Introduction

In this paper, we will explore several prediction methods in order to best predict the qualitative variable *High*, which equals one for *Income* equal to or above $50,000 and zero otherwise. The dataset, *marketing* is obtained from the R-package **ElemStatLearn**. We will use an experimental approach, where we will look at different metrics in order to find the best method. In order to make sure that the models are fitted on the same basis and to make sure that the variable importance discussed in Task D is calculated the same way, we have used the **caret** package throughout this paper. By doing so, we ensure that the models utilize the same metrics when fitting the models and we are able to better compare the results from the different methods.

We are aware that our train/test split impacts our predictions. Ideally, we could run the methods multiple times with different train/test seeds in order to investigate the impact of our chosen seed. However, this requires a lot of computing power and we have therefore chosen to set a specific seed. We are using the same seed for all of the methods presented in this paper.

Further, we will first present descriptive statistics of the initial dataset, *marketing*. Next, we will look at the different prediction methods we are using in order to predict the qualitative variable *High* and the evaluation metrics we use when evaluating the predictions. We will describe the application of the methods and our model tuning, before we briefly present the prediction result from each method. In addition, we will elaborate on the importance of the different predictors in each model and lastly, a summary of all the methods with regards to the metrics we use will be presented and a corresponding conclusion will be derived.

Task A

Descriptive statistics

Data set

The marketing data set consists of 14 variables and 8.993 observations based on questions filled out by shopping mall customers in the San Francisco Bay area. The data set is a combination of categorical and continuous variables.

Initially our dependent variable is *Income*, which is the annual income of household (personal income if single). This variable is divided into nine subcategories, with five or ten thousand dollars intervals, all from "less than $10.000" up to "$75.000 or more". The rest of the variables, such as *Age*, *Education* and *Occupation*, are all identