,x}\Psi(y, f(x))$. However, instead of only performing function parameter optimisation (like in gradient descent), the optimisation procedure in gradient boosting models is held in the function space. Consider the function estimate $\hat{f}(x)$ in the additive function form:

$$\hat{f}(x) = \sum_{m=0}^{M} \hat{f}_m(x) \tag{3.23}$$

In this representation (see Equation 3.23), M is the number of iterations, $\hat{f}_0(x)$ is the initial guess and $\{\hat{f}_m(x)\}_{m=1}^{M}$ are the function increments, also called "*boosts*". In order to distinguish the function estimate $\hat{f}(x)$ and the function increments, we introduce the notion of "*base-learner*" functions $h(x)$. To make the problem tractable, we parameterise the base-learners, resulting in $h(x, \theta)$. We can now start by formulating the gradient boosting algorithm. The general idea is greedily incrementing the function estimate with the base-learners. For this purpose, the optimal step size ρ and function parameters of the base-learner should be determined at each iteration. Therefore, the optimisation problem at iteration m is as follows.

$$(\rho_m, \theta_m) = \arg\min_{\rho,\theta} \sum_{i=1}^{N} \Psi(y_i, \hat{f}_{m-1}(x)) + \rho h(x_i, \theta) \tag{3.24}$$

Since Equation 3.24 does not restrict the loss function $\Psi(y, f)$ and the base-learner $h(x, \theta)$, one can arbitrarily choose one. In practice, given some loss function and/or a base-learner, the solution to the parameter estimates can be hard to obtain. To solve this, Friedman proposed to choose a new function $h(x, \theta_m)$ to be the most parallel to the negative gradient along the observed data. Thus, the gradient along the observed data is specified as follows (see Equation 3.25).

$$g_m(x) = E_y \left[\frac{\delta\Psi(y, f(x))}{\delta f(x)} \mid x \right]_{f(x)=\hat{f}_{m-1}(x)} \tag{3.25}$$

Therefore, we can simply choose a base-learner to be most correlated with $-g_m(x)$. This simplifies Equation 3.24 to Equation 3.26, which is equal to the classic least-squares minimisation.

$$(\rho_m, \theta_m) = \arg\min_{\rho,\theta} \sum_{i=1}^{N} \left[-g_m(x_i) + \rho h(x_i, \theta)\right]^2 \tag{3.26}$$

This lets us specify Friedman's gradient boost algorithm as follows (see Algorithm 2). First, $\hat{f}_0(x)$ is initialised with a constant. For each iteration, the negative gradient is computed and a new base-learner function $h(x, \theta)$ is fitted. Afterwards, the optimal step-size increment is computed and the function estimate is updated.

1 Initialise $\hat{f}_0(x)$ with a constant.
2 **for** $m = 1$ *to* M **do**
3 | Compute the negative gradient $-g_m(x)$.
4 | Fit a new base-learner function $h(x, \theta_m)$.
5 | Find the best gradient descent step-size ρ_m:

$$\rho_m = \arg\min_{\rho} \sum_{i=1}^{N} \Psi[y_i, \hat{f}_{m-1}(x_i) + \rho h(x_i, \theta_m)]$$

6 | Update the function estimate $\hat{f}_m(x) \leftarrow \hat{f}_{m-1}(x) + \rho_m h(x, \theta_m)$
7 **end**

Algorithm 2: Friendman's Gradient Boosting algorithm

There are different choices to choose a base-learner. The most common base-learners can be categorised into three categories: linear models, smooth models and decision trees. In this thesis we choose decision trees as the base-learners as these produce competitive, robust and interpretable models, therefore making them appropriate for data mining [21, p. 1]. The idea behind a decision tree base-learner is to partition the space of input variables into homogeneous rectangle areas by a tree-based rule system. Each tree split corresponds to an if-else clause over some input variable. This structure naturally encodes and models interactions between predictor variables. These trees are generally parameterised with the number of splits, or commonly referred to as *interaction depth* s and the minimum terminal node size $|R|_{min}$. In many practical applications, however, small trees and tree stumps provide accurate results [41]. A more detailed discussion regarding decision trees can be found in Appendix A.1. Since we are using additive base-learners, the learning algorithm differs slightly from Algorithm 2. Furthermore, as this thesis focusses on classification problems, the loss function that will be used is the K class *multinomial deviance loss function* and is defined as:

$$\Psi(y, p(x)) = -\sum_{k=1}^{K} I(y = k) \log p_k(x) \tag{3.27}$$

The algorithm starts by initialising a constant model (see Line 1). Then for each class k, we compute the negative gradient of the multinomial deviance loss function (see Line 5), which will be the working response. The working response can also be interpreted as a vector of pseudo-residuals. Subsequently, we fit the decision tree to the working response

and compute the optimal step size. Finally, we update the ensemble $f_{km}(x)$. The output of this algorithm is a classifier that predicts class probabilities rather than a single class (see Line 11). To predict a single class, we take the class with the highest probability.

1 Initialise $f_{k0}(x) = 0$, $k = 1, \ldots, K$
2 **for** $m = 1$ *to* M **do**
3 Compute $p_k(x) = \frac{\exp f_k(x)}{\sum_{l=1}^{K} \exp f_l(x)}$
4 **for** $k = 1$ *to* K **do**
5 Compute $r_{ikm} = I(y_i = k) - p_k(x_i)$, for $i = 1, \ldots, N$
6 Fit a decision tree to the working response r_{ikm}, $i = 1, \ldots, N$, giving terminal regions R_{jkm}, $j = 1, \ldots, J_m$
7 Compute

$$\gamma_{jkm} = \frac{K-1}{K} \cdot \frac{\sum_{x_i \in R_{jkm}} r_{ikm}}{\sum_{x_i \in R_{jkm}} |r_{ikm}|(1 - |r_{ikm}|)}$$

for all $j = 1, \ldots, J_m$
8 Update $f_{km}(x) = f_{k,m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jkm} I(x \in R_{jkm})$.
9 **end**
10 **end**
11 Output $\hat{f}_k(x) = f_{kM}(x)$, $k = 1, \ldots, K$.

Algorithm 3: Gradient Boosting Algorithm for classification [29, p.387]

Similar to regularizing logistic regression models, regularization can also be applied on gradient boosting models [21]. This is done by introducing a shrinkage parameter to reduce the contribution of each tree by a factor v. This changes Line 8 to $f_{km}(x) = f_{k,m-1}(x) + v \cdot \sum_{j=1}^{J_m} \gamma_{jkm} I(x \in R_{jkm})$, where $0 < v < 1$. Smaller values of v result in larger training risk for the same number of iterations M. Both parameters are dependent as smaller v leads to larger M for the same training risk. Using empirical tests, Friedman found that smaller values of v leads to better test errors and require correspondingly larger values of M [21]. In fact, the best strategy appears to be setting v to be very small and then choose M.

Another method to improve noisy classifiers is by a concept derived from *bagging* [5]. This reduces the variance by averaging across different models. Employing the bagging concept to gradient boosting results in *stochastic gradient boosting*. With each iteration in stochastic gradient boosting, a fraction, η of the features are sampled without replacement[5] and used as the training set to grow the tree. This leads to improved computing time and provides a more accurate model. A typical value for η is $\frac{1}{2}$, which is also used primarily throughout this thesis.

3.2.1 Interpretation Methods

Single decision trees are highly interpretable. Unfortunately, the stochastic gradient boosting model consists of linear combination of trees, which loses this important property.

[5] Note: this is different from *bagging*, which samples with replacement because it uses samples of the same size as the training set

Therefore, different ways are developed to interpret these models. We will discuss two interpretation methods that are commonly used throughout this thesis.

Variable Importance

For a single decision tree T, Breiman et al. proposed the following variable importance metric for feature j [6]:

$$\mathcal{I}_j^2(T) = \sum_{t=1}^{J-1} \hat{i}_t^2 I(v(t) = j) \tag{3.28}$$

Where $J-1$ is the number of internal nodes, v a function that returns the feature index given a node index t and $\hat{i}_t^2$ is the maximum estimated improvement in squared error risk over the entire region. The squared relative importance $\mathcal{I}_j^2(T)$ for node j in tree T is then the sum of squared improvements for all nodes for which j is chosen as the splitting variable. The generalisation of this concept to additive trees is straightforward, it is averaged over the trees (see Equation 3.29).

$$\mathcal{I}_j^2(T) = \frac{1}{M} \sum_{m=1}^{M} \mathcal{I}_j^2(T_m) \tag{3.29}$$

For a K-class classification, stochastic gradient boosting creates K separate sub-models, $\hat{f}_k(x)$ (see Algorithm 3 Line 11). Each of these sub-models consists of a sum of trees (see Equation 3.30).

$$\hat{f}_k(x) = \sum_{m=1}^{M} T_{km}(x) \tag{3.30}$$

Thus, after averaging over the trees, it should also be averaged over all K classes. Formally, this can be written as follows:

$$\mathcal{I}_{jk}^2 = \frac{1}{M} \sum_{m=1}^{M} \mathcal{I}_j^2(T_{km}) \tag{3.31}$$

$$\mathcal{I}_j^2 = \frac{1}{K} \sum_{k=1}^{K} \mathcal{I}_{jk}^2 \tag{3.32}$$

Considering that the variable importance measure is relative, it is customary to assign the largest a value of 100 and then scale the other importance measures accordingly [29, p. 368].

Partial Dependence Plots

In order to understand the complex behaviour between predictors and outcome, *partial dependence plots* are used. These plots are graphical visualisations of the marginal effect of a given variable (or multiple variables) on an outcome. Typically, these are restricted to only one or two variables due to the limits of human perception, and thus may be misleading due to hidden higher-order interactions. Despite this, partial dependence plots can still be extremely useful for knowledge discovery in large data sets, especially when the model is dominated by lower-order interactions and main effects. Following the framework provided by Hastie et al. [29, p. 369], partial dependency plots can be mathematically defined as follows. Consider a subset S of p features, such that $S \subset \{1, \ldots, p\}$. Let S^C be the complement to S, such that $S \cup S^C = \{1, \ldots, p\}$. The stochastic gradient boosting predictor function $f(X)$ will depend upon all p predictor variables. Thus, $f(X_S, X_{S^C})$. The partial dependence of the S predictors on the predictive function $f(X)$ is:

$$f_S(X_S) = \mathbb{E}_{X_{S^C}}[f(X_S, X_C)] \tag{3.33}$$

This can be estimated with:

$$\bar{f}_S(X_S) = \frac{1}{N}\sum_{i=1}^{N}[f(X_S, X_{S_i^C})] \tag{3.34}$$

where $\{x_{S_1^C}, x_{S_2^C}, \ldots, x_{S_N^C}\}$ are the values of X_{S^C} occurring over all observations in the training data. Hence, to calculate the partial dependence of a given variable or variables, the entire training set is utilised for every set of joint values in X_S. In some cases this is computationally intensive when the data set is large. Fortunately for decision trees, this computation can be done relatively efficiently without reference to the training data [62]. For every point in the graph, a weighted tree traversal is performed. Concretely, if a split node involves a feature in X_S, the corresponding branch is followed. In all other cases, both branches are followed. Each branch is weighted by the fraction of training samples that entered the branch and finally, the partial dependence is given by a weighted average of all visited leaf nodes. For a tree ensemble, such as stochastic gradient boosting, the result of each tree is averaged.

3.3 Support Vector Machines

The last machine learning technique we discuss are support vector machines (SVM). Support vector machines are state-of-the-art classification methods introduced in 1992 by Boser, Guyon and Vapnik [4]. First, consider a linear two-class classification problem of the form:

$$y(x) = \beta^T \phi(x) + \beta_0 \tag{3.35}$$

where $\phi(x)$ denotes a fixed feature space transformation, β_0 the bias parameter and β is the weights vector. The training set comprises of N input vectors, $x_1, \ldots, x_N$, with the corresponding target values $t_n \in \{-1, 1\}$. A new data point x is classified according to the sign of $y(x)$. For now, let us assume that the training data set is linearly separable in the feature space $\mathcal{Z}$. This means that there are values for β and β_0 that satisfies $y(x_n) > 0$ for points having $t_n = 1$ and $y(x_n) < 0$ for points having $t_n = -1$. Thus, all data points satisfy $t_n y(x_n) > 0$. If there are multiple solutions, the support vector machine will choose the one with the largest margin, which is defined as the smallest distance between the decision boundary and any of the observations. The perpendicular distance from point x_n to a hyperplane, defined by $y(x) = 0$, can then be calculated as follows (see Equation 3.36).

$$\frac{t_n y(x_n)}{||\beta||} \tag{3.36}$$

The margin is then given by the perpendicular distance to the closest point x_n from the data set and we optimise the parameters β and β_0 to maximise this distance:

$$\text{margin} = \arg\max_{\beta,\beta_0} \min_n \frac{t_n y(x_n)}{||\beta||} \tag{3.37}$$

Unfortunately, the direct solution of this optimisation problem is rather difficult. Therefore, we rescale the constraint for all data points to $t_n y(x_n) \geq 1$, where $t_n y(x_n) = 1$ for all points that are the closest to the decision surface. It can be shown that by rescaling β and β_0, the distance from any point to the decision surface is not changed [29, p. 328]. This representation is also known as the *canonical representation of the decision hyperplane.*

Now the optimisation problem can be rewritten to:

$$\begin{aligned} \arg\min_{\beta} \quad & \tfrac{1}{2}||\beta||^2 \\ \text{subject to} \quad & t_n y(x_n) \geq 1 \qquad \text{for } n = 1, \ldots, N \end{aligned} \tag{3.38}$$

To solve this constrained optimisation problem, we introduce Lagrange multipliers $a_n \geq 0$ with one multiplier for each constraint, giving the Lagrangian function:

$$L(\beta, \beta_0, \mathbf{a}) = \frac{1}{2}||\beta||^2 - \sum_{n=1}^{N} a_n \{t_n y(x_n) - 1\} \tag{3.39}$$

where $\mathbf{a} = (a_1, \ldots, a_N)^T$. Note that $a_n = 0$ for points that are not the closest to the decision surface and $a_n > 0$ for points that are. Taking the first derivative of the Lagrangian function gives us the following conditions:

$$\begin{aligned} \frac{\partial L}{\partial \beta} &= \beta - \sum_{n=1}^{N} a_n t_n \phi(x) = 0 \\ \frac{\partial L}{\partial \beta_0} &= \sum_{n=1}^{N} a_n t_n = 0 \end{aligned} \tag{3.40}$$

From the first condition, we can see that β is equal to the linear sum of some of the vectors in the data set. If we plug β into the Lagrangian function (see Equation 3.39), we get the dual representation:

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m \phi(x)^T \phi(x) \tag{3.41}$$

This allows us to rewrite the optimisation problem to:

$$\begin{aligned} \arg\max_{\mathbf{a}} \quad & \tilde{L}(\mathbf{a}) \\ \text{subject to} \quad & \sum_{n=1}^{N} a_n t_n = 0 \\ & a_n \geq 0 \qquad \text{for } n = 1, \ldots, N \end{aligned} \tag{3.42}$$

One of the advantages of this representation is that $\tilde{L}(\mathbf{a})$ depends only on the dot product of pairs of samples. This allows us to reformulate the model using kernel functions, which will be elaborated in a later section.

In order to solve Equation 3.42, quadratic programming techniques will be used. However, due to memory and computational requirements, traditional quadratic programming techniques cannot be used. Vapnik proposed to solve it using a technique called *chunking*, which exploits the fact that the value of the Lagrangian is unchanged if only the rows and columns corresponding to non-zero Lagrange multipliers are used in the inner product [78]. Platt took this concept to the extreme by only considering two Lagrange multipliers at the same time [64]. In this case, the subproblem can be solved analytically, thus avoiding numerical quadratic programming altogether. Heuristics are used for choosing the pair of Lagrange multipliers considered at each step.

After solving for $\mathbf{a}$, we proceed to solve for β_0. This can be done by taking a point for which $a_n \neq 0$ and solve for β_0 using the equation $t_n y(x_n) = 1$. A numerically more stable solution can be obtained by solving:

$$\beta_0 = \frac{1}{N_s} \sum_{n \in S} t_n - \sum_{m \in S} a_m t_m \phi(x_n) \phi(x_m) \tag{3.43}$$

This basically multiplies $t_n y(x_n) = 1$ with t_n on both sides and then averages the equation over all support vectors and solving for β_0. Here N_S is the number of support vectors and S the set of indices for the support vectors.

To classify new data points, such as x, we evaluate $y(x)$, which is similar to Equation 3.35 with β plugged in from the first equation in Equation 3.40. This results in the following equation (see Equation 3.44).

$$y(x) = \sum_{n=1}^{N} a_n t_n \phi(x)\phi(x_n) + \beta_0 \tag{3.44}$$

This shows that any data point which has $a_n = 0$, does not play a role in the classification process and therefore, can be omitted. The remaining points are called support vectors and satisfy $t_n y(x_n) = 1$. These points lie on the maximum margin hyperplanes in the feature space.

So far, we have only considered data sets that are linearly separable in the feature space $\phi(x)$, however this is not always the case. In case with overlapping distributions, we would like to allow for misclassified data points to be penalised. To achieve this, we introduce slack variables ξ_n for every data point, where $\xi_n = 0$ for data points that lie inside the correct margin boundary and $\xi_n = |t_n - y(x_n)|$ for all other points. As a result, points for which $0 < \xi_n \leq 1$ are correctly classified, but lie inside the margin and points with $\xi_n > 1$ are misclassified.

$$\xi_n = \begin{cases} 0 & \text{if correctly classified} \\ |t_n y(x_n)| & \text{otherwise} \end{cases} \tag{3.45}$$

The optimisation Equation 3.38, changes to:

$$\begin{aligned} \arg\min_{\beta} \quad & \tfrac{1}{2}||\beta||^2 + C\textstyle\sum_{n=1}^{N} \xi_n \\ \text{subject to} \quad & t_n y(x_n) \geq 1 - \xi_n \qquad \text{for } n = 1, \ldots, N \\ & \xi_n \geq 0 \end{aligned} \tag{3.46}$$

where C controls the trade-off between the slack variable penalty and the margin. The C parameter can be considered analogous to the inverse of a regularization parameter, as it controls the trade-off between minimising training errors and controlling the model complexity. From Equation 3.46 it can be shown that this formulation of the SVM is equivalent to Tikhonov regularization problem [68]. First, observe that one can rewrite the constraints using the Hinge loss function as $H(y(x_n), t_n) = \max(1 - t_n y(x_n), 0)$. If we define $\lambda = \frac{1}{2C}$, then we can rewrite Equation 3.46 to Equation 3.47, which is equivalent to the Tikhonov regularization problem.

$$\arg\min_{\beta} \sum_{n=1}^{N} H(y(x_n), t_n) + \lambda ||\beta||_2 \tag{3.47}$$

The corresponding Lagrangian can then be given by Equation 3.48.

$$L(\beta, \beta_0, \mathbf{a}) = \frac{1}{2}||\beta||^2 + C\sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} a_n\{t_n y(x_n) - 1 + \xi_n\} - \sum_{n=1}^{N} \mu_n \xi_n \tag{3.48}$$

where μ_n and a_n are the Lagrange multipliers. Again, we take the first derivative with respect to β, β_0 and ξ_n. The first order conditions of β and β_0 corresponds exactly to the Equation 3.40, the derivative with respect to ξ_n is given as:

$$\frac{\delta L}{\delta \xi_n} = C - \mu_n - a_n = 0, \text{for } n = 1, \ldots, N \tag{3.49}$$

Using the first order conditions (see Equation 3.40 and 3.49), we can eliminate β, β_0 and $\{\xi_n\}_{n=1}^{N}$ from the Lagrangian (see Equation 3.48). The resulting dual is exactly the same as Equation 3.41, except the constraints differ. The corresponding constraints are:

$$0 \leq a_n \leq C \tag{3.50}$$

$$\sum_{n=1}^{N} a_n t_n = 0 \tag{3.51}$$

These are also known as box constraints. The resulting solution can then also be used similarly as the SVM model with a hard margin.

Inherently support vector machines are binary classifiers. In practice however, we wish classify for multiple classes. Various methods are suggested for combining multiple two-class SVMs in order to build a multiclass classifier. In this thesis a pairwise classification method is used [48]. This method constructs $\binom{K}{2}$ classifiers where each one is trained on data from two classes. Prediction is then done by voting, where each classifier gives a prediction and the class which is predicted most wins. It has been shown that this method produce robust results when used with SVMs [36].

A different issue that needs to be addressed is in achieving probabilistic output. Probabilistic output can be used to gain further insights into model evaluation. Probabilistic output is not straightforward in SVMs and generally achieved by using Platt scaling [54]. Platt scaling works by fitting a logistic regression model to the classifiers scores using Newtons backtracking with line search.

3.3.1 Kernels

As noted in the previous section, a support vector machine can be written such, that it depends only on the inner products of two vector points. This allows us to use the *kernel trick*, which is an implicit mapping of the input data into a high dimensional feature space defined by a kernel function. The kernel function returns the inner product between the images of two data points in the feature space [43]. The learning then takes place in the feature space assuming that the learning algorithm can be written such that the data points only appear inside dot products with other points. The most basic kernel is the *linear kernel*, which is also the *kernel function* that is used in the previous section. This is simply the inner product between two data points x and x' (see Equation 3.52).

$$\begin{aligned} k(x, x') &= \phi(x)^T \phi(x') \\ k(x, x') &= x^T x' \end{aligned} \tag{3.52}$$

In order to illustrate the concept, let $x = (x_1, x_2)$ and $x' = (x'_1, x'_2)$ be two points in the input space $\mathcal{X}$, we wish to compute $z^T z'$, which is the inner product of x and x' transformed in the feature space $\mathcal{Z}$. Now suppose the transformation is to the second order polynomial, i.e. $z = \phi(x) = (1, x_1, x_2, x_1^2, x_2^2, x_1 x^2)$. The inner product of x and x' transformed into feature space $\mathcal{Z}$ is then:

$$z^T z' = 1 + x_1 x'_1 + x_2 x'_2 + x_1^2 x_1'^2 + x_2^2 x_2'^2 + x_1 x'_1 x_2 x'_2 \tag{3.53}$$

The question now becomes whether it is possible to compute $z^T z'$, without first transforming x and x'. Fortunately, this is possible and can be achieved with the kernel $K(x, x') = (1 + x^T x')^2$. This gives the following output:

$$(1 + x^T x')^2 = 1 + x_1 x'_1 + x_2 x'_2 + 2x_1^2 x_1'^2 + 2x_2^2 x_2'^2 + 2x_1 x'_1 x_2 x'_2 \tag{3.54}$$

Although the transformation is not exactly as previously defined, it is however a transformation in the feature space $\mathcal{Z}$. The transformation is namely, $\phi(x) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x^2)$. Generalising the linear kernel into a d degree *polynomial kernel* can be done by the following kernel:

$$K(x, x') = (ax^T x' + b)^d \tag{3.55}$$

Where a is the scaling factor, b the offset and d the degree of the kernel. Similarly one can also define a kernel in an infinite feature space. This can be achieved by the following *Gaussian radial basis kernel* function:

$$K(x, x') = \exp\left(-\sigma |(||x - x'||)|^2\right) \tag{3.56}$$

To show that this kernel is in fact the inner product after transforming the data points x and x' into an infinite dimensional $\mathcal{Z}$ feature space, observe that the Taylor expansion of this kernel is as follows:

$$K(x, x') = \sum_{k=0}^{\infty} \frac{(x^T x')^k}{\sigma^k k!} \tag{3.57}$$

In this thesis, we will examine the performance of these three kernels. The linear kernel is chosen as it is usually chosen in text categorisation and when dealing with large sparse data

[42]. The polynomial kernel is selected as it allows for non-linear models and the Gaussian radial kernel function is selected due to its general purpose usage [38].

3.4 Model Evaluation

The most widely used method for estimating the prediction error for each combination of different hyperparameters is cross-validation. K-fold cross-validation is defined as the process of splitting the data into K roughly equal-sized sets randomly and then use one of the sets as a validation set for the model trained using the remaining data sets. Based on the validation set, different performance measures, such as accuracy can be computed. This gives a reasonable indication of the generalised performance measure. Based on the performance measure of the validation set, a set of values are chosen for each hyperparameter that maximises the performance measure and the final model is trained on the full data set. Alternative methods for model evaluation have been explored, such as withholding one data set as a validation set, instead of doing cross-validation. This would reduce the computational power that is needed, but may also lead to a higher bias. Bootstrapping and information criteria techniques have been considered, but simulation studies and empirical evidence showed that cross-validation outperforms these methods [29]. For those reasons, cross-validation seems to be the best choice. In this thesis, the number of folds K is set to $K = 5$ and the validation sets for each fold are pre-generated, which helps in comparing different models.

3.4.1 Model Performance

In supervised machine learning there are several ways of assessing model performance. Most performance measures are usually built from the confusion matrix, which records correctly and incorrectly classified data points. In this thesis, the confusion matrix is created using the test set. The values of the target variable of the test set are referred to as the reference classes, while the predictions made from the model are referred to as simply the predictions.

The most commonly used performance measure is accuracy. It is defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.58}$$

where TP stands for true positive, TN true negative, FP false positive and FN false negative. Unfortunately, accuracy itself does not take into account random chance. Therefore it should always be compared with a different classifier. In our case, we use the naive classifier as the base case, which assigns every sample to the majority class.

Cohens Kappa statistic resolves this issue by comparing the expected accuracy with the observed accuracy of the model. This removes the need for the naive classifier in the case with accuracy. The formula for the Kappa statistic is:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{3.59}$$

where $p_o = Accuracy$ is the observed accuracy and p_e is the expected accuracy. The expected accuracy can be expressed as follows. Let p_{i+} be the proportion of observations that are classified by the classification algorithm as class i and p_{+i} the proportion of observations that are classified as class i. The expected accuracy can then be defined as:

$$p_e = \sum_{k=1}^{|C|} p_{k+} p_{+k} \tag{3.60}$$

Magnitude guidelines for the Kappa statistic were suggested. Landis and Koch [51] for example characterised $[0.61, 0.81]$ as substantial and $[0.81, 1]$ as perfect. While, Fleiss [20] characterised values over 0.75 as excellent and $[0.4, 0.75]$ as fair to good.

A third method that has become increasingly popular is using the area under the *receiver operating characteristic* (ROC) curve. ROC curves assess predictive behaviour independent of error costs or class distribution and therefore, offer a more robust evaluation of the relative prediction performance of alternative models than traditional comparison methods [67]. ROC graphs are used in signal detection theory to depict the tradeoff between true and false positives of the classifier [14]. Let us consider a binary classifier, if a classifier is run against a training set, it produces a confusion matrix and can be used to compute the sensitivity and the specificity (see Equations 3.61).

$$\begin{aligned} sensitivity &= \tfrac{TP}{TP+FN} \\ specificity &= \tfrac{TN}{TN+FP} \end{aligned} \tag{3.61}$$

These are then used as coordinates and corresponds to a single point in the ROC space (see Figure 3.2). The lower left point $(0, 0)$ represents a classifier that never issues a true or false positive. The upper right corner $(1, 1)$ represents a classifier that issues positive classifications unconditionally. The top left corner $(0, 1)$ represents perfect classification. The ROC curve is created by combining all points into a curve, which is presented in blue and can be used to visualise the trade-off between specificity and sensitivity. When the threshold of classifying an observation to a specific class is low, the sensitivity is increased, since many observations will be classified as positive and this also include those that should be classified negative. Thus, increase in sensitivity is met by a decrease in specificity (i.e. an increase in $1 - Specificity$). The reverse relationship also holds.

To produce a ROC curve, the posterior probabilities of the model should be computed. By varying the decision threshold, a series of sensitivity and specificity pairs can be made, which defines the ROC curve. The *area under the curve* (AUC) can then be used to quantify the overall ability of the model to discriminate between the two classes. There are several different methods to calculate the AUC. The first one is using the formula to calculate

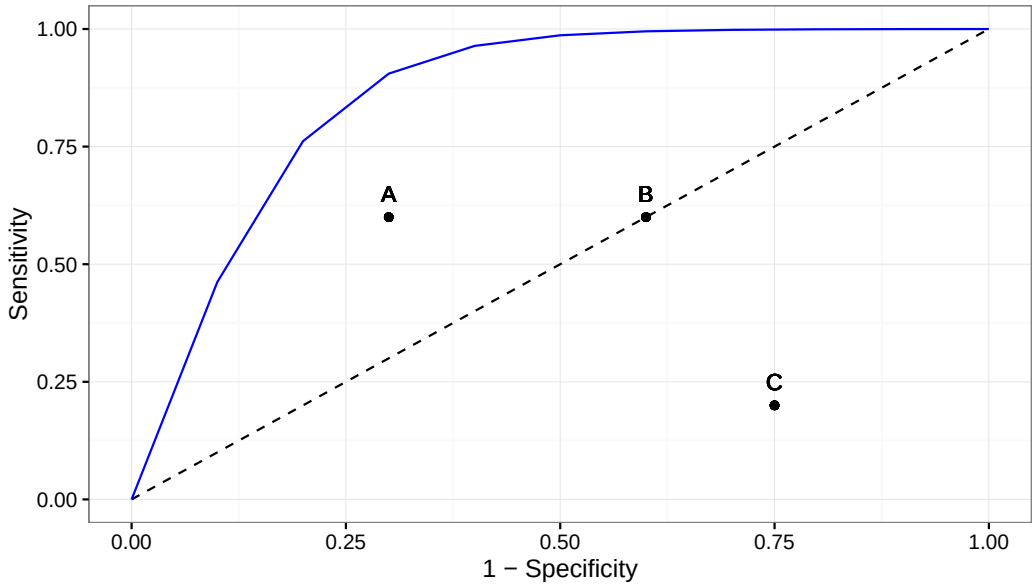


Fig. 3.2.: ROC space, where x-axis represents $1 - Specificity$ and y-axis represent the *Sensitivity*. The dotted line represent the performance of a random guess classifier. The blue curve is an example of a ROC.

the trapezoid to approximate the AUC. A different approach is by calculating the Mann-Whitney U statistic and transform it to the AUC [27]. In this thesis, the first approach is used, as this has the advantage to display the actual ROC curve rather than only calculating the AUC. Analysis of the ROC curves for a multiclass problem should be met with caution, as the ROC curves are only well defined for binary classifiers. Therefore, each of the ROC curve can only represent a binary classification case, which is between the class itself and its complement.

3.4.2 Algorithm Comparison

Cross-validation or resampling techniques provide point estimates of the performances which can be used to compare and identify algorithms with good properties. Validating the model using a test set can give some indication of the accuracy of the point estimates. But to compare different learners, benchmark experiments are required. This thesis uses a theoretical framework for inference problems in benchmark experiments which allows standard statistical test procedures to be used to test for differences in performances [34, 16].

Let $\mathcal{L} = \{z_1, \ldots, z_n\}$ be a learning sample of n observations and a set of $k = (1, \ldots, K)$ candidate algorithms. Each of the candidates is a two-step algorithm a. In the first step, a model is fitted based on a learning sample $\mathcal{L}$ yielding function $a(\cdot|\mathcal{L})$, which in a second step can be used to compute certain objects of interest (in this case, predictions of the response, based on the input variables). In order to compare the candidates, some performance measure $p(a, \mathcal{L})$ is used, which assesses the performance of the function $a(\cdot|\mathcal{L})$. That is the performance of algorithm a based on the learning sample $\mathcal{L}$. Since $\mathcal{L}$ is a random learning sample, $p(a, \mathcal{L})$ is a random variable whose variability is induced by the variabil-

ity of learning samples following the same data generating process as $\mathcal{L}$. Therefore, it is natural to compare the distribution of the performance measures when we need to decide whether any of the candidate algorithms performs superior to all others. The idea is to draw independent random samples from the distribution of the performance measure for an algorithm a by evaluating $p(a, \mathcal{L})$. Statistical test procedures are then used to test different hypotheses about the distributions of the performance measures. Consequently, this gives a possibility to control the error probability of falsely declaring any candidates as the best. The theoretical basis of the idea is given in The Design and Analysis of Benchmark Experiments by Hothorn et al. [34].

4 Experimental set-up

In this research, we examine whether it is possible to determine the household composition based on program level television viewing behaviour. Klein defined household composition into 4 categories, namely a man, woman, adult family and an adult family with children [47]. This definition proves to be insufficient for the Nielsen data set, as there are households with more than two adults or one adult with a child. Upon closer examination, it cannot be determined with certainty that all members of the group of adults are family members[1], as some households consists of only respondents with a similar age. Furthermore, it can be argued that a household containing similarly aged relatives have a viewing behaviour more similar to a group of similarly aged adults than to families. This observation leads us to add an extra class to the household composition definition, namely households (`H`). It is also possible for this class to have children in the household. Consequently, we also add household with children (`HC`). These additional classes still does not make our definition of household composition collectively exhaustive because certain households are not categorised, namely single adults with children. Therefore, single parents are added to include all possible household compositions that are present in the data set. The final set of classes of our definition of household composition is as follows:

- Family (`F`): a household that consists of two adults, irrespective of their gender.
- Family with children (`FC`): a household that consists of two adults, irrespective of their gender, with at least one child[2].
- Household (`H`): a household that consists of more than two adults.
- Household with children (`HC`): a household that consists of more than two adults with at least one child.
- Single female (`SF`): a household with only one adult female.
- Single male (`SM`): a household with only one adult male.
- Single female parent (`SPF`[3]): a household with only one adult female with at least one child.
- Single male parent (`SPM`[4]): a household with only one adult male with at least one child.

[1] This is sometimes possible if there is a large variance in the age distribution of the respondents in a household.
[2] A child is defined as a person younger than 18 years old.
[3] The letters `SPF` stand for single parent female.
[4] The letters `SPM` stand for single parent male.

This is very similar to the definition by eurostat[5] [35]. Eurostat also uses categories similar to household (`H`) and household with children (`HC`), but they call it "Other type of household without children" and "Other type of household with children". The only major difference between the definition used in this thesis and eurostat is the inclusion of gender in single adult households (i.e. single adults and single parents). Eurostat ignores gender and therefore, has only 6 classes, while this thesis uses a household composition definition with 8 classes. We decide not to implement the classes defined by eurostat, but keep our current definition as this is more in line with the research done by Klein.

The data processing and machine learning is mainly run on an $i5-4210$ 1.7GHz laptop with 8GB memory in R[6]. Every algorithm is run in parallel on a single machine, while multiple machines are used to experiment with different parameters and/or algorithms. The main hardware problem that we face is the memory limitation, since in R each thread during one cross-validation step will need roughly 3GB of memory. To alleviate this problem, we use sparse data structures, although not all packages in R support them. Therefore, some tweaking is required. The primary packages that provide an implementation of algorithms used in this research are `glmnet`[7] for regularized logistic regression, `gbm`[8] for stochastic gradient boosting and `kernlab`[9] for support vector machines.

4.1 Data Processing

This thesis uses the Nielsen National People Meter (NNPM) data. This data is loaded into a database that is subjected to change during the period of the thesis. Nielsen delivers the data in specialised files and these are read into the database. Occasionally, Nielsen also sends additional files that change previously loaded data or adds data to an earlier period. This makes it difficult to work with the most recent database. Thus, this thesis is based on a snapshot of the database in May 2015. For this thesis, two parts of the data set are used, namely the socio-demographics and the television viewing behaviour part. These are stored in different tables. A simplified version of the relevant tables is given in Figure 4.1. The socio-demographics are stored in the `Respondent Demographics` table. This table contains the `Household ID`, which shows the respondents living in the same household and the number of children in that household (`Number of Children`)[10]. Since each respondent can have multiple viewing statements, the `Respondent Demographics` has a (1:m) relationship with `Tv Program Viewing`. The `Tv Program Viewing` table contains the `Telecast ID`, which links a respondent to a broadcast of a program. It also contains the date[11] (`Date`) at which the respondent had watched the telecast, the offset (`Offset Seconds`) and the duration (`Duration Seconds`). The offset (`Offset Seconds`) is used to measure how much of the telecast elapsed before the respondent started watching the program and the duration (`Duration Seconds`) is how long the respondent was watching

[5] Apparently the name should not be capitalised as this is also not the case on their website.
[6] see `https://cran.r-project.org/`.
[7] see `https://cran.r-project.org/web/packages/glmnet/index.html`.
[8] see `https://cran.r-project.org/web/packages/gbm/index.html`.
[9] see `https://cran.r-project.org/web/packages/kernlab/index.html`.
[10] This variable is used, because there are no viewing statements of children with an age smaller than 3.
[11] Specifically, the start date at which the respondent started watching the telecast.

the program. The `Telecast Id` has a (m:1) relationship with each entry of the table `Telecast`. The `Telecast` table contains information regarding the television broadcast, such as program duration (`Duration Seconds`), the broadcast date (`Date`), `Channel Name` and `Program Name`.

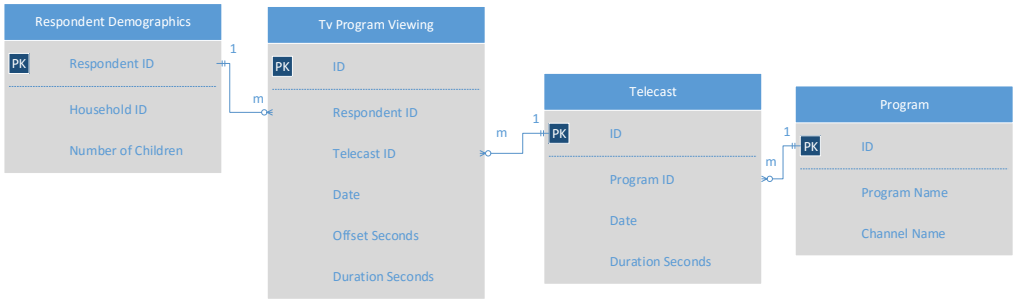


Fig. 4.1.: An extremely simplified schema of the database containing the NNPM data used in this thesis.

The `Respondent Demographics` table consists of 150 076 respondents, which are constantly moving in and out the panel. In Section 4.1.1, we will discuss how we deal with this problem. These respondents are living in 46 205 households. The television viewing behaviour data set (`Tv Program Viewing`) consists of almost 260 million viewing statements, where each viewing statement consist of one respondent watching one program at a certain date time and duration. As a consequence, when multiple respondents in the same household watch a program at the same time, it will generate the same number of viewing statements as the number of respondents watching that program.

4.1.1 Filtering and Unification

The data set presented in the previous section is too large to be handled and does not fit into memory. Thus, we use only the first month of 2014 as our data set. This narrows the number of respondents down to 84 199 corresponding to 21 614 households. Time-shifted viewing statements[12] and viewing statements of visitors are not included to narrow the scope of this research. Syndicated programs are also removed, since at the time of writing this data set was incomplete and the programs do not have a starting time or ending time. However, they do have a duration. Additionally, programs that are not watched or only watched by one person are removed. This reduces the number of programs from 7 109 to 6 168.

The respondents are constantly subjected to change. Hence, Nielsen uses the unification process to define a consistent sample universe for a specified reporting interval. This ensures that all respondents have the same opportunity to view a program [73]. In this thesis, we are also using the default unification process as defined by Nielsen. This requires that each respondent watches at least 75% of the days in January 2014. In addition, 16 households have inconsistent household level socio-demographics, i.e. one respondent reports he/she is living with one child and the other with two children, even though they live in the same

[12] Time-shifted viewing are recorded programs that are watched on a different time than the actual broadcast.

household. These households are also removed. The final data set after unification contains 6 168 programs, 32 131 respondents and 15 591 households.

4.1.2 Aggregation and Feature Creation

In order to train our learning algorithms on this data set, some aggregation method should be used to aggregate the viewing statements at a respondent level to a household level. Several exploratory experiments are conducted to determine which features should be included. The exploratory experiments are conducted using 200 households, with a distribution similar to the full data set. The features that are used varies with each experiment.

A trivial way to aggregate the viewing statements is by summing the durations of viewing statements per program for each household. This creates *total duration* features for each program that is watched at least by one household. Another way to aggregate the viewing statements is by counting the number of viewing statements per program for each household. This creates *total frequency* features for each program. We argue that *total frequency* features contain information that cannot be observed with only *total duration* features. For example, a household that spend a lot of time watching a program, does not need to do this frequently. Similarly, households that watch short programs do not necessarily have a high total duration for that program. The downside of using these features is that many of these features probably do not contain information on the household structure. This can be observed in unpopular programs, where households with many respondents, still do not have a high duration nor frequency. It is also possible for a popular program to have a high duration, even though there is only one adult in the household.

A different type of feature we also consider is combining the total duration and total frequency features into the *total average duration* features. This is calculated by dividing the total duration feature by the total frequency feature for each program. The disadvantage of using these features instead of total duration/frequency features is that it is less informative, as it will not possible to determine the effect of the duration of a program and the frequency individually. Exploratory research also shows that the models using only the total average duration features have very similar cross-validation accuracies as using only total duration features.

We also consider incorporating a time element in the feature set. One way to achieve this is by calculating the total duration for each weekday instead of the whole period. Programs that do not air nor have any viewers are removed. Experiments using these features drastically reduce the cross-validation accuracies compared to using only the total duration features. This suggests that all algorithms prefer less sparse features. Additional experiments using total duration per hour or per hour and weekday also shows a decrease in the cross-validation accuracy.

A different method of incorporating a time element into the feature set is by adding time-based features. A trivial way to do this is by adding binary features that indicate whether a household has watched TV at a certain hour and weekday. We use this new feature set to conduct our experiments and observed that very few of these time-based features were selected. The features that were selected had only a small effect on the prediction.

The cross-validation accuracies were also similar to a feature set without these time-based features. Therefore, we do not include the time-based features in our feature set.

We also examine whether it is possible to create features from *viewing sessions*, instead of only viewing statements. Viewing sessions are defined as intervals in which respondents or households watch programs sequentially. Viewing sessions differ from viewing statements as viewing sessions capture the duration one sits in front of the television, while viewing statements only shows the duration of a respondent watching a program. Unfortunately, we are unable to extract this data using viewing statements as we miss data regarding commercials. We use an example to illustrate this problem. Suppose we have viewing statements V_1 and V_2 with intervals $i_1 = [s_1, e_1]$ and $i_2 = [s_2, e_2]$ from some household, where s_i and e_i indicate the start and end time of viewing statement i. We also assume that $s_1 < s_2$, meaning that the viewing statement V_1 is watched before V_2. If $e_1 = s_2$, it can be reasonably assumed that someone had watched V_1 and then V_2. Thus, V_1 and V_2 belong to the same viewing session. The problem is when $s_2 - e_1 < \epsilon$, where ϵ is an unknown threshold. In this case, it is unknown whether V_1 and V_2 should belong to the same viewing session as we have no information what happens between s_2 and e_1. If commercials were shown during s_2 and e_1, then V_1 and V_2 should belong to the same viewing session. However, as multiple commercials were shown, it makes determining ϵ very difficult. We also observe that in many households overlapping viewing statements can be found, where the household watched different programs at the same time. This indicates that many American households have at least two televisions, which complicates the matter even more.

Two other interesting features are the number of unique programs and channels that a household watches. One can argue that if more people are present in the household, more complicated viewing behaviour can be expected. This includes a more diverse viewing behaviour (i.e. more different programs/channels). Therefore, these features are included in our feature set.

Our final feature set contains three types of features:

- The total duration, which is calculated by summing the durations of viewing statements per program and household.
- The total frequency, which is calculated by counting the number of viewing statements per program and household.
- Different number programs and channels viewed by a household.

In the following sections, the total duration and total frequency will be referred to as *duration* and *frequency*. Any references to the duration per viewing statement will be made explicitly.

4.1.3 Data Splitting

The next step is dividing the data into a training and test set. We use the training set to determine the hyperparameters with cross-validation and in construct the model, while we use the test set to validate the model. We choose a training set such that we maximise the training set size, while still allowing for parallel processing. In addition, all data that is not used during the training phase is removed from memory. This results in a training set of 10% of the data, which corresponds to 1 563 households, while the test set contains 90% of the data or 14 028 households. A larger training set can be used, if we are willing to sacrifice parallel processing. This is not feasible in our case, as this would lengthen the model training process to probably several months in total.

4.2 Data Characteristics

In this section, we present the interesting characteristics of our data. We first examine the household composition data and its class distribution. Subsequently, we examine the duration and frequency data. We do this by showing the data distributions as a whole and per household structure. We also perform correlation analysis between the durations/frequencies of different programs. Finally, we present the distribution of the number of programs (`NPrograms`) and channels (`NChannels`) over different household compositions.

4.2.1 Household Composition

The target variable in this research is household composition. The most common household structure in the data are families (`F`) with 5 150 households, followed by families with children (`FC`) with 3 103 households (see Figure 4.3 and Table 4.1). Households (`H`) and households with children (`HC`) have 1 783 and 1 487 observations. Among the single adult household structures, the largest are single females (`SF`) with 2 050 observations. This is 55% more than single males (`SM`), with only 1 321 observations. The smallest groups are single parents households (`SPF` and `SPM`)with 413 classified as single female parents (`SPF`) and 284 single male parents (`SPM`). The training data set is constructed such that it has a similar class distribution as the test data set (see Figure 4.3).

Tab. 4.1.: Distribution of household compositions in January 2014.

Household Structure	`F`	`FC`	`H`	`HC`	`SF`	`SM`	`SPF`	`SPM`
Frequency	5 150	3 103	1 783	1 487	2 050	1 321	413	284

4.2.2 Duration and Frequency Features

The program duration data given by Nielsen is presented in seconds; however all duration data is dividable by 60, so it can be safely assumed that it is either rounded off to the nearest minute or only reported in minute intervals. The resulting data matrix $\mathbf{D}$ for the duration data is $15\,591 \times 6\,168$. The frequency count is defined as the number of viewing statements for a given program in a given household. This results in a matrix $\mathbf{F}$ of the same

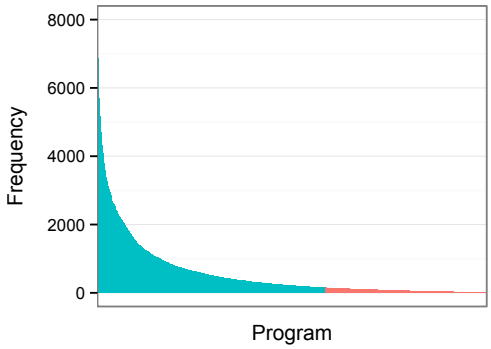


Fig. 4.2.: Duration data distribution per program, where the collection of programs containing the top 95% of the data is coloured blue.

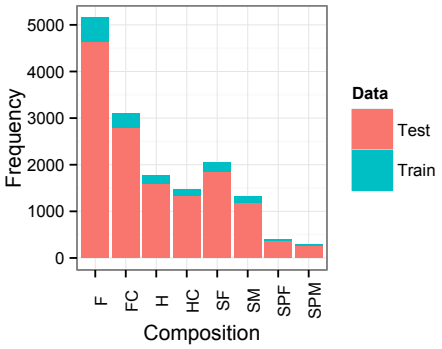


Fig. 4.3.: Household distribution in the train and test data set.

size, giving the full data matrix a size of 15 591 × 12 326. In addition, the total number of unique programs and channels viewed are also added, resulting in a final data matrix of 15 591 × 12 328.

This data set is extremely sparse with only 3.5% of the matrix containing non-zero data. Sorting it based on which program has the highest number of non-zero duration data gives us Figure 4.2. 95% of the non-zero data reside in 3 698 programs, while the other programs account for 5% of the data. The top 5 programs with the most duration data are: `AMC MOVIE` (7 995), `FOX NFC CHAMPIONSHIP` (7 942), `FX MOVIE PRIME` (7 687), `BIG BANG THEORY, THE` (7 528) and `NCIS` (7 516). These programs account for 1.1% of the non-zero duration data. The same plot can be made using only the frequency data, but due to the large similarity, the plot is not included.

Average Duration and Frequency

We now continue to examine the *average duration* and the *average frequency* of the data. Average duration is calculated by the duration of a program divided by the number of households, which is 15 591. Also, we also define the contribution of each household structure to the average duration of a program. This is done by summing the duration per household structure and then dividing it by the total number of households.

The top 3 programs with the longest average durations are `LAW & ORDER: SVU`, `NCIS` and `SPONGEBOB` (see Figure 4.4a). For `LAW & ORDER: SVU`, we see that families (`F`) contribute the most towards the average duration, namely 72.62 minutes. Households with children (`HC`) contribute 35.07 minutes. It is very remarkable that families (`F`) contribute twice as much towards the average duration of `LAW & ORDER: SVU` than households with children (`HC`), considering that a family (`F`) only contains two respondents, while a household contains at least 3 respondents. Households (`H`), families with children (`FC`) and single females (`SF`) contribute 33.76, 31.09 and 29.70 minutes, respectively. It is also surprising, that single females (`SF`) contribute almost as much as households (`H`). Single males (`SM`), single female parents (`SPF`) and single male parents (`SPM`) contribute significantly less to `LAW & ORDER: SVU` with only 8.67, 5.08 and 1.37 minutes. If we compare a single female (`SF`) with a

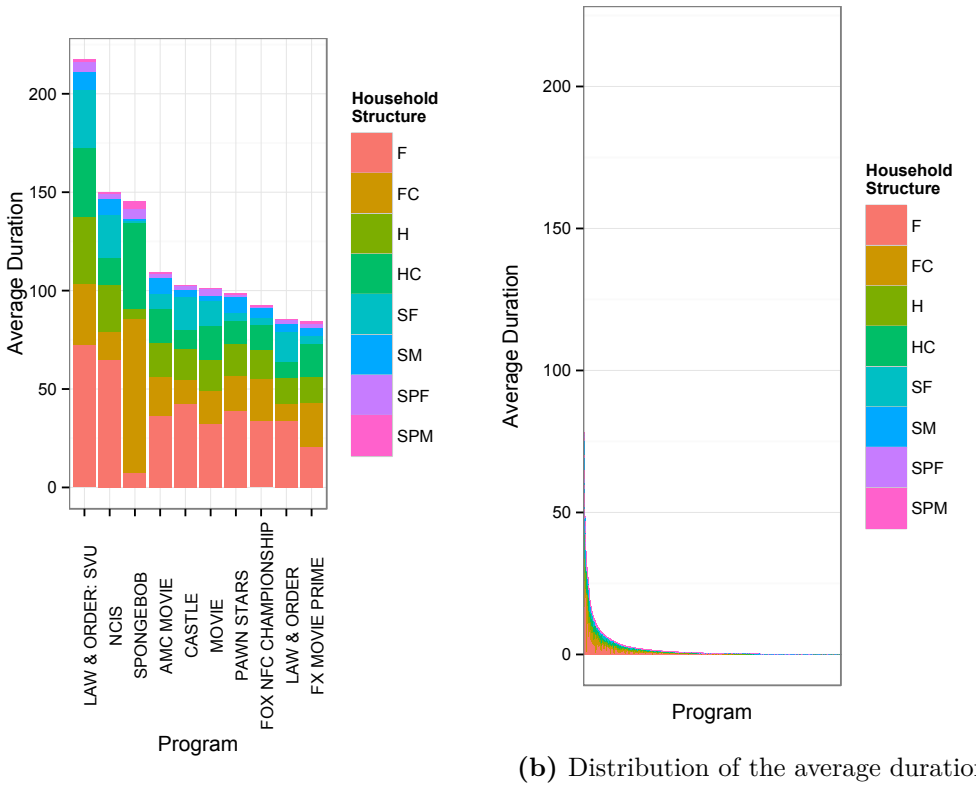
single female parent (SPF), we observe that a single female parent (SPF) contributes 80% less towards the average duration of LAW & ORDER: SVU. For the second program NCIS, the average duration is about 153 minutes. Again, the families (F) contribute the most with 64.91 minutes. Households (H) come second with 23.46 minutes. Single females (SF) are ranked third with 21.75 minutes. Families and households with children (FC, HC) are ranked fourth and fifth, contributing 14.49 and 13.96 minutes, respectively. Again, we observe that single females (SF) contribute more than households (H). Single males contribute 8.31 minutes. Single female and male parents (SPF, SPM) are ranked at the bottom, with both structures contributing less than 3 minutes. For SPONGEBOB, families with children (FC) and households with children (HC) contribute the most with 78.31 and 43.27 minutes, as expected from a children program. Though, single parents (SPF and SPM) do not seem to contribute significantly more to SPONGEBOB than their childless counterparts (SF, SM). If we look beyond the top three programs, we observe that the average duration decreases slightly for lower ranked programs. The top 100 programs with the longest duration are similar to the top 100 programs with the most data entries. Both sets seem to be almost equal, though programs are not ranked at the same positions. For example, AMC MOVIE, FOX NFC CHAMPIONSHIP and FX MOVIE PRIME have the most data entry points, but they appear lower based on their average duration.

We now turn to the contributions of each household structure to the average duration of each program. We see that in the top 10 programs with the highest average duration, families (F) contribute the most, except for SPONGEBOB (see Figure 4.4a). This suggests that every household structure probably has a different ranking based on the average duration. These rankings will be examined in detail in later parts of this section. We also note that there are some large differences between families (F) and families with children (FC). Their contributions to some programs are very different as seen by the long lines in the soft red area representing a family (F) (see Figure 4.4b).

For the *average frequency*, similar plots can be made. The average frequency is defined as the summed frequency per program divided by the number of households. Similarly to the average duration, the contributions per household structure can also be defined by the summed frequency per household structure divided by the number of households.

The highest average viewing frequency is equal to 9.11 for LAW & ORDER: SVU (see Figure 4.5a). Largely, this is contributed by families (F). The contributions from different classes to average viewing frequency of LAW & ORDER: SVU is similar to the contributions to the average duration.

If we compare the top 10 rankings of the average duration and frequency, we notice that only three programs differ (see Figure 4.4a and 4.5a). These are WEATHER CENTER LIVE, BIG BANG THEORY, THE and COLLEGE BKBL REG SSN. BIG BANG THEORY, THE is positioned at rank 13 with 78.89 minutes, based on the ranking of the average duration. WEATHER CENTER LIVE is also positioned rather high at position 41 based on average duration of 46.83 minutes. COLLEGE BKBL REG SSN is a bit odd, as it is positioned at 2 015 based on the average duration of 43.71 minutes. This suggests that households watch COLLEGE BKBL REG SSN frequent but not very long.



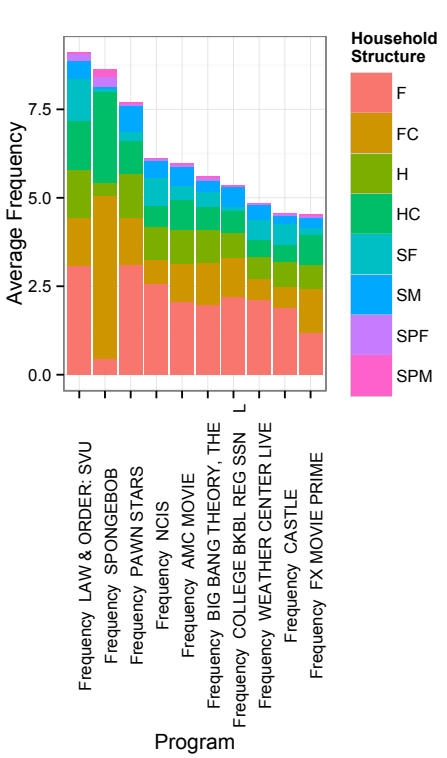
(a) Top 10 programs with the highest average duration.

(b) Distribution of the average duration per program over different household compositions.

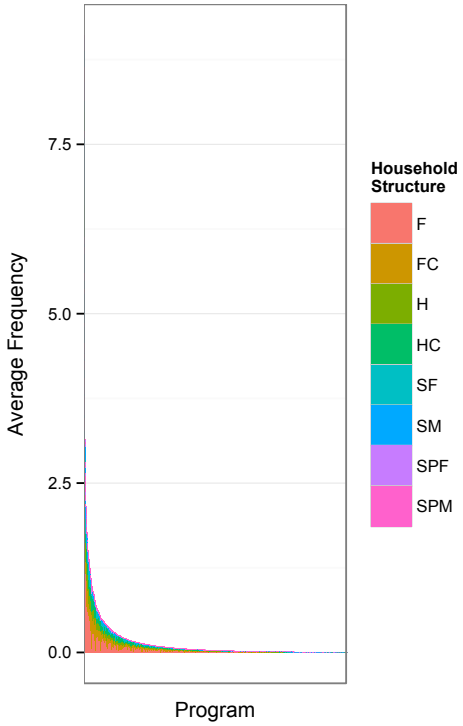
Fig. 4.4.: Distribution of the average duration of programs.

To get a better understanding of the data, we construct a scatter plot of the two programs with the highest duration/frequency. The top 2 programs with the highest average duration are chosen as the x and y-axes (see Figure 4.6). It can be observed that most points cluster at the lower left corner of the graph for all household structures. Creating additional graphs based on the top 25 programs with the highest average duration yield 300 similar graphs, with less variance in the lower quadrant. We also observe that the graphs among different classes are similar, albeit with less data points. The same scatter plots can also be made with the top 2 programs with the highest average frequency. Again, the graph contains households that are clustered towards the lower bottom corner (see Figure 4.7). For household structures with no child present, we observe very low frequencies for the program `SPONGEBOB`. This suggests that we can use the frequency of `SPONGEBOB` to identify whether a household has a child. The points in the duration scatter plots also seem to be more distributed, while the points in the frequency scatter plots are more concentrated in the axes. Further creating these scatter plots based on the top 25 programs with the highest average frequency yield 300 similar graphs, but no obvious patterns are be observed.

By computing the average duration and frequency for all programs, we can examine the distributions of the average duration and frequency. From the distribution of the average duration, we observe that the top 2 520 programs with the highest average duration account for 95% of the total average duration (see Figure 4.8a). For frequency, 2 791 programs with the highest frequency account for 95% of the total average frequency (see Figure 4.8b). Thus, about 40% of the programs account for 95% of the duration and frequency. This



(a) Top 10 programs with the highest average frequency.



(b) Distribution of the average frequency per program over different household compositions.

Fig. 4.5.: Distribution of the average frequency per program.

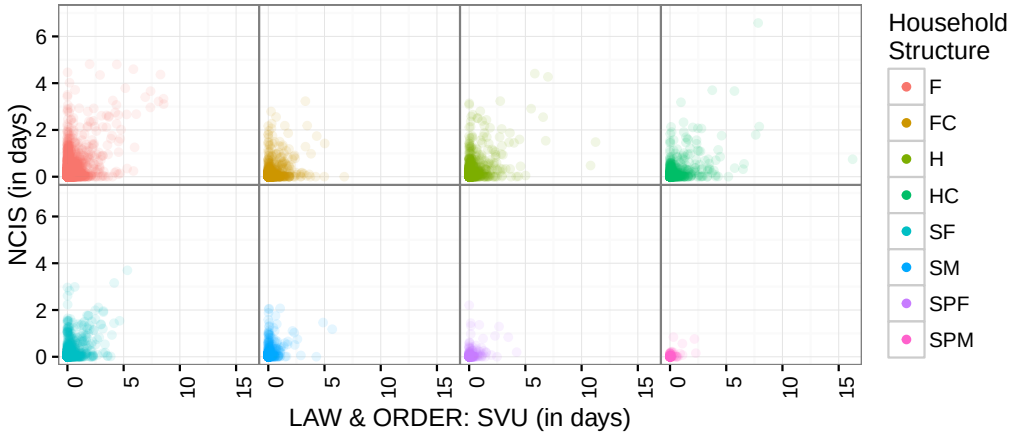


Fig. 4.6.: Scatter plot of the duration in days of each household for the top 2 programs with the longest average duration.

is also in line with the conclusions drawn by observing the non-zero matrix elements from the duration and frequency data set. The *overall average duration*[13] is 2.52 minutes, with a median of 0.42 minutes, indicating that the duration data is left skewed (see Table 4.2).

[13]The overall average duration is defined as the mean of the average durations.

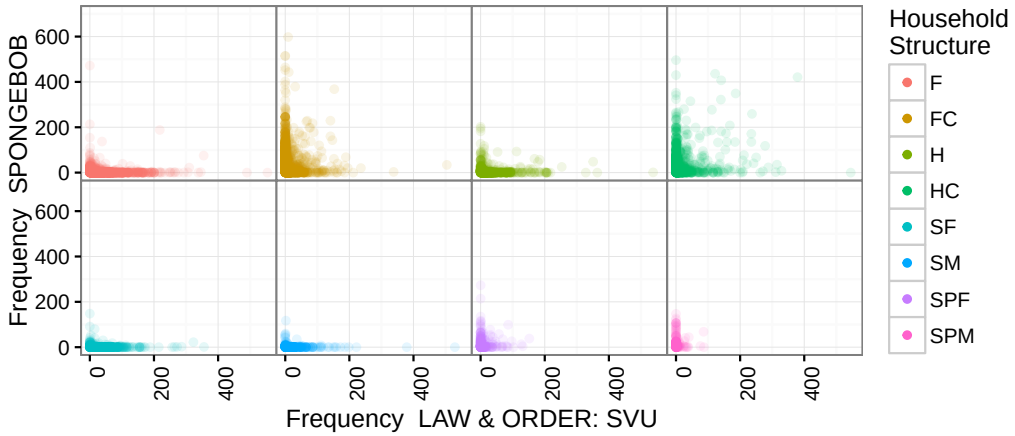
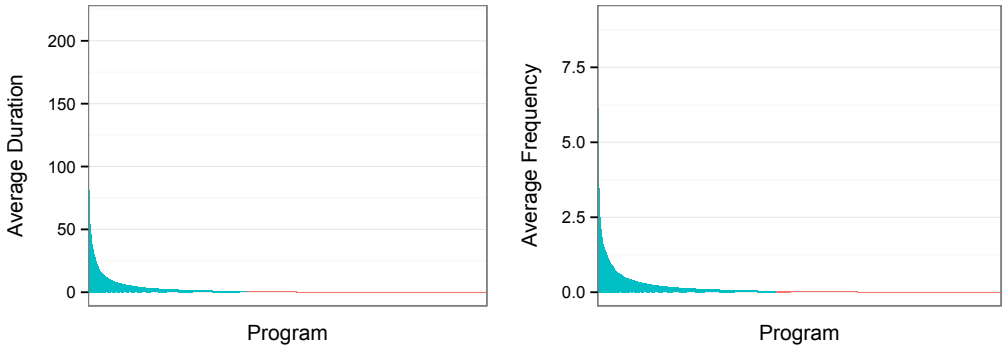


Fig. 4.7.: Scatter plot of the frequency of each household for the top 2 programs with the highest average frequency.

The *overal average frequency*[14] is 0.37, with a median of 0.07, which indicates that most programs do not have viewing statements and the frequency data is also left skewed.

Tab. 4.2.: Summary statistics of the average duration and frequency of programs.

Type	Min	Median	Mean	Max	Variance
Duration	0.00	0.42	2.52	217.33	0.13
Frequency	0.00	0.07	0.37	22.52	1.10



(a) Distribution of the average durations of all programs, where the top 2 520 programs account for 95% of the total average duration.

(b) Distribution of the average frequency of all programs, where the top 2 791 programs account for 95% of the total average frequency.

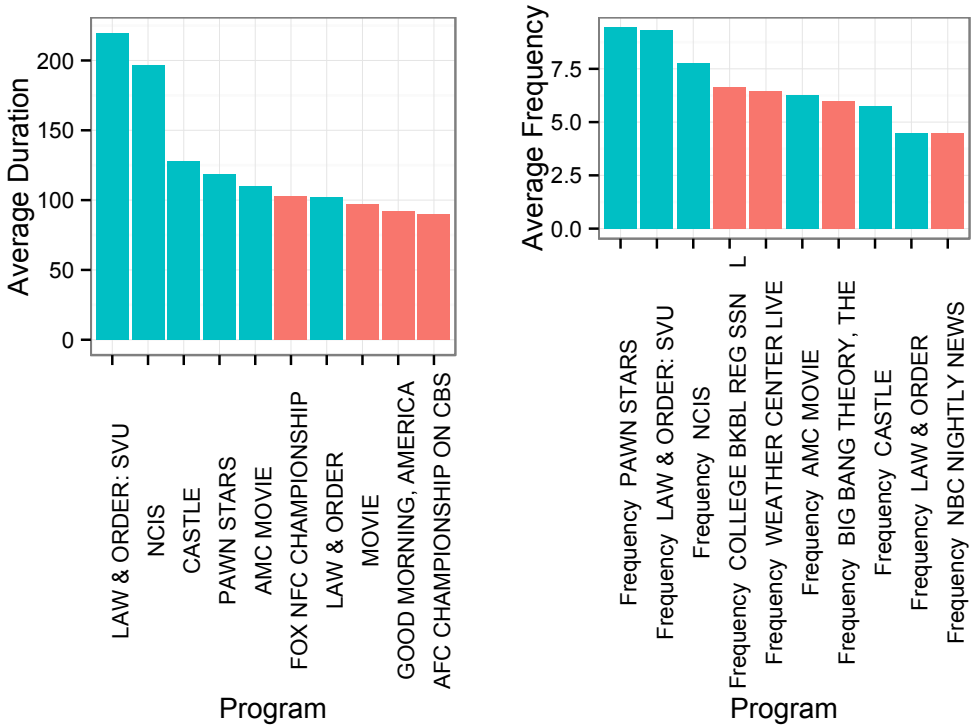
Fig. 4.8.: Distribution of the average duration and frequencies of all programs.

Data Characteristics per Household Composition

We now examine the data grouped by their household structure. Since we have 8 classes and each of those classes have probably a different ranking of their top 10 programs with the highest average duration and frequency, we get 16 different graphs.

[14]The overall average frequency is defined as the mean of the average frequency.

The program with the highest average duration for all household structures is exactly the same as for families (F) (see Figure 4.9a). We also observe that in families (F), children programs, such as SPONGEBOB, are not present in the top 10 average duration/frequency. If we compare the top 10 average durations with the top 10 frequencies, we observe that the ranking has changed. In addition, four programs that were present in the top 10 average duration ranking do not show up in the top 10 average frequency ranking (see Figure 4.9a and 4.9b). Informative programs such NBC NIGHTLY NEWS and WEATHER CENTER LIVE are frequently watched, but their average duration is only 81.44 and 59.04 minutes. This places them at position 13 and 31 based on average duration. The programs COLLEGE BKBL REG SSN L and BIG BANG THEORY, THE are placed at position 15 and 12 based on average duration. The programs MOVIE, GOOD MORNING, AMERICA and FOX NFC CHAMPIONSHIP are ranked 15, 25 and 26 based on average frequency. This suggests that the ranking between programs based on average duration and frequency are similar. The only outlier seems to be AFC CHAMPIONSHIP ON CBS, which is placed on position 55 based on average frequency, while it is placed on rank 10 based on average duration. The average frequency of AFC CHAMPIONSHIP ON CBS is 2.21.



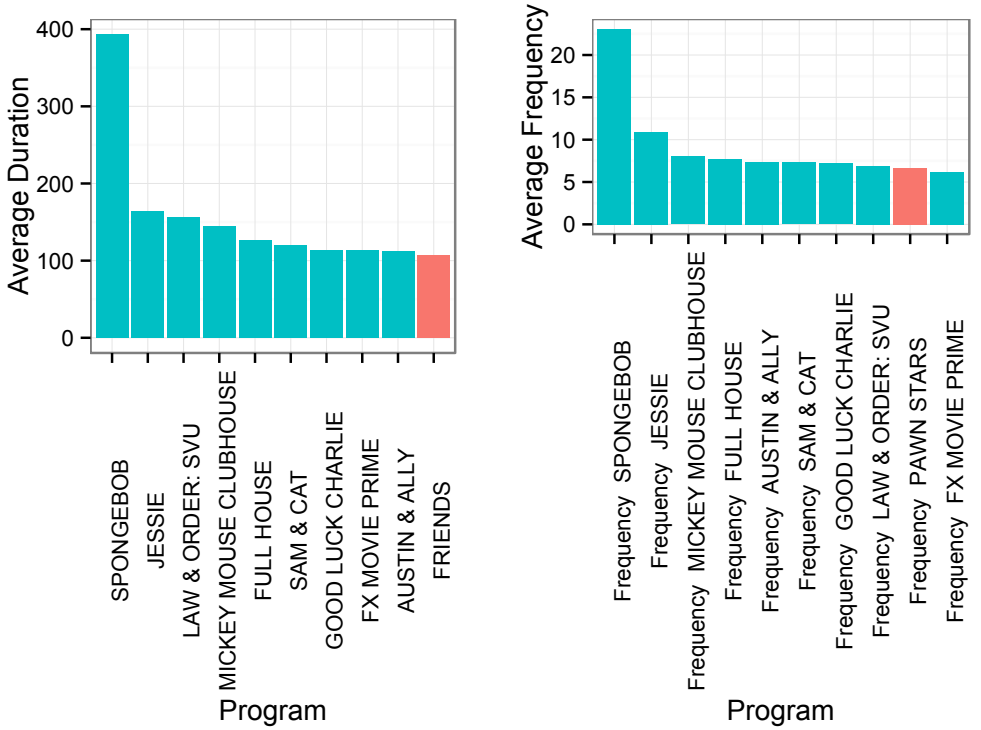
(a) Top 10 programs based on average duration for families (F).

(b) Top 10 programs based on average frequency for families (F).

Fig. 4.9.: Top 10 programs based on average duration and frequency for families (F). The blue bars represent programs that are common to both graphs, while red bars represent the differences.

In families with children (FC), we immediately notice that the top two programs based on average duration and frequency are SPONGEBOB and JESSIE. SPONGEBOB has an average duration of 393.49 minutes and JESSIE 163.65 minutes (see Figure 4.10a). Comparing the top 10 rankings based on average duration and frequency, we notice that only one program differs (see Figure 4.10a and 4.10b). In the top 10 average duration ranking, PAWN STARS

is not not present. This program is ranked at position 15 based on its average duration. In the top 10 average frequency ranking, FRIENDS is not present. This program is ranked at position 13 based on its average frequency. We also note that a significant portion of the families with children (FC) should contain teenagers since teenage programs are also visible in the top 10 ranking, namely, SAM & CAT, GOOD LUCK CHARLIE and AUSTIN & ALLY. We now compare the top 10 average durations between a families (F) and a families with children (FC). We notice that most programs are displaced by children programs, such as SPONGEBOB, JESSIE and MICKEY MOUSE CLUBHOUSE. This is also the case if we compare their average frequencies. The differences in the y-axis seem to suggest that families with children (FC) watch a lot more television. This is however not the case. The overall average duration for families (F) is 2.32 minutes, while for families with children (FC) it is 2.92 minutes. This suggests that families with children (FC) only watch slight more television. Furthermore, children seem to be less diverse in their program choice than adults as SPONGEBOB seems to be the favourite by far, while the average duration of LAW & ORDER: SVU and NCIS are relatively close. The overall average frequency for families (F) is 0.14, while for families with children (FC) it is 0.18. This supports our findings that families with children (FC) only watch slightly more television.



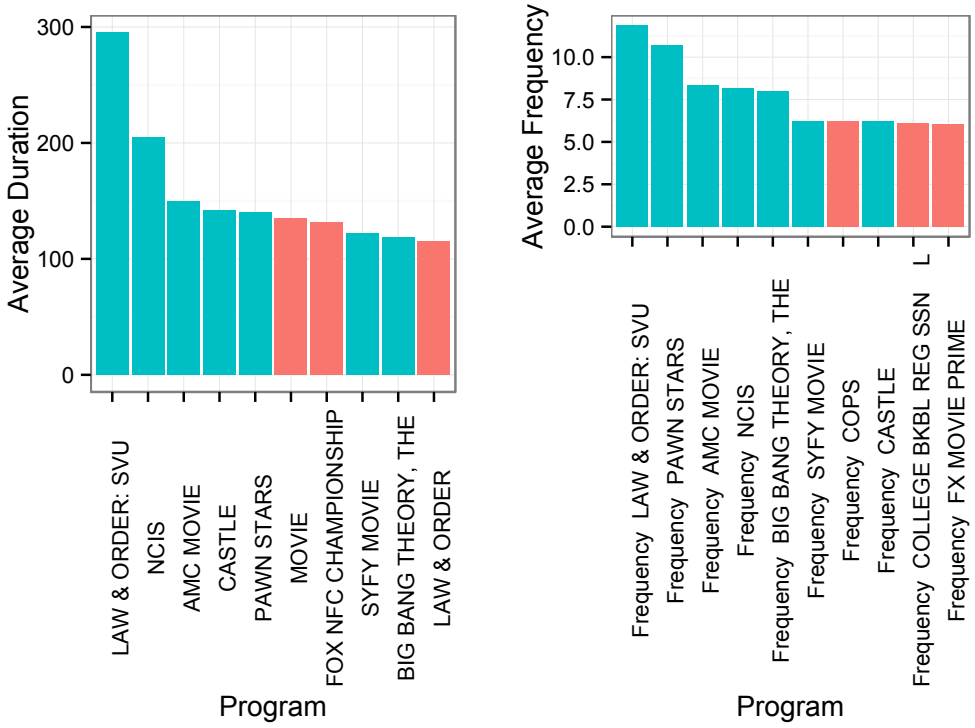
(a) Top 10 programs based on average duration for families with children (FC).

(b) Top 10 programs based on average frequency for families with children (FC).

Fig. 4.10.: Top 10 programs based on average duration and frequency for families with children (FC). The blue bars represent programs that are common to both graphs, while red bars represent the differences.

Households (H) have a slightly higher overall average duration than families (F), with 3.19 minutes. This is most likely due to households (H) having more members than families (F). The top 10 programs based on the average duration between households (H) and families (F)

are very similar (see Figure 4.11a). Families (F) do not seem to watch BIG BANG THEORY, THE very often, while households (H) do. Households do not have SYFY MOVIE in their top 10 average duration ranking, while families (F) do. Households (H) also have a slightly higher overall average frequency than families (F) with an overall frequency of 0.18. Again, this can be attributed to the size of the household structure. By comparing the ranking based on average duration and frequency for households (H), we see that three programs differ in each graph. In the average duration ranking, these are MOVIE, FOX NFC CHAMPIONSHIP and LAW & ORDER. These are ranked 14, 28 and 17 based on the average frequency (see Figure 4.11b). In the average frequency plot, the three programs that differ are COPS, COLLEGE BKBL REG SSN L and FX MOVIE PRIME. These are ranked 32, 24 and 11 with a duration of 69.53, 73.50 and 112.91 minutes, respectively. The programs that differ seem to be ranking rather close to the top 10.



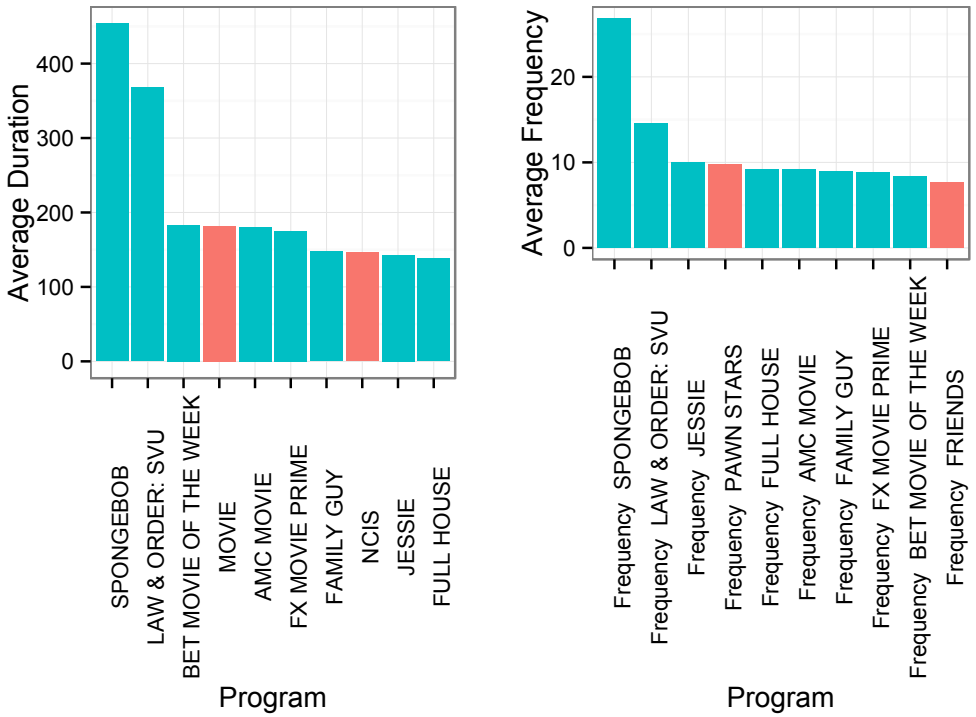
(a) Top 10 programs based on average duration for households (H).

(b) Top 10 programs based on average frequency for households (H).

Fig. 4.11.: Top 10 programs based on average duration and frequency for households (H). The blue bars represent programs that are common to both graphs, while red bars represent the differences.

The program with the highest average duration for households with children (HC) is SPONGE BOB, closely followed by LAW & ORDER: SVU (see Figure 4.12a). This is different from families with children (FC), where SPONGEBOB is leading by far (see Figure 4.10a). The overall duration for households with children (HC) is 4.12 minutes, which is the highest overall duration across all household structures. If we compare households with children (HC) to households (H), we observe the same displacement as between families with (FC) and without children (F). The displacement of programs with children programs is the most in families with children (FC). SPONGEBOB also dominates all other programs based on the average frequency in households with children (HC) (see Figure 4.12b). SPONGEBOB has an average

frequency of 26.86, while LAW & ORDER: SVU has only an average frequency of 14.52. This is a recurring pattern, as this was also the case in families with children (FC). This leads us to conclude that although each episode of SPONGEBOB is relatively short, it is watched often. For shows that have a longer duration, such as LAW & ORDER: SVU, we see that their average frequency is relatively low.



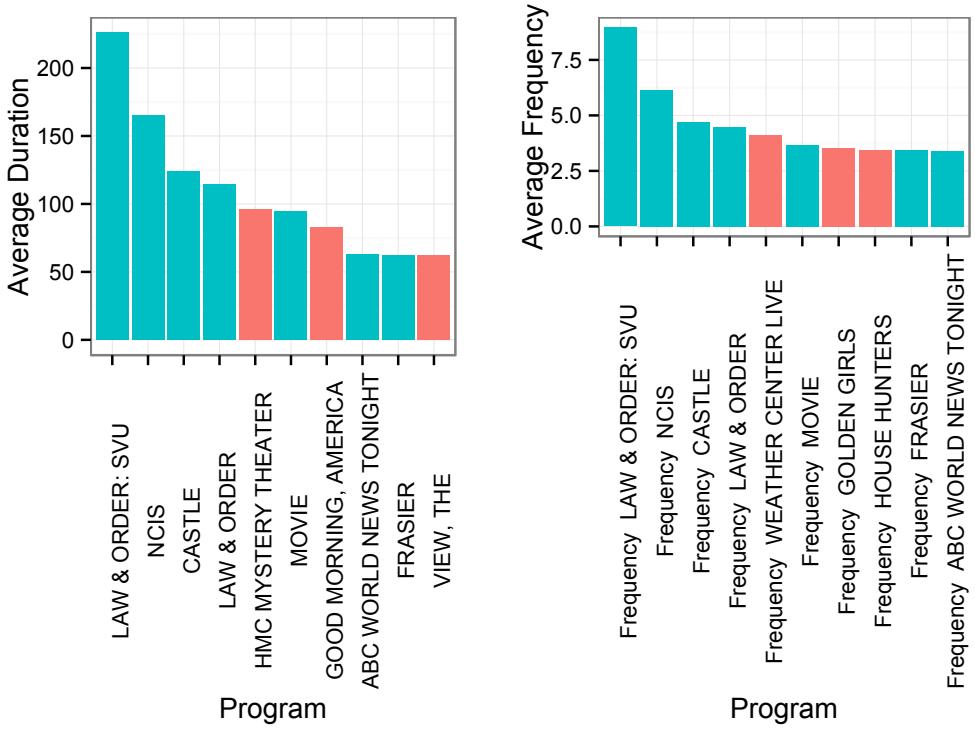
(a) Top 10 programs based on average duration for households with children (HC).

(b) Top 10 programs based on average frequency for households with children (HC).

Fig. 4.12.: Top 10 programs based on average duration and frequency for households with children (HC). The blue bars represent programs that are common to both graphs, while red bars represent the differences.

We now turn our attention to single adult household structures. We start with single females (SF) and see that their viewing behaviour is similar to that of families (F) (see Figure 4.13a). Single females have an average duration of 225.90 minutes for LAW & ORDER: SVU, while families (F) have an average duration of 219.86 minutes. The similarity in the average duration can also be seen in the programs NCIS and CASTLE. We also note the lack of sport programs such as FOX NFC CHAMPIONSHIP and AFC CHAMPIONSHIP ON CBS in the top 10 programs with the highest average durations. FOX NFC CHAMPIONSHIP and AFC CHAMPIONSHIP ON CBS are placed at position 60 and 65. This confirms the stereotype image that females in general do not often watch sports. The average duration of both programs are 28.73 and 27.22 minutes, respectively. Based on empirical evidence, we can only hypothesise that the average durations of sports programs in families (F and FC) and households (H and HC) are primarily contributed by males. The overall average duration for single females (SF) is 1.64 minutes. This is less than households structures with more than one adult. The top 10 programs with the highest frequencies for the single females (SF) also shows very similar behaviour to families (F) (see Figure 4.13b). Single females (SF)

have an average frequency of 8.96 for LAW & ORDER: SVU, which is similar to the average frequency in families (F) of 9.33. The overall average frequency for single females (SF) is 0.085. This suggests that single females (SF) only watch a select number of programs, while households with multiple adults watch a larger variety of programs. If we are to compare the average duration and frequency ranking for single females (SF), we observe that three programs differ. In the average duration ranking, we miss WEATHER CENTER LIVE, GOLDEN GIRLS and HOUSE HUNTERS. WEATHER CENTER LIVE is placed at position 25 in the average duration ranking, with an average duration of only 43.98 minutes. GOLDEN GIRLS and HOUSE HUNTERS are placed at positions 12 and 23 with an average duration of 60.20 and 51.02 minutes. In the average frequency top 10 ranking, we miss HMC MYSTERY THEATER, GOOD MORNING, AMERICA and VIEW, THE. They are placed at position 11, 14 and 16, with an average frequency of 3.17, 2.89 and 2.68, respectively.



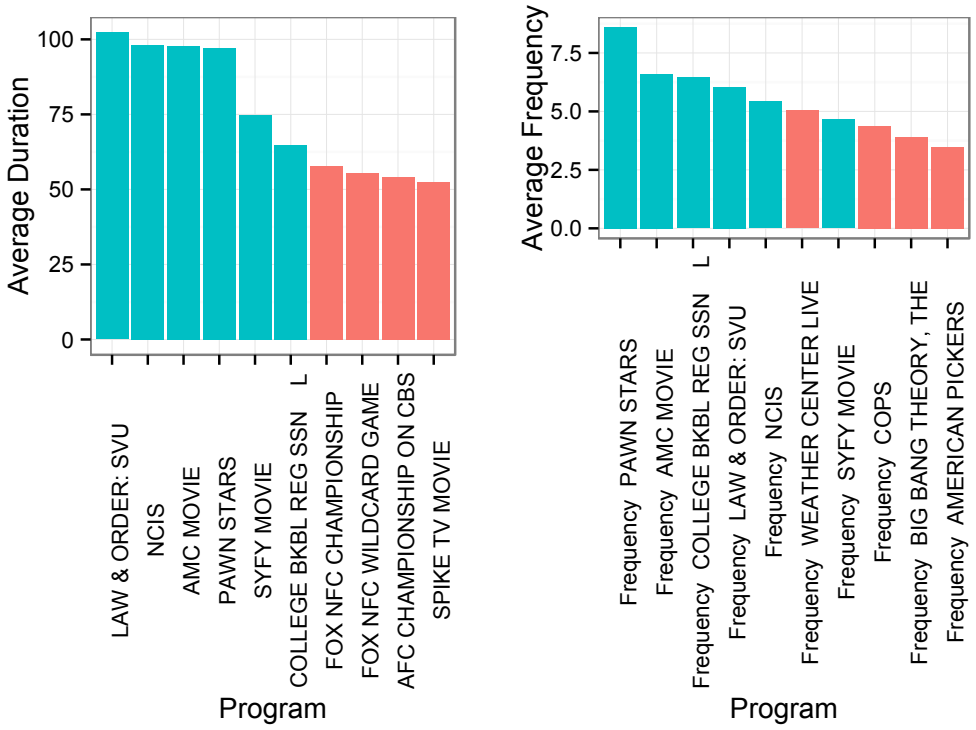
(a) Top 10 programs based on average duration for single females (SF).

(b) Top 10 programs based on average frequency for single females (SF).

Fig. 4.13.: Top 10 programs based on average duration and frequency for single females (SF). The blue bars represent programs that are common to both graphs, while red bars represent the differences.

Single males (SM) in general watch less television. This can be seen by the overall average duration, which is only 1.49 minutes. The most popular show based on duration is LAW & ORDER: SVU with an average duration of 102.30 minutes (see Figure 4.14a). This suggests that both genders like to watch LAW & ORDER: SVU. In the top 10 average duration ranking we also observe sports programs, such as COLLEGE BKBL REG SSN L, AFC CHAMPIONSHIP ON CBS and FOX NFC WILDCARD GAME. This strengthens the stereotype viewing behaviour of men. The overall average frequency of single males (SM) is 0.10, indicating that single males (SM) have more viewing statements than single females (SF) (see Figure 4.14b). If we compare the top 10 average duration ranking with the top 10 frequency ranking, we

notice that four programs differ. The main differences are primarily sport programs. FOX NFC CHAMPIONSHIP, FOX NFC WILDCARD GAME and AFC CHAMPIONSHIP ON CBS not in the top 10 ranking based on average frequency, but are ranked at positions 26, 28 and 40, respectively. This indicates that single males (SM) probably watch the entire program. The programs that are present in the top 10 ranking based on durations are WEATHER CENTER LIVE, COPS, BIG BANG THEORY, THE and AMERICAN PICKERS. These are placed at position 24, 23, 16 and 12, respectively.



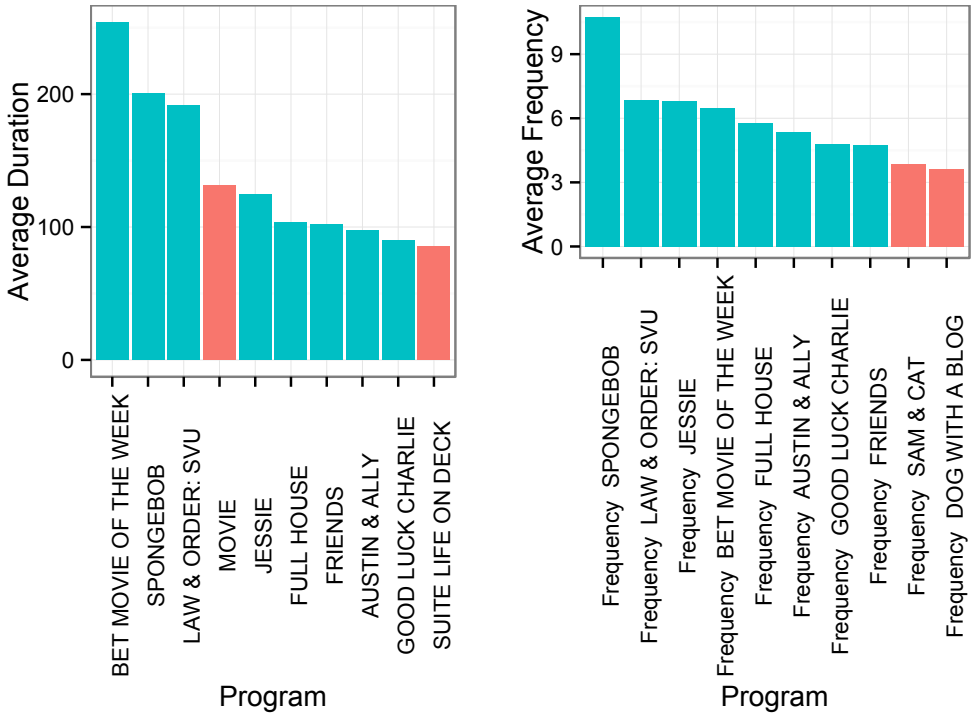
(a) Top 10 programs based on average duration for single males (SM).

(b) Top 10 programs based on average frequency for single males (SM).

Fig. 4.14.: Top 10 programs based on average duration and frequency for single males (SM). The blue bars represent programs that are common to both graphs, while red bars represent the differences.

The overall average duration for single female parents (SPF) is 2.03 minutes. This is a bit higher than for single females (SF). We also observe that the ranking of programs are different when compared to single females (SF). The program with the highest duration is BET MOVIE OF THE WEEK with 254.11 minutes, followed by SPONGEBOB with 200.75 minutes. While single females (SF) watch the most LAW & ORDER: SVU. Single female parents (SPF) rank LAW & ORDER: SVU at position 3 with an average duration of 191.62 minute. We also note that the average duration of SPONGEBOB is the highest as with the previous household structures with children. This suggests that the children of single female parents (SPF) watch less SPONGEBOB. Five children shows are listed in the top 10 programs based on duration. These are SPONGEBOB, JESSIE, AUSTIN & ALLY, GOOD LUCK CHARLIE and SUITE LIFE ON DECK. We now turn to the average frequency. The overall average frequency for single female parents (SPF) is 0.09. This is not very different from single females (SF), which have an overall average frequency of 0.085. SPONGEBOB is ranked first based on average frequency, which is consistent with our suggestion that SPONGEBOB has a short duration,

but is watched frequently. We also observe that BET MOVIE OF THE WEEK is ranked rather low compared to its ranking using average duration. We observe in the top 10 ranking of programs by their duration and frequency, that only two programs differ. The average duration has MOVIE and SUITE LIFE ON DECK, which are placed at position 12 and 13 based on average frequency. The average frequency has SAM & CAT and DOG WITH A BLOG, which are placed 13 and 18 based on average duration. Thus, we can conclude that both rankings are relatively similar.

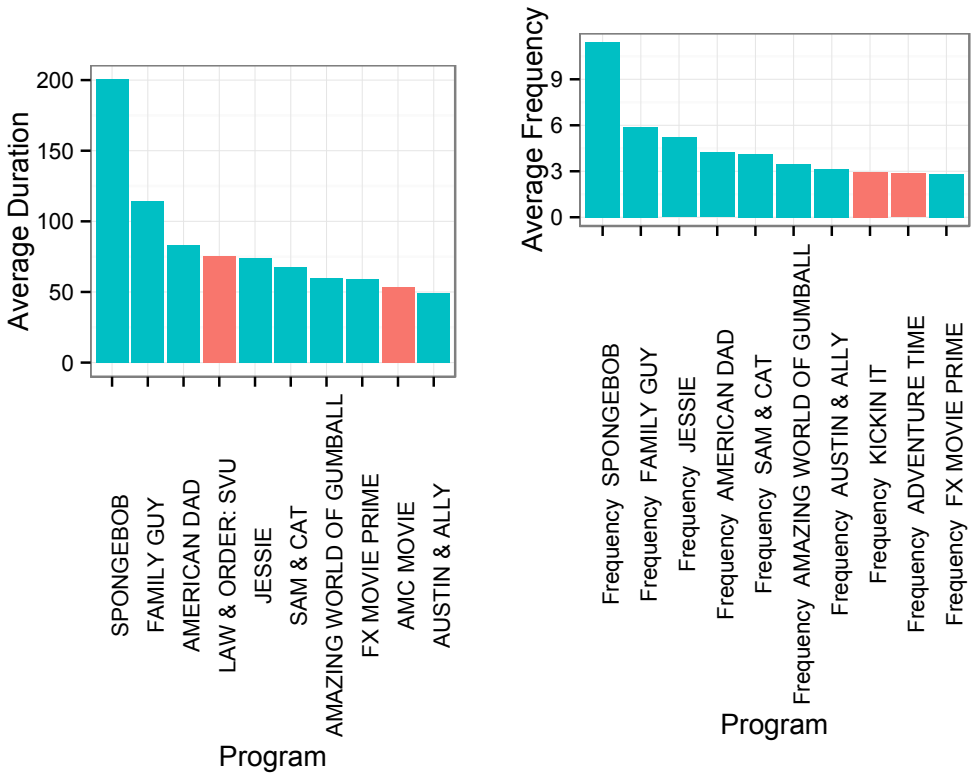


(a) Top 10 programs based on average duration for single female parents (SPF).

(b) Top 10 programs based on average frequency for single female parents (SPF).

Fig. 4.15.: Top 10 programs based on average duration and frequency for single female parents (SPF). The blue bars represent programs that are common to both graphs, while red bars represent the differences.

Single male parents (SPM) have a ranking that differs significantly from single males (SM) (see Figure 4.16a). All sports programs that were in the top 10 based on average duration, have now been shifted to rank 21 and further. The number of children shows increase drastically with now 7 programs in the top 10 based on average duration. The overall average duration has decreased to 1.25, which suggests that single male parents (SPM) themselves are watching less television. The average frequency top 10 ranking also seems to be more agreeable with the ranking given by the duration as now only two programs differ (see Figure 4.16b). The ranking based on the average frequency shows that all programs in the top 10 are children programs except for FX MOVIE PRIME. LAW & ORDER: SVU is placed 11 based on average frequency. The overall average frequency of single male parents (SPM) is 0.069, which is also much lower than the 0.10 of single males (SM).



(a) Top 10 programs based on average duration for single male parents (SPM).

(b) Top 10 programs based on average frequency for single male parents (SPM).

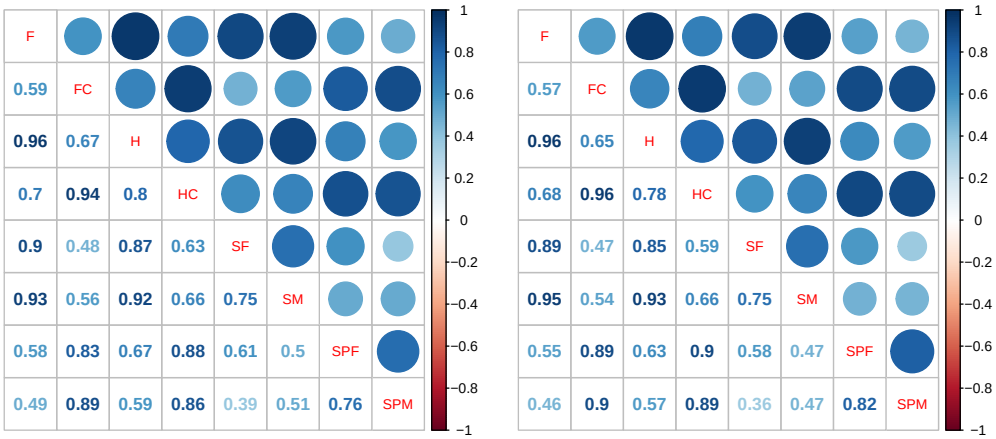
Fig. 4.16.: Top 10 programs based on average duration and frequency for single male parents (SPM). The blue bars represent programs that are common to both graphs, while red bars represent the differences.

In conclusion, families (F) have a mixed viewing behaviour. If families have children (FC), the average viewing duration increases and some programs are replaced by children programs. The average frequencies also increase. Households (H) tend have a higher average duration, but this can be attributed to the number of respondents in a household. The overall average frequency is also higher compared to families (F). If households have children (HC), the overall average duration increases, but this increase is less when compared to the effect in families. Single females (SF) have the highest overall average duration among all single adult households. If the single female is a parent (SPF), the overall average duration increases slightly and some programs are watched less by the female. This effect is still small compared to single males (SM). For single males (SM), we observe that some programs in the top 10 programs with the highest average durations are sport programs. These programs are replaced by children programs if a child is present. In addition, the overall average duration and frequency is drastically reduced.

Correlation Analysis

Using the average duration and frequency per household, it is also possible to determine the correlation between household structures. This is done by computing the correlations between the average duration per program per household structure. The correlations show that many household structures have viewing behaviours that are largely correlated (see

Figure 4.17a). Only 3 pairs have a correlation less than 0.5. The most correlated household structures are families (F) and households (H). Families with children (FC) and households with children (HC) also have a high correlation. If we compare the correlation of household structures with and without children, we notice that it should be easier to distinguish whether a child is present or not than distinguishing between the number of adults in a household. This can be seen by the lower correlations between the household structures with and without a child. The correlations between the average frequencies of household structures show that the correlations are higher than those using average durations (see Figure 4.17b).



(a) Average duration **(b)** Average frequency

Fig. 4.17.: Correlation between household structures measured by the average duration/ frequency, where the lower triangular matrix show the correlation coefficients and the upper triangular matrix show the magnitude of the correlation.

It is also possible to calculate the correlations between programs. This is done by computing the correlations between two features of the same type. The correlations between program durations are presented in Figure 4.18. The distribution of the correlations between all program duration pairs show that the majority of programs are not heavily correlated. None of the programs are perfectly correlated and are only perfectly correlated due to rounding. The most correlated programs based on viewing durations are between `MINNESOTA GOPHER HOCKEY L` and `GOPHERS LIVE PREGAME L`. The programs have a correlation of 0.996. Two households watch `GOPHERS LIVE PREGAME L`, while 8 households watch `MINNESOTA GOPHER HOCKEY L`. The average durations between the two programs are not very similar; therefore, we conclude that the high correlation between this pair can be explained by the low number of viewers. The correlation between `AQUA TEEN HUNGERFORCE` and `SQUIDBILLIES` paint a different picture. This is the second most correlated pair, with 1 248 and 1 168 households watching them and an average duration of roughly 8 minutes. As both shows are part of the late night programming and are about $11 - 12$ minutes each, this suggests that most people watch the entire episode. Though, some programs are genuinely correlated, the majority of the correlated programs are due to the low number of viewers. This is also shown when investigating the correlations between program viewing frequencies and therefore the graph is not shown.

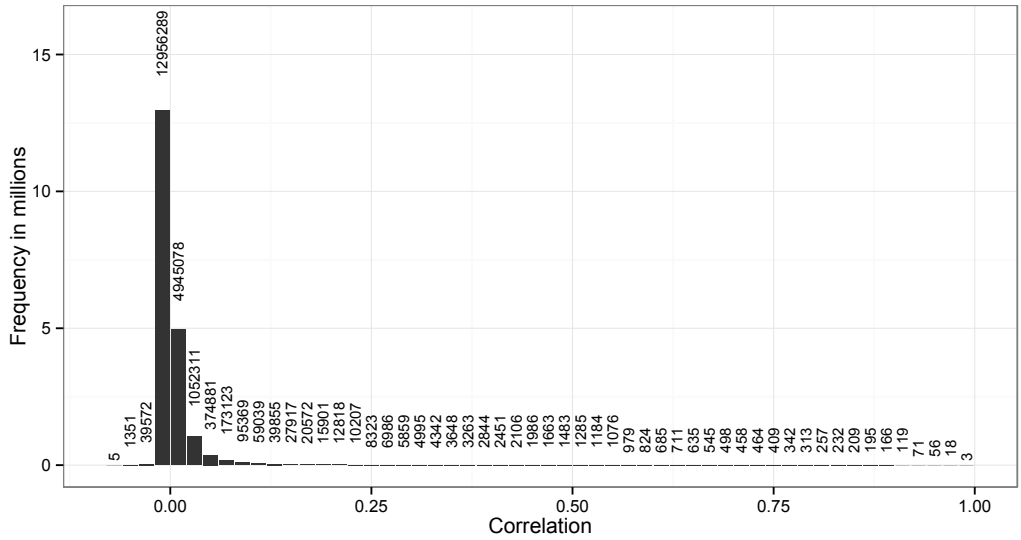


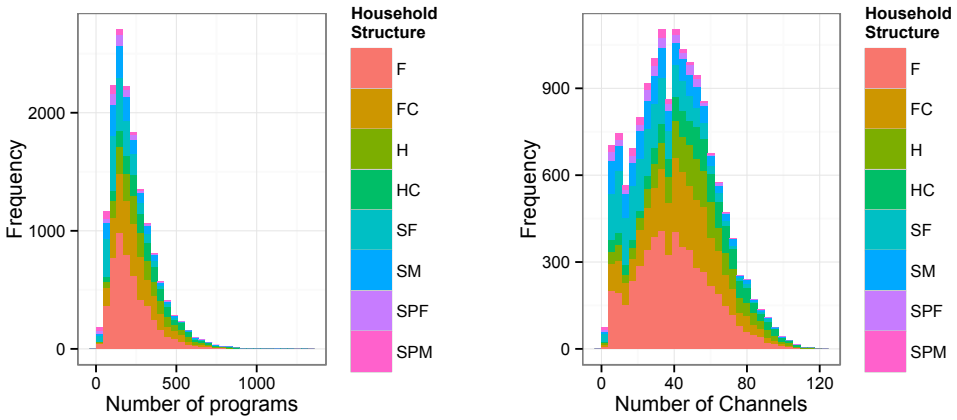
Fig. 4.18.: Distribution of correlations of the program durations between all programs.

4.2.3 Number of Unique Programs and Channels

Two additional features are added that are hard to capture using only duration and frequency features. These are the number of distinct program (`NPrograms`) and channels (`NChannels`) that a household watches. In addition, their effects are also interesting as these features might be used to identify household structures with more members as they have more complicated viewing patterns. The average number of programs that a household watch is 231.1, with a maximum of 1 277 and a minimum of 1 (see Table 4.3). The variance of the number of distinct programs is also quite high. Using a histogram, we observe that the number of distinct programs are positively skewed (see Figure 4.19a). For different household structures, the distributions of the number of distinct programs watched are also roughly the same. The second additional feature is the number of distinct channels (`NChannels`). The average is 40.8 and the median is 40, suggesting a less skewed distribution (see Table 4.3). The maximum number of channels that a household watches is 118, while the minimum is 1. The distribution of the number of distinct channels show that there is a peak at around 10, suggesting that many households only watch a select number of channels (see Figure 4.19a). The top is divided between two peaks around 40, suggesting that some people either watch a set of channels or do not.

Tab. 4.3.: Summary statistics of the distinct number of programs and channels watched.

Type	Min	Median	Mean	Max	Variance
`NPrograms`	1.0	198.0	231.1	1 277.0	19 580.8
`NChannels`	1.0	40.0	40.8	118.0	456.8



(a) Distribution of the number of distinct programs (`NPrograms`).

(b) Distribution of the number of distinct channels (`NChannels`).

Fig. 4.19.: Distributions of the number of distinct programs (`NPrograms`) and channels (`NChannels`).

5 Results

In this chapter we present the results of this thesis, which is divided into three sections. In the first section, we discuss the base models, which are the models created by the algorithms discussed in Chapter 3. These models are trained and validated using the data sets presented in Chapter 4.

In the second section, we discuss support vector machines with dimension reduction. To reduce the dimensions, we use the models created in Section 5.2 to perform feature selection on the data set. Next, we rank these features based on a feature importance measure and we introduce a new hyperparameter that selects the top most important features prior to training the support vector machines. This approach is motivated by multiple reasons. First, although support vector machines can handle a large number of features, it does not perform feature selection. This makes it relatively hard observe the relevant features that can be used to classify the household composition. Second, the presence of many irrelevant features in the data set can lead to a poor performance of support vector machines [84]. Lastly, Navot et al. [59] show that support vector machines cannot handle a large number of weakly relevant features, as this could lead to a performance comparable to a naive classifier.

The last section discusses a variant of a cascading classification model. Cascading classification models have been used successfully in object detection models [80]. Cascading classification models consist of multiple stages, where at the first stage one aspect of the target variable is classified and at each following stage a different aspect is classified with a data set that is extended by the predictions of the previous stage. The motivation of using this approach is that our target variable consists of three aspects, namely the presence of a child, the number of adults in the households and the gender of adults if only one adult is present. We do not report the gender of households when more than one adult is present, therefore, we group the two latter aspects. This allows us to train two different models, each one specialising in classifying one aspect, which can be combined to get the original target variable. In addition, this also allows us to study which aspects of the target variable can pose difficulties for our algorithms.

5.1 Naming Conventions

Before proceeding with the results, some naming conventions in this chapter should be discussed. Each algorithm can be abbreviated as follows: elastic net regularized multinomial logistic regression ($GLMNET$[1]), stochastic gradient boosting (SGB) and support vector machines (SVM). Support vector machines behave differently with different kernels. There-

[1] It is called GLMNET due to the R package used.

fore, we append the first letter of the kernel to its abbreviation (i.e. *SVML* for a linear kernel, *SVMP* for a polynomial kernel and *SVMR* for a radial basis kernel). These abbreviations are primarily used in subsections where algorithms are compared. In subsections where only one learning algorithm is used, models can be distinguished by the performance measure that is used in determining the hyperparameters. Thus, models in these sections are generally referred to using the performance measure and the learning algorithm used is omitted (e.g. accuracy model). For stochastic gradient boosting, we have decided to use the abbreviation *GBM* (gradient boosting model), as it is more commonly referred to in the literature[2]. In situations where ambiguity may arise when referring to a certain model, both the performance measure and the learning algorithms are stated (e.g. Kappa SVML model).

5.2 Base Models

This section presents the main results of the learning algorithms run on the data set. The base models use the standard features that have been discussed in the previous chapter (see Chapter 4). Each algorithm uses 5-fold cross-validation to determine the hyperparameters associated with the maximum performance measures. This results in at most three models, which is the case when different performance measures give different hyperparameter values. Next, we validate each model against the test set and compare their performances. Finally, if there are multiple models, we choose one model to study its properties and elaborate on its classification results. This allows us to gain more insights into the workings and possible improvements of the models. The selection procedure is purely subjective and by no means an indication which performance measure selects the best hyperparameters[3]. Another reason for this selection is that presenting a detailed analysis for each model seems to be excessive.

5.2.1 Multinomial Logistic Models

The first learning algorithm that will be discussed is the multinomial logistic model combined with elastic net regularization (*GLMNET* model). Using a 5-fold cross-validation, we train three different models, each using the hyperparameters that maximise a different performance measure. The three performance measures we use are: accuracy, Kappa statistic (*Kappa*) and area under the curve (*AUC*).

Hyperparameter Search Space

The model with the highest cross-validation accuracy is at $\alpha = 0.55$ and $\lambda = 0.066$. With these hyperparameters a cross-validation accuracy of 0.406 is achieved (see Figure 5.1). This suggests a generalisation error of 59.38%, which is an improvement of 7.58 percentage points over the naive classification, where all observations are classified in the majority class, family (`F`). The accompanying cross-validation Kappa and AUC values at this level are 0.140 and 0.652. Based on the guidelines discussed in Section 3.4.2, these values suggest that

[2]The word "model" is already incorporated in the abbreviation
[3]For this, additional research is needed, which is outside the scope of this thesis.

the model performs poorly. The hyperparameter search space using accuracy as the performance measure shows a stable area. There are no large differences in the cross-validation accuracies when changing the hyperparameters slightly. The lowest cross-validation accuracies are achieved by setting both the regularization and mixing parameter high, $\alpha \geq 0.55$ and $\lambda \geq 0.09$. These cross-validation accuracies are the same as the naive classifier. Another interesting aspect of the search space is that in each row the same behaviour can be seen, with the exception of the last row. For example, in the first row, the cross-validation accuracies increase, as the regularization parameter λ increases. This behaviour continues up to a certain threshold. After that threshold, the cross-validation accuracy quickly diminishes to levels corresponding to a naive classifier. In addition, the maximum cross-validation accuracy in every row is only slightly lower than the optimal cross-validation accuracy. This suggests that the cross-validation accuracy of 0.406 can be achieved by almost every mixing percentage. Further searching does not seem to be necessary, since it is very unlikely that significantly higher cross-validation accuracy levels are present in the area around the optimal cross-validation accuracy.

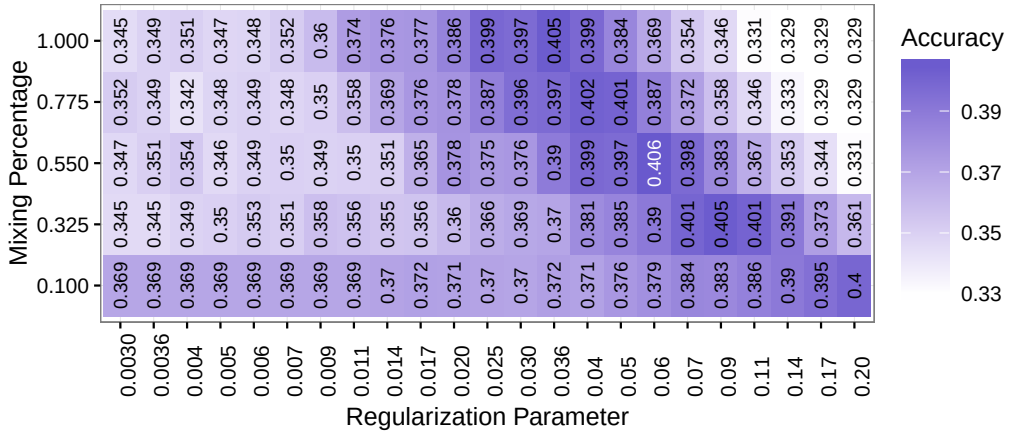
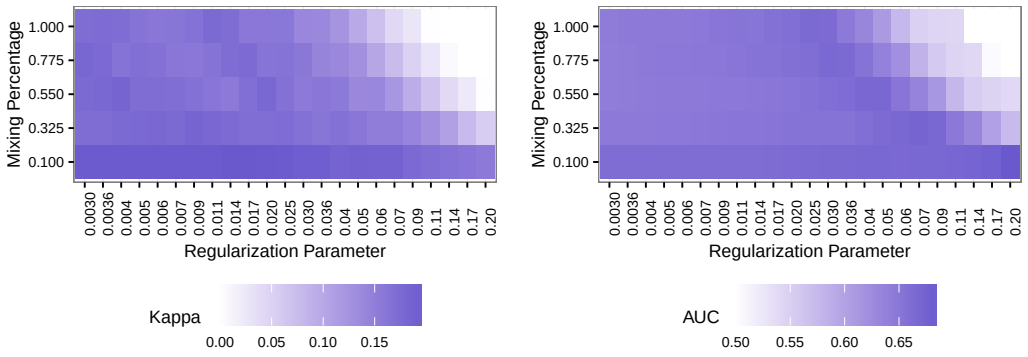


Fig. 5.1.: Level plot for cross-validation accuracy values over different combinations of the mixing (α) and regularization (λ) hyperparameters.



(a) Kappa **(b)** AUC

Fig. 5.2.: Level plot for cross-validation Kappa and AUC values over different combinations of the mixing (α) and regularization (λ) hyperparameters.

Changing the performance measure to Kappa shows a different hyperparameter search space (see Figure 5.2a). The maximum cross-validation Kappa of 0.197 is reached at $\alpha =$

0.1 and $\lambda = 0.017$. The cross-validation accuracy and AUC at this level are 0.372 and 0.663. Both cross-validation Kappa and AUC values are increased at this level, though the cross-validation accuracy went down. The Kappa performance measure also seems to favour the ridge approach as $\alpha = 0.1$, while accuracy favours a combined approach. The hyperparameter search space shows that in all cases $\alpha = 0.1$ delivers a superior cross-validation Kappa than all other values. The Kappa search space is not very regular. At $\alpha \geq 0.775$ and $\lambda = 0.2$, the cross-validation Kappa indicates that there is no agreement other than what would be expected by chance.

The AUC performance measure produces a hyperparameter search space similar to the ones using the accuracy and Kappa performance measures. The thresholding property of the accuracy search space and the high performance measure values of the Kappa search space at $\alpha = 0.1$ are both present in the AUC hyperparameter search space (see Figure 5.2b). The cross-validation AUC is at its maximum of 0.686 when $\alpha = 0.1$ and $\lambda = 0.21$. The cross-validation accuracy and Kappa at this point are 0.400 and 0.156 respectively. The cross-validation accuracy at this point is only slightly less than the accuracy model, while the cross-validation Kappa is increased. The AUC model also seems to agree with the Kappa model, as lower values of the mixing hyperparameter α gives a better a cross-validation AUC. All performance measures also seem to agree on which hyperparameter values produce the worst performing models, but this region seems to be the smallest using AUC.

Validation

Proper evaluation of the generalisation error is done by using the accuracy, Kappa and AUC models to predict the test data set and compare them to the true classification using a confusion matrix. We also compare the predictions made by the models created using the hyperparameters that are selected by different performance measures. This can be done using the performance measures itself (i.e. we can calculate the accuracy and Kappa between two sets of predictions), but to avoid confusion, we use a more general association measure, namely Cramer's V. Cramer's V is a measure of association between two nominal variables. 0 corresponds to no association between the variables, while 1 corresponds to complete association. A more detailed discussion regarding Cramer's V can be found in the Appendix C.

The confusion matrix for the accuracy model is given in Figure 5.3a. The accuracy model does not seem to classify any of the 626 single parent households into the single parents (`SPM` and `SPF`) categories (see Figure 5.3a). The majority of every household, irrespectively of their reference class, is classified as family (`F`) as indicated by the blue colours. For example, there are 255 single male parents (`SPM`) of which 215 are classified as families (`F`), which is 84.31%. This is also indicated by the level of darkness of the cell. The accuracy model achieves an accuracy of 0.406 with a 95% confidence interval of $(0.398, 0.414)$ over the test set (see Table 5.1). This is almost the same as the maximum cross-validation accuracy. The Kappa and AUC of this model on the test set are 0.137 and 0.646, which indicates a poor performance based on the guidelines discussed in Section 3.4.2 (see Table 5.1). It can be speculated that this poor performance is due to the model using only 113 features.

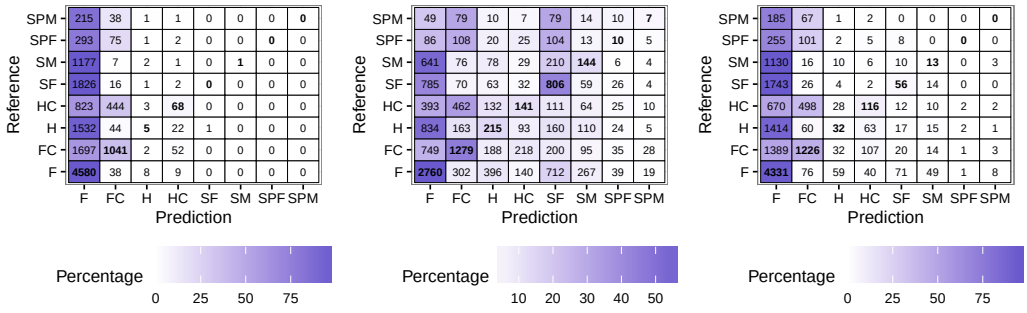
[4] Confidence interval

Tab. 5.1.: Performance measures of different models on the test set.

Model	Performance Measure		
	Accuracy (95% CI[4])	Kappa	AUC
Accuracy	0.406 (0.398, 0.414)	0.137	0.646
Kappa	0.382 (0.374, 0.390)	0.203	0.659
AUC	0.412 (0.403, 0.420)	0.163	0.688

The Kappa model is expected to perform better, as it favours the ridge approach and thus will use more features. The trained Kappa model uses 5 060 features, but this only gives an accuracy of 0.382 (see Figure 5.3b and Table 5.1). The Kappa and AUC are 0.203 and 0.659, respectively (see Table 5.1). Based on these two measurements, the Kappa model is favoured over the accuracy model. Also, the Kappa model makes predictions in more classes than the accuracy model, though the majority of the households are still classified incorrectly. Predictions of single females (`SF`) are very good compared to the accuracy model. Nonetheless, predictions made in the family category suffers severely, with only 2 760 of the 4 635 families (`F`) classified correctly.

The AUC model gives the best accuracy, with an accuracy of 0.412 and a 95% confidence interval between $(0.403, 0.420)$ (see Figure 5.3c and Table 5.1). It uses 1 259 features, which is less than 25% of the features used by the Kappa model. The confusion matrix shares similarities with the accuracy model, but it is slightly worse in classifying families (`F`) than the accuracy model. However, this is compensated by more accurate predictions in all other household structures. It also classifies less randomly than the Kappa model, which can be seen by the decrease in blue areas.

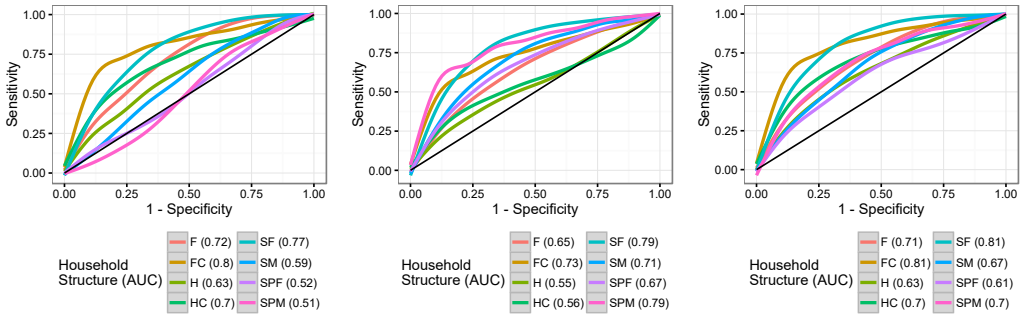


(a) Accuracy **(b)** Kappa **(c)** AUC

Fig. 5.3.: Confusion matrices of different GLMNET models on the test set.

In order to examine the AUC performance measure, we breakdown the multiclass AUC performance measure into individual ROC curves for all classes (see Figure 5.4). Since ROC curves are only well defined for the binary case, each of the ROC curves of multiclass problem should be interpreted as the ability of the algorithm to distinguish between the class and its complement. For example, the ROC curve of families (`F`), shows that the algorithm is able to distinguish between families (`F`) and non-families in 72% of the cases. The accuracy model seems to be able to classify all classes relatively well, with the exception of single parents (`SPF` and `SPM`) (see Figure 5.4a). This can be seen by the AUC values of the ROC curves of single parents (`SPF` and `SPM`), as these are very close to 0.5. This

indicates that the algorithm predicts single parents (SPF and SPM) only slightly better than random. According to the ROC curves of the Kappa model, it performs much better than the accuracy model. The lowest AUC is for households (H) and is still higher than the AUCs for single parents (SPF and SPM) in the accuracy model (see Figure 5.4b). Yet, the AUC model can still be considered the best as all its ROC curves are concave and it has the highest minimal AUC among all classes (see Figure 5.4c).



(a) Accuracy **(b)** Kappa **(c)** AUC

Fig. 5.4.: ROC curves of different classes using different models.

To compare the predictions between different models selected by different performance measures, we use Cramer's V. The accuracy and Kappa model have a Cramer's V of 0.259. Between the Kappa and AUC model, the Cramer's V is 0.362 and for the last pair this is 0.451. This indicates that the AUC and accuracy model have the highest associations, while the accuracy and Kappa model have the lowest associations. A more detailed comparison of prediction results is created by confusion matrices (see Appendix D.1.1, Figure D.1). These show that Kappa and accuracy models do not share a lot of common predictions (see Appendix D.1.1, Figure D.1a), while the AUC and accuracy models have many common predictions in families (F). Note that even though both models predict families (F), the reference classification is not stated and cannot be inferred from the displayed confusion matrix (see Appendix D.1.1, Figure D.1b). The AUC and Kappa models share common predictions in all classes, although the number of common predictions are rather low (see Appendix D.1.1, Figure D.1c).

Model Properties

To narrow the scope of this thesis, we only consider one model and study its model properties and predictions. The AUC model seems to be the most favourable in this case. It has a good accuracy and it also classifies minority households, although it does this incorrectly. Therefore, this is in our opinion the most interesting model to study.

To visualise the entire AUC model, variable trace plots are created. Variable trace plots are created by setting $\alpha = 0.1$ and varying the lambda over the x-axis. The y-axis represents the magnitude of the coefficients for the specific values of λ. For the AUC model, the variable trace plots are shown in Figure 5.5. The family equation uses 280 features (281 non-zero coefficients when including the intercept) (see Figure 5.5a). At most 1 726 features are used when the regularization parameter λ is the smallest. The families with children (FC) equation uses a bit less features, only 247 (see Figure 5.5b). Both equations have

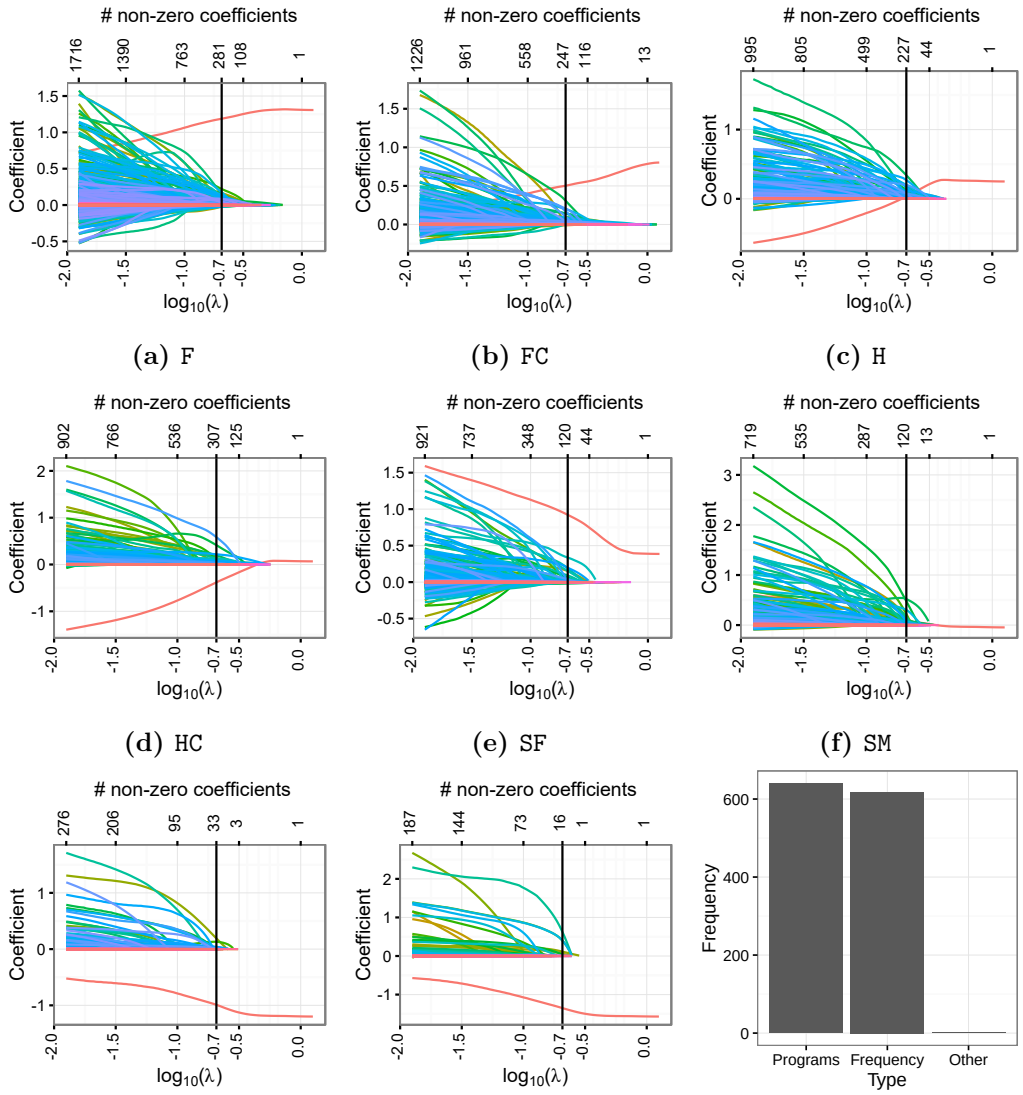
only 31 features in common. The household (H) equation uses 227 features (see Figure 5.5c). The household with children (HC) equation uses the most features (see Figure 5.5d). The single female (SF) equation is the only equation of the single households that has a large number of negative coefficients (see Figure 5.5e). The single male (SM) equation is characterised almost only positive coefficients (see Figure 5.5f). Both equations have also the same number of coefficients, although they do not share any features. The last two trace plots are characterised by their low number of coefficients (see Figure 5.5g and 5.5h). The distribution of feature types (see Figure 5.5i) shows that both feature types (i.e. frequency and program duration) are roughly equally used. Also, both of the variables `NChannels` and `NPrograms` are selected.

It can be shown that many features that are used are only present in one equation (see Appendix D.1.1, Table D.1). No feature is used in more than three equations, while only 14 features are only used in exactly three equations. These features are primarily children programs with the exception being `NPrograms` and `NChannels`. Examining each equation and the coefficients of the features can lead to the definition of a variable/feature importance. Feature importance is defined as the absolute marginal impact a feature has on the log odds of being a certain household structure. For the family (F) equation, the first features that were selected by the algorithm were `Frequency FARMERS INSURANCE OPEN-SA` and `Frequency CBS EVENING NEWS` (see Appendix D.1.1, Table D.2). These are not the most important ones, as they are only ranked at position 26 and 88 in the equation with the optimal lambda. In all equations, the first selected features are not the most important ones. The most important feature is `Frequency DEATH ROW: FINAL 24 HOURS`. Another interesting observation is that in the family with children (FC) equation has no children programs in the top three most important features, although these were selected first (see Appendix D.1.1, Table D.3). This is also the case for the household with children (HC) equation (see Appendix D.1.1, Table D.5). Single parent household (SPF and SPM) equations do have children shows in their top three most important features (see Appendix D.1.1, Tables D.8 and D.9).

Incorrectly Classified Households

In order to provide a better idea of why household are incorrectly classified, we first investigate the incorrectly classified household as an aggregate. This is done by examining the classification results of the AUC GLMNET model on the test set. We compute the average durations and frequencies for each correctly and incorrectly classified household structure, where the average duration is expressed in minutes.

We observe that correctly classified households have a higher average duration than those that are incorrectly classified, with the exception of families (F) (see Table 5.2). Families (F) that are correctly classified seem to have a duration that is half of the duration of incorrectly classified families (F). For all other household structures, the average durations of correctly classified household structures are higher than their incorrectly classified counterparts. This suggests that incorrectly classified households in general, watch less television than correctly classified households, which can contribute to their incorrect classification. The low average durations and frequencies in families (F) can be explained by it being the majority class. Households that have a different viewing behaviour compared to the general behaviour in



(a) F (b) FC (c) H

(d) HC (e) SF (f) SM

(g) SPF (h) SPM (i) Distribution of features

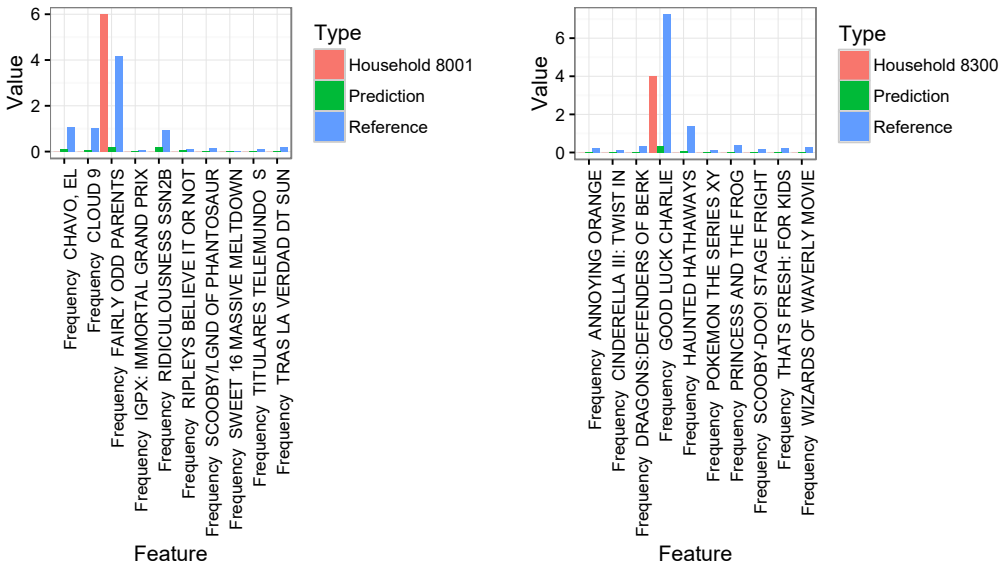
Fig. 5.5.: Trace plots of all equations of the AUC model showing the estimated coefficients against the transformed regularization parameter with the number of non-zero coefficients. The vertical line indicates the optimal transformed lambda, where $\lambda = 0.207$. The colours indicate different features.

their household structure, will be most likely classified as a family (F). This is either due to families (F) being the majority class or insufficient stimulus is given to the model to classify it to a different household structure. A combination of both reasons is very likely to be the case. Single parents (SPF and SPM) are not correctly classified by the AUC model, thus we do not have any information what the average durations and frequencies of those households are when they are correctly classified.

By inspecting households individually, a better picture can be given on why these households are classified incorrectly. Household 8 001 is a household with children (HC), but is classified as a family (F). If we take the features with the largest effect (i.e. largest absolute β) from the equation of the reference class (in this case household with children (HC)), we can graph

Tab. 5.2.: Average durations and frequencies of correctly and incorrectly classified households by the AUC GLMNET model.

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	14 501.8	27 902.5	875.4	1 690.8
FC	26 014.8	13 697.3	1 581.6	813.6
H	35 637.6	20 268.1	1 266.6	1 173.6
HC	42 003.2	25 084.4	2 659.8	1 466.4
SF	13 411.2	10 346.6	673.2	539.4
SM	21 647	9 277.9	799.2	661.2
SPF		12 951.2		603.6
SPM		7 709.2		419.4



(a) Household 8001 **(b)** Household 8300

Fig. 5.6.: Individual household viewing behaviours compared to the feature averages of the reference and prediction class.

the actual values from household 8 001 and compare them with the average values of the household with children (`HC`) and families (`F`) class. The household with children (`HC`) is referred to as the reference class, while family (`F`) the prediction class. It can be clearly seen that although the household with children (`HC`) has a high frequency of `FAIRLY ODD PARENTS`, it lacks viewing frequency with all the other shows (see Figure 5.6a). This makes Household 8 001 more similar to the prediction class, which also lacks viewing frequency, than the reference class. For household 8 300, the household structure is family with children (`FC`), but it is classified as family (`F`). Again, we see that the viewing behaviour is more similar to the prediction class, rather than the reference class, even though household 8 300 is viewing `GOOD LUCK CHARLIE` frequently (see Figure 5.6b).

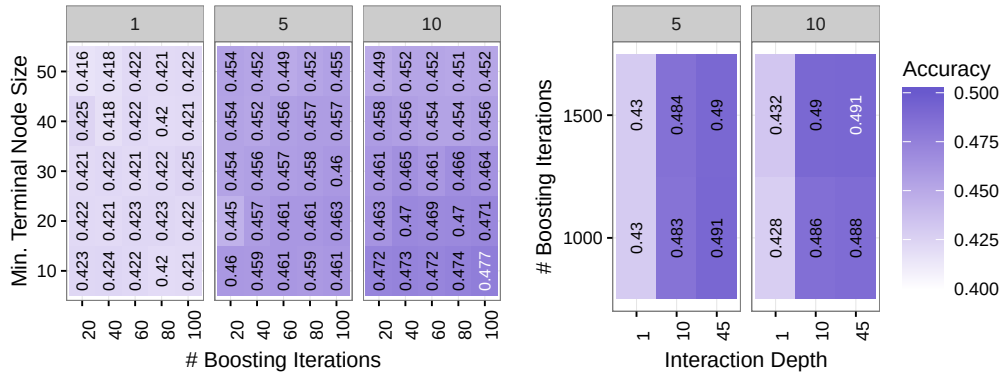
5.2.2 Stochastic Gradient Boosting Models

The second algorithm that is used is the stochastic gradient boosting (SGB) algorithm. The same 5-fold cross-validation is run with the three different performance measures. These folds are exactly the same as the previous model for comparability reasons.

Hyperparameter Search Space

Prior to running the stochastic gradient boosting algorithm, a few additional parameters have to be selected. First, the sampling size η needs to be selected. In this case, literature suggests that $\eta = 0.5$ gives the best performance, as shown in simulation studies [22, p. 5]. The second parameter that needs to be addressed is the shrinkage parameter v. In general, this parameter is not trained using cross-validation, but set prior to training as the number of boosting iterations depends on this parameter. Simulation studies show that a shrinkage of $v < 0.1$ increases performance significantly and it is generally accepted that a smaller shrinkage is better [21]. Thus, the shrinkage parameter v is set at 0.001. Any further decrease in the shrinkage parameter would exceed the computational time and memory that is available.

A 5-fold cross-validation reveals that the maximum cross-validation accuracy of 0.477 can be achieved with $M = 100$ boosting iterations, an interaction depth $s = 10$ and a minimum terminal node size $|R|_{min} = 10$. Inspection of the hyperparameter search space reveals the direction in which further improvements can be made. First, number of boosting iterations M can be increased, as this generally would also increase the accuracy as seen in Figure 5.7a. Second, an increase in interaction depth s would lead to a large increase in the cross-validation accuracy, as the maximum cross-validation accuracy with an interaction depth $s = 1$ is 0.425, while increasing the interaction depth to $s = 10$ gives a cross-validation accuracy of 0.463. The minimum terminal node size $|R|_{min}$ does not seem to have a large impact on the cross-validation accuracy, but the search space suggests that smaller values are better. The other performance measures, Kappa and AUC, agree on these findings (see Appendix D.1.2, Figure D.2 and D.3).



(a) 100 boosting iterations **(b)** 1 500 boosting iterations

Fig. 5.7.: Level plot for accuracy over different values of boosting iterations (M), interaction depths (s) and minimum observations per node ($|R|_{min}$). The shrinkage (v) and sampling fraction (η) stay constant at $v = 0.001$ and $\eta = 0.5$.

A second 5-fold cross-validation with a different search area reveals a 0.014 increase in the maximum cross-validation accuracy. A further small increase of 500 in the boosting iterations M or an increase of 5 of the interaction depth s do not seem to be very rewarding. Combining both only increases the maximum cross-validation accuracy slightly less than $1 \cdot 10^{-5}$. We also observe that a smaller minimum terminal node size $|R|_{min}$ does not seem to make a big difference, as the maximum cross-validation accuracy that can be reached with a smaller minimum terminal node size is also 0.491[5]. Therefore, further searching does not seem to be necessary. Other performance measurements agree on the selected hyperparameter values and thus we only have one model. The gradient boosting model (GBM) with only $M = 100$ boosting iterations is not further examined.

Although simulations show that $\eta = 0.5$ is optimal, the question is whether it also applies to this data set, as only 30% of the features contain 95% of the data. Increasing $\eta > 0.5$ would allow the algorithm to select better features and thus it can increase the accuracy. Experimenting with a higher η, shows that the increase in computational time does not outweigh the potential increase in accuracy. Using an $\eta = 0.9$, we can show that the maximum cross-validation accuracy is 0.475 with $M = 100$ boosting iterations, interaction depth $s = 10$ and minimum terminal node size $|R|_{min} = 10$ (see Appendix D.1.2, Figure D.4). This is slightly less than the 0.477 cross-validation accuracy achieved with $\eta = 0.5$ with the same set of hyperparameters. Experiments were also done with $v = 0.1$ and $v = 0.01$, but during training small convergence issues can be observed. The training deviance increased in several iterations, while using smaller v values a more stable convergence can be observed. This is most likely due to overfitting. The maximum cross-validation accuracy is 0.425[6] with hyperparameters set at $M = 700$, $s = 10$, $|R|_{min} = 10$ and $v = 0.01$.

Validation

We proceed to examine the GBM with $M = 1\,500$ boosting iterations, interaction depth $s = 45$ and a minimum terminal node size of $|R|_{min} = 10$. This model has an accuracy of 0.488 with a 95% confidence interval between $(0.479, 0.496)$, Kappa of 0.305 and an AUC of 0.766. From the confusion matrix it can be seen that the model performs well, as it correctly classifies the majority of families (`F` and `FC`) (see Figure 5.8). 84.06% of the families (`F`) are classified as families (`F`) and 66.33% of the families with children (`FC`) are classified correctly. This can also be seen by the dark blue colours in their respective cell. The model does not seem to be able to differentiate households (`H` and `HC`) and families (`F` and `FC`), as most households (`H` and `HC`) are classified as families (`F` and `FC`). The model also performs well on predicting whether a household contains a child, as the majority of these household are classified in the family with children (`FC`) rather than families (`F`). From the ROC curve it can be seen that the model is very good in differentiating between single females (`SF`) and other groups, as well as family with children (`FC`) and families (`F`) (see Figure 5.9).

Model Properties

The GBM uses 3 854 features, which is slightly more than three times the number of features used by the AUC GLMNET model. Computing the variable importance as described

[5]The value has been rounded.
[6]The accuracy based on the test set is 0.415.

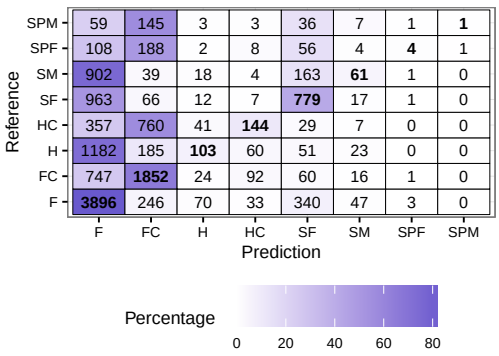


Fig. 5.8.: Confusion matrix of the GBM on the test set.

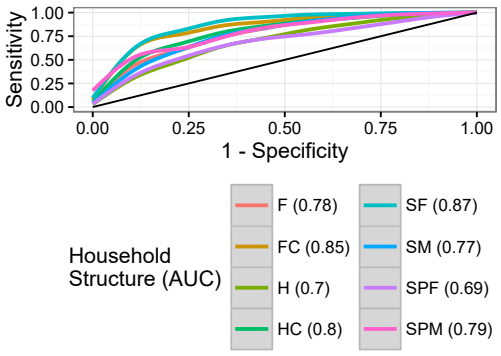
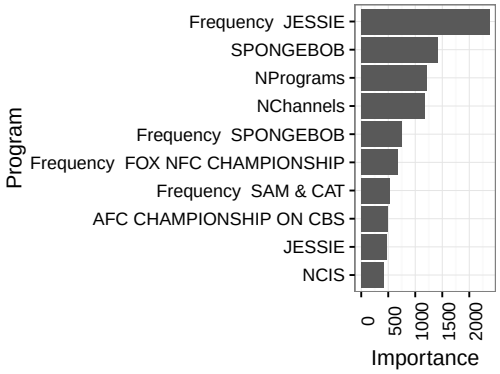
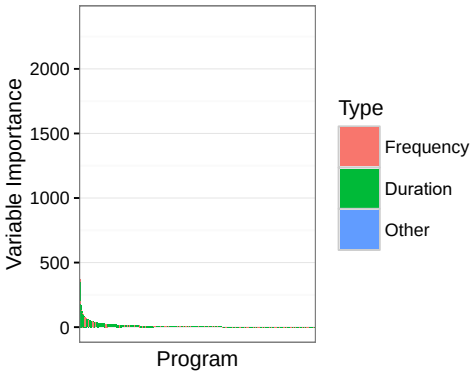


Fig. 5.9.: ROC curves of different classes using the GBM.

in Section 3.2.1 shows that the majority of the important features are program duration features (see Figure 5.10b). Furthermore, both the `NChannels` and `NPrograms` are selected, but since there are only two of them, these features are hardly visible in the graph. We also note that 5 out of the 10 most important programs are children programs (see Figure 5.10a).



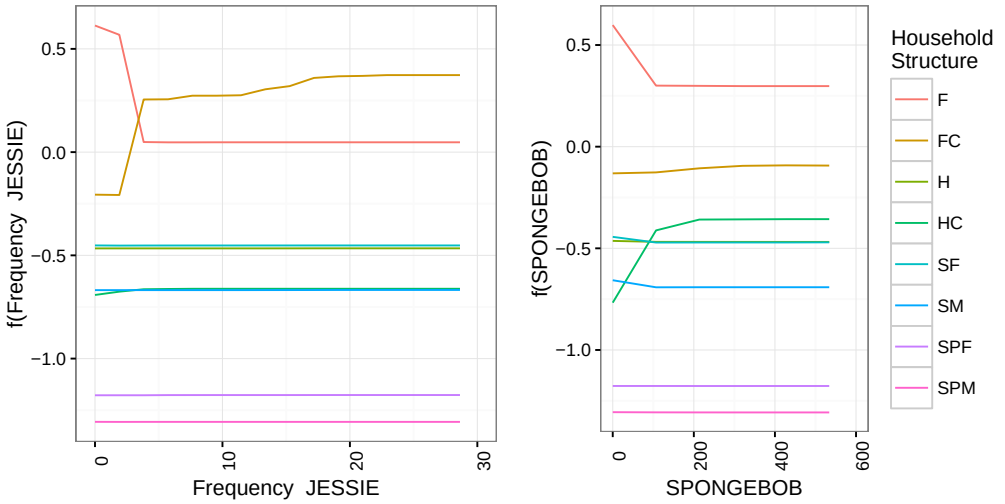
(a) The top 10 important features.



(b) The variable importance distribution with all features.

Fig. 5.10.: Variable importance of the GBM with $M = 1500$ boosting iterations, interaction depth of $s = 45$ and a minimum terminal node size of $|R|_{min} = 10$.

In order to observe the marginal effects of an important feature on the target variable, we construct partial dependency plots as discussed in Section 3.2.1. The most important feature is `Frequency JESSIE`. The majority of the household structures are families (`F`), which is also reflected in the partial dependency plot, as the line is positioned higher than other lines (see Figure 5.11a). We observe that if the frequency of watching Jessie increases to more than 4, the household would be more likely to be considered a family with children (`FC`), rather than just a family (`F`) (see Figure 5.11a). For households (`H`) we do not observe any changes in their marginal effects regarding `Frequency JESSIE`. The feature `Frequency JESSIE` has also a proportional relationship with the probability of a household being classified as household with children (`HC`). This relationship is much smaller than the relationship in families with children (`FC`). For single adults with or without children (`SF`, `SM`, `SPF`, `SPM`), an increase in `Frequency JESSIE` has almost no visible effect.



(a) Partial dependency plot of the feature: `Frequency JESSIE`.

(b) Partial dependency plot of the feature: `SPONGEBOB`.

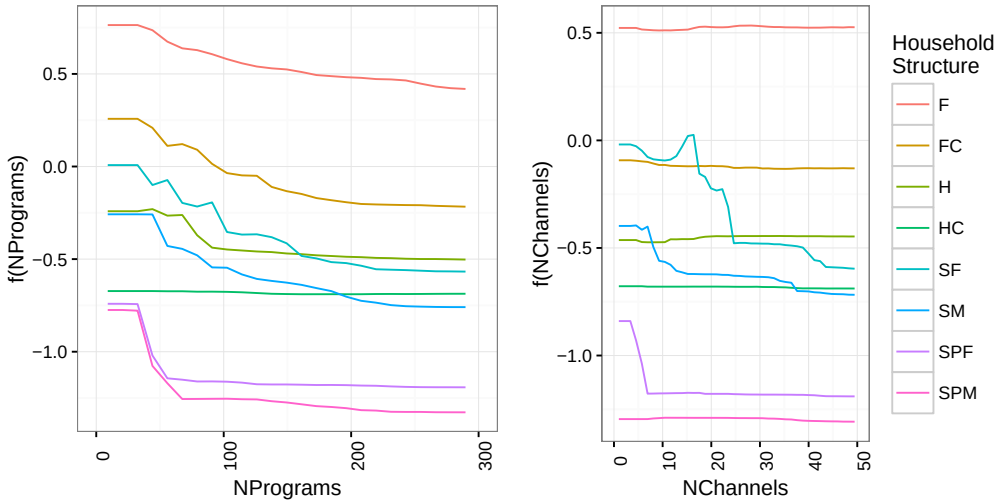
Fig. 5.11.: Partial dependency plots of children programs.

The second most important feature is `SPONGEBOB`. This feature is more interesting, as the marginal effects seem to be larger (see Figure 5.11b). We observe that if a household watches `SPONGEBOB` more than 100 minutes, the probability of it being a family (`F`) decreases, while the probability of it being a family with children (`FC`) slowly increases. This effect seems to be weaker compared to the effect on families (`F` and `FC`) regarding the feature `Frequency JESSIE`. There, we observe that the probability of the household being a family (`F`) decreases, while the probability that it is a family with children (`FC`) increases. Here, we only observe that the probability of it being a family (`F`) decreases. For households (`H` and `HC`), we observe that the probability that a household is classified as a household with children (`HC`) increases, as the duration of watching `SPONGEBOB` increases. We also observe that an increase in the duration of watching `SPONGEBOB` decreases the probability that a household is classified as a single adult household (`SF` and `SM`). For single parents (`SPF` and `SPM`) no marginal effect is visible.

The marginal effect of `NPrograms` seems to be decreasing for all classes except households with children (`HC`) (see Figure 5.12a). If the number of programs (`NPrograms`) exceeds 38, the probability that the household is a single parent (`SF` and `SM`) decreases. This effect can also be seen in family (`F` and `FC`) household structures.

The `NChannels` feature primarily affects the probability of single adults (`SF` and `SM`) (see Figure 5.12b). If `NChannels` exceeds 40, a household will be less likely to be classified as a single adult (`SF` and `SM`). This effect is much stronger for single females (`SF`) than single males (`SM`). The marginal effect for single female parents (`SPF`) also seems rather strong. If a household watches more than 4 different channels, the probability that the household is classified as a single female parent (`SPF`) is small.

The next feature on the list is `Frequency FOX NFC CHAMPIONSHIP`. This is the first feature that contains data from a program aimed at adults. As expected, we observe that an increase in `Frequency FOX NFC CHAMPIONSHIPS` results in a higher probability that the



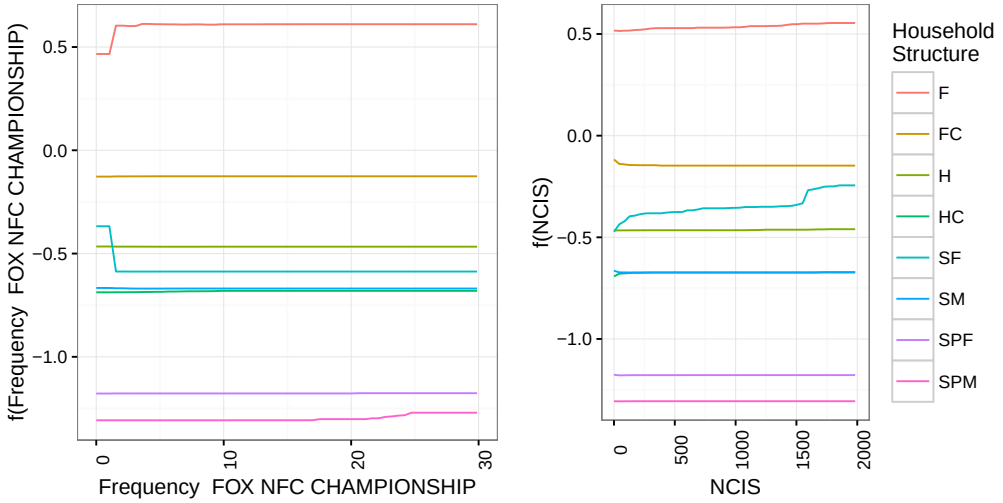
(a) Partial dependency plot of the feature: `NPrograms`.

(b) Partial dependency plot of the feature: `NChannels`.

Fig. 5.12.: Partial dependency plots of the number of different programs (`NPrograms`) and channels watched (`NChannels`).

household is classified as a single male (`SM`). The same effect can also be seen for single male parents (`SPM`). We also observe that if `Frequency FOX NFC CHAMPIONSHIPS` exceeds 1, the probability that the household is classified as a single female (`SF`) decreases. While at the same time, the probability that the household is classified as a family (`F`) increases.

For the `NCIS` duration feature, we observe that if a household watches `NCIS` longer, it is more likely classified as a single female (`SF`). A gradual increase can also be seen in the probability of the household being classified as a family (`F`). We also observe that if a household watches `NCIS` for an extended period of time, it is also less likely to be classified as a family with children (`FC`) or single male (`SM`).



(a) Partial dependency plot of the feature: `Frequency FOX NFC CHAMPIONSHIP`.

(b) Partial dependency plot of the feature: `NCIS`.

Fig. 5.13.: Partial dependency plots of programs aimed at adults.

The remaining partial dependency plots are given in the Appendix D.1.2. An increase in the feature `Frequency SPONGEBOB` leads to a decrease in the probability that a household being classified as family (`F`), while it increases the probability that the household is classified as family (`FC`) or household with children (`HC`) (see Appendix D.1.2, Figure D.5). An increase in `Frequency SPONGEBOB` does not lead to visible changes in the probability of other household structures. An increase in `Frequency SAM & CAT` leads only to an increase in the probability of a household being classified as a family with children (`FC`) (see Appendix D.1.2, Figure D.6) For other household structures, this has almost no effect, except for single females (`SF`). The probability of the household being a single female (`SF`) decreases only slightly due to watching `Frequency SAM & CAT`. The partial dependency plot of `AFC CHAMPIONSHIP ON CBS` shows that when the duration exceeds about 500 minutes, the probability of classifying the household as a single man (`SM`) increases, while other classes only have a small marginal effect (see Appendix D.1.2, Figure D.7). The partial dependency plot for `JESSIE` shows a positive effect for only households with children `HC`, although families with children (`FC`) have also this program in their top 10 longest duration programs (see Figure 4.10a and Appendix D.1.2, Figure D.8).

Incorrectly Classified Households

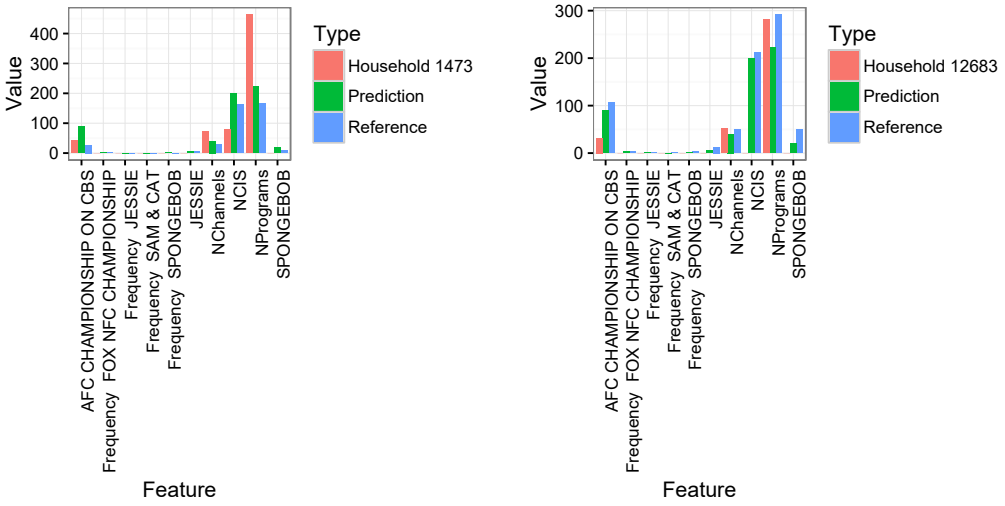
The average durations and frequencies of the test set predictions using the GBM are given in Table 5.3. We observe that families (`F`), single females (`SF`) and single male parents (`SPM`) have a lower average duration than their incorrectly classified counterparts. The difference between the average duration of correctly and incorrectly households is the smallest for families (`F`) and the largest for single male parents (`SPM`). The latter can be attributed to the fact that only one single male parent (`SPM`) is correctly classified (see Figure 5.8). In the column of the average frequencies, families (`F`) that are correctly classified have a higher average frequency than incorrectly classified families (`F`). This is also true for every household structure with the exception of single females (`SF`) and single male parents (`SPM`).

Tab. 5.3.: Average durations and frequencies of correctly and incorrectly classified households by the GBM.

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	14 624.1	14 854.3	893.9	837.65
FC	20 433.5	14 066.5	1 246.3	811.56
H	33 328.2	19 423.2	2 081.2	1 111.84
HC	47 055.5	23 398.2	2 903.7	1 360.86
SF	9 068	11 280.9	404.3	638.15
SM	11 801.1	9 152.3	682.3	660.32
SPF	13 885.8	12 941	788.8	601.36
SPM	661	7 736.9	49	421.11

Using the important variables from the GBM (see Figure 5.10a), we are able to plot the feature values of households to better determine why these households are incorrectly classified. These feature values are then compared to average values of the reference and prediction class. Household 1 473 is a single female (`SF`), but is classified as a family (`F`). Household 1 473 barely watches `SPONGEBOB`, making it more similar to the single female (`SF`) than a fam-

ily (F) (see Figure 5.14a). The number of programs (NPrograms) and channels (NChannels) suggest that it is more similar to a family (F). Household 12 683 is a household (H), but is classified as a family (F). The lack of viewing SPONGEBOB, AFC CHAMPIONSHIP ON CBS and NCIS suggest that it should be classified as a family (F) (see Figure 5.14b).



(a) Household 1473 **(b)** Household 12683

Fig. 5.14.: Individual household viewing behaviours compared to the duration averages of the reference and prediction class.

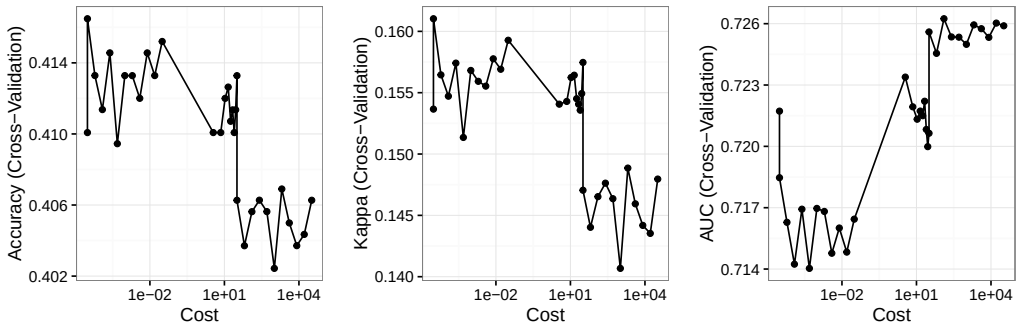
5.2.3 Support Vector Machines

The last machine learning algorithm will be discussed is the support vector machine (SVM). We try three different kernels, namely the linear ($SVML$), polynomial ($SVMP$) and the radial basis kernel ($SVMR$). Again, the same 5-fold cross-validation is used in order to determine the hyperparameters.

Linear Kernel

Hyperparameter Search Space We start by training a support vector machine with a linear kernel. This is the fastest algorithm to train compared to GLMNET and stochastic gradient boosting. This can be attributed due to the optimisations possible and the one-dimension search space, which involves only the cost hyperparameter C. The maximum cross-validation accuracy that is achieved is 0.417, which is achieved by setting $C = 3.052 \cdot 10^{-5}$ (see Figure 5.15a). At this point the cross-validation Kappa is also at its maximum of 0.161 (see Figure 5.15b). The cross-validation AUC at this point is 0.719. It can be seen in the three graphs that the accuracy measure and the Kappa statistic favour a low cost parameter, while AUC favours higher ones (see Figure 5.15). The turning point seems to be around $C = 32$. The maximum cross-validation AUC of 0.726 is achieved at $C = 128$, with a cross-validation accuracy of 0.406 and Kappa of 0.147 (see Figure 5.15c).

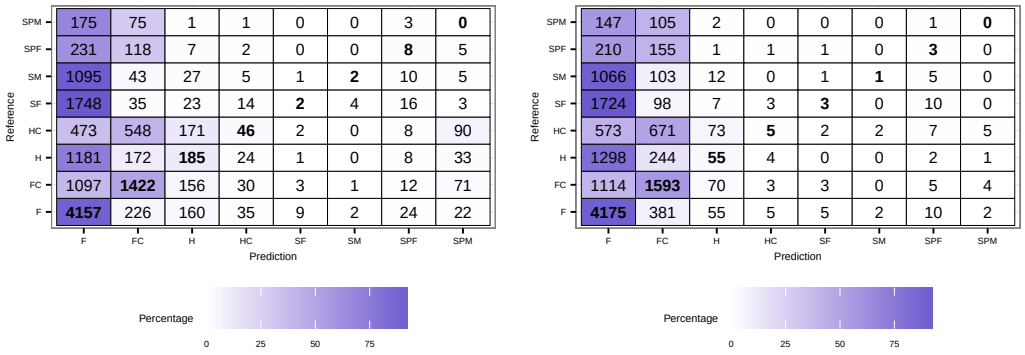
Validation The accuracy model with $C = 3.052 \cdot 10^{-5}$ achieves an accuracy of 0.415 on the test set with a 95% confidence interval between $(0.407, 0.423)$ (see Figure 5.16a). The associated Kappa and AUC values are 0.183 and 0.689. A notable observation is that



(a) Accuracy **(b)** Kappa **(c)** AUC

Fig. 5.15.: Level plot for different cross-validation performance measures over different cost (C) values.

the model classifies single female parents (`SPF`) surprisingly well compared to the previous confusion matrices. The second observation is that the model does not seem to be able to classify single male parents (`SPM`), as many of the single male parents (`SPM`) are actually families (`F`) and `FC`) or households (`H` and `HC`). The model selected using AUC gives a comparable model, 11 665 of the 14 028 predictions are the same. The accuracy of the AUC model is 0.416 with a 95% confidence interval between $(0.408, 0.424)$ (see Figure 5.16b). An AUC of 0.704 also suggests that the AUC model is preferred over the accuracy model. The Kappa statistic of 0.174 is the only performance measure that suggests otherwise. Although both models have more than $\frac{2}{3}$ predictions in common, the Cramer's V for both is only 0.364. The main difference between both models lies in the prediction of households (`H` and `HC`). The accuracy model classifies a more diverse number of classes, while the AUC model focusses more on the classification of families (`F`) and families with children (`FC`).

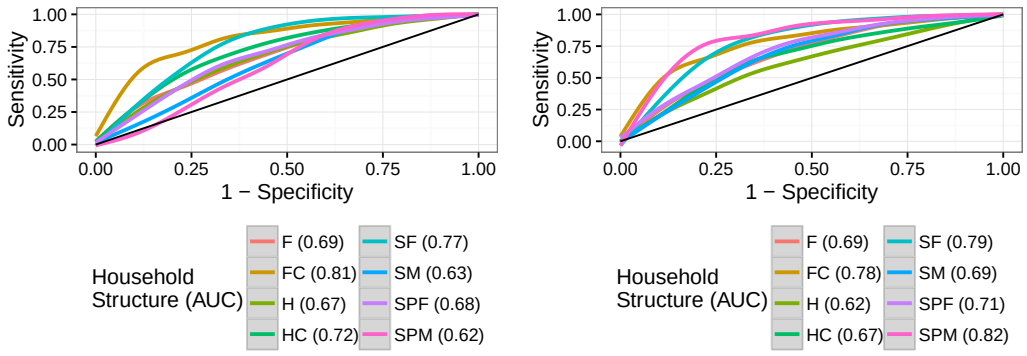


(a) Accuracy **(b)** AUC

Fig. 5.16.: Confusion matrices of different SVML models on the test set.

The ROC curves provide a better understanding of behaviour of the models. The accuracy model is superior in distinguishing families with children (`FC`) and households (`H` and `HC`) (see Figure 5.17a), while the AUC model sacrifices the ability to distinguish those classes in favour of better distinguishing all the other classes (see Figure 5.17b).

Incorrectly Classified Households The AUC model will be used for further analysis as it classifies a more diverse number classes. The summary statistics of correctly and incorrectly



(a) Accuracy **(b)** AUC

Fig. 5.17.: ROC curves of different classes using different models on the test set.

classified households using the AUC model are presented in Table 5.4. It can be clearly observed that most of the average durations of the correctly classified households are much higher compared to the incorrectly classified households. Again, with the exception of family (`F`). The frequencies show similar behaviour, with an addition that single female parents (`SPF`) also have a lower average frequency than their incorrectly classified counterparts. Individually households cannot be visualised, as no clear definition exist for important features in a SVM model.

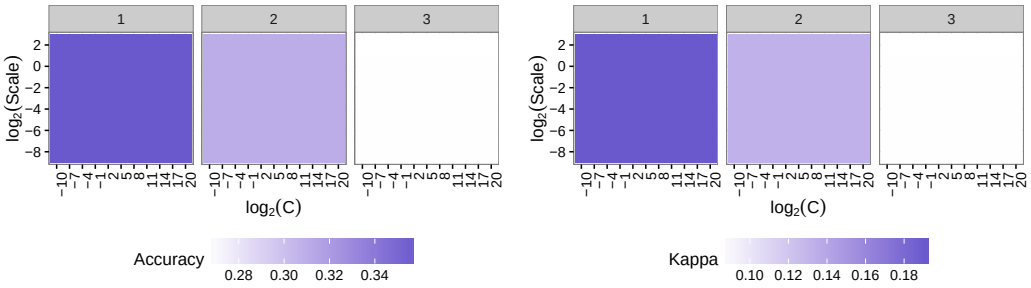
Tab. 5.4.: Average durations and frequencies of correctly and incorrectly classified households by the AUC SVML model.

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	13 598.4	23 900.3	818.2	1 465.7
FC	21 877.4	14 566.2	1 327.4	863.8
H	32 506.6	18 726.7	2 136.8	1 048.6
HC	52 450.4	25 000.5	3 372.9	1 461.2
SF	35 963	10 318.8	2 798	537
SM	33 653.5	9 247.2	919	661
SPF	15 627.1	12 892.3	547.9	604.6
SPM		7 709.2		419.7

Polynomial Kernel

Hyperparameter Search Space The next kernel we use is a generalisation of the linear kernel, which is the polynomial kernel. Although the polynomial kernel adds many additional degrees of freedom, it seems that the best model based on cross-validation accuracy is a simple linear model with a degree $d = 1$, scale of $a = 2^{-8}$ and cost $C = 2^{-10}$ (see Figure 5.18a). With those hyperparameters a maximum cross-validation accuracy of 0.359 is achieved and a cross-validation Kappa of 0.191. It is not possible to compute the AUC for a large portion of the hyperparameter combinations, as the line search algorithm fails to find a good fit for the logistic regression model for calculating SVM class probabilities. Thus, it has been opted to exclude using the AUC performance measure for the SVM with a polynomial kernel. The hyperparameter search space using accuracy shows a flat surface,

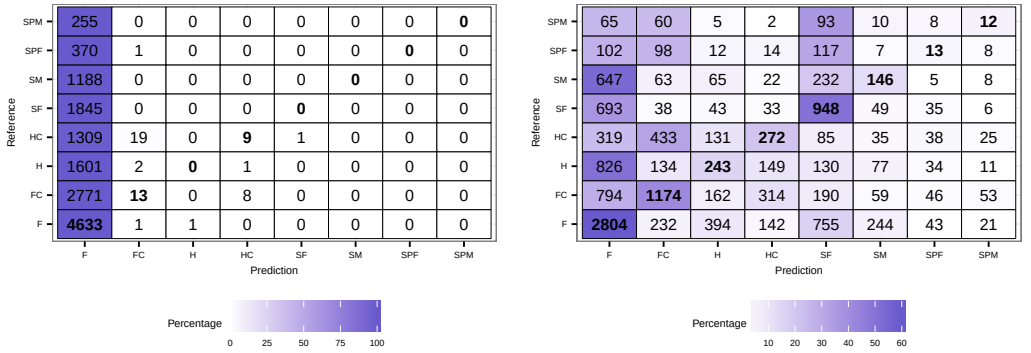
which indicates that the degree is the most important tuning parameter. If $d = 2$, the cross-validation accuracy becomes 0.310, while at $d = 3$ the cross-validation accuracy is only 0.267. The parameter search space for the Kappa performance measure is very similar, as it is also flat (see Figure 5.18b). Selecting the model based on the optimal cross-validation Kappa value gives the same model as selecting with cross-validation accuracy.



(a) Accuracy **(b)** Kappa

Fig. 5.18.: Level plot for cross-validation accuracy values over different combinations of the degree (d), scale (a) and cost (C) hyperparameters.

Validation We now proceed to validate the SVMP model using the test set. The confusion matrix is shown in Figure 5.19a. We observe that almost every household is predicted as a family (`F`). In addition, only five classes are actually predicted. This gives an accuracy of 0.332 with a 95% confidence interval between $(0.324, 0.340)$. The Kappa value of this model is 0.003, which suggests the model performs extremely poor. These results leads us to re-evaluate the hyperparameter search space.



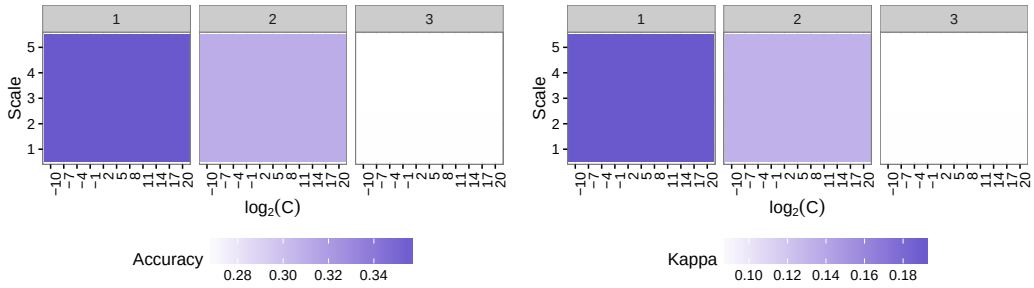
(a) General model **(b)** Restricted model

Fig. 5.19.: Confusion matrices of SVMP models selected by a general and restricted hyperparameter search space.

The degree hyperparameter d and cost hyperparameter C do not seem to be the problem, as with a degree $d = 1$ is a linear kernel and the linear kernel did perform much better. The cost hyperparameter C also does not seem to be the problem, as these cost values were also present in the hyperparameter search space of the support vector machine using a linear kernel. This leaves us with the scale hyperparameter a, which is set to a very small value. If the scale hyperparameter a is set to a very small value, all points irrespectively of their value will get the same value out of the kernel. This also explains why the kernel is classifying households as families (`F`). The libSVM package suggests the scale hyperparameter a should

have a value of $\frac{1}{p}$, where p is the number of features [8]. Unfortunately, with these settings the problem still persists and therefore, a scale of at least one is chosen ($a \geq 1$). We refer to this model as the *restricted model*, while the SVMP model introduced in the previous paragraph is referred to as the *general model*.

The restricted hyperparameter search space seems to be similar as the hyperparameter search space with very small scales (see Figure 5.20a). The highest cross-validation accuracy is 0.359 and Kappa is 0.191. These are the same optimal cross-validation values as in the general hyperparameter search space. The optimal hyperparameter values are again similar, with $d = 1$ and $C = 2^{-10}$. The only difference is of course the scale hyperparameter a, as that dimension of the search space is restricted to $a \geq 1$. We find that the optimal scale parameter is $a = 1$. The Kappa hyperparameter search space in Figure 5.20b shows exactly the same hyperparameter search space as Figure 5.18b and also agrees with the selected hyperparameter values using accuracy.



(a) Accuracy **(b)** Kappa

Fig. 5.20.: Level plot for cross-validation accuracy and Kappa values over different combinations of the degree (d), scale (a) and cost (C) hyperparameters. The hyperparameter search space is restricted to $a \geq 1$.

Validating our restricted model with the test set shows us a major improvement in the accuracy. The accuracy of the restricted model is 0.400, with a 95% confidence interval of $(0.392, 0.408)$ (see Figure 5.19b). The Kappa is 0.229. All performance measures seem to favour this model over the one with a smaller scale. In addition, we also observe that the restricted model classifies more classes accurately. Comparing both prediction results also reveals that only 44% of the predictions are the same. The Cramers V between both models is 0.068.

Incorrectly Classified Households Since the restricted SVMP model performs much better in the validation set, this model is used to examine the average duration and frequencies of the correctly and incorrectly classified households.

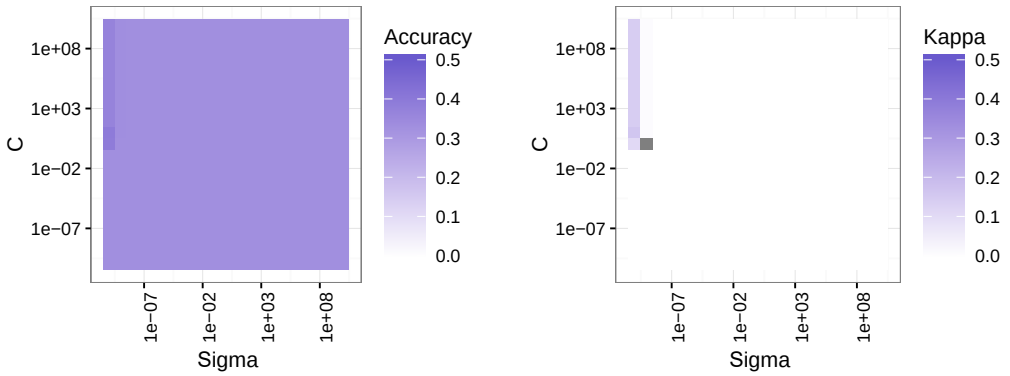
The average duration and frequency of correctly and incorrectly classified households show similar values as with the SVML model (see Table 5.5). The average durations of correctly classified households in general have a higher average durations than incorrectly classified households, with the exception of families (`F`) and single females (`SF`). The average frequencies shows a similar effect.

Tab. 5.5.: Average durations and frequencies of correctly and incorrectly classified households by the restricted SVMP model.

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	13 316.5	16 719.4	819.1	985.8
FC	19 456.1	17 443.7	1 193.3	1 032.2
H	25 322.1	19 422.2	1 553.0	1 106.4
HC	40 806.3	22 152.0	2 497.0	1 279.4
SF	8 815.4	11 964.9	432.8	652.1
SM	12 754.9	8 802.6	901.1	627.9
SPF	18 225.2	12 759.7	835.7	594.9
SPM	12 744.2	7 460.6	706.3	405.50

Radial Basis Kernel

Hyperparameter Search Space The radial basis kernel results in a relatively flat accuracy hyperparameter search space (see Figure 5.21a). The cross-validation accuracy for most of the time is equal to the naive classifier. At $\sigma = 10^{-10}$ and $C = 0.1$ both the maximum cross-validation accuracy and Kappa of 0.382 and 0.121 are achieved (see Figure 5.21a and 5.21b). One notable observation for the Kappa search space is the negative cross-validation Kappa value of -0.012 at $\sigma = 10^{-9}$ and $C = 0$. This value has been depicted grey.



(a) Accuracy **(b)** Kappa

Fig. 5.21.: Level plot for cross-validation accuracy and Kappa values over different combinations of the cost (C) and sigma ($Sigma/\sigma$)[7] hyperparameters.

Validation Validating the support vector machine with radial basis kernel against the test set results in the naive classifier. Since the model predicts all households as families (`F`), we only achieve an accuracy of 0.330 with a 95% confidence interval of $(0.323, 0.338)$. The corresponding Kappa is also 0. This suggests that there are some problems with the hyperparameter search space. The cost C hyperparameter does not seem to be the problem, as this was also not an issue using our previous kernels. This implies that the problem is with the σ hyperparameter. The σ can be interpreted as how far the influence of a single training example reaches [62] (see Equation 3.56). Low values correspond to a far influence, while

[7]Both $Sigma$ and σ are used interchangeably.

high values correspond to only a close influence. However, we already consider σ values of $1 \cdot 10^{-10}$ to $1 \cdot 10^{10}$. We expand the hyperparameter search space to $C \in \{2^{-5}, 2^{-4}, \ldots, 2^{15}\}$ and $\sigma \in \{2^{-15}, 2^{-14}, \ldots, 2^{3}\}$ in the hope it improves the classification results. This results in the optimal hyperparameter of $\sigma = 2^{15}$ and $C = 2^{-15}$. Unfortunately, this does not solve the problem and the model still predicts only families (`F`). Changing the sigma to $\sigma = 1$ also does not prove to be beneficial, as we still get a naive classifier on the validation data set. The accompanying optimal cost hyperparameter C is in all cases $C = 2^{-10}$.

Comparing Kernels

We now proceed to compare SVMs with different kernels. We exclude the radial basis kernel, as this one only gives results similar to the naive classifier. Additionally, some caution has to be taken to interpret these results, as we restrict the scale hyperparameter a of the polynomial kernel to be larger than one. This deviates from the usual cross-validation procedure, as the optimal solution is not achieved by only cross-validation, but also due to restricting the hyperparameter search space. As discussed in Chapter 3.4.2, we compare the performance of different models using their resampling distributions. This results in Figure 5.22.

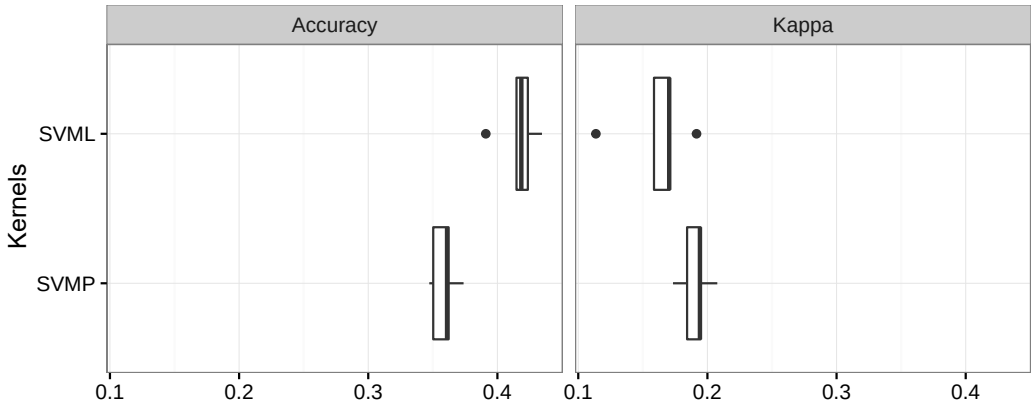


Fig. 5.22.: The resampling distributions with different performance measures and kernels.

We observe that SVML performs better in terms of accuracy than the restricted SVMP. The cross-validation accuracy values of the SVML also seems to be higher than the SVMP in all cases. Using Kappa as the performance measure, we observe that the restricted SVMP only performs better in some folds. Using a paired t-test we conclude that both kernels are not significantly different from one another, as the p-value is 0.226. However, this is not in agreement with the Wilcoxon signed rank test, which shows that we can reject the null hypothesis that both medians between pairs of observations are equal to zero using a significance level of 10%. The p-value of the Wilcoxon signed rank test is 0.063.

5.2.4 Algorithm Comparison

We now proceed to compare the different algorithms using their resampling distributions as presented in Section 3.4.2. The algorithms that are in our comparison are: GLMNET, stochastic gradient boosting (*SGB*) and SVML. We do not include SVMs with other kernels,

because these algorithms tend to predict primarily families (F). The only algorithm that improved by restricting the hyperparameter search space is the SVMP. However, based on the optimal hyperparameter values that were selected, the SVMP still seems to favour a linear kernel. Therefore, SVMs with other kernels than linear are excluded from this comparison.

From the resampling distributions we observe that SGB is superior with respect to the cross-validation accuracy and Kappa (see Figure 5.23). The minimum cross-validation accuracy and Kappa from SGB exceeds the maximum cross-validation accuracy and Kappa values from GLMNET and SVML. This visually suggests that SGB is better than the rest.

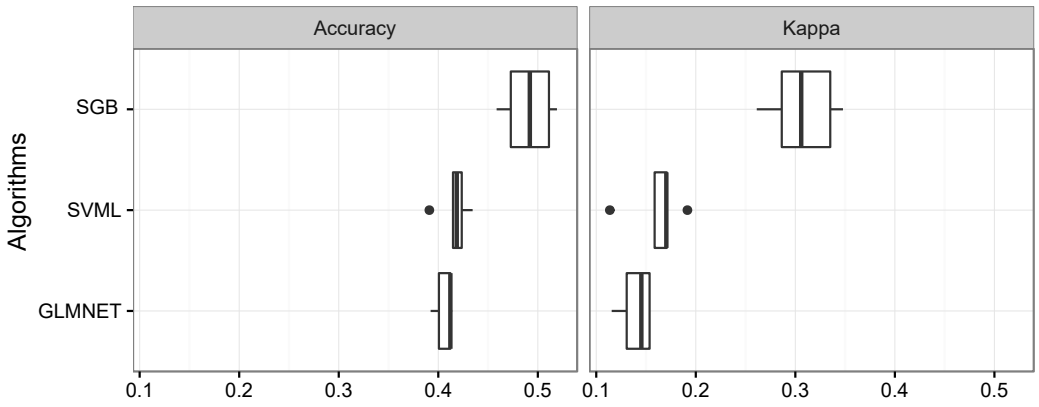


Fig. 5.23.: The resampling distributions with different performance measures and algorithms.

In order to show that SGB is statistically significant different from the other algorithms, a parametric and non-parametric test are done on the means/medians between all pairs of models[8]. The parametric test used is simply the t-test, while the non-parametric test that is used is the Wilcoxon signed rank test. A non-parametric test is included as it is not evident that cross-validation accuracy and Kappa follow a normal distribution and the number of resamples are low in our case. The null hypothesis of the t-test states that the population means from both groups are equal, while the null hypothesis of the Wilcoxon signed rank test states that the population median from both groups are equal. The p-values from both tests for all pairs are presented in Table 5.6. This shows that between SGB and GLMNET the mean and the median cross-validation accuracies are statistically significant different under a significance level of 10%. This is also the case when using the Kappa statistic. The difference is also significant between SGB and SVML, while GLMNET and SVML are not significantly different. It can also be observed that the Wilcoxon signed rank test produces higher p-values than the t-test.

5.2.5 Overview

In section we present a short overview of the base models and compare them with each other and with the results from Klein[47]. We use a mapped accuracy approach, which can be

[8]The non-parametric test tests the difference between the medians between pairs of observations

Tab. 5.6.: P-values for testing the differences between the means and medians of the cross-validation performance measures.

Pair		Accuracy		Kappa	
		T-test	Wilcoxon	T-test	Wilcoxon
SGB	GLMNET	0.004	0.063	0.002	0.063
SGB	SVML	0.010	0.063	0.004	0.063
GLMNET	SVML	0.332	0.313	0.243	0.188

constructed using the following mapping. Families (`F`) and households (`H`) are combined, as Klein only defines a family category. Next, we also combine families with children (`FC`) and households with children (`HC`) into families with children category. Klein does not define a single parents category and therefore, single parents (`SPF` and `SPM`) are combined with their single adult (`SF` and `SM`) counterparts. This gives us a total of four classes and allows us to compare our research with that of Klein.

The multinomial logistic model seems to be relatively fast to train, though exact timings are not presented in this thesis. The accuracy of the GLMNET models vary from 0.382 to 0.412, which are better than the naive classifier (see Table 5.7). The accuracy and AUC performance measures suggest that the AUC model is the best, but the Kappa statistic prefers the Kappa model. However, it is not sure whether this difference is significant and more research needs to be conducted in order to determine which performance measure is the best. The GLMNET models in general suggest that duration and frequency features are equally favourable, as both types of features are selected in roughly equal amounts[9]. The classification errors of the AUC GLMNET model also seem to indicate that correctly classified households in general have a higher average duration and frequency than their incorrectly classified counterparts, with the exception of families (`F`) (see 5.2). We argue that this is due to the fact that if the algorithm cannot decide with enough certainty that a household has a particular structure, that it will be assigned to the family (`F`) class. This would lead to an increase in the average duration of incorrect classified families (`F`). The mapped accuracy of the accuracy GLMNET model seems to be the highest, with 0.551 (see Table 5.7). This is however slightly worse than the logit model from Klein [47], which has an accuracy of 0.596. In addition, we also observe that all GLMNET models improve due to the mapping, which leads us to believe that our models perform worse due to training on eight different household structures. In addition, the accuracy of Klein is the result of a 10-fold cross-validation process, therefore, it is possible that it provides an optimistic accuracy value.

The GBM is one of the slowest to train, but it performs the best, with an accuracy of 0.488 (see Table 5.7). The other performance measures also agree on this. From the confusion matrix, we observe large similarities with the AUC GLMNET model (see Figure 5.3 and 5.8). The GBM performs worse in classifying families (`F`), but this is largely compensated by its better performance in all other household structures. In fact, the GLMNET algorithm only seems to be better in classifying families (`F`) irrespective of the performance measure used in selecting the model. The average duration and frequencies also show a similar behaviour as with the AUC GLMNET model, except that the average duration of correctly classified

[9] Only one of the distributions is presented in this thesis.

single females (SF) is also lower than the duration of its incorrectly classified counterpart (see Table 5.2 and 5.3). This suggests that single females (SF) are classified due to the lack of certain viewing behaviours. Single females (SF) that have a high average duration are then probably classified as families (F), as 340 of the 1 514 single females (SF) are classified as such. The mapped accuracy of the GBM of 0.642 is better than the random forest accuracy of Klein, which is 0.629.

The SVMs with different kernels are not very successful, with the exception of the linear kernel. The linear kernel is the fastest to train, due to its many implemented optimisations and a simple search space with only one hyperparameter. The accuracies of the accuracy SVML and AUC SVML are very similar, ranging from 0.415 to 0.416 (see Table 5.7). The accuracy and Kappa seem to favour the accuracy SVML model, while the AUC favours the AUC SVML model. This is expected, since the cross-validation accuracy and Kappa agree on the selected hyperparameter value. The confusion matrices of both SVML models are very similar, with the only difference being their performance on households (H and HC) (see Figure 5.16). That is also where the accuracy SVML model seems to excel, whereas the AUC SVML excels at classifying families (F and FC). The average durations and frequencies for the AUC SVML model shows a very high average duration for correctly classified households with children (HC), which is not seen before (see Table 5.4). In addition, it also shares the same low average durations for families (F) and single females (SF) similar to the GBM. The mapped accuracies also show that the SVML models have a slightly lower accuracy than the logit model from Klein [47]. The unrestricted SVMP model seems to have a flat hyperparameter search space, which results in a naive classifier (see Figure 5.18). From the confusion matrix, we observe that it does not even perform very well in the other classes, as it confuses families with children (FC) with households with children (HC) (see Table 5.19). The restricted SVMP model fares much better, but from the confusion matrix, we observe that it fails to classify families (F). Only 2 804 of the 4 635 families (F) are correctly classified. It fails to distinguish between families (F) and all other classes. This is very similar to the Kappa GLMNET model, but this model seems to better based on accuracy and Kappa performance measure (see Figure 5.3 and 5.19). The SVMP models also performs worse than the logit model from Klein [47]. The SVMR model in our case is unfortunately exactly the same as the naive classifier. Therefore, it seems unnecessary to discuss it further.

Tab. 5.7.: Performance measures for the base models.

Algorithm	Performance Measure	Accuracy	Mapped Accuracy[10]	Confidence Interval		Kappa	AUC
				Min	Max		
Naive	-	0.330	0.445	0.323	0.338	0.000	0.500
GLMNET	accuracy	0.406	0.551	0.398	0.414	0.137	0.646
GLMNET	Kappa	0.382	0.527	0.374	0.390	0.203	0.659
GLMNET	AUC	0.412	0.529	0.403	0.420	0.163	0.688
SGB	-	0.488	0.642	0.479	0.496	0.330	0.766
SVML	accuracy	0.415	0.553	0.407	0.423	0.183	0.689
SVML	AUC	0.416	0.561	0.408	0.424	0.174	0.704
SVMP	-	0.332	0.448	0.324	0.340	0.003	N.A.
SVMP[11]	-	0.400	0.552	0.392	0.408	0.229	N.A.
SVMR	-	0.330	0.445	0.323	0.338	0.000	N.A.

[10]Mapped to classes as defined by Klein [47].
[11]Restricted SVMP

From the algorithm comparison, we observed that the SGB algorithm is the best. This is also visible in the performance measure on the validation set. Thus, the GBM can be considered the best among the base models.

5.3 Support Vector Machines With Dimension Reduction

Support vector machines in itself do not provide feature selection. Thus, they will use the whole feature set to train the model. In some cases, this can lead to a bad performance if many irrelevant features are included [84]. This can be remedied by using feature selection. In this section, we introduce support vector machines with a dimension reduction filter based on the variable importance of the previous models. The procedure is as follows. First, the important variables are extracted and ranked from the regularized multinomial logistic regression (*GLMNET*) model or stochastic gradient boosting model (*GBM*). Only these features will be used in the training and testing process. Next, a filter approach is included in the 5-fold cross-validation, resulting in an extra hyperparameter T, namely the top most important variables that are selected. In total, two dimension reductions steps are performed, where the first step reduces the number of features to only the features used in the feature selection model and the second step determines which subset of these features should be used. The advantage of incorporating the second step is to eliminate less important features. As with the base models, the same 5-fold cross-validation is used.

5.3.1 Support Vector Machines With Dimension Reduction Using a Regularized Multinomial Logistic Model

We first start with the models that we have created in the previous section using GLMNET (see Section 5.2). Since we have examined the AUC GLMNET model in-depth and it has good properties, we will extract important features from this model. A total of 1 259 features are used in this model (see Section 5.2.1). These features are then ranked according to the absolute value of their corresponding coefficients. If a feature is used in multiple equations, the sum of the absolute values is taken. Next, we introduce a new hyperparameter, which selects the top T features from this ranking and drops the other features prior to training the support vector machine models.

Selected Features

Before presenting the models, we first examine the features that are selected by our dimension reduction approach. We start examining the distribution of the feature types for each subset of features from the AUC GLMNET model. We do this by computing the average and median of the sum for each feature type (i.e. duration and frequency) and subset of features.

The top 100 most important features consist of only frequency features, with an average summed frequency of 1.74 over the entire data set (see Table 5.8). The median summed

frequency of 0 suggests that the majority of the reduced feature set consists of programs that are not watched by households. The top 250 most important features still consist of only frequency features, but the median summed frequency changes to 6 and the average summed frequency increases to 10.16. Doubling the number of the most important features results in 494 frequency features, 5 duration features and one other feature (in this case the feature is `NChannels`). The average summed duration also increases from 0 to 7.17 minutes. Using the top 1 000 features leads to a substantial increase in the number of duration features. It also adds `NPrograms` to the feature set. The average summed frequency becomes 1 160.99 and the average summed duration becomes 118.97 minutes. If all features are used from the AUC GLMNET model, it results in a distribution of feature types as presented in Section 5.2.1.

Tab. 5.8.: Summary statistics of different subsets of selected features.

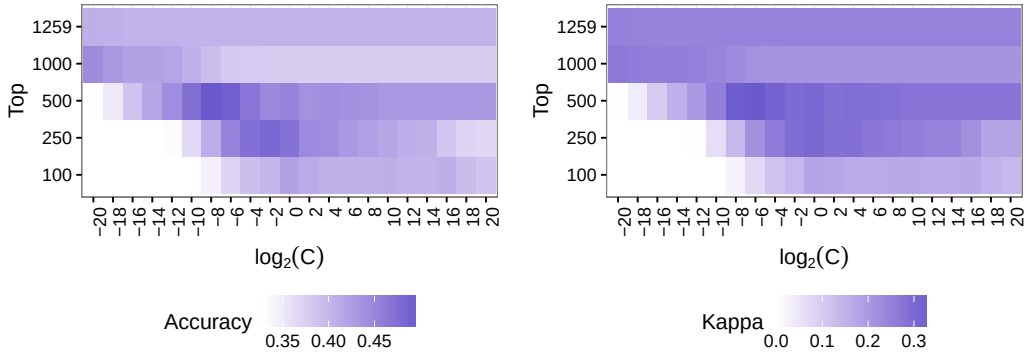
Top	Number of features			Average[12]		Median[13]	
	Duration	Frequency	Other	Duration	Frequency	Duration	Frequency
100	0	100	0	0	1.74	0	0
250	0	250	0	0	10.16	0	6
500	5	494	1	7.17	161.81	7.22	118
1 000	381	617	2	118.97	1 160.99	118.69	453
1 259	640	617	2	120.49	1 160.99	120.20	453

Linear Kernel

Hyperparameter Search Space The support vector machine with a linear kernel and dimension reduction produces the following hyperparameter search spaces (see Figure 5.24). We observe that the maximum cross-validation accuracy of 0.498 is achieved at $T = 500$ and $C = 2^{-8}$ (see Figure 5.24a). The cross-validation Kappa at this point is 0.330. From the hyperparameter search space it can be seen that the full feature set does not result in a model with the highest cross-validation accuracy. The highest cross-validation accuracy achieved with the most important 1 259 features is 0.408, while with 500 features the cross-validation accuracy is 0.498. We also observe that the variance of the cross-validation accuracies changes drastically when adding the 259 least important features. At that point, the variance changes from $4.81 \cdot 10^{-4}$ to $8.01 \cdot 10^{-7}$. The Kappa hyperparameter search space shows a similar surface, where the highest Kappa of 0.339 is reached at $T = 500$ and $C = 2^{-6}$ (see Figure 5.24b). Both performance measures seem to favour lower cost values and the top 500 most important features from the AUC GLMNET model.

Validation We now validate the models against the test set, which results in the following confusion matrices (see Figure 5.25a and 5.25b). The accuracy of the accuracy model is 0.411, with a 95% confidence interval of $(0.403, 0.419)$. This is a lot less than the cross-validation accuracy of 0.498, suggesting that the model is overfitted. The same effect can also be seen in the Kappa statistic, which is only 0.202. From the confusion matrix, we observe that the model attempts to classify all household structures, though many of its predictions seem to be wrong (see Figure 5.25a). The only class in which no household is predicted correctly is single male parent (`SPM`). The confusion matrix of the Kappa model

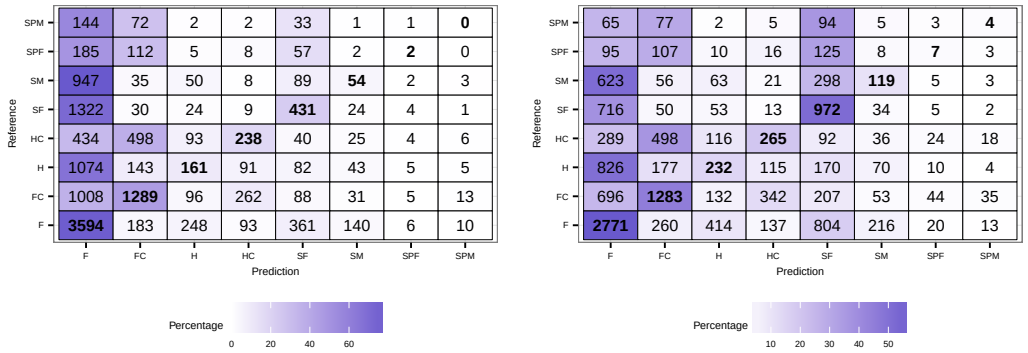
[12]The average sum of the respective feature type.
[13]The median sum of the respective feature type.



(a) Accuracy **(b)** Kappa

Fig. 5.24.: Level plot for cross-validation accuracy and Kappa values over different combinations of the cost (C) and Top (T) hyperparameters. The *Top* label indicates the number of top most important variables used in training the model.

shows a similar behaviour as the accuracy model, but it predicts more households in different classes (see Figure 5.25b). Unfortunately, the majority of these predictions are wrong, resulting in a slightly lower accuracy of 0.403 with a 95% confidence interval of $(0.395, 0.411)$. The Kappa statistic of the Kappa model is 0.232, which suggests that based on the Kappa performance measure that this model is slightly better. Another observation is that the Kappa model only correctly classifies 2 771 households as family (`F`), this is 823 less than the accuracy model. However, this shortcoming is partly overcome by the correctly classification of all non-family household structures. This can be seen by comparing the values in the diagonal of the confusion matrices. For example, the Kappa model correctly classifies 972 households as single females, while the accuracy model only correctly classifies 431. Both models also share a lot of similar predictions, as both models predict the same classes for 72% of the households and have a Cramer's V of 0.561.



(a) Accuracy **(b)** Kappa

Fig. 5.25.: Confusion matrices of different dimension reduced SVML models on the test set.

Incorrectly Classified Households In our opinion, the Kappa model seems to be the most interesting to examine, as it predicts more classes. In the accuracy model, 8 708 households are classified as families (`F`), while in the Kappa model, only 6 081 households are classified as families (`F`). This is also more similar to the actual distribution of families

(F), because the test data set contains 4 635 families (F). However, using a Chi-squared test, we find that the all distributions are significantly different from one another under a significance level of less than 1%[14].

The average duration and frequencies are tabulated in Table 5.9. We observe the same behaviour as with the previous models. All correctly classified households have an average duration higher than incorrectly households, with the exception of families (F) and single females (SF). If we compare the average durations with the ones we computed in Table 5.4, we observe that most correctly classified households have a slightly lower average duration than the AUC SVML model. For example, the average duration of households with children (HC) for the dimension reduced Kappa SVML model is 35 905.34, while for the AUC SVML model this is 52 450.4. These large differences, can be seen when comparing the average duration of households (H and HC) and single adults (SF and SM). This suggests that the dimension reduced Kappa SVML model seems to be better in coping with households with short viewing durations. The average frequency shows that all correctly classified households have a higher average frequency than incorrectly classified households, with the exception of families (F) and single females (SF). Comparing Table 5.9 with Table 5.4 shows that only 2 out of 7 classes have a higher average frequency[15]. This suggests that the dimension reduced Kappa SVML model also is better able to cope with viewing behaviours that have a low average frequency.

Tab. 5.9.: Average duration and frequencies of correctly and incorrectly classified households by the dimension reduced Kappa SVML model.

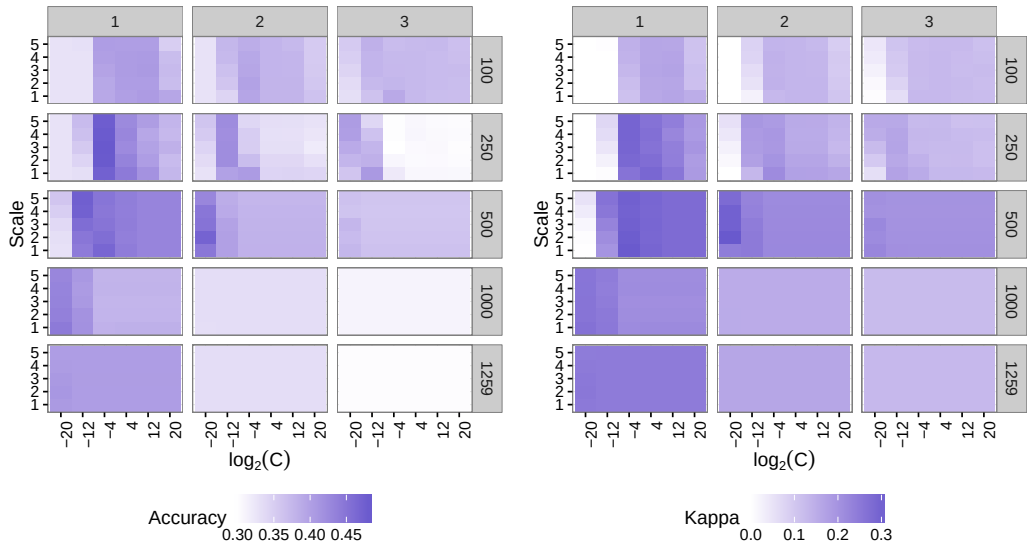
Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	14 143.48	15 429.85	848.37	939.31
FC	20 476.82	16 430.48	1 241.22	979.80
H	24 139.55	19 669.51	1 485.25	1 121.47
HC	35 905.34	23 484.12	2 175.95	1 366.61
SF	8 867.17	11 993.78	422.65	669.46
SM	12 156.43	8 969.03	903.35	634.52
SPF	18 053	12 853.12	1 021.14	595.35
SPM	17 054	7 560.28	860.50	412.63

Polynomial Kernel

Hyperparameter Search Space The support vector machine with dimension reduction and a polynomial kernel produces the hyperparameter search space as presented in Figure 5.26. We observe that if all features are used (i.e. $T = 1\,259$), the hyperparameter search space becomes relatively flat. The cross-validation accuracy at degree $d = 1$ and top $T = 1\,259$ varies from 0.404 to 0.407 (see Figure 5.26a). If the degree changes to degree $d = 2$, we get cross-validation accuracies all equal to 0.335. Increasing the degree to $d = 3$, gives a cross-validation accuracy of 0.302. The cross-validation Kappa also changes similarly. The cross-validation Kappa at degree $d = 1$ and $T = 1\,259$ varies from 0.252 to 0.248 (see Figure 5.26b). Increasing the degree to degree $d = 2$ gives a cross-validation

[14] We are aware that the Chi-squared test approximations might be incorrect due to the small expected values, but a Monte-Carlo approximation of the p-value also showed the same results[60].
[15] We are unable to compare single male parents (SPM) as it has no values in Table 5.4.

Kappa of 0.164 and at degree $d = 3$, the Kappa is 0.130. We also observe that reducing the number of features increases the variance of the performance measure. In addition, we also observe that if the dimension is reduced, the effect of the other hyperparameters becomes larger. This is clearly visible with the cost hyperparameter C, but also with the scale hyperparameter a to some extent. The maximum cross-validation accuracy of 0.482 is found with the hyperparameters degree $d = 1$, scale $a = 3$, $C = 2^{-4}$ and top $T = 250$. The cross-validation Kappa at this point is 0.293. Maximising the cross-validation Kappa results in the hyperparameters degree $d = 2$, scale $a = 2$, $C = 2^{-20}$ and $T = 500$. The highest cross-validation Kappa is 0.318. This is associated with an accuracy of 0.470.

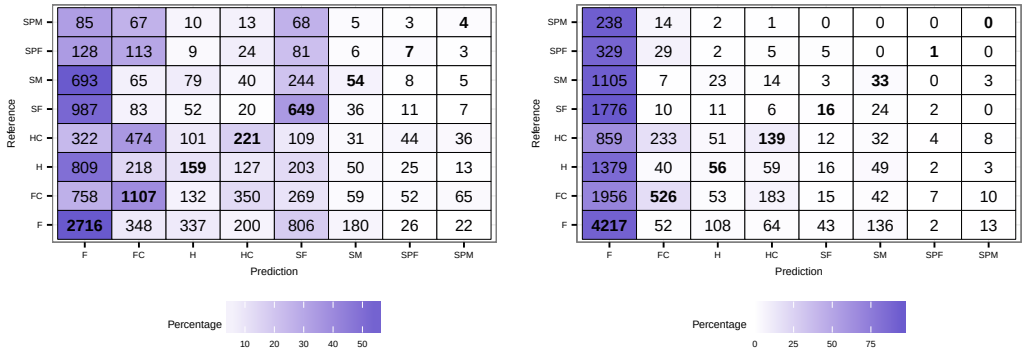


(a) Accuracy **(b)** Kappa

Fig. 5.26.: Level plot for cross-validation accuracy and Kappa values over different combinations of the degree (d), scale (a), cost (C) and Top (T) hyperparameters. The rows indicate the number of important features that are selected (T), while the columns indicate the degree (d).

Validation Validating the dimension reduced accuracy and Kappa model gives the following confusion matrices (see Figure 5.27). We observe that the dimension reduced accuracy SVMP model performs relatively well for all classes except for families (`F`) (see Figure 5.27a). The model does not seem to be able to distinguish families (`F`) from households (`H`) and single adults (`SF` and `SM`). The accuracy of the model is 0.351 with a 95% confidence interval of $(0.343, 0.359)$. The Kappa value of the model is 0.159. The validation accuracy seems to be significantly worse than our cross-validation accuracy. The dimension reduced Kappa SVMP model performs slightly better. It has an accuracy of 0.356 and a 95% confidence interval of $(0.347, 0.364)$ (see Figure 5.27b). The Kappa value however indicates that the Kappa model performs worse than the accuracy model, as it only has a Kappa value of 0.078. The Kappa model seems to be more focussed on classifying families (`F`) correctly and does not seem to be able to distinguish between families (`F`) and all other classes. Therefore, the high number of correctly classified families (`F`) is more due to the large number of household classified as families (`F`), rather than actually being able to classify families (`F`). Almost 85% of all households are classified as family (`F`). Although both models do not seem to be very

similar in their predictions, roughly 52% of their predictions are the same. The Cramers V between both models is 0.239.



(a) Accuracy **(b)** Kappa

Fig. 5.27.: Confusion matrices of different dimension reduced SVMP models on the test set.

Incorrectly Classified Households The dimension reduced accuracy SVMP model seems to be better in distinguishing between households than the Kappa version. This can be seen by the prediction results, as it attempts to classify a more diverse range of household structures. The dimension reduced Kappa SVMP model only seems to classify families (`F`). Therefore, we use the dimension reduced accuracy SVMP model for further examination.

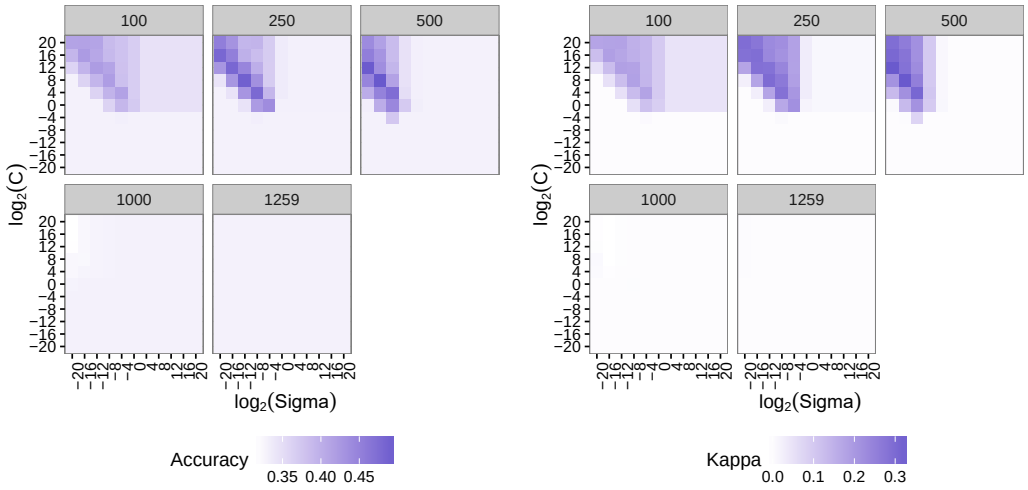
We observe that correctly classified families (`F`), single females (`SF`) and single parents (`SPF` and `SPM`) have a lower average duration than their incorrectly classified counterparts (see Table 5.10). However, using the average frequency, we only observe that families (`F`) and single female (parents) (`SF` and `SPF`) have a lower average frequency than their incorrectly classified counterparts. The dimension reduced accuracy SVMP model has a lower average duration than the restricted SVMP model. This is true for all classes, except for families (`F`) and families with children (`FC`) (see Table 5.5).

Tab. 5.10.: Average total duration and frequencies of correctly and incorrectly classified households by the dimension reduced accuracy SVMP model.

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	14 307.74	15 160.50	859.96	920.31
FC	21 141.64	16 416.35	1 285.62	977.94
H	22 955.3	20 025.64	1 329.82	1 156.95
HC	36 283.74	23 898.54	2 190.85	1 395.55
SF	8 783.92	11 194.56	437.26	594.88
SM	10 737.19	9 219.31	758.74	656.82
SPF	12 382	12 962.17	510.57	605.17
SPM	7 506.75	7 712.42	779.75	413.92

Radial Basis Kernel

Hyperparameter Search Space The hyperparameter search space of the dimension reduced SVMR model is presented in Figure 5.28. We observe that if more than 1 000 important features are selected, we get the naive classifier, with a cross-validation accuracy of around 0.330 (see Figure 5.28a). However, this improves if we only select 500 important features. The "area of interest"[16] seems to move to lower sigma values as the number of important features increase. This can also be seen in the Kappa hyperparameter search space (see Figure 5.28b). The maximum cross-validation accuracy of 0.498 can be found at $C = 2^8$, $Sigma = 2^{-16}$ and $T = 500$. The cross-validation Kappa value with these hyperparameters is 0.340. This is also the optimal cross-validation Kappa value in our hyperparameter search space.



(a) Accuracy **(b)** Kappa

Fig. 5.28.: Level plot for cross-validation accuracy and Kappa values over different combinations of the cost (C), sigma ($Sigma$) and Top (T) hyperparameters. The *Top* label indicates the number of top most important variables used in training the model.

Validation Next, we validate the dimension reduced SVMR model against the test set and achieve an accuracy of 0.408 with a 95% confidence interval between $(0.400, 0.416)$ (see Figure 5.29). The Kappa is 0.225. This is a major improvement compared to the SVMR model without dimension reduction (see Section 5.2.3). From the confusion matrix we observe that the dimension reduced SVMR model classifies families (`F`) and families with children (`FC`) very well. This also applies to single females (`SF`), but not to single males (`SM`). Many of the households that are classified as single males (`SM`) are actually families (`F`). This is also similar for single parents (`SPF` and `SPM`), but the majority of the single parents are families with children (`FC`). In addition, the model also has some problems with classifying households (i.e. households (`H`) and households with children (`HC`)) as it classifies more families (`F`) as a household (`H`) than actual households (`H`) itself. This is also the case when children are present.

[16] In this context, we define the area of interest as an area in the hyperparameter search space that has a cross-validation performance measure value that is higher than either the naive classifier or zero.

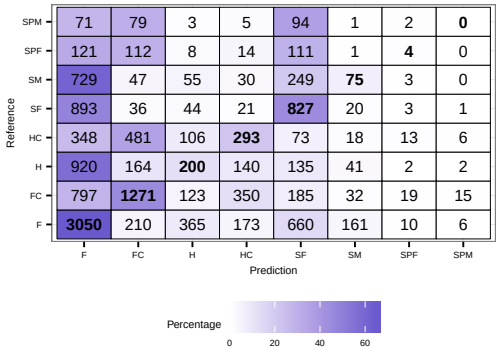


Fig. 5.29.: Confusion matrix of the dimension reduced SVMR model on the test set.

Incorrectly Classified Households From Table 5.11, we observe that the average total duration is higher for correctly classified household except for families (`F`) and single females (`SF`). This is also the same case for the average total frequencies. Unfortunately, since the SVMR model results in a naive classifier, we do not compare the classification results of the SVMR model and its dimension reduced version.

Tab. 5.11.: Average total duration and frequencies of correctly and incorrectly classified households by the dimension reduced SVMR model

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	13 832.66	16 254.40	820.81	1 008.36
FC	20 758.41	16 227.10	1 249.32	975.10
H	26 570.61	19 425.09	1 642.97	1 107.29
HC	38 048.36	22 550.44	2 350.08	1 296.11
SF	8 881.4	11 536.87	419.86	636.57
SM	12 899.41	9 044.97	930.22	643.34
SPF	19 739.75	12 877.24	867.75	600.50
SPM		7 709.20		419.65

Comparing Kernels

To determine which kernels performs the best, we analyse resampling distribution of the cross-validation performance measures as discussed in Section 3.4.2. From the boxplots, we observe that all support vector machine kernels perform similarly (see Figure 5.30). For all pairs, the difference between the means and medians of the cross-validation accuracies do not seem to be significant (see Table 5.12). For the cross-validation Kappa, we do observe some significant differences. This is most likely due to the bad performance of the polynomial kernel. The difference between the cross-validation Kappa mean and median seem to be significant between the linear and polynomial kernel under a significance level of 10%. For the polynomial and radial basis kernel pair, only the difference between the mean seems to be significant. These results suggest that, based on the cross-validation accuracy resampling distributions all kernels perform similarly, while based on the cross-validation Kappa resampling distributions, the linear and radial basis kernel are better than the polynomial kernel. This strengthens the results presented in the validation phase of each kernel. The linear kernel reported accuracies of 0.403 and 0.411, polynomial kernel

0.351 and 0.356 and radial basis kernel 0.408. For the Kappa values, we observe for the polynomial kernel values of 0.078 and 0.159, while for the remaining kernels, these were at least 0.2.

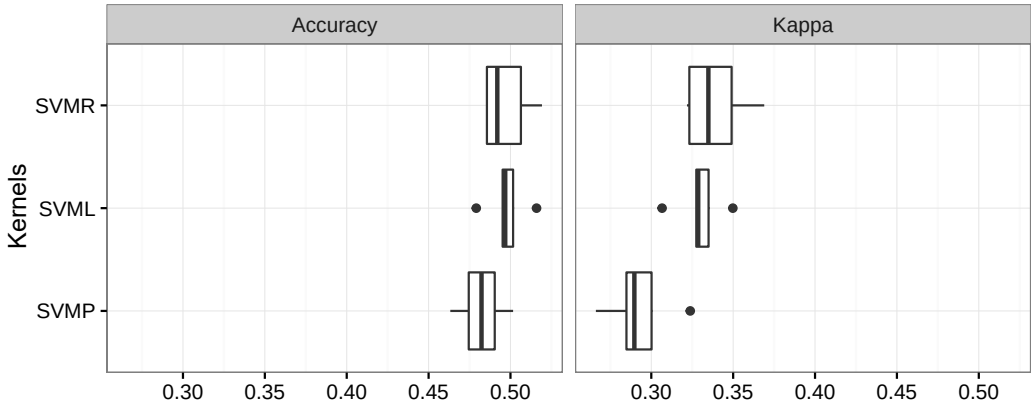


Fig. 5.30.: The resampling distributions with different performance measures and kernels.

Tab. 5.12.: P-values for testing the differences between the means and medians of the cross-validation performance measures.

Pair		Accuracy		Kappa	
		T-test	Wilcoxon	T-test	Wilcoxon
SVML	SVMP	0.210	0.313	0.048	0.063
SVML	SVMR	0.996	1.000	0.380	0.813
SVMP	SVMR	0.153	0.188	0.022	0.125

Overview

In this section we present a short overview of the AUC GLMNET dimension reduced SVM models and compare them with each other and with the models from Klein[47].

The support vector machine with a linear kernel performs relatively well. The accuracy SVML has the highest accuracy, while the Kappa SVML has the highest Kappa (see Table 5.13). The confusion matrices from both models seems to share some similarities with the Kappa GLMNET model from the previous section, as both seem to have difficulties with distinguishing families (`F`) from other classes (see Figure 5.3b and 5.25). From the average durations and frequencies, we observe our reoccurring pattern. Correctly classified families (`F`) and single females (`SF`) still have lower average durations and frequencies than their incorrectly classified counterparts (see Table 5.9). Comparing the AUC GLMNET dimension reduced Kappa SVML model with the AUC SVML model shows that the average frequencies are lower for most classes, which could indicate that the dimension reduced model is better in classifying households with lower frequencies. The mapped accuracy of the AUC GLMNET dimension reduced SVML models do not exceed the SVML models in the base model section, but the differences are not very large. This also implies that the AUC GLMNET models do not seem to perform better than the logit model discussed by Klein [47].

The AUC GLMNET dimension reduced SVMP model performs only slightly better than the naive classifier, as the naive classifier has an accuracy of 0.330. The AUC GLMNET dimension reduced SVMP models have an accuracy ranging from 0.351 to 0.356 (see Table 5.13). The AUC GLMNET dimension reduced Kappa SVMP model seems to performing slightly better than the AUC GLMNET dimension reduced accuracy SVMP model, while the AUC GLMNET dimension reduced accuracy SVMP model performs better with respect to the Kappa statistic[17]. The confusion matrix of the AUC GLMNET dimension reduced accuracy SVMP has problems in distinguishing between families (`F`) and other household structures. Only 2 716 out of 4 635 are classified correctly (see Figure 5.27). The AUC GLMNET dimension reduced Kappa SVMP fares a bit better, but lacks the ability of classifying other classes than families (`F`). For example, the AUC GLMNET dimension reduced Kappa SVMP model is only able to classify 16 single females (`SF`), while the AUC GLMNET dimension reduced accuracy SVMP is able to classify 649 single females (`SF`). The average duration and frequencies of the AUC GLMNET dimension reduced accuracy SVMP show a slightly different pattern, where the average duration for correctly classified families (`F`), single females (`SF`) and single female parents (`SPF`) are lower than their incorrectly classified counterpart (see Table 5.10). The difference between the average duration of the correctly and incorrectly classified single female parents (`SPF`) is very small though ($\Delta = 580$ minutes). In addition, the average durations and frequencies appear to be similar to the AUC GLMNET dimension reduced Kappa SVML model, with the exception of single parents (`SPF` and `SPM`). The AUC GLMNET dimension reduced SVMP model is performing better than its unrestricted counterpart in the previous section (see Table 5.7 and 5.13). However, the restricted SVMP model seems to perform much better than the AUC GLMNET dimension reduced SVMP models. This is also visible in the mapped accuracies and thus does not perform any better than the logit model by Klein [47].

We do observe some improvements in the AUC GLMNET dimension reduced SVMR model as it performs much better than the naive classifier (see Table 5.13). From the confusion matrix the SVMR model is also able to predict many different classes, but still fails to predict any single male parent (`SPM`) correctly (see Figure 5.29). The average durations and frequencies show again the same pattern, with correctly classified families (`F`) and single females (`SF`) having a lower average duration and frequency than their incorrectly classified counterparts (see Table 5.11). The mapped accuracy of the AUC GLMNET dimension reduced SVMR model is 0.567, but this is still not an improvement over the logit model of Klein [47].

Tab. 5.13.: Performance measures for the AUC GLMNET dimension reduced models.

Algorithm	Performance Measure	Accuracy	Mapped Accuracy[18]	Confidence Interval		Kappa
				Min	Max	
SVML	accuracy	0.411	0.564	0.403	0.419	0.202
SVML	Kappa	0.403	0.559	0.395	0.411	0.232
SVMP	accuracy	0.351	0.498	0.343	0.359	0.159
SVMP	Kappa	0.356	0.492	0.347	0.364	0.078
SVMR	-	0.408	0.567	0.400	0.416	0.225

[17] Even though this is strange, it has been verified multiple times, that this is not an error.
[18] Mapped to classes as defined by Klein [47].

From the algorithm comparison, we observe that the AUC GLMNET dimension reduced SVMP algorithm performs the worst and is significantly worse than the SVML and SVMR. The AUC GLMNET dimension reduced SVML and SVMR algorithms do not seem to be significantly different. This can also be observed from the test set, as the accuracy SVML performs better than the SVMR model, but the Kappa SVML performs slightly worse (see Table 5.13).

5.3.2 Support Vector Machines With Dimension Reduction Using a Stochastic Gradient Boosting Model

We now proceed with dimension reduction using the model created in the previous section using stochastic gradient boosting (see Section 5.2.2). Again, we use the same model as during the validation process. This is also a better model than with only $M = 100$ boosting iterations. In addition, all performance measures select only one set of hyperparameters. This eliminates the fact that we have to choose from different sets of hyperparameters selected by different performance measures. The stochastic gradient boosting model with $M = 1\,500$ boosting iterations contains a total of 3 854 features (see Section 5.2.2). These features are then ranked based on their variable importance as defined in Section 3.2.1. This ranking will be used to introduce a new hyperparameter T, which selects the top T most important features from this ranking and drops the remaining features in order to train the support vector machine models.

Selected Features

We now proceed to examine the features that are selected by our dimension reduction approach. The top 100 features consist of 77 duration and 22 frequency features (see Table 5.14). The remaining two features are `NPrograms` and `NChannels`. This gives an average duration of 31 706.17 minutes and an average frequency of 139.48. Taking the top 250 most important features, we observe more than a doubling in the average duration, while the average frequency quadruples. Doubling the number of important features to 500 results in a doubling of duration and frequency features. The average frequency is also doubled, while the average duration increases to 93 817.93. The majority of the most important features are duration features. However, this changes to frequency features when more features are added that are used in the GBM model. When all features of the GBM model are used, it results in a data set with 3 854 features, where 1 268 are duration features, 2 584 are frequency features and 2 features for counting the number of channels and programs watched.

Tab. 5.14.: Summary statistics of different subsets of selected features.

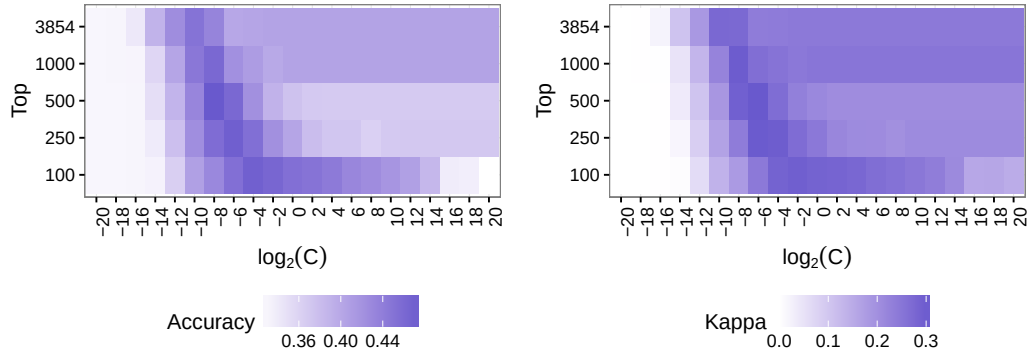
Top	Number of features			Average		Median	
	Duration	Frequency	Other	Duration	Frequency	Duration	Frequency
100	77	21	2	31 706.17	139.48	449	11 823
250	196	52	2	64 340.09	568.799	1 112	36 932.5
500	393	105	2	93 817.93	1 339.58	1 950	62 989.5
1 000	761	237	2	130 903.69	2 342.52	3 755	93 915
3 854	1 268	2 584	2	228 892.13	7 123.80	8 333.28	168 273

If we compare the selected features between the GBM and AUC GLMNET model, we arrive at the conclusion that the main difference lies in the duration and the frequency of the selected features. The GBM model uses features that have a high duration/frequency, while the AUC GLMNET model uses features with a lower duration/frequency. In addition, the most important features consist mainly of frequency features in the AUC GLMNET model, while the GBM model considers durations more important. In AUC GLMNET model, all features in the top 100 most important features are frequency features, while in the GBM model, only 21 are frequency features (see Table 5.8 and 5.14). The AUC GLMNET model also seems to favour features that are relative sparse. The features used by the AUC GLMNET model contain 14 492 zeroes on average, while the GBM "only" contain 14 202. This is still much better than without dimension reduction, as the average number of zeroes in a feature is 15 031.05.

Linear Kernel

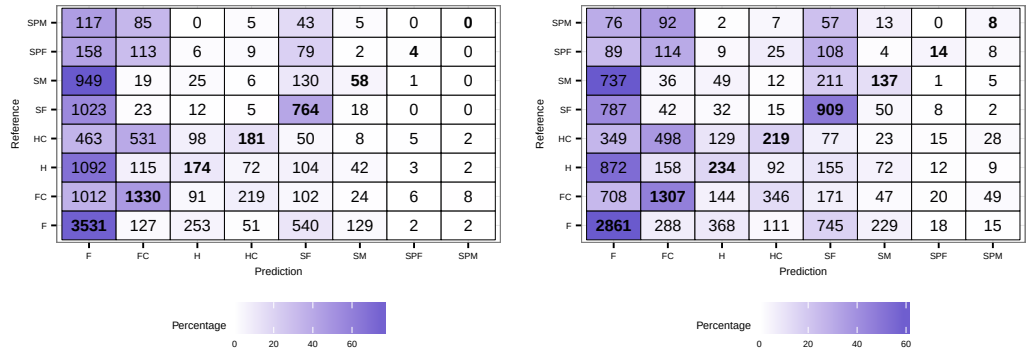
Hyperparameter Search Space The support vector machine with a linear kernel and the dimension reduction procedure with the GBM model produces the following hyperparameter search spaces as presented in Figure 5.31. Reducing the features to only the top 100 important features gives a maximum cross-validation accuracy of 0.47 with the cost hyperparameter set to $C = 2^{-4}$ (see Figure 5.31a). The corresponding cross-validation Kappa is 0.289 (see Figure 5.31b). The maximum cross-validation Kappa with 100 important features is 0.297 with $C = 2^{-2}$. Increasing the number of important features to 250 leads to a similar maximum cross-validation accuracy of 0.47 with $C = 2^{-6}$. The maximum cross-validation Kappa of 0.305 using only 250 features is also at this point. Increasing the number of features to 500, gives us the maximum overall cross-validation accuracy of 0.477 at $C = 2^{-8}$. At $C = 2^{-6}$ the maximum overall cross-validation Kappa of 0.309 is reached. Continuing adding more features does not appear to be beneficial. With 1 000 features, the maximum cross-validation accuracy decreases slightly to 0.462 at $C = 2^{-8}$. The maximum cross-validation Kappa decreases to 0.302, which also happens to be at $C = 2^{-8}$. We also observe a pattern for each subset of important features. For each subset of important features, we observe that if the cost hyperparameter C is set too low, we will get a bad performance. Increasing the cost hyperparameter C slightly, results in a major improvement of the cross-validation performance measures. A further increase of the cost hyperparameter C would result in a constant cross-validation performance measure value.

Validation We now proceed to validate both dimension reduced models against the test set. This gives us the resulting confusion matrices (see Figure 5.32). The accuracy model has an accuracy of only 0.4307 with a 95% confidence interval between $(0.4225, 0.4390)$ (see Figure 5.32a). The Kappa of this model is 0.2331. All performance measures seem to be less than what would be expected based on the cross-validation. The Kappa model fares slightly worse, with an accuracy of 0.4055 and a 95% confidence interval between $(0.3974, 0.4137)$ (see Figure 5.32b). The Kappa is 0.2293, which is also slightly lower than the Kappa of the accuracy model. The Kappa model does predict more classes. However, this is at the expense of predicting families (`F`) accurately. Although the confusion matrices are very different in both models, $\frac{2}{3}$ of the predictions are the same. The Cramers V between both models is 0.577.



(a) Accuracy **(b)** Kappa

Fig. 5.31.: Level plot for cross-validation accuracy and Kappa values over different combinations of the cost (C) and Top (T) hyperparameters. The *Top* label indicates the number of top most important variables used in training the model.



(a) Accuracy **(b)** Kappa

Fig. 5.32.: Confusion matrices of different dimension reduced SVML models on the test set.

Incorrectly Classified Households We argue it is more interesting if a model predicts more classes, therefore, we will examine the dimension reduced Kappa SVML model. The average duration and frequencies are tabulated in Table 5.15. All correctly classified households have a higher average duration than their incorrectly classified counterparts, except families (`F`) and single females (`SF`). The same goes for the average frequencies. Also, we observe very similar summary statistics as with the GLMNET dimension reduced Kappa SVML model. This suggests that the GBM dimension reduced Kappa SVML model is also able to cope with low average frequencies and durations. This is confirmed by comparing both tables (see Table 5.4 and 5.15).

Polynomial Kernel

Hyperparameter Search Space A support vector machine with a polynomial kernel and dimension reduction with the GBM model produces the following hyperparameter search space (see Figure 5.33). We observe that using a degree of $d = 3$ in general produces models that do not exceed a cross-validation accuracy of 0.379, which is reached with scale $a = 2$, cost $C = 2^{-20}$ and $T = 100$ (see Figure 5.33a). Increasing the number of features seems to

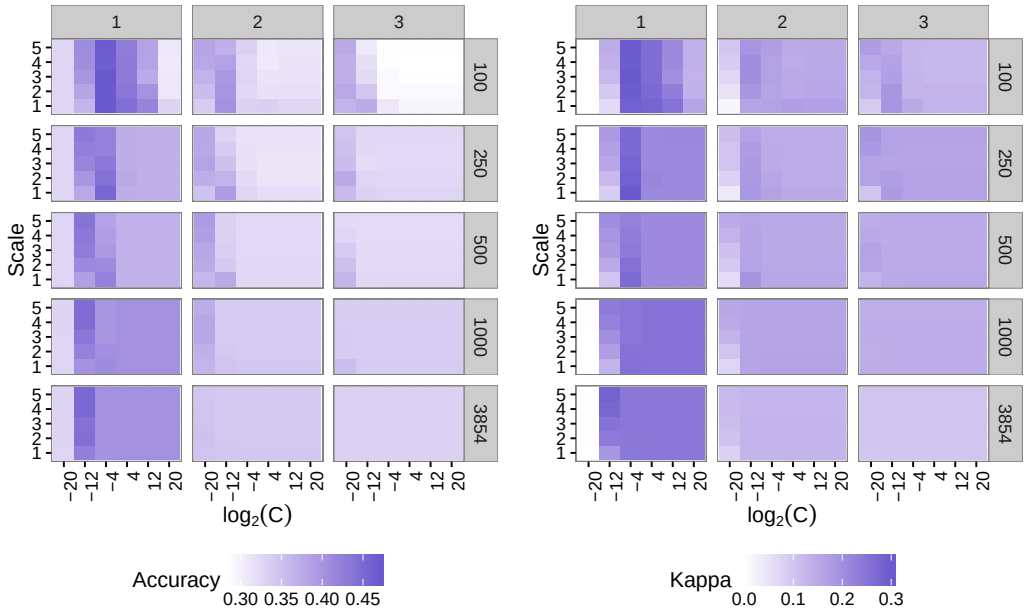
Tab. 5.15.: Average duration and frequencies of correctly and incorrectly classified households by the dimension reduced Kappa SVML model.

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	14 068.89	15 615.40	879.39	893.90
FC	19 446.57	17 271.84	1 201.56	1 010.48
H	24 061.63	19 676.29	1 370.79	1 140.49
HC	37 368.57	23 708.37	2 274.45	1 380.61
SF	9 141.77	11 516.65	454.11	622.30
SM	12 268.03	8 899.89	907.74	629.35
SPF	20 495.29	12 655.38	940.93	590.15
SPM	19 163.38	7 338.21	842	405.98

worsen the model, but stabilises the hyperparameter search space. If all features are used, the hyperparameter search space seems to stabilise at a cross-validation accuracy between 0.333 to 0.334. Using Kappa as the performance measure, we get similar results with a degree $d = 3$. The highest cross-validation Kappa with degree $d = 3$ is 0.189 with scale $a = 1$, cost $C = 2^{-12}$ and $T = 100$ (see Figure 5.33b). The cross-validation Kappa also stabilises when more features are added. When all features are used, the cross-validation Kappa is between 0.1 and 0.102. With degree $d = 2$, we get a maximum cross-validation accuracy of 0.404 with scale $a = 1$, cost $C = 2^{-12}$ and $T = 100$. The highest overall cross-validation accuracy of 0.47 is reached at degree $d = 1$, scale $a = 3$ and $T = 100$. These hyperparameters also give the maximum cross-validation Kappa of 0.303. It can also be observed that only with a degree $d = 1$ a naive classifier is produced. This is at a $C = 2^{-20}$. However, we also believe that with a small enough cost hyperparameter C at different degrees, a naive classifier can be produced, as it would cost nothing to classify incorrectly. The top 100 most important features seems to give the best cross-validation performance measures and adding more only seems to decrease the performance.

Validation Since the accuracy and Kappa performance measures agree on the hyperparameters, we only have one model with degree $d = 1$, scale $a = 3$, $C = 2^{-4}$ and $T = 100$. Validating the dimension reduced SVMP model against the test set gives us an accuracy of 0.435 with a confidence interval between $(0.427, 0.444)$ and a Kappa of 0.2523 (see Figure 5.34). We observe that the model can classify at least some households correctly for all classes.

Incorrectly Classified Households From the average durations of the GBM dimension reduced SVMP model, we observe the reoccurring pattern that every correctly classified household has a higher average duration than their incorrectly classified counterparts, except families (`F`) and single females (`SF`) (see Figure 5.16). Considering the average frequencies, the exception seems only to be single females (`SF`). The GBM dimension reduced SVMP model seems to be similar to the GLMNET dimension reduced SVMP model. For example, the average duration of correctly classified families (`F`) is 14 459.75 for the GBM dimension reduced version, while 14 307 for the GLMNET dimension reduced version. This is observed in the average duration of all classes except for single parents (`SPF` and `SPM`). The average duration of correctly classified single female parents (`SPF`) is more than twice



(a) Accuracy **(b)** Kappa

Fig. 5.33.: Level plot for cross-validation accuracy and Kappa values over different combinations of the degree (d), scale (a), cost (C) and Top (T) hyperparameters. The rows indicate the number of important features that are selected (T), while the columns indicate the degree (d).

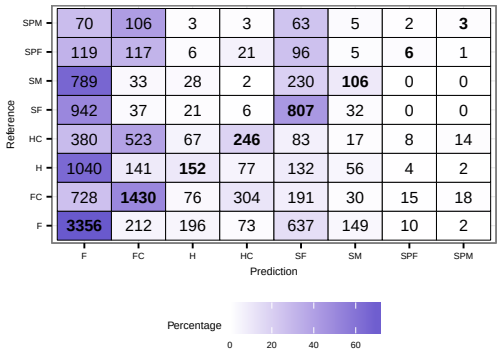


Fig. 5.34.: Confusion matrix of the dimension reduced SVMP model on the test set.

as high as the average duration of the same class in the GLMNET dimension reduced accuracy SVMP model. Further investigation shows that this is not due to only one household being correctly classified, as the GBM dimension reduced SVMP model classifies 6 single female parents correctly, while the GLMNET dimension reduced SVMP model classifies 7 (see Figure 5.34 and 5.27a). This suggests that the GBM features allow the SVMP model to classify single female parents (`SPF`) correctly that have long durations. The same can be seen for single male parents (`SPM`). Although the average durations of correctly classified single parents (`SPF` and `SPM`) are much higher, the average durations of their incorrectly classified counterparts are similar to those of the GLMNET dimension reduced SVMP model.

Tab. 5.16.: Average duration and frequencies of correctly and incorrectly classified households by the dimension reduced SVMP model.

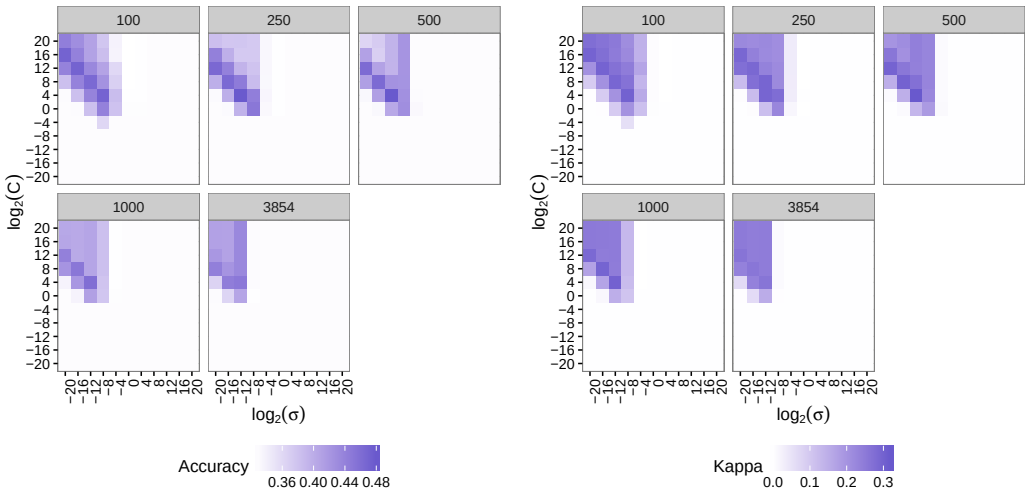
Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	14 459.75	15 188.36	893.13	863.48
FC	20 147.51	16 339.51	1 230.13	963.23
H	28 402.59	19 469.53	1 607.67	1 128.70
HC	37 976.78	23 233.60	2 268.89	1 359.76
SF	8 838.59	11 518.99	413.64	637.24
SM	12 984.63	8 926.19	1 114.22	617.10
SPF	30 238.17	12 667.06	1 440.5	589.62
SPM	15 302	7 618.81	942.67	413.43

Radial Basis Kernel

Hyperparameter Search Space The hyperparameter search spaces of the GBM dimension reduced SVMR model are shown in Figure 5.35. In the accuracy hyperparameter search space, we observe that there are large regions that perform similar to the naive classifier. Using the top $T = 100$ features, a maximum cross-validation accuracy of 0.471 can be achieved by setting $C = 2^{16}$ and $\sigma = 2^{-20}$. This is also the location for the maximum cross-validation Kappa of 0.299 with only $T = 100$ features. Increasing the number of features to $T = 250$ gives us a smaller "area of interest", similar as before. The maximum cross-validation accuracy is 0.479, with $C = 2^4$ and $\sigma = 2^{-12}$. Again, these hyperparameters also give the maximum cross-validation Kappa of 0.295 with $T = 250$ features. Adding more features to $T = 500$ gives us the highest overall maximum cross-validation accuracy of 0.48, with $C = 2^4$ and $\sigma = 2^{-12}$. With the same hyperparameter values, the overall maximum cross-validation Kappa of 0.32 is also reached. Increasing the number of features further does not seem to increase the maximum cross-validation accuracy nor Kappa. In all graphs the "area of interest" also seems to be moving, similar to the previous dimension reduced SVMR model. For example, with $T = 100$ features, we observe that the cross-validation accuracy with $\sigma = 2^{-4}$ ranges from 0.329 to 0.386. If we were to increase the number of features to $T = 250$, the range becomes narrower, ranging only from 0.329 to 0.335. This is also visible given a fixed cost value.

Validation Validating the GBM dimension reduced SVMR model against the test set gives us an accuracy of 0.429 with a 95% confidence interval between $(0.421, 0.437)$ and a Kappa of 0.249 (see Figure 5.36). The model is able to classify some households correctly for all classes. This model also seems to be better at classifying all classes than the GLMNET dimension reduced SVMR model, with the exception of households with children (`HC`).

Incorrectly Classified Households The average durations of correctly classified households are in most cases higher than their incorrectly classified counterparts, with the exception of families (`F`) and single females (`SF`) (see Figure 5.17). This is also the case for the average frequencies. These values are also very similar to the GLMNET dimension reduced SVMR model.



(a) Accuracy **(b)** Kappa

Fig. 5.35.: Level plot for cross-validation accuracy and Kappa values over different combinations of the cost (C), sigma ($Sigma$) and Top (T) hyperparameters. The *Top* label indicates the number of top most important variables used in training the model.

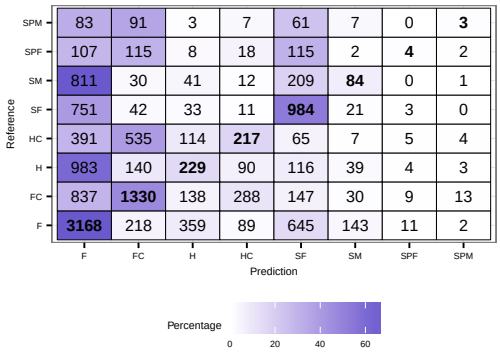


Fig. 5.36.: Confusion matrix of the dimension reduced SVMR model on the test set.

Tab. 5.17.: Average duration and frequencies of correctly and incorrectly classified households by the dimension reduced SVMR model.

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	14 038.41	16 004.87	869.42	918.47
FC	20 815.21	15 992.56	1 275.69	940.04
H	27 084	19 188.88	1 476.05	1 123.80
HC	37 796.73	23 649.86	2 302.76	1 376.72
SF	8 919.96	11 977.02	446.45	645.70
SM	13 394.57	8 975.87	961.54	638.62
SPF	21 214.75	12 861.16	785.5	601.40
SPM	18 298.33	7 583.14	964	413.17

Comparing Kernels

In order to determine which kernels perform the best, we examine the resampling distribution of the different dimension reduced kernels. We observe that the variance in the cross-validation accuracy for the polynomial and radial basis kernel are similar, while the variance in the cross-validation accuracy for the linear kernel is larger (see Figure 5.37). The average cross-validation of the linear kernel seems to be the highest, while the polynomial kernel has the lowest average cross-validation accuracy. The cross-validation Kappa shows a slightly different result. Based on the average cross-validation Kappa, the radial basis kernel is preferred, although the differences are not very large.

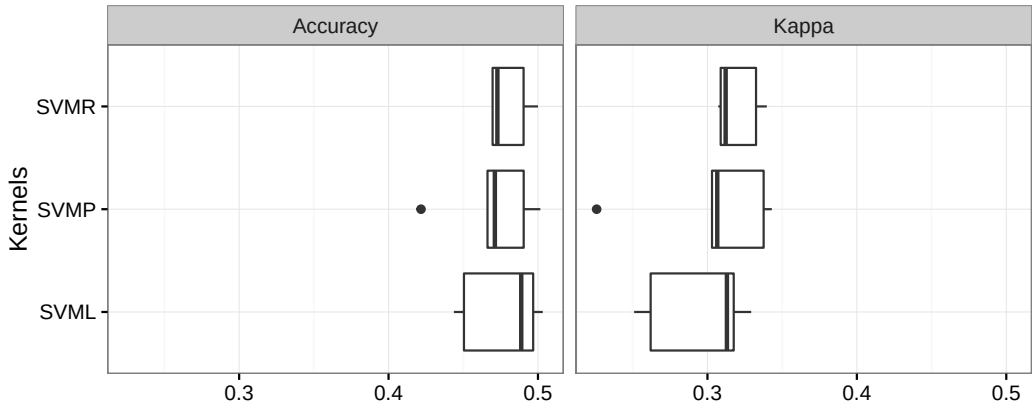


Fig. 5.37.: The resampling distributions with different performance measures and kernels.

Unfortunately, though there are some differences in the variances and averages, it can be shown using the t-test and the Wilcoxon signed rank test that none of them are significantly different from another (see Table 5.18).

Tab. 5.18.: P-values for testing the differences between the means and medians of the cross-validation performance measures.

Pair		Accuracy		Kappa	
		T-test	Wilcoxon	T-test	Wilcoxon
SVML	SVMP	0.569	0.625	0.604	0.813
SVML	SVMR	0.679	0.813	0.107	0.125
SVMP	SVMR	0.487	0.438	0.433	0.438

Overview

In this section we present a short overview of the GBM dimension reduced SVM models and compare them with each other and with the models present in Klein [47].

The GBM dimension reduced SVML is an improvement over the SVML models in the base models section. The accuracies from the GBM dimension reduced model ranges from 0.406 to 0.431 (see Table 5.19). All performance measures seem to agree that the GBM dimension reduced accuracy SVML model is better than the GBM dimension reduced Kappa SVML model. The GBM dimension reduced accuracy SVML model seems to better at predicting families (`F`) and families with children (`FC`), while the GBM dimension reduced Kappa SVML

model is better at predicting other classes (see Figure 5.32). This is however insufficient to compensate for the misclassifications made in in families (F) and families with children (FC) (see Figure 5.32). The confusion matrices of the GBM dimension reduced SVML models are very similar to the confusion matrices of the AUC GLMNET dimension reduced SVML models, even though they use a different subset of features (see Figure 5.25 and 5.32). The AUC GLMNET dimension reduced accuracy SVML model seems to be slightly better at classifying families (F), while the GBM dimension reduced counterpart is better at classifying other household structures (see Figure 5.25a and 5.32a). The GBM dimension reduced Kappa SVML model is also better at classifying every household structure with the exception of households with children (HC) and single females (SF) compared to its AUC GLMNET dimension reduced counterpart (see Figure 5.25b and 5.32b). However, these differences are relatively small. The average durations and frequencies of both models are also very similar (see Table 5.9 and 5.15). Although, the accuracy of the SVML model has improved, the mapped accuracy is still lower than the accuracy of the logit model by Klein [47], which is 0.596 (see Table 5.19).

The GBM dimension reduced SVMP model also shows some improvements over its AUC GLMNET dimension reduced counterpart. It has the highest accuracy among the GBM dimension reduced SVM models, which is 0.435 (see Table 5.19). The confusion matrix shows that this model also has trouble distinguishing between families and other household structures (see Figure 5.34). For example, the GBM dimension reduced SVMP model classifies 162 households (H) correctly, but it also classifies 196 families (F) as households (H). The GBM dimension reduced SVMP model shows some differences with its AUC GLMNET dimension reduced counterparts. The GBM dimension reduced SVMP model outperforms the AUC GLMNET dimension reduced SVMP models in all classes, except for families (F) in the AUC GLMNET dimension reduced Kappa model (see Figure 5.27 and 5.34). The average durations and frequencies of the GBM dimension reduced SVMP model also shows many similarities with the average durations and frequencies of the GLMNET dimension reduced accuracy SVMP model, with the exception of single female parents (SPF) (see Table 5.10 and 5.16). Correctly classified single female parents have an average duration of 30 238.17 in the GBM dimension reduced SVMP model, while they have an average duration of 12 382 in the GLMNET dimension reduced accuracy SVMP model. Both models correctly classify roughly the same number of single female parents (SPF) (6 versus 7), therefore, we can conclude that the GBM dimension reduced SVMP model correctly classifies single female parents (SPM) that have long durations, while the GLMNET dimension reduced accuracy SVMP model is able to correctly classify single female parents (SPF) with lower average durations. The mapped accuracy of the GBM dimension reduced SVMP model comes close to the accuracy of the logit model by Klein [47], but still performs slightly worse (see Table 5.19).

The GBM dimension reduced SVMR model also shows a slight improvement, from 0.408 in the GLMNET dimension reduced version to 0.429 (see Table 5.19). The confusion matrix shows more correctly classifications in all household structures compared its GLMNET dimension reduced counterpart, with the exception of households with children (HC) (see Figure 5.29 and 5.36). The average durations and frequencies are similar, with the only major difference being that the GBM dimension reduced model is able to classify some single male parents (SPM) correctly (see Table 5.11 and 5.17). The GBM dimension reduced

SVMR model has the highest mapped accuracy from all dimension reduced models, however it is still slightly worse than the logit model by Klein [47].

Tab. 5.19.: Performance measures for the GBM dimension reduced models.

Algorithm	Performance Measure	Accuracy	Mapped Accuracy[19]	Confidence Interval		Kappa
				Min	Max	
SVML	accuracy	0.431	0.586	0.423	0.439	0.233
SVML	Kappa	0.406	0.564	0.397	0.414	0.229
SVMP	-	0.435	0.590	0.427	0.444	0.252
SVMR	-	0.429	0.592	0.421	0.437	0.249

From the resampling distributions we cannot conclude that the algorithms perform significantly different with respect to the performance measures accuracy and Kappa. From the validation set, we observe that the GBM dimension reduced SVMP model performs the best with respect to accuracy and Kappa, however the mapped accuracy of the GBM dimension reduced SVMP model is still slightly lower than that of the GBM dimension reduced SVMR model (see Table 5.19).

5.4 Cascading Classification Models

A different method of classifying households is using a cascading classification approach. This means that we first classify the household on a certain aspect and in the second stage, we classify the household on a different aspect with an augmented data set containing the predictions made in the first stage. In our case there are two different aspects. The first aspect is regarding whether a household contains a child. The second aspect is regarding the household structure, which can be a family (`F`), household (`H`), single female (`SF`) or a single male (`SM`). To determine the order of aspects to be classified, we first examine the majority classes for each aspect. From the distribution, we observe that the 66.09% of the households have no children and the largest household structure (irrespectively whether children are present) are families (`F`), which is 52.28% of the households. Since the naive classifier can be seen as a lower bound on the performance of a classifier, classifying whether a child is present should give better results in the worst case. In addition, a binary classification is much easier than a multinomial classification. For these reasons, we choose to classify the presence of a child in the first stage and the household structure in the second stage. The predictions made in the first stage can then be used to augment the data set for the second stage classification. Therefore, it is also important that the first stage classification should have a high accuracy[20].

[19] Mapped to classes as defined by Klein [47].

[20] It is also possible for the second stage algorithm to simply not use the first stage predictions, but as this is not certain, it is beneficial to have an accurate additional feature.

5.4.1 First Stage Classification

Multinomial Logistic Model

Hyperparameter Search Space With a multinomial logistic model, the highest cross-validation accuracy is 0.808. This is a 0.148 improvement over naive classifications, where every household is classified as having no children. The highest cross-validation accuracy results in the hyperparameters $\alpha = 0.775$ and $\lambda = 0.0179$ (see Figure 5.38). The AUC and Kappa at this point are 0.856 and 0.531. Examining the hyperparameter search space shows a relatively stable accuracy search space, with a cross-validation accuracy of at least 0.78 for small regularization values ($\lambda \leq 0.04$). This can also be seen with different performance metrics, such as Kappa and AUC (see Appendix D.2.1, Figure D.9b and D.9a). The optimal hyperparameter values for AUC is $\alpha = 0.1$ and $\lambda = 0.262$ with an accuracy of 0.786 and a Kappa of 0.45. This suggests that the optimal AUC models leans more towards a ridge regression than a Lasso. The optimal hyperparameter values for Kappa is $\alpha = 0.55$ and $\lambda = 0.0148$, which is similar to the hyperparameter values selected using accuracy.

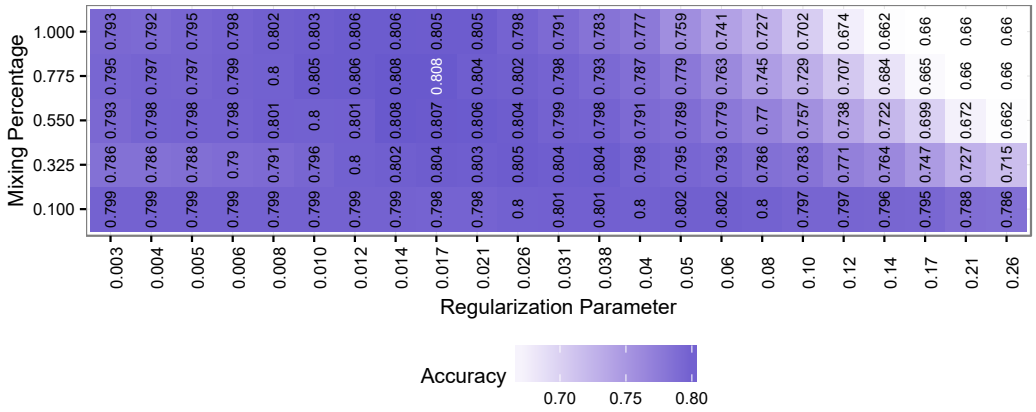
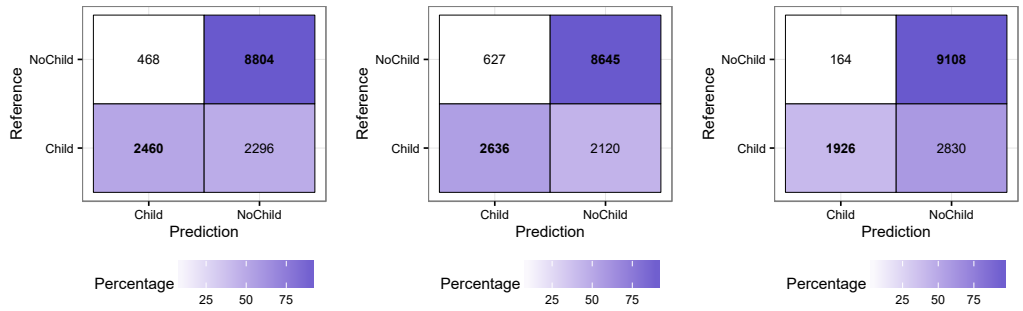


Fig. 5.38.: Level plot for cross-validation accuracy values over different combinations of the mixing (α) and regularization (λ) hyperparameters.

Validation Using these models to classify the test data set results in the following confusion matrices (see Figure 5.39). The accuracy model achieves an accuracy of 0.803, with a 95% confidence interval of $(0.763, 0.810)$ (see Figure 5.39a). The Kappa and ROC of the accuracy model are 0.515 and 0.862. It can be seen from the confusion matrix that the accuracy model performs very well, as only 468 households are classified incorrect as having children, while they do not have any children and 2 296 household are classified as having no children even though they are present. The Kappa model performs slightly better than the accuracy model, with a test accuracy of 0.804, with a 95% confidence interval between $(0.798, 0.811)$ (see Figure 5.39b). The Kappa and AUC of the Kappa model are 0.527 and 0.851. With the AUC model an accuracy of 0.787 is achieved, with a 95% confidence interval of $(0.7797, 0.7933)$ (see Figure 5.39c). This is slightly worse than the accuracy model. This is also confirmed by the Kappa of 0.449. The AUC value of 0.883 however suggest that the AUC model is preferred.



(a) Accuracy **(b)** Kappa **(c)** AUC

Fig. 5.39.: Confusion matrices of different GLMNET models on the test set.

Stochastic Gradient Boosting

Hyperparameter Search Space For stochastic gradient boosting the shrinkage parameter and sampling size are set to $v = 0.001$ and $\eta = 0.5$, respectively. This results in the following hyperparameter search space (see Figure 5.40). The search space is relatively flat. When using less than $M = 250$ boosting iterations, the stochastic gradient boosting model simply does not perform any better than the naive classifier. After $M = 500$ boosting iterations, the accuracy quickly jumps up to around 0.8 irrespectively of the interaction depth s or minimum observations per node $|R|_{min}$. Increasing the number of trees to $M = 1\,250$ does slightly improve the accuracy with 0.02. The optimal hyperparameter values are $M = 1\,250$ boosting iterations, interaction depth of $s = 40$ and minimum observations in node of $|R|_{min} = 20$. This also gives the optimal Kappa and AUC of 0.613 and 0.878, respectively. The hyperparameter search space of Kappa is very similar to accuracy and is also relatively stable (see Appendix D.2.1, Figure D.10a). The AUC hyperparameter search space only differs slightly with more irregularities (see Appendix D.2.1, Figure D.10b).

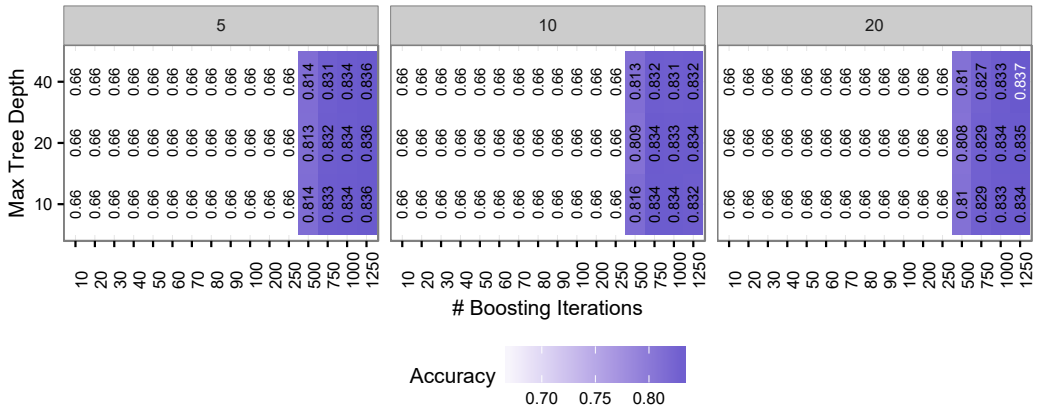
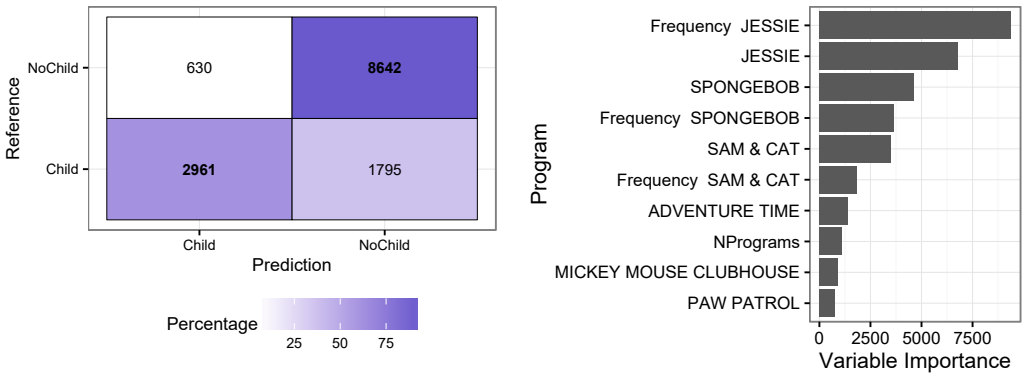


Fig. 5.40.: Level plot for accuracy over different values of boosting iterations (M), interaction depths (s) and minimum observations per node ($|R|_{min}$). The shrinkage (v) and sampling fraction (η) stay constant at $v = 0.001$ and $\eta = 0.5$.

Validation The test set accuracy of this model is 0.827, with a 95% confidence interval between $(0.8208, 0.8334)$. The Kappa and AUC values are 0.590 and 0.886. The GBM model does seem to make some compromises, as it has a slightly higher number of incor-

rectly predictions regarding the presence of a child than the previous GLMNET models[21]. However, this is compensated by the correctly prediction of the presence of a child (see Figure 5.41a). From the variable importance, we observe that children programs are among the most important features (see Figure 5.41b). This can also be expected, as we are classifying whether a child is present. From the partial dependency plots, we also observe a trivial pattern, where the probability increases as more children programs are watched[22]. The existence of `NPrograms` in the top 10 most important features, suggests that children watch a lot of different programs or very few programs. Based on the summary statistics, we are more inclined to belief that children watch more programs.

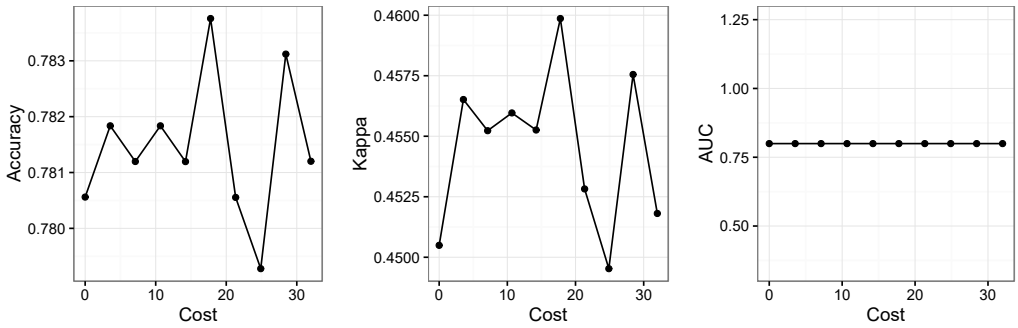


(a) Confusion matrix of the GBM.

(b) Variable importance

Support Vector Machines

Hyperparameter Search Space The hyperparameter search space for binary classification with SVM and a linear kernel is presented in Figure 5.42. The maximum cross-validation accuracy of 0.784 is achieved at a cost of $C = 17.78$. This also gives the highest cross-validation Kappa of 0.460. The AUC on the other hand does not seem to change with different cost values.



(a) Accuracy

(b) Kappa

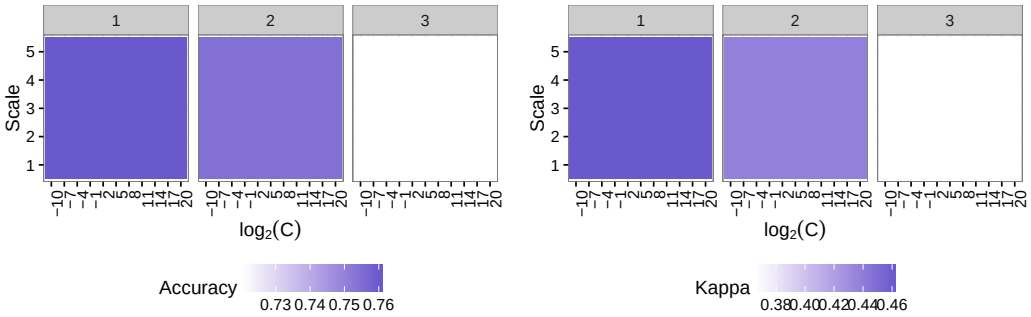
(c) AUC

Fig. 5.42.: Level plot for different cross-validation performance measures over different cost (C) values.

Using a polynomial kernel does not seem to improve the cross-validation performance metrics. As with the previous SVMs with a polynomial kernel, the majority of class probabilities

[21] Compared to all GLMNET models from the previous subsection.
[22] Due to the trivial nature, these are not included

could not be fitted and therefore, the AUC could not be computed. Based on the hyperparameter search space, it seems that increasing the polynomial degree to 3 worsen the cross-validation accuracy performance. Changing the scaling a or cost C also does not seem to have a large effect. This is also true for the cross-validation Kappa. The optimal hyperparameter values are a degree $d = 1$, scale $a = 1$ and cost $C = 2^{-10}$. This gives a cross-validation accuracy of 0.76 and a Kappa of 0.462.



(a) Accuracy **(b)** Kappa

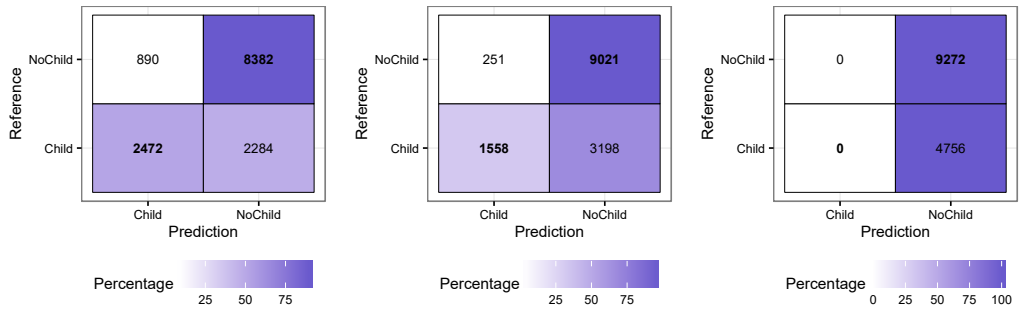
Fig. 5.43.: Level plot for cross-validation accuracy and Kappa values over different combinations of the degree (d), scale (a) and cost (C) hyperparameters.

Switching to the radial basis kernel does not improve the cross-validation performance measures. The cross-validation accuracy and Kappa are 0.66 and 0. These values are similar to the naive classifier. In addition, the hyperparameter search space is completely level. Changing the cost C and sigma σ do not show any changes in any performance measure.

Validation Validating the three different support vector machine models against the test set gives us the following confusion matrices (see Figure 5.44). The support vector machine model with the linear kernel achieved an accuracy of 0.774, with a 95% confidence interval between $(0.767, 0.781)$ and a Kappa value of 0.456 (see Figure 5.44a). The polynomial kernel performs similar, with an accuracy of 0.754 and a 95% confidence interval of $(0.747, 0.761)$ (see Figure 5.44b). This is most likely due to the fact that the polynomial kernel is actually a linear kernel, since the optimal degree is one. The Kappa value of 0.354 does seem to indicate that the linear kernel is much better than the polynomial kernel. The radial basis kernel performs the worst, as the accuracy is similar to the naive classification (see Figure 5.44c). It classifies all households as having no children present. The accuracy of the radial basis kernel is 0.661, with a 95% confidence interval of $(0.653, 0.669)$.

Algorithm Comparison

Based on the resampling distributions of the different algorithms, it can be shown that the stochastic gradient boosting (SGB) algorithm performs the best (see Figure 5.45). The average cross-validation accuracy of the SGB algorithm is higher than the regularized multinomial regression (GLMNET) and the support vector machines (SVML and SVMP). However, there are some cross-validation accuracies from the GLMNET algorithm that exceeds the average cross-validation accuracy of the SGB algorithm. The support vector machine with a polynomial kernel performs the worst on average, but when the maximum cross-validation



(a) Linear kernel **(b)** Polynomial kernel **(c)** Radial basis kernel

Fig. 5.44.: Confusion matrices of support vector machines with different kernels.

performance measures are considered, we observe that there are some values that exceed the maximum cross-validation measures of the support vector machine with a linear kernel. The largest variances are visible the SVMP algorithm, while the SVML algorithm has the smallest. The cross-validation Kappa shows the similar results, however the variances and the differences between the average cross-validation Kappas are larger.

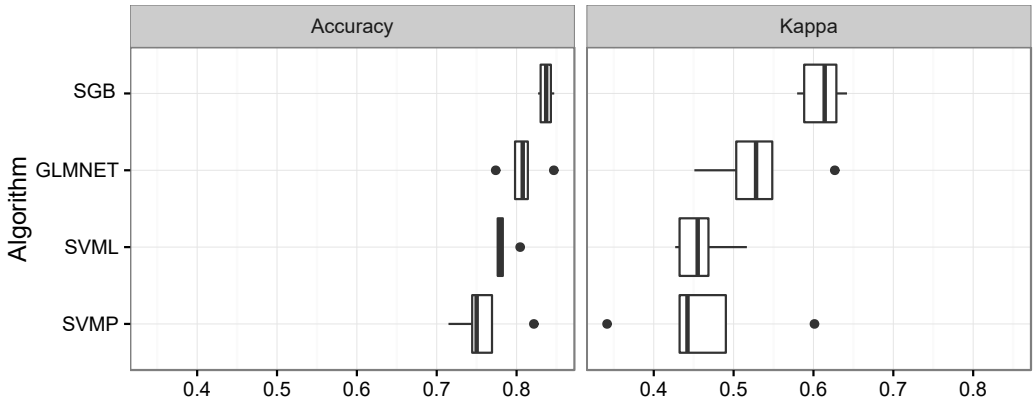


Fig. 5.45.: The resampling distributions with different performance measures and algorithms.

In order to determine whether the averages and medians are significantly different, we use a standard t-test and Wilcoxon signed rank test. The results of both tests for different cross-validation performance measures are presented in Table 5.20. We observe that the SGB algorithm has an average that is significantly different from the other algorithms under a significance level of 10%, irrespectively of the performance measure used. The accuracy averages between the other pairs do not appear to be significant, but using Kappa as the performance measure, we can conclude that GLMNET and SVML have a significant difference. The Wilcoxon signed rank test also shows that the medians of the SGB and SVML are significantly different in all performance measures under a significance level of 10%. The medians of the remaining pairs are not significantly different regardless of the performance measure used.

Tab. 5.20.: P-values for testing the differences between the means and medians of the cross-validation performance measures.

Pair		Accuracy		Kappa	
		T-test	Wilcoxon	T-test	Wilcoxon
SGB	GLMNET	0.095	0.125	0.073	0.125
SGB	SVML	0.000	0.063	0.063	0.063
SGB	SVMP	0.007	0.063	0.014	0.063
GLMNET	SVML	0.106	0.125	0.063	0.125
GLMNET	SVMP	0.140	0.188	0.312	0.313
SVML	SVMP	0.278	0.313	0.972	1.000

Overview

In this section we present a short overview of the models created in the first stage of our cascading classification model, which predict the presence of a child. We compare the performance among the models and also compare it with the first classification problem of Klein [47]. Note that we do not compare the model created by the support vector machine with a radial basis kernel as it predicts only families (`F`).

The GLMNET models have a relatively good performance in predicting the presence of a child. All models have an accuracy of varying from 0.797 to 0.804 (see Table 5.21). This is at least a 0.136 improvement over the naive classifier. We observe that the AUC GLMNET model makes 50% less errors in its prediction of the presence of a child than the other GLMNET models (see Figure 5.39). Only 7.84% of its "Child" predictions are wrong, while in the other models this error varies from 15.98% to 19.22%. The errors made in the predictions with respect to the absence of children however, show a slightly different picture. The Kappa GLMNET model performs the best, having only an error rate of 19.63%. The AUC GLMNET performs the worst, with an error rate of 23.71%. The accuracy levels of the GLMNET models are slightly worse than all classification methods discussed by Klein [47]. The differences vary from 0.067 to 0.102. It should be noted that the distribution in the dataset by Klein is different and thus has an accuracy of 0.724 using only a naive classifier [47].

The GBM has the best performance with respect to all performance measures (see Table 5.21). The confusion matrix of the GBM shows similarities with the Kappa GLMNET model, with only slightly better predictions (see Figure 5.39b and 5.41a). The accuracy of the GBM does not exceed the accuracy of any of the models created by Klein [47], but if we compare it using the differences between the accuracies between the model and the naive classifier, we see that the GBM is better than the classification tree and random forest with 2 000 trees from Klein [47].

The SVML model is one of the worst models with respect to the performance measures (see Table 5.21). The accuracy and Kappa are only 0.774 and 0.456. From the confusion matrix, we observe that it predicts the absence of children in households where children present (see Figure 5.44a). 2 284 households are classified incorrectly in this way. This is similar to the accuracy GLMNET model, which classifies 2 296 households incorrectly in the same manner (see Figure 5.39a). The difference lies in the classification of households without

children. The SVML model seems to be classifying 890 without children incorrectly, while the accuracy GLMNET model only classifies 468 households incorrectly. If we compare the SVML model with the models by Klein, we notice that the SVML model lacks in accuracy. The same conclusion is reached when comparing the differences between the naive classifier and the model.

The SVMP model is the worst performing model in predicting the presence of children. This can be seen in the accuracy and Kappa (see Table 5.21). The confusion matrix shows a decent performance in its prediction of the presence of children ("Child") (see Figure 5.44b). Only 13% of its predictions are wrong. The major issue is its inability to predict the absence of children correctly. 26.17% of its predictions of "NoChild" are wrong. Therefore, it can also be expected that the SVMP model is not better than the models created by Klein [47].

Tab. 5.21.: Performance measures for models predicting child presence.

Algorithm	Performance Measure	Accuracy	Confidence Interval		Kappa	AUC
			Min	Max		
Naive	-	0.661	0.653	0.669	0.000	0.500
GLMNET	accuracy	0.803	0.763	0.810	0.515	0.862
GLMNET	Kappa	0.804	0.798	0.811	0.527	0.851
GLMNET	AUC	0.797	0.780	0.793	0.449	0.883
SGB	-	0.827	0.821	0.833	0.590	0.886
SVML	-	0.774	0.767	0.781	0.456	N.A.
SVMP	-	0.754	0.747	0.761	0.354	N.A.

From the resampling distribution, we observe clearly that the SGB algorithm performs the best. Using the t-test and Wilcoxon signed rank test, we are able to show that the SGB algorithm performs significantly better than other algorithms. In addition, the SGB algorithm also is the only algorithm able to create a model that is comparable to the models created by Klein [47].

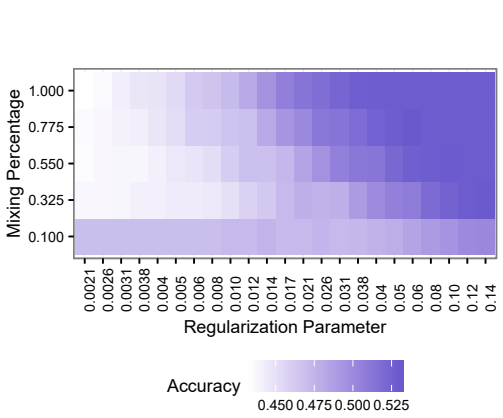
5.4.2 Second Stage Classification

For the second stage classification, we will augment the data set with information whether a child is present and use it to classify the household structures. The target variable becomes a variable with only four classes, namely: family (`F`), household (`H`), single female (`SF`) and single male (`SM`). During the training phase, information of the original target variable is used to extract the information regarding the presence of a child. Therefore, during the training and validation of the second stage, no errors are propagated from the first stage.

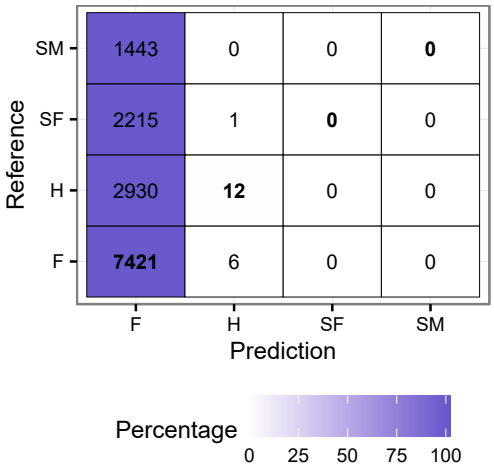
Multinomial Logistic Model

Hyperparameter Search Space The GLMNET algorithm performs a lot worse due to the increase in the number of classes. The majority class of the four household structures is family (`F`), which makes up 52.93% of the training sample. The highest cross-validation accuracy achieved was 0.530 with an $a = 0.775$ and a $\lambda = 0.0689$ (see Figure 5.46a). The

Kappa and AUC values at those values are 0.006 and 0.636. A notable observation from the hyperparameter search space is that using some hyperparameter values, the cross-validation accuracy is lower than the naive classifier. The cross-validation accuracy also seems to favour a high regularization parameter (λ) value, while in the first stage GLMNET model small regularization values are preferred. The cross-validation Kappa and AUC however, seem to favour the opposite, as low regularization parameter (λ) values are preferred (see Figure D.11b and D.11a). The maximum cross-validation Kappa is 0.138 and is achieved at $\alpha = 0.1$ and $\lambda = 0.010$. The cross-validation accuracy and AUC with these hyperparameter values are 0.473 and 0.646, respectively. The maximum cross-validation AUC is 0.667 and is reached at $\alpha = 0.1$ and $\lambda = 0.148$. The cross-validation accuracy and Kappa with those hyperparameters are 0.502 and 0.060.



(a) Level plot for cross-validation accuracy values over different combinations of the mixing (α) and regularization (λ) hyperparameters.



(b) Confusion matrix of the accuracy model.

Fig. 5.46.: Multinomial logistic model with accuracy as the performance measure.

Validation The Kappa and AUC models do not perform any better than the naive classifier, as both models only predicted families (`F`). These models are not further examined. The accuracy model only slightly outperforms the naive classifier, as it predicts some households (`H`) correctly (see Figure 5.46b). 13 households are classified as households (`H`), while all the other are classified as families (`F`). This gives an accuracy of 0.530, with a 95% confidence interval of $(0.522, 0.538)$. The Kappa and AUC of the accuracy model are 0.002 and 0.639.

Stochastic Gradient Boosting

Hyperparameter Search Space The GBM achieves a maximum cross-validation accuracy of 0.582 with $M = 1\,250$ boosting iterations, interaction depth $s = 20$ and a minimum observations per node $|R|_{min} = 10$ (see Figure 5.47). The cross-validation Kappa and AUC values are 0.197 and 0.764. The shrinkage hyperparameter and sampling size are fixed at $v = 0.001$ and $\eta = 0.5$. From the accuracy hyperparameter search space we observe that the GBM performs better than the naive classifier in the entire search space. Maximising the cross-validation Kappa shows a slightly different model, with $M = 1\,250$ boosting iterations,

interaction depth $s = 40$ and a minimum observations per node $|R|_{min} = 5$ (see Appendix D.2.2, Figure D.12a). The corresponding cross-validation accuracy, Kappa and AUC are 0.578, 0.197 and 0.759, respectively. This suggests that the Kappa model is very similar to the accuracy model. Maximising the cross-validation AUC gives almost the same hyperparameter values as the Kappa model, where only the number of minimum observations per node is $|R|_{min} = 10$ (see Appendix D.2.2, Figure D.12b). The accuracy, Kappa and AUC with these hyperparameter values are 0.576, 0.192 and 0.768, respectively. This suggests a slightly worse performance compared to the accuracy and Kappa model.

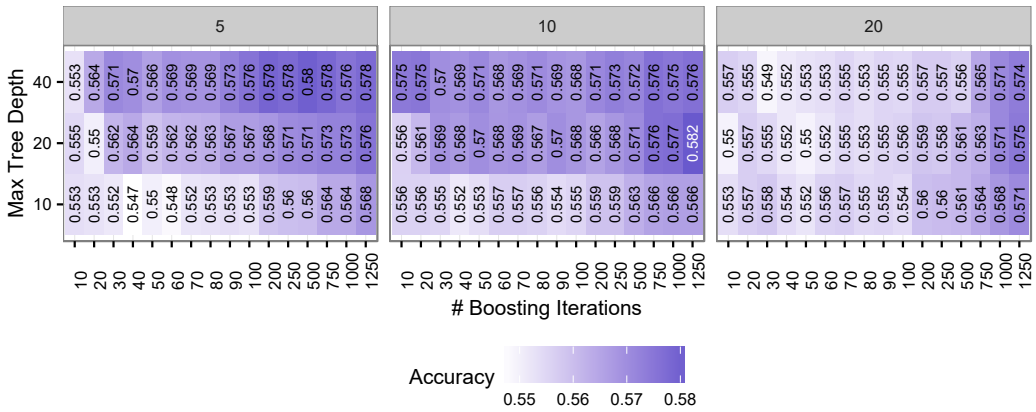
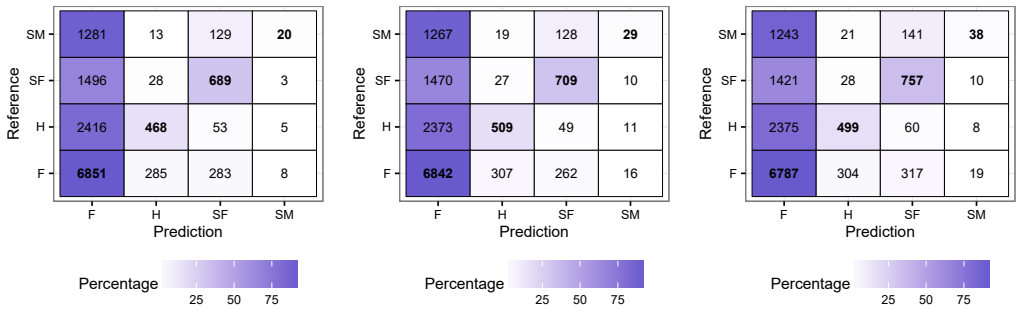


Fig. 5.47.: Level plot for accuracy over different values of boosting iterations (M), interaction depths (s) and minimum observations per node ($|R|_{min}$). The shrinkage (v) and sampling fraction (η) stay constant at $v = 0.001$ and $\eta = 0.5$.

Validation The accuracy model achieves an accuracy of 0.572, with a 95% confidence interval between $(0.564, 0.581)$ (see Figure 5.48a). The Kappa and AUC values are 0.178 and 0.768. From the confusion matrix, we observe that this model is the best in classifying families (`F`). The Kappa model has an accuracy of 0.577, with a 95% confidence interval between $(0.568, 0.585)$ (see Figure 5.48b). The Kappa and AUC values are 0.190 and 0.762. This is also the model with the highest accuracy, even though it performs the worst in classifying families (`F`). The Kappa model seems to be able to compensate it, by a better classification of households (`H`) and single female (`SF`). The AUC model performs slightly worse and has similar performance measures as during cross-validation. The accuracy of the AUC model is 0.576, with a 95% confidence interval between $(0.568, 0.584)$ (see Figure 5.48c). The Kappa and AUC are 0.194 and 0.768. The AUC value from the AUC model is equal to that of the accuracy model. Surprisingly the Kappa value of the AUC model is the highest, although it does not differ a lot. Comparing the predictions between the GBM models, we observe that the accuracy and AUC model have a Cramer's V of 0.840. This shows that their predictions have a large association and thus are very similar. The remaining Cramer's V between the accuracy and Kappa model is 0.772. This is very similar to the Cramer's V between the AUC and Kappa model, which is 0.775. All Cramer's V values seem to indicate that all models do not differ significantly.

Model Properties In this GBM, we deal with multiple classes and the effect of features are not trivial. Therefore, we opt to conduct a more detailed analysis on the effect of the



(a) Accuracy **(b)** Kappa **(c)** AUC

Fig. 5.48.: Confusion matrices of GBM models on the test set.

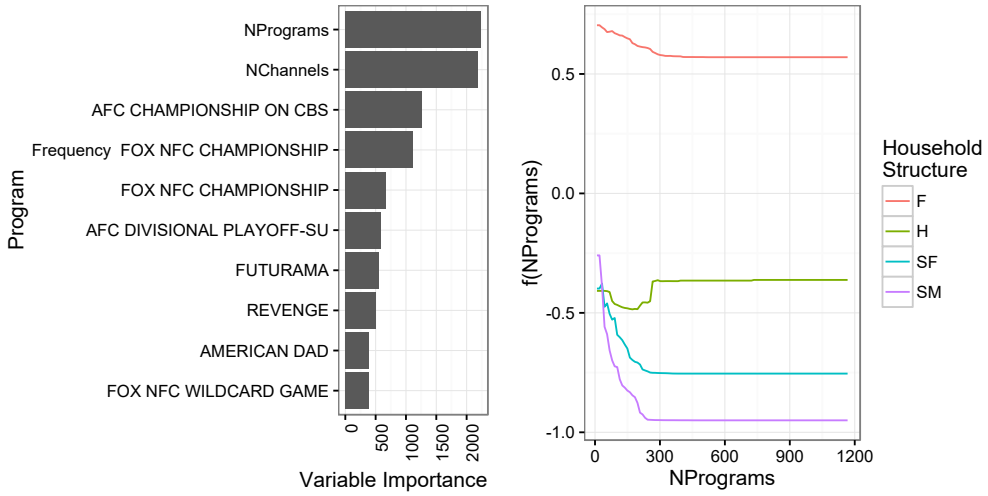
important features of this model. We choose to analyse the Kappa GBM, as it predicts a more diverse range of classes[23].

A more detailed analysis of the Kappa GBM shows us that the most important variables are `NPrograms`, `NChannels` and `AFC CHAMPIONSHIP ON CBS` (see Figure 5.49a). It is also evident that the children present variable is not one of the most important variables, although it is used in the model. From the partial dependency plot, we observe that when the number of programs (`NPrograms`) increases, the probability that the household is a single adult (`SF` and `SM`) decreases rapidly. This can also be seen in the probabilities that the household is a family (`F`) or household (`H`), although the decrease is less (see Figure 5.49b). Roughly the same effect can be seen in the number of channels (`NChannels`) (see Appendix D.2.2, Figure D.13a). As the numbers of channels increase, the probability that a household is a single adult (`SF` and `SM`) becomes extremely low, while the probability that a household is a family (`F`) or household (`H`) increases slightly. An increase in the duration of `AFC CHAMPIONSHIP ON CBS` is accompanied with a large increase in the probability that the household is a household (`H`) (see Appendix D.2.2, Figure D.13b). The effect of the duration in `AFC CHAMPIONSHIP ON CBS` on the other household structures is small. The child feature, there does not seem to be a large effect in general (see Appendix D.2.2, Figure D.14a). We do observe that if a child is present (which is indicated by a value of 1), the probability of the household being classified as a family (`F`) or single female (`SF`) drops slightly. For the most important animated program, `FUTURAMA`, we observe that if the total duration of watching `FUTURAMA` increases, the effect of the household being a household (`H`) increases (see Appendix D.2.2, Figure D.14b).

Support Vector Machines

Hyperparameter Search Space The optimal hyperparameter value for the support vector machine with a linear kernel is a cost of $C = 7.111$. This gives us the following cross-validation accuracy, Kappa and AUC of 0.531, 0.0176 and 0.662 respectively. These are also the maximum values for every cross-validation performance measure (see Figure 5.50). Switching to a polynomial kernel or radial basis kernel does not improve the cross-validation accuracy and leaves us with a model that only classifies families (`F`), which is the same as

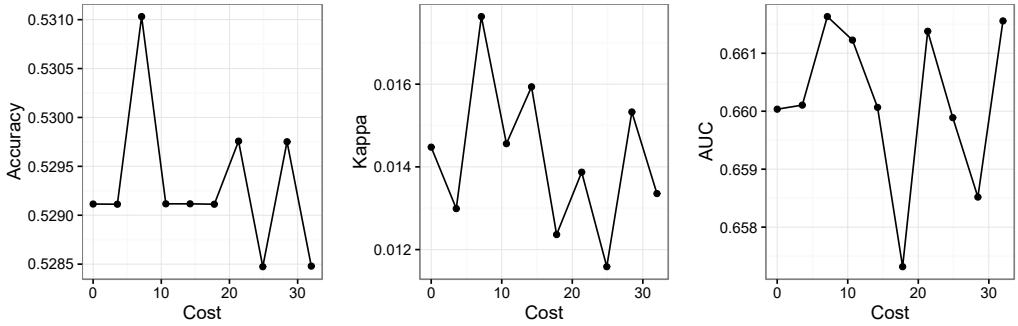
[23] It should be noted that top 10 most important features of the three models only differ in one or two features.



(a) Variable importance **(b)** NPrograms

Fig. 5.49.: Variable importance and partial dependency plot of the most important variable of the Kappa GBM model.

the naive classifier. Using a different hyperparameter search space does not improve the performances as the entire surface for both models are flat.



(a) Accuracy **(b)** Kappa **(c)** AUC

Fig. 5.50.: Level plot for cross-validation accuracy, Kappa and AUC values over different cost (C) hyperparameter.

Validation The validation shows rather disappointing results. The SVML model produces the confusion matrix in Figure (see Figure 5.51). This gives an accuracy of 0.5237, with a 95% confidence interval of $(0.5154, 0.532)$. This is slightly lower than a naive classifier. 7201 out of the 7427 families (`F`) are correctly classified, however this is not compensated by sufficiently correct classifications in other classes. The Kappa of the SVML model is 0.018. The SVMP and SVMR models perform slightly better, but they only classify every household as a family (`F`). Therefore, no confusion matrices of those models are presented.

Algorithm Comparison

From the resampling distributions, we observe that based on the cross-validation accuracy, the algorithms do not differ very much (see Figure 5.52). The SVML and GLMNET produce a similar cross-validation accuracy distribution with a very small variance. The SGB

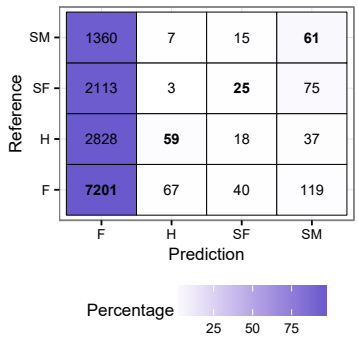


Fig. 5.51.: Confusion matrix of the SVM model with a linear kernel.

algorithm has the largest variance, but produces higher cross-validation accuracy values. From the cross-validation Kappa values, we are able to observe a larger difference between the algorithms. The SVML and GLMNET still are very similar, but the SGB algorithm performs much better. The variances of the cross-validation Kappa are also slightly higher than the variances of the cross-validation accuracy.

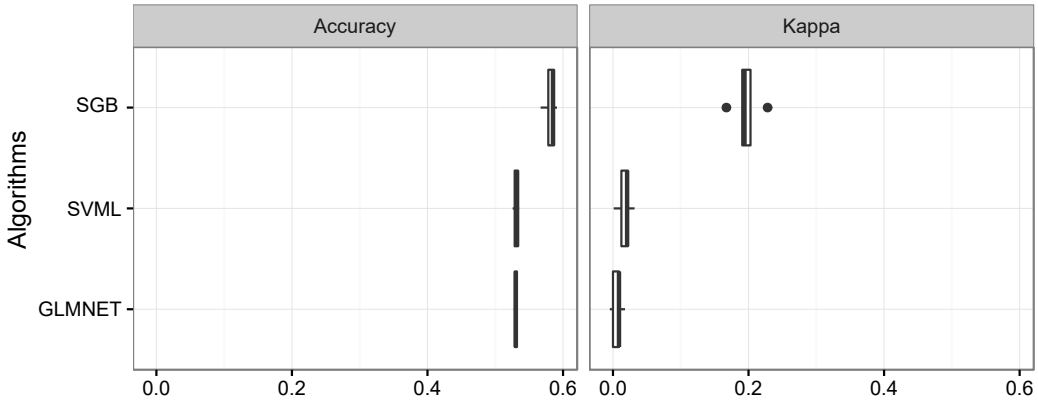


Fig. 5.52.: The resampling distributions with different performance measures and algorithms.

To determine whether the averages and medians are significantly different, we use a t-test and a Wilcoxon signed rank test. From the t-test we observe that the average cross-validation accuracy between the SGB and the other algorithms are significantly different under a significance level of less than 1% (see Table 5.22). The Wilcoxon signed rank test shows that the median cross-validation accuracies between SGB and the rest of the algorithms are also significantly difference, albeit under a significance level of 10%. The same can also be seen in the average and median cross-validation Kappa values. GLMNET and SVML do not appear to be significantly different at all.

Overview

In this section, we also present a short overview of all the models created in the second stage of our cascading classification model, which predict the household structure. Unfortunately, there is no mapping possible from our household structures to the household structures

Tab. 5.22.: P-values for testing the differences between the means and medians of the cross-validation performance measures.

Pair		Accuracy		Kappa	
		T-test	Wilcoxon	T-test	Wilcoxon
SGB	GLMNET	0.0003	0.0625	$9.82 \cdot 10^{-5}$	0.0625
SGB	SVML	0.0004	0.0625	$3.30 \cdot 10^{-5}$	0.0625
GLMNET	SVML	0.7965	0.8125	0.2277	0.3125

used in the second classification problem by Klein [47]. This is due to the fact that we do not include children in the target variable, whereas Klein has the household structure adult family with children. In this overview we will also omit the SVMP and SVMR models as their predictions are exactly the same as those of the naive classifier.

The GLMNET model does not seem to perform satisfactory. The accuracy is similar to the naive classifier and from the confusion matrix, we also observe that almost every household is classified as a family (`F`) (see Figure 5.46b).

The GBMs are again one of the best performing models (see Table 5.23). It does seem to differ slightly which performance measure is used to select the parameters, but they all have an accuracy that exceeds 0.572. The confusion matrices show only small differences in the predictions in the GBMs (see Figure 5.48). All models seem to be classifying the majority of households as families (`F`), even though they are not. Only about 57% of the family (`F`) predictions are correct. From the variable importance, we observe that `NPrograms` and `NChannels` are important features in distinguishing between household structures, which shows that our initial reasoning is correct (see Figure 5.49a).

The support vector machines do not seem to perform very well on multinomial classification problems. The SVML model performs slightly worse than the naive classifier (see Table 5.23). From the confusion matrix, we observe that the SVML model does predict other classes than families (`F`), however it fails to make sufficient correct predictions to compensate for the incorrect ones (see Figure 5.51).

Tab. 5.23.: Performance measures for models predicting household structures.

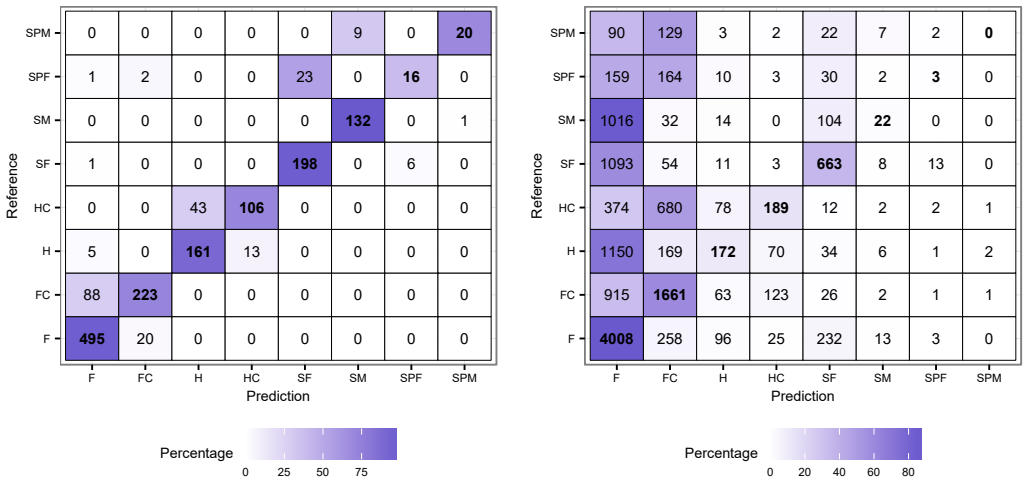
Algorithm	Performance Measure	Accuracy	Confidence Interval		Kappa	AUC
			Min	Max		
Naive	-	0.529	0.521	0.538	0.000	0.500
GLMNET	accuracy	0.530	0.522	0.538	0.002	0.639
SGB	accuracy	0.572	0.564	0.581	0.178	0.768
SGB	Kappa	0.577	0.568	0.585	0.190	0.762
SGB	AUC	0.576	0.568	0.584	0.194	0.768
SVML	-	0.524	0.515	0.532	0.018	N.A.

The resampling distributions shows that the SGB algorithm outperforms all the other algorithms. The difference between averages and medians cross-validation performance measures of SGB with the other algorithms are all significant. Therefore, we can conclude that the SGB algorithm performs the best in classifying household structures.

5.4.3 Combining Predictions

Validation Since the SGB algorithms performs the best in both stages, we will use two gradient boosting models to predict the presence of a child and the household structure. We will be using the accuracy GBM to predict the presence of children and the Kappa GBM to predict the household structure. The choices are made primarily based on their validation results as both models predict a broad range of classes, instead of focussing on predicting the majority group. These predictions are then combined, which results in two models predicting a total of eight classes, similar to the previous sections.

We first use our cascading model to make predictions on the entire training set to observe the performance of our cascading model in the training data. This gives us an accuracy of 0.864, with a 95% confidence interval of $(0.846, 0.881)$ and a Kappa of 0.830 (see Figure 5.53a). Both performance measures suggest that the cascading model performs good. We are unable to compute the AUC values as there is no standard way to combine both class probabilities. Unfortunately, this is most likely to be an overestimation of the performance, as 80% of the data is used to select the hyperparameters in both models and the entire data set is used in the construction of the final models. From the training set, we observe that the model has some problems in determining whether a child is present. For example, 495 families (`F`) are classified correctly, but 88 of the predicted families (`F`) are actually families with children (`FC`). The same can also be seen in households (`H`), where 43 of the predicted households (`H`) are actually households with children (`HC`). Validating the model against the test set shows a different result. The accuracy is only 0.479 with a 95% confidence interval between $(0.471, 0.487)$ (see Figure 5.53b). In the confusion matrix, we observe the same pattern that the model seems to struggle in distinguishing families (`F`) and other household structures. The cascading model also has some problems with determining whether a child is present, although this is less visible in rare household structures (i.e. household structures that have a low number of observations). In addition, we also observe that the model is unable to correctly classify single male parents (`SPM`).



(a) Training set **(b)** Test set

Fig. 5.53.: Confusion matrix of the predictions of the two GBM models on different data sets.

Incorrectly Classified Households From the average durations and frequencies of correctly and incorrectly classified households, we observe the same pattern as before. All correctly classified household structures have an average duration higher than their incorrectly classified counterparts, with the exception of families (`F`) and single females (`SF`) (see Table 5.24). The average duration of households with children (`HC`) is more than twice as much as its incorrectly classified counterpart. This is similar what is observed in the GBM and AUC SVML base models. The average frequencies show a slightly different pattern. The average frequencies of the correctly classified households are higher, with the exception of families (`F`) and single adult families (`SF`, `SM` and `SPF`). For single male parents (`SPM`) no averages could be computed for the correctly classified households as no households were correctly classified.

Tab. 5.24.: Average duration and frequencies of correctly and incorrectly classified households by the cascading model.

Household	Duration		Frequency	
	Correct	Incorrect	Correct	Incorrect
F	14 206.4	17 597.8	862.91	1 025.8
FC	20 766.5	14 664.6	1 261.7	862.3
H	30 234.5	19 131.1	1 880.2	1 089.3
HC	46 274.9	22 607.3	3 011.8	1 282.7
SF	9 571.4	10 786.4	422.7	604.9
SM	15 106.0	9 182.4	624.2	662.2
SPF	15 616.8	12 933.0	545	603.9
SPM		7 711.8		419.7

6 Discussion

In this thesis, we have seen three different machine learning algorithms tacking the difficult classification problem of classifying households into eight different household structures. The stochastic gradient boosting algorithm creates clearly better models than the other algorithms. The base GBM has an accuracy of 0.488, but in predicting the correct class, it is no better than a coin toss (i.e. 48.8% of its predictions are correct.). In terms of application, this accuracy level might be insufficient. Therefore, we propose two different ways in which the models might be applied.

First, we can compute the positive predictive value (PPV), which is simply the proportion of correctly classified households of class c among all households that are predicted in class c. In other words, given a certain prediction, it is the probability that the prediction is correct. Since this information can be computed from the confusion matrices presented in Chapter 5, we only present the maximum PPV for each class and each set of models.

Among the base models, we already established that the GBM performs the best. This is also reflected in the PPV, as the GBM has the highest PPV for the majority of household structures (see Table 6.1). The highest overall PPV comes from the accuracy GLMNET model, which is 1 for single males (`SM`), however it should be taken into account that only one out of 14 028 households is classified as such in the test set. The second highest PPV for single males (`SM`) comes from the GBM and is 0.335.

Tab. 6.1.: Maximum PPVs among the base models.

Algorithm	Performance Measure	Household Structure	Maximum PPV
GBM	-	F	0.474
GLMNET	accuracy	FC	0.611
GBM	-	H	0.377
GLMNET	accuracy	HC	0.433
GBM	-	SF	0.515
GLMNET	accuracy	SM	1.000
GBM	-	SPF	0.364
GBM	-	SPM	0.500

The dimension reduced models fare slightly worse than the base models. The maximum PPV of the AUC GLMNET dimension reduced models is 0.577, which is for the class family with children (`FC`) (see Table 6.2a). The AUC GLMNET dimension reduced accuracy and Kappa SVML models have the best predictions in 6 classes. However, the majority of the AUC GLMNET dimension reduced PPVs are still lower than those of the GBM dimension reduced PVVs, except for families (`F` and `FC`) (see Table 6.2b).

Tab. 6.2.: Maximum PPVs among the dimension reduced models.

(a) AUC GLMNET dimension reduced models

Algorithm	Performance Measure	Household Structure	Maximum PPV
SVML	Kappa	F	0.456
SVMP	Kappa	FC	0.577
SVML	accuracy	H	0.237
SVML	accuracy	HC	0.335
SVML	accuracy	SF	0.365
SVML	Kappa	SM	0.220
SVMR	-	SPF	0.071
SVML	Kappa	SPM	0.049

(b) GBM dimension reduced models

Algorithm	Performance Measure	Household Structure	Maximum PPV
SVMP	-	F	0.452
SVML	accuracy	FC	0.568
SVMP	-	H	0.277
SVMP	-	HC	0.336
SVML	accuracy	SF	0.422
SVMP	-	SM	0.265
SVML	accuracy	SPF	0.190
SVMR	-	SPM	0.107

We proceed to consider the positive predictive values of the models created in the first and second stage of the cascading classification model. The maximum PPVs from the first stage in the cascading classification model are much higher. The AUC GLMNET model is able to correctly predict the presence of a child in 92% of its "Child" predictions (see Table 6.3a). The GBM has a PPV of 82.8% for "NoChild". This allows us to determine with a high certainty whether a household contains child. The second stage has a slightly lower maximum PPV, but compared to the base and dimension reduced models, though it does perform better in predicting all classes (see Table 6.3b). This can be seen for example in single males (`SM`), where 55.6% of its single male (`SM`) predictions are correct. It should be noted that these models are not directly comparable to the base and dimension reduced models as the models from the second stage only classify four classes. Therefore, we also show the PPVs from the cascading classification model.

Tab. 6.3.: Maximum PPVs in the first and second stages of the cascading classification model.

(a) First stage

Algorithm	Performance Measure	Household Structure	Maximum PPV
GLMNET	AUC	Child	0.922
GBM	-	NoChild	0.828

(b) Second stage

Algorithm	Performance Measure	Household Structure	Maximum PPV
GBM	AUC	F	0.574
GLMNET	-	H	0.632
GBM	Kappa	SF	0.618
GBM	accuracy	SM	0.556

We observe that the cascading classification model is better at predicting households (`H`), households with children (`HC`), single females (`SF`) and males (`SM`) (see Table 6.4). For example, 59% of its single females (`SF`) predictions are correct.

Tab. 6.4.: Positive predictive values of the cascading classification model.

Household Structure	PPV
F	0.455
FC	0.528
H	0.385
HC	0.455

Household Structure	PPV
SF	0.590
SM	0.355
SPF	0.120
SPM	0.000

A second way, in which these models might be applied is by reducing the number of classes. This can be done by a simple mapping, which improves the accuracy (as seen in the comparison with Klein [47]). For example, the advertisement of food items targeted at single

households could benefit by combining the classes single females (SF) and males (SM), as single households are increasing [70].

6.1 Limitations

In this thesis, we have seen a broad range of techniques and algorithms that are applied on predicting household compositions, but there still remain several limitations in this thesis that needs to be overcome.

The main limitation in this thesis, which in our opinion severely affected the attainable maximum accuracy is the usage of only 10% of the households in January 2014 as the training set. There are several solutions to this problem, which are also shortly examined, but not extensively researched in this thesis. One simple solution is by not using multiple cores, this allows us to increase the training set to roughly 40%, but this would also increase the computational time. Given the unknown search space, the decision was made to use only 10% of the data instead of 40% without parallelization. In addition, the extra computational time that is needed quadruples without parallelization. This is in our opinion unacceptable, given that the stochastic gradient boosting model already takes 2 weeks to run on one machine. Another solution would be using packages in R to handle the big data[1]. The disadvantage of this is that it uses the hard drive to store the data, which is extremely slow, which also increases the computational time. The third solution would be using dimension reduction methods to reduce the number of features. There are several ways to accomplish this. First, we can use the features that contain 95% of the data, which reduces the number of features enormously and speeds up the computational requirements. However, this does not solve the memory issue, since only 5% of the data is dropped and the zero entries do not contribute to the memory usage due to the use of sparse matrix representations. A second method would be using principal component analysis to reduce the feature set. The problem with this method is that it increases the computational time significantly, as the principal component analysis step should be done for each hyperparameter set and for each resampling iteration. In addition, this also prevents us from using a larger training set, as it is impossible to do principal component analysis with large sparse matrices. There are techniques for computing the first few principal components[2], but these are only approximations and they do not allow us to compute the variance explained by these principal components. Furthermore, exploratory research done in this area shows the problem of including noise into the data. This resulted in a drop in accuracy in the range of $0.05 - 0.2$ compared to the base models. The third method, which is used in this thesis is simply by using the features selection and use those features in the subsequent models.

Another major limitation of this thesis is the selection of important features in a regularized multinomial logistic model. We used the absolute coefficients corresponding to the features as a proxy to measure feature importance. Though this method does measure the largest effect, it does not take the lack of fit into consideration. In addition, we have not studied the stability of the model. In order to study the stability of the regularized multinomial

[1] For example: `bigmemory`, see `https://cran.r-project.org/web/packages/bigmemory/`.
[2] For example using implicitly restarted Lanczos bi-diagonalization algorithm, see `https://cran.r-project.org/web/packages/irlba/irlba.pdf`.

logistic model, one needs to train the model over different data sets and keep track of the selected features [58]. If our model is not stable, different data sets could end up with a totally different set of features. Unfortunately, analysing the stability of our models is computationally very expensive in our case and therefore has been excluded in this thesis.

7 Conclusion

In this thesis we examined whether it was possible to predict household composition using granular television viewing data. We defined household compositions into eight classes and we achieved a maximum test accuracy of 0.488 using the stochastic gradient boosting model, which is 0.158 better than the naive classifier. In addition, we also examined the multinomial logistic model with regularization and support vector machines with different kernels. The multinomial logistic models with regularization achieved a maximum accuracy of 0.412. The support vector machine model works best using a linear kernel and also achieved a comparable accuracy of 0.415. Using statistical tests, we come to the conclusion that the stochastic gradient boosting model is significantly different from the other models. This suggests that the stochastic gradient boosting model is the best.

We observed that the support vector machines with polynomial and radial basis kernel did not perform satisfactory, as their results were too similar to the naive classifier. Therefore, we used the multinomial logistic model with regularization and stochastic gradient boosting for feature selection prior to training the support vector machines. In addition, we also added an additional hyperparameter, which can remove less important features. The AUC GLMNET dimension reduction improved the accuracy of the radial basis kernel, while it worsened the accuracy of the other two kernels. The GBM dimension reduction improved the accuracy for all kernels, up to a maximum accuracy of 0.431.

The cascading classification model in its whole did not result in a better model than the base GBM. However, it did provide several insights in different aspects of our target variable. In the first stage, most models did not seem to have any problems with predicting the presence of a child. The second stage was more challenging, as the accuracy dropped severely compared to the first stage. In addition, we managed to provide empirical evidence that `NPrograms` and `NChannels` can be effectively used to predict household composition.

Comparing our results with Klein his research, showed that our models have comparable accuracies, although the models presented in this thesis seem to do slightly worse [47]. However, taken into account that our training set is much smaller and the distribution of our classes are different, some of the models presented in this thesis can be considered an improvement over the models from Klein. In addition, the GBM presented in this thesis uses less iterations than the number of trees in Klein his random forest.

As to answer our research question, we believe that it is possible to predict household composition using granular television viewing data, though memory usage and computational requirements are challenging and additional research is needed.

7.1 Future Research

In this section, we present several ways in which this research can be improved upon or extended.

One way this thesis can be extended upon is by using feature hashing to reduce the size of the data. It works by applying a hash function to the features and using their hash values as indices directly, rather than looking up in an associative array [83]. It should be noted that there are not a lot of common duration values among programs that are frequently watched, however for many programs that are infrequently watched, there are many common duration values. Thus it could lead to even smaller sparse matrices. In conjunction, QR decomposition can also be used to eliminate linear combinations of features if they exist, but it requires a significant amount of memory that was not available for this thesis [44].

Reduction in noise in the data might be possible since the viewing data can be viewed as an analogy of an image, noise reduction techniques in image recognition can also be used for viewing data. One method for example is by using robust recovery of subspace structures by low-rank representation [56] or robust principal component analysis [55]. Unfortunately, the analogy breaks down when determining the parameters for these methods, as in an image. The quality of noise reduction in an image can be observed, while noise reduction in the viewing data cannot be easily observed. But if viewing behaviour of visitors are included in the data (which are explicitly excluded in this research to narrow the scope), noise reduction techniques might be able to differentiate between the viewing behaviour of the household and visitors. The processed features can then be used in more complicated machine learning algorithms, such as convolutional neural networks to capture the complicated patters in the viewing data [52].

Due to the large number of basic features that are used in this data, more advanced features or observations are not included. These can be included in subsequent studies, provided with enough memory and computational time. For example, in Chapter 4, viewing behaviours of visitors have been omitted to reduce the scope of this thesis. These observations could be used to create visitor features. More complicated features, can also be created using feature extraction algorithms, such as an autoencoder [79].

Lastly, in the near future, complex models can be run in the open source Spark using H2O[1]. It provides a much better environment to store large data sets and provides a distributed framework. The only disadvantage is that during the writing of this thesis, only a limited number of machine learning algorithms are supported and in-depth analysis of the models are limited. Furthermore, there are limited tools for visualising the data and the model itself and the models are not compatible with R yet. It does however provide a neat interface to R in which several functions of H2O can be called upon.

[1] See `http://www.h2o.ai/`.

Bibliography

8

[1] *25 Weirdly Specific Facebook Ads Targeting Ideas You Didnt Know Existed*. `http://connectio.io/25-facebook-ads-targeting-ideas/`. Accessed: 2016-01-01 (cit. on p. 2).

[2] E. Alpaydin. *Introduction to Machine Learning*. 2nd ed. Massachusetts, 2009 (cit. on p. 2).

[3] Pamela L Alreck and Robert B Settle. *The survey research handbook*. McGraw-Hill, 1994 (cit. on p. 127).

[4] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. „A training algorithm for optimal margin classifiers“. In: *Proceedings of the fifth annual workshop on Computational learning theory*. ACM. 1992, pp. 144–152 (cit. on p. 17).

[5] Leo Breiman. „Bagging predictors“. In: *Machine learning* 24.2 (1996), pp. 123–140 (cit. on p. 15).

[6] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984 (cit. on pp. 16, 121, 122).

[7] M.J. Carey, G.D. Tattersall, H. Lloyd-Thomas, and M.J. Russell. „Inferring identity from user behaviour“. In: *IEE Proceedings - Vision, Image, and Signal Processing* 150.6 (2003), p. 383 (cit. on p. 3).

[8] Chih-Chung Chang and Chih-Jen Lin. „LIBSVM: A library for support vector machines“. In: *ACM Transactions on Intelligent Systems and Technology* 2 (3 2011). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`, pp. 1–27 (cit. on p. 68).

[9] K. Chang, J. Hightower, and B. Kveton. „Inferring identity using accelerometers in television remote controls“. In: *Pervasive Computing*. Springer, 2009, pp. 151–167 (cit. on p. 3).

[10] S. Derksen and H. J. Keselman. „Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables“. In: *British Journal of Mathematical and Statistical Psychology* 45.2 (1992), pp. 265–282 (cit. on p. 7).

[11] N. R. Draper and R. Craig van Nostrand. „Ridge Regression and James-Stein Estimation: Review and Comments“. English. In: *Technometrics* 21.4 (1979), pp. 451–466.

[12] N. R. Draper and H. Smith. *Applied regression analysis*. John Wiley and Sons, 2014 (cit. on p. 8).

[13] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. „Least angle regression“. In: *The Annals of statistics* 32.2 (2004), pp. 407–499 (cit. on p. 10).

[14] James P Egan. „Signal detection theory and ROC analysis“. In: (1975) (cit. on p. 24).

[15] eMarketer. *US TV Ad Market Still Growing More than Digital Video*. June 2014. URL: `http://www.emarketer.com/Article/US-TV-Ad-Market-Still-Growing-More-than-Digital-Video/1010923` (visited on Dec. 12, 2014) (cit. on p. 1).

[16] Manuel JA Eugster and Friedrich Leisch. „Exploratory analysis of benchmark experiments an interactive approach“. In: *Computational Statistics* 26.4 (2011), pp. 699–710 (cit. on p. 25).

[17] Jianqing Fan and Runze Li. „Variable selection via nonconcave penalized likelihood and its oracle properties“. In: *Journal of the American statistical Association* 96.456 (2001), pp. 1348–1360 (cit. on p. 11).

[18] Tom Fawcett. „An introduction to ROC analysis“. In: *Pattern recognition letters* 27.8 (2006), pp. 861–874.

[19] R. A. Fisher. „On the mathematical foundations of theoretical statistics“. In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* (1922), pp. 309–368 (cit. on p. 124).

[20] Joseph L Fleiss. „The measurement of interrater agreement“. In: *Statistical methods for rates and proportions* 2 (1981), pp. 212–236 (cit. on p. 24).

[21] Jerome H Friedman. „Greedy function approximation: a gradient boosting machine“. In: *Annals of statistics* (2001), pp. 1189–1232 (cit. on pp. 13–15, 58).

[22] Jerome H Friedman. „Stochastic gradient boosting“. In: *Computational Statistics & Data Analysis* 38.4 (2002), pp. 367–378 (cit. on p. 58).

[23] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. „Regularization paths for generalized linear models via coordinate descent“. In: *Journal of statistical software* 33.1 (2010), p. 1 (cit. on p. 12).

[24] Michael Friendly. „The Generalized Ridge Trace Plot: Visualizing Bias and Precision“. In: *Journal of Computational and Graphical Statistics* (2012). In press; Accepted, 2/1/2012 (cit. on p. 9).

[25] D. J. Hand and W. E. Henley. „Statistical Classification Methods in Consumer Credit Scoring: a Review“. In: *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 160.3 (Sept. 1997), pp. 523–541 (cit. on p. 2).

[26] David J Hand and Robert J Till. „A simple generalisation of the area under the ROC curve for multiple class classification problems“. In: *Machine learning* 45.2 (2001), pp. 171–186.

[27] James A Hanley and Barbara J McNeil. „The meaning and use of the area under a receiver operating characteristic (ROC) curve.“ In: *Radiology* 143.1 (1982), pp. 29–36 (cit. on p. 25).

[28] F. E. Harrell. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. Graduate Texts in Mathematics. Springer, 2001 (cit. on p. 7).

[29] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. 2nd ed. 1. Springer, 2009 (cit. on pp. 5, 8, 15–18, 23, 121).

[30] A. E. Hoerl and R. W Kennard. „Ridge regression: Biased estimation for nonorthogonal problems“. In: *Technometrics* 12.1 (1970), pp. 55–67 (cit. on p. 8).

[31] Arthur E Hoerl, Robert W Kannard, and Kent F Baldwin. „Ridge regression: some simulations“. In: *Communications in Statistics-Theory and Methods* 4.2 (1975), pp. 105–123 (cit. on p. 9).

[32] Arthur E Hoerl and Robert W Kennard. „Ridge regression: applications to nonorthogonal problems“. In: *Technometrics* 12.1 (1970), pp. 69–82.

[33] Arthur E Hoerl and Robert W Kennard. „Ridge regression iterative estimation of the biasing parameter“. In: *Communications in Statistics-Theory and Methods* 5.1 (1976), pp. 77–88 (cit. on p. 9).

[34] Torsten Hothorn, Friedrich Leisch, Achim Zeileis, and Kurt Hornik. „The design and analysis of benchmark experiments“. In: *Journal of Computational and Graphical Statistics* 14.3 (2005), pp. 675–699 (cit. on pp. 25, 26).

[35] *Household composition statistics*. `http://ec.europa.eu/eurostat/statistics-explained/index.php/Household_composition_statistics`. Accessed: 2016-04-04 (cit. on p. 28).

[36] Chih-Wei Hsu and Chih-Jen Lin. „A comparison of methods for multiclass support vector machines“. In: *Neural Networks, IEEE Transactions on* 13.2 (2002), pp. 415–425 (cit. on p. 21).

[37] Shuai Huang, Jing Li, Liang Sun, et al. „Advances in Neural Information Processing Systems“. In: 2009, pp. 808–816.

[38] Mohammed F Hussain, Russel R Barton, and Sanjay B Joshi. „Metamodeling: radial basis functions, versus polynomials“. In: *European Journal of Operational Research* 138.1 (2002), pp. 142–154 (cit. on p. 23).

[39] E. L. James and I. Cunningham. „A profile of direct marketing television shoppers“. In: *Journal of Direct Marketing* 1.4 (1987), pp. 12–23 (cit. on p. 3).

[40] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning*. Springer, 2013 (cit. on pp. 7, 10).

[41] Wenxin Jiang. „On weak base hypotheses and their implications for boosting regression and classification“. In: *Annals of statistics* (2002), pp. 51–73 (cit. on p. 14).

[42] Thorsten Joachims. *Text categorization with support vector machines: Learning with many relevant features*. Springer, 1998 (cit. on p. 23).

[43] Alexandros Karatzoglou, Alex Smola, Kurt Hornik, and Achim Zeileis. „kernlab-an S4 package for kernel methods in R“. In: (2004) (cit. on p. 22).

[44] Hyunsoo Kim, Peg Howland, and Haesun Park. „Dimension reduction in text classification with support vector machines“. In: *Journal of Machine Learning Research*. 2005, pp. 37–53 (cit. on p. 114).

[45] B. Kitts, D. Au, and B. Burdick. „A High-Dimensional Set Top Box Ad Targeting Algorithm including Experimental Comparisons to Traditional TV Algorithms“. In: *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE. 2013, pp. 370–378.

[46] B. Kitts, L. Wei, D. Au, et al. „Targeting Television Audiences using Demographic Similarity“. In: *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*. IEEE. 2010, pp. 1391–1399.

[47] F. Klein. „Empirical study of household TV viewing behaviour“. MA thesis. Erasmus University Rotterdam, May 2014 (cit. on pp. 1, 2, 7, 27, 71–73, 82, 83, 91–93, 99, 100, 106, 110, 113).

[48] Stefan Knerr, Léon Personnaz, and Gérard Dreyfus. „Single-layer learning revisited: a stepwise procedure for building and training a neural network“. In: *Neurocomputing*. Springer, 1990, pp. 41–50 (cit. on p. 21).

[49] Yehuda Koren. „The bellkor solution to the netflix grand prize“. In: *Netflix prize documentation* 81 (2009).

[50] Max Kuhn and Kjell Johnson. *Applied predictive modeling*. Springer, 2013.

[51] J Richard Landis and Gary G Koch. „The measurement of observer agreement for categorical data“. In: *biometrics* (1977), pp. 159–174 (cit. on p. 24).

[52] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. „Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations“. In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM. 2009, pp. 609–616 (cit. on p. 114).

[53] Jeongyeon Lim, Munjo Kim, Bumshik Lee, et al. „A target advertisement system based on TV viewers profile reasoning“. In: *Multimedia Tools and Applications* 36.1-2 (2008), pp. 11–35 (cit. on p. 3).

[54] Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C Weng. „A note on Platts probabilistic outputs for support vector machines“. In: *Machine learning* 68.3 (2007), pp. 267–276 (cit. on p. 21).

[55] Zhouchen Lin, Minming Chen, and Yi Ma. „The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices“. In: *arXiv:1009.5055* (2010) (cit. on p. 114).

[56] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, et al. „Robust recovery of subspace structures by low-rank representation“. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35.1 (2013), pp. 171–184 (cit. on p. 114).

[57] S. Loeb. „Architecting Personalized Delivery of Multimedia Information“. In: *Commun. ACM* 35.12 (Dec. 1992), pp. 39–47 (cit. on p. 3).

[58] Nicolai Meinshausen and Peter Bühlmann. „Stability selection“. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72.4 (2010), pp. 417–473 (cit. on p. 112).

[59] Amir Navot, Ran Gilad-Bachrach, Yiftah Navot, and Naftali Tishby. „Is feature selection still necessary?“ In: *Subspace, latent structure and feature selection*. Springer, 2006, pp. 127–138 (cit. on p. 49).

[60] Bernard V North, David Curtis, and Pak C Sham. „A note on the calculation of empirical P values from Monte Carlo procedures“. In: *The American Journal of Human Genetics* 71.2 (2002), pp. 439–441 (cit. on p. 77).

[61] G. J. Nowak. „TV viewer characteristics and Results beyond response“. In: *Journal of Direct Marketing* 6.2 (1992), pp. 18–31 (cit. on p. 3).

[62] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. „Scikit-learn: Machine Learning in Python“. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on pp. 17, 69).

[63] Martin Piotte and Martin Chabbert. „The pragmatic theory solution to the netflix grand prize“. In: *Netflix prize documentation* (2009).

[64] John Platt et al. „Fast training of support vector machines using sequential minimal optimization“. In: *Advances in kernel methodssupport vector learning* 3 (1999) (cit. on p. 19).

[65] John C. Platt. „Fast Training of Support Vector Machines Using Sequential Minimal Optimization“. In: *Advances in Kernel Methods - Support Vector Learning*. MIT Press, Jan. 1998.

[66] K. Pongiannan. „An Exploratory Comparison On The Impact Of TV And Web Advertisements Towards The Buying Behaviour“. In: *European Journal of Business and Management* 3.4 (2011), pp. 244–257 (cit. on p. 1).

[67] Foster J Provost, Tom Fawcett, and Ron Kohavi. „The case against accuracy estimation for comparing induction algorithms.“ In: *ICML*. Vol. 98. 1998, pp. 445–453 (cit. on p. 24).

[68] Ryan Michael Rifkin. „Everything old is new again: a fresh look at historical approaches in machine learning“. PhD thesis. MaSSachuSettS InStitute of Technology, 2002 (cit. on p. 20).

[69] Robert E Schapire. „The boosting approach to machine learning: An overview“. In: *Nonlinear estimation and classification*. Springer, 2003, pp. 149–171.

[70] E.J. Schultz. *As Single Becomes New Norm, How to Market Without Stigma*. 2010. URL: `http://adage.com/article/news/advertising-market-singles-stigma/146376/` (visited on Oct. 11, 2010) (cit. on p. 111).

[71] Mark R Segal, Kam D Dahlquist, and Bruce R Conklin. „Regression approaches for microarray data analysis“. In: *Journal of Computational Biology* 10.6 (2003), pp. 961–980 (cit. on p. 11).

[72] W. E. Spangler, M. Gal-Or, and J. H. May. „Using Data Mining to Profile TV Viewers“. In: *Commun. ACM* 46.12 (Dec. 2003), pp. 66–72 (cit. on p. 4).

[73] The Nielsen Company. *NPOWER Unification Overview Reference Guide*. Accessed: 2015-06-09. 2011. URL: `http://en-us.nielsen.com/sitelets/cls/documents/npower/tips/NPOWER-Unification-v1.pdf` (cit. on p. 29).

[74] The Nielsen Company. *Shifts in Viewing: The Cross-Platform Report | Q2 2014*. Tech. rep. 2014, pp. 4–18 (cit. on p. 1).

[75] R. Tibshirani. „Regression Shrinkage and Selection via the Lasso“. English. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1 (1996), pp. 267–288 (cit. on pp. 7, 9).

[76] A. N. Tikhonov. *Numerical methods for the solution of ill-posed problems*. Vol. 328. Springer Science and Business Media, 1995 (cit. on p. 7).

[77] Laura Toloi and Thomas Lengauer. „Classification with correlated features: unreliability of feature ranking and solutions“. In: *Bioinformatics* 27.14 (2011), pp. 1986–1994 (cit. on p. 11).

[78] Vladimir Naumovich Vapnik and Samuel Kotz. *Estimation of dependences based on empirical data*. Vol. 40. Springer, 1982 (cit. on p. 19).

[79] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. „Extracting and composing robust features with denoising autoencoders“. In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 1096–1103 (cit. on p. 114).

[80] Paul Viola and Michael Jones. „Rapid object detection using a boosted cascade of simple features“. In: *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2001, pp. 1–511 (cit. on p. 49).

[81] Sergiy A Vorobyov, Yue Rong, Nicholas D Sidiropoulos, and Alex B Gershman. „Robust iterative fitting of multilinear models“. In: *Signal Processing, IEEE Transactions on* 53.8 (2005), pp. 2678–2689 (cit. on p. 10).

[82] A. R. Webb. *Statistical Pattern Recognition*. 2nd ed. West Sussex, England: John Wiley & Sons Ltd, 2002, p. 514 (cit. on p. 2).

[83] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. „Feature hashing for large scale multitask learning“. In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM. 2009, pp. 1113–1120 (cit. on p. 114).

[84] Jason Weston, Sayan Mukherjee, Olivier Chapelle, et al. „Feature selection for SVMs“. In: *NIPS*. Vol. 12. Citeseer. 2000, pp. 668–674 (cit. on pp. 49, 74).

[85] H. Zou and T. Hastie. „Regularization and variable selection via the elastic net“. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.2 (2005), pp. 301–320 (cit. on p. 11).

Decision Trees

A

One of the most commonly used method in data mining are decision trees. A decision tree is a classifier expressed as a partitioning of the feature space. A decision tree consists of one root node, which has no incoming edges, intermediate notes, which are split using a single attribute, and a leaf node, where each leaf is assigned to one class representing the most appropriate target value. Alternatively, a leaf node can also hold a probability indicating the probability of the target having a certain value. These decision trees are called binary recursive decision trees, though more complicated decision trees exist, these are in most cases not a good strategy, as it fragments the data too quickly [29, p. 311]. In addition, multi-way splits can be achieved by a series of binary splits. Therefore, binary splits are usually preferred.

A decision tree that partitions the feature space into M regions can be modelled as follows (see Equation A.1).

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \tag{A.1}$$

where c_m is a certain constant, which can be the class probability in the leaf node or the class itself. R_m stands for region m and the $I(\dots)$ is an indicator function that returns 1 if the observation is in region R_m and 0 otherwise. To grow a decision tree, it is infeasible to consider all possible binary partitions, therefore a greedy approach is used to decide on which variable a split should occur. The gradient boosting model in this thesis uses the *classification and regression tree* (CART) algorithm to construct a decision tree and this will also be the algorithm on which we will focus. The CART algorithm uses the *Gini impurity* metric to decide on which feature a split should occur [6]. The Gini impurity for node m can be defined as:

$$Gini(m) = \sum_{k=1}^{|C|} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{A.2}$$

where $\hat{p}_{mk}$ is the proportion of class k in node m (see Equation A.3).

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k) \tag{A.3}$$

The quality of a split s can then be given by the difference between the Gini impurity measurement before the split of node m and the Gini impurity measurements after the split into the left node l and right node r. Formally this is written as:

$$\Delta Q(s,m) = Gini(m) - \frac{N_{m_l}}{N_m} Gini(l) - \frac{N_{m_r}}{N_m} Gini(r) \tag{A.4}$$

where $\Delta Q(s,m)$ is the quality of the split s, N_{m_l} and N_{m_r} the number of observations from node m that goes into node l or r respectively. After choosing the best feature to split on, the last issue is to decide on which value a split should be. For ordered values this is rather straightforward. Consider feature $\mathbf{x}_j$ that contains only ordered values, with d distinct values. The split can therefore only be at $d-1$ locations. For unordered categorical values with binary outcomes, we can order the predictor classes according to the proportion falling in class 1, then split the predictor as an ordered predictor [6].

In most cases cost complexity pruning, is used to limit overfitting. It starts by constructing a large tree until some stopping criterion is reached, for example, the minimum number of observations in a leaf node. Then it collapses its nodes based on a cost complexity criterion. However, this method is not used by gradient boosting machines, since it relies on boosting to limit overfitting and reduce the variance.

Logistic Regression

B

Consider a categorical response variable Y that is dichotomous (i.e. $C = 2$) and takes two values: 0 and 1. This problem can be modelled as a binary logistic regression. In order to provide the intuition behind logistic regression, first consider a linear regression model (see Equation B.1).

$$P(Y = 1|X = x_i) = X^T\beta \tag{B.1}$$

Where x_i is the augmented observation vector (i.e. $x_i^T = (1, x_{i,1}, \ldots, x_{i,p})$) and β the weights associated with each feature. This is sometimes called the *linear probability model* and is often estimated using the *ordinary least squares* (OLS). The problem with this model is that the probability $P(Y = 1|X = x_i)$ on the left-hand-side (LHS) has to be between zero and one, but the linear predictor on the right-hand-side (RHS) can take on any value. A simple solution for this case would be transforming the probabilities into *odds*, defined as the ratio of the probability to its complement.

$$odds_i = \frac{\pi_i}{1 - \pi_i} \tag{B.2}$$

Where π_i is the probability $P(Y = 1|X = x_i)$. This transformation however, is insufficient to map the probabilities to the entire real line. To achieve this, logarithms are taken, resulting in the *logit* or *log-odds*.

$$\log odds_i = \log \frac{\pi_i}{1 - \pi_i} = X^T\beta \tag{B.3}$$

This effectively removes the floor restriction and successfully maps the probabilities onto the entire real line. Probability models are, as a rule estimated from survey data, which in most cases provide large samples of independent observations with a wide range of variation of the features.

Generalising Equation B.3 to the multinomial case, we equate the linear component to the log of the odds of a k-th observation compared to the baseline class C (i.e. we consider the C class to be omitted). This gives the following model.

$$\log \frac{\pi_{i,k}}{1 - \sum_{k=1}^{C-1} \pi_{i,k}} = X^T\beta \qquad \forall k = 1, 2, ..., C - 1 \tag{B.4}$$

Here $\pi_{i,K}$ is the probability that $P(Y = K|X = x_i)$.

B.1 Maximum likelihood estimation

The preferred method of estimating the logit model is using a *maximum likelihood estimation*. This permits the estimation of parameters of almost any specification of the probability function. The general concept is as follows: given the data, the goal is to learn something about the model. In other words, the distribution of the unknown parameters θ conditional on the observed data $\mathbf{X}$ are of interest. This can be written as Equation B.5.

$$p_{\mathbf{y}}(\theta|\mathbf{X}) \tag{B.5}$$

Where $p_{\mathbf{y}}(\theta|\mathbf{X}) = P(Y = \mathbf{y}|X = \mathbf{X}; \theta)$ and $\theta = \{\beta_0, \beta\}$. This is also known as the *inverse probability problem*. Using the Bayes' theorem, Equation B.5 can be written as Equation B.6).

$$p_{\mathbf{y}}(\theta|\mathbf{X}) = \frac{p_{\mathbf{y}}(\theta \cup \mathbf{X})}{p_{\mathbf{y}}(\mathbf{X})} \tag{B.6}$$

$$= \frac{p(\theta)}{p_{\mathbf{y}}(\mathbf{X})} p_{\mathbf{y}}(\mathbf{X}|\theta) \tag{B.7}$$

As it only makes sense to compare the conditional densities for the same data, the denominator can be basically be ignored. This results in Equation B.8 which shows that the posterior is proportional to the prior times the likelihood.

$$p_{\mathbf{y}}(\theta|\mathbf{X}) = p(\theta) p_{\mathbf{y}}(\mathbf{X}|\theta) \tag{B.8}$$

In most cases, the prior is unknown and it is difficult to make assumptions about it, thus it is impossible to calculate the inverse probability directly. Fisher [19] solved this problem by introducing the notion of likelihood and the Likelihood Axiom, which uses relative measures of uncertainty. The Likelihood Axiom can be defined as follows (see Equation B.9).

$$\begin{aligned} \mathcal{L}(\theta) &= k_{\mathbf{y}}(\mathbf{X}) p_{\mathbf{y}}(\mathbf{X}|\theta) \\ &\propto p_{\mathbf{y}}(\mathbf{X}|\theta) \end{aligned} \tag{B.9}$$

where $k_{\mathbf{y}}(\mathbf{X}) = \frac{p(\theta)}{p_{\mathbf{y}}(\mathbf{X})}$ and a constant, since it is an unknown function of the data. This transforms the problem into maximising the likelihood $\mathcal{L}$, as this would also maximise the likelihood of observing the data, due to the proportional relationship. Thus the likelihood $\mathcal{L}$ is a relative measure of uncertainty of different values of θ given a particular dataset. This also implies that it is not possible to compare likelihoods across datasets.

In most cases, to simplify the calculations, a log-likelihood function is maximised rather than the likelihood function itself. To derive the log-likelihood function ℓ for a binary logistic model, we start with the joint probability density function (see Equation B.10).

$$f(\mathbf{X}|\theta) = \prod_{i=1}^{N} \binom{n_i}{y_i} \pi_i (1-\pi_i)^{n_i - y_i} \tag{B.10}$$

where n_i is the number of trials for observation i, y_i the number of successes and π_i the probability of success. This is very similar to the likelihood function, except the parameters of the function are reversed (i.e. the likelihood function expresses the values of θ in terms of fixed values for y. This results in the following log likelihood function ℓ (see Equation B.11).

$$\ell(\theta) = \sum_{i=1}^{N} \left(y_i \theta^T x_i\right) - \log\left(1 + \exp(\theta^T x_i)\right) \tag{B.11}$$

Similarly, the log likelihood function for the multinomial logistic model can be constructed.

Cramer's V

C

Measures of association provide a means of summarising the size of the association between two variables. Most measures are scaled such that they reach a maximum numerical value of one when two variables have a perfect relationship with each other. One way to determine whether there is a statistical relationship between two variables is to use the chi square test for independence. Unfortunately, these tests could be incorrect due to the small expected values and therefore the approximations of the p-values could be incorrect [3]. As a result, in this thesis only the measure of association is given between predictions of different models. Several different measures of association exist, ϕ, contingency coefficient and Cramer's V. Although they are all related, Cramer's V is preferred as it is scaled from zero to one and it can be applied to variables with different dimensions[1]. Cramer's V is defined as:

$$V = \sqrt{\frac{\chi^2}{nt}} \tag{C.1}$$

where $\chi^2 = \sum \frac{(f_o - f_e)^2}{f_e}$ and f_o is the observed frequency and f_e is the expected frequency. It should also be noted that it is possible to use "accuracy" as a measure of association between two different predictions, but since this is not widely used and to avoid confusion, accuracy levels are not used and Cramer's V are used to express the measure of association.

[1] ϕ is not scaled properly and the contingency coefficient does not equal one if both variables are perfectly related to each other.

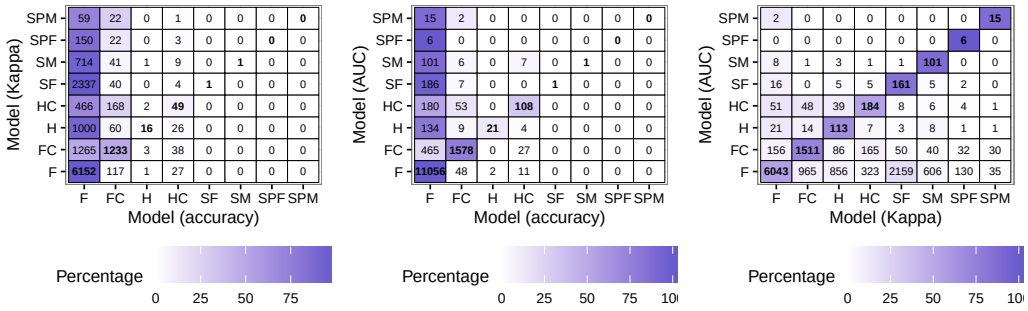
Additional graphs

D

These are additional graphs accompanying the results in Chapter 5.

D.1 Base Model

D.1.1 Multinomial Logistic Model

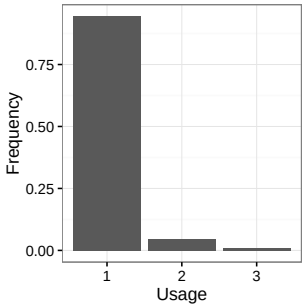


(a) Models selected using Kappa and accuracy.

(b) Models selected using AUC and accuracy.

(c) Models selected using AUC and Kappa.

Fig. D.1.: Confusion matrix of the predictions made between different logistic models where the hyperparameters α and λ of the models are selected with different performance measures.

Usage Distribution	Most Common Features
	`JESSIE`
	`Frequency AUSTIN & ALLY`
	`Frequency MICKEY MOUSE CLUBHOUSE`
	`NChannels`
	`NPrograms`

Tab. D.1.: The usage of features among all class equations of the logistic model.

Rank	Feature	Coefficient
1	(Intercept)	1.19
2	Frequency DEATH ROW:FINAL 24 HOURS	0.22
3	Frequency REDEMPTION	0.18
4	Frequency ALCATRAZ: LIVING HELL	0.16
26	Frequency FARMERS INSURANCE OPEN-SA	0.07
88	Frequency CBS EVENING NEWS	0.01

Tab. D.2.: Features used by the family (F) equation in the AUC logistic regression model ranked by the absolute value of the coefficient.

Rank	Feature	Coefficient
1	(Intercept)	0.50
2	Frequency FREESKIING	0.31
3	Frequency VECINOS III	0.21
4	Frequency SECRET MILLIONAIRES CLUB	0.20
56	Frequency GOOD LUCK CHARLIE	0.01
67	Frequency DOG WITH A BLOG	0.01

Tab. D.3.: Features used by the family with children (FC) equation in the AUC logistic regression model ranked by the absolute value of the coefficient.

Rank	Feature	Coefficient
1	Frequency EROTIC TRAVELER 01:MOLDED	0.34
2	Frequency SPAWN (1997)	0.27
3	Frequency UDN F?TBOL CLUB AM	0.20
110	ROSWELL:FLYING SAUCERS OV	0.00
112	AFC CHAMP POST GUN ON CBS	0.00

Tab. D.4.: Features used by the household (H) equation in the AUC logistic regression model ranked by the absolute value of the coefficient.

Rank	Feature	Coefficient
1	Frequency WMNS COLL LACROSSE	0.58
2	Frequency GHOSTLAND TENNESSEE	0.41
3	(Intercept)	-0.38
4	Frequency TITULARES TELEMUNDO S	0.22
72	Frequency FAIRLY ODD PARENTS	0.01
142	NPrograms	0.00
196	NATIONAL TEAMS (PT) WED	0.00

Tab. D.5.: Features used by the household with children (HC) equation in the AUC logistic regression model ranked by the absolute value of the coefficient.

Rank	Program	Value
1	(Intercept)	0.92
2	Frequency MANNAFEST W/PERRY STONE	0.34
3	Frequency TRIP FLIP	0.20
4	Frequency LATE SHOW/DL-SUS-1/1	0.19
62	NPrograms	-0.00

Tab. D.6.: Features used by the single female (SF) equation in the AUC logistic regression model ranked by the absolute value of the coefficient.

Rank	Feature	Coefficient
1	Frequency HIP HOP POWER HOUR	0.52
2	Frequency FXM FILLER	0.48
3	Frequency ROAD TO ROLEX 24	0.30
4	Frequency DESIGNING WOMEN	0.22
86	SVP & RUSSILLO L	0.00

Tab. D.7.: Features used by the single male (SM) equation in the AUC logistic regression model ranked by the absolute value of the coefficient.

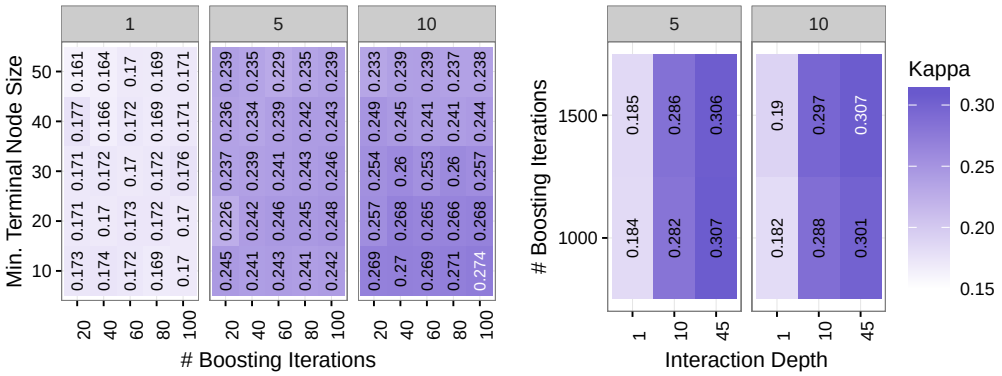
Rank	Program	Value
1	(Intercept)	-0.99
2	Frequency 2 TURNTABLES & MICROPHONE	0.19
3	Frequency CATCH 21 1/14 6:30P	0.13
4	Frequency TEEN KIDS NEWS (B)	0.04
12	CATCH 21 1/14 6:30P	0.00
16	TEEN KIDS NEWS (B)	0.00

Tab. D.8.: Features used by the single female parent (SPF) equation in the AUC logistic regression model ranked by the absolute value of the coefficient.

	Program	Value
1	(Intercept)	-1.35
2	Frequency NFL SUPER BOWL KICKOFF	0.62
3	Frequency VENOM (2005)	0.40
4	Frequency GOLDEN BOY ON FOX	0.40
5	Frequency D2: MIGHTY DUCKS	0.11

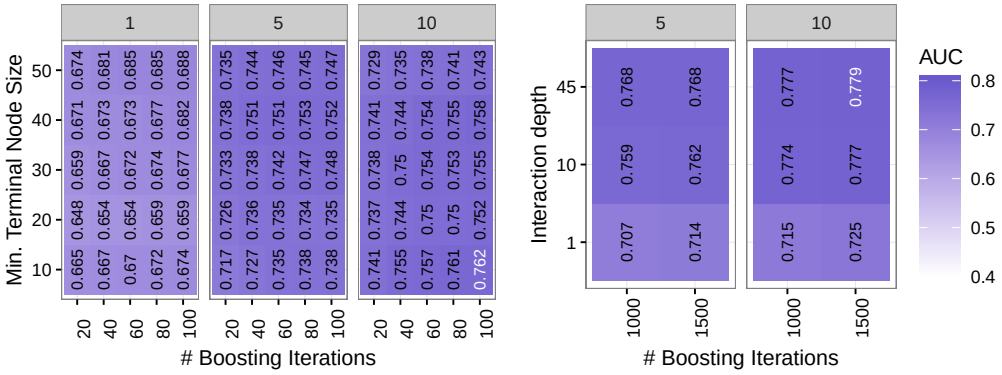
Tab. D.9.: Features used by the single male parent (SPM) equation in the AUC logistic regression model ranked by the absolute value of the coefficient.

D.1.2 Stochastic Gradient Boosted Models



(a) 100 boosting iterations **(b)** 1500 boosting iterations

Fig. D.2.: Level plot for Kappa over different combinations of number of boosting iterations, M, interaction depth and minimum observations per node. The shrinkage v stays constant at 0.001 and the sampling fraction $\eta = 0.5$.



(a) 100 boosting iterations **(b)** 1500 boosting iterations

Fig. D.3.: Level plot for AUC over different combinations of number of boosting iterations, M, interaction depth and minimum observations per node. The shrinkage v stays constant at 0.001 and the sampling fraction $\eta = 0.5$.

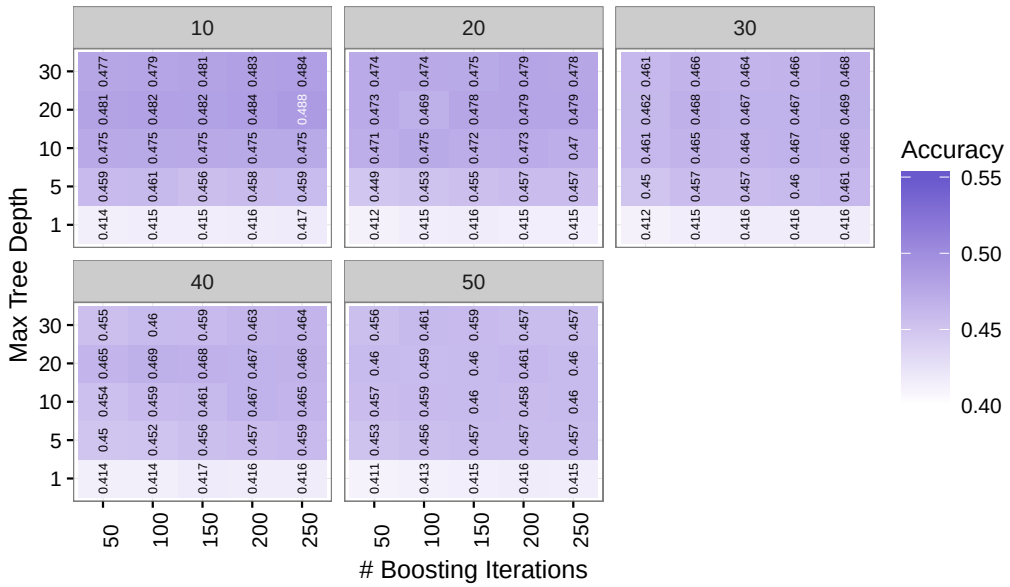


Fig. D.4.: Level plot for accuracy over different combinations of number of boosting iterations, M, interaction depth and minimum observations per node. The shrinkage v stays constant at 0.001 and the sampling fraction $\eta = 0.9$.

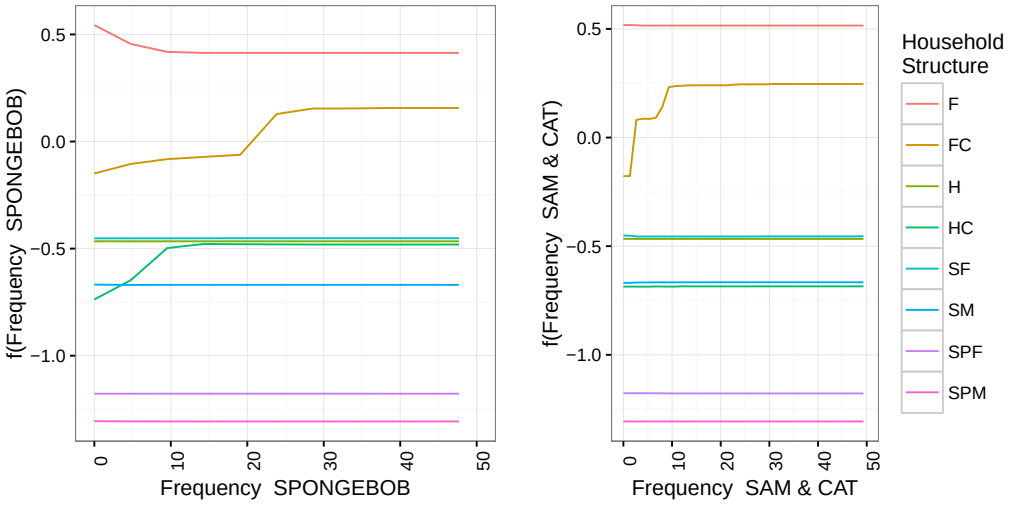


Fig. D.5.: Partial dependency plot of the feature: `Frequency SPONGEBOB`.

Fig. D.6.: Partial dependency plot of the feature: `Frequency SAM & CAT`.

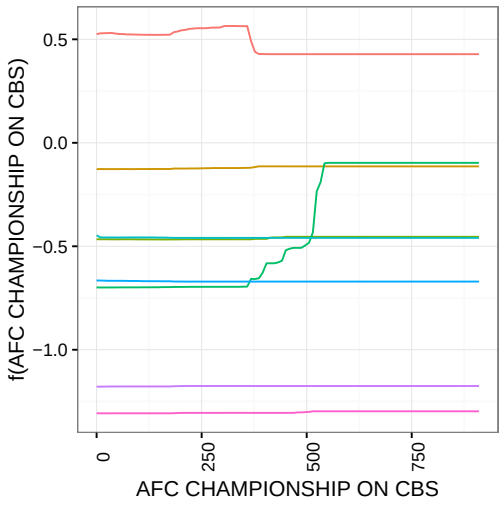


Fig. D.7.: Partial dependency plot of the feature: `AFC CHAMPIONSHIP ON CBS`.

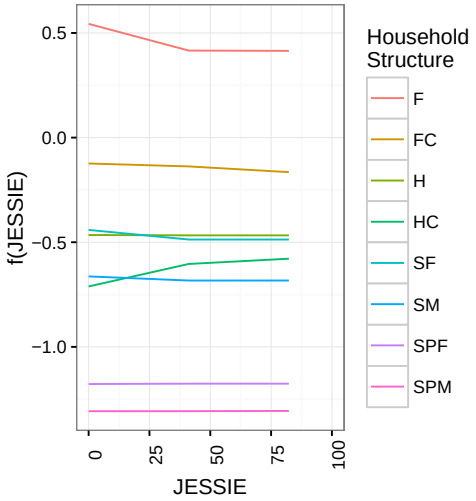
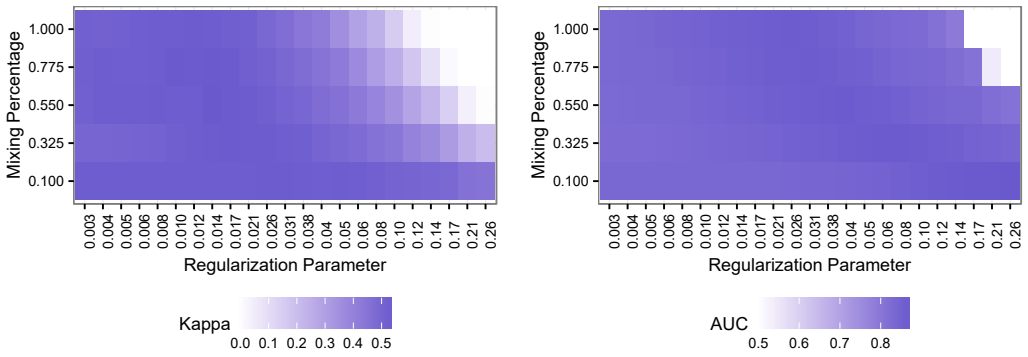


Fig. D.8.: Partial dependency plot of the feature: `JESSIE`.

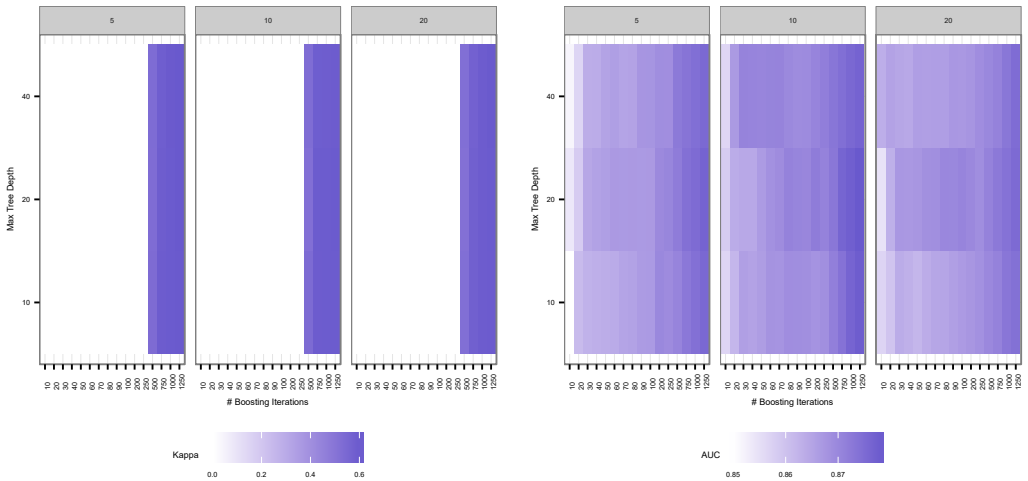
D.2 Cascading Classification Models

D.2.1 First Stage Classification



(a) Kappa **(b)** AUC

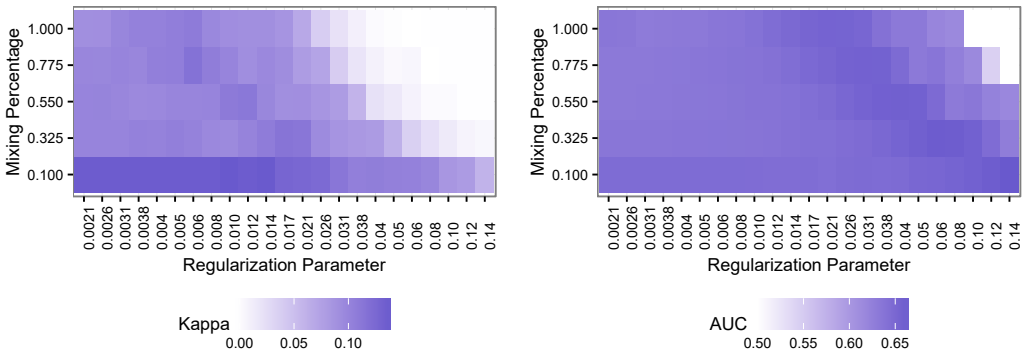
Fig. D.9.: Level plot for cross-validation Kappa and AUC values over different combinations of the mixing (α) and regularization (λ) hyperparameters.



(a) Kappa **(b)** AUC

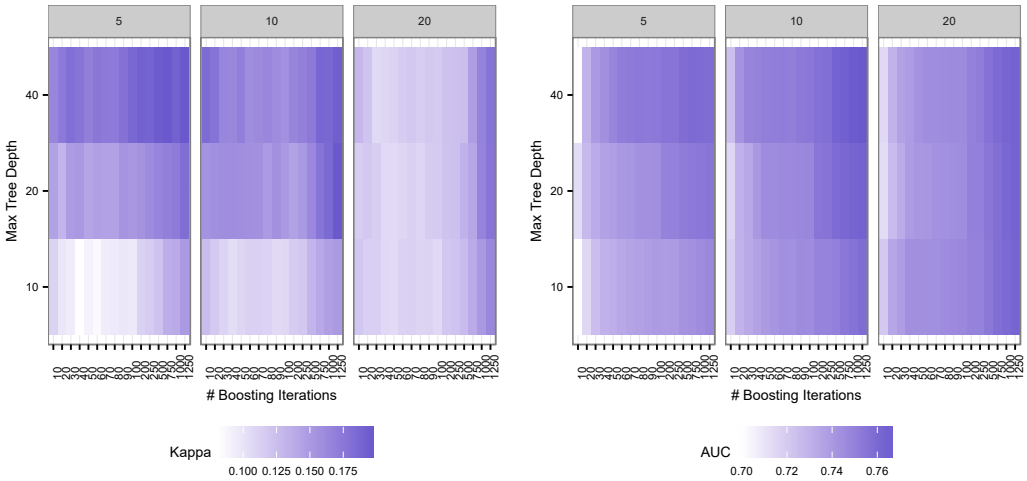
Fig. D.10.: Level plot for Kappa and AUC over different values of boosting iterations (M), interaction depths (s) and minimum observations per node ($|R|_{min}$). The shrinkage (v) and sampling fraction (η) stay constant at $v = 0.001$ and $\eta = 0.5$.

D.2.2 Second Stage Classification



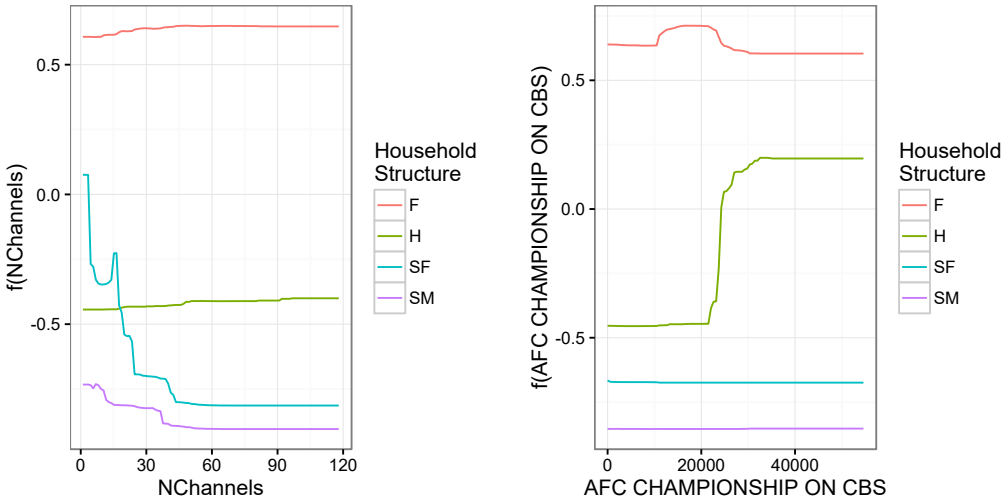
(a) Kappa **(b)** AUC

Fig. D.11.: Level plot for cross-validation Kappa and AUC values over different combinations of the mixing (α) and regularization (λ) hyperparameters.



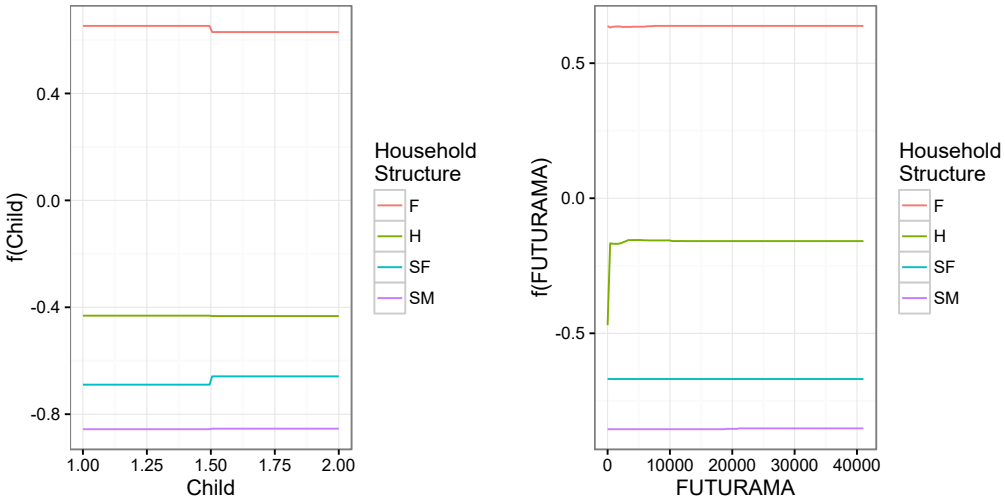
(a) Kappa **(b)** AUC

Fig. D.12.: Level plot for Kappa and AUC over different values of boosting iterations (M), interaction depths (s) and minimum observations per node ($|R|_{min}$). The shrinkage (v) and sampling fraction (η) stay constant at $v = 0.001$ and $\eta = 0.5$.



(a) NChannels **(b)** AFC CHAMPIONSHIP ON CBS

Fig. D.13.: Partial dependency plots of different important variables of the Kappa GBM model.



(a) Child **(b)** FUTURAMA

Fig. D.14.: Partial dependency plots of different important variables of the Kappa GBM model.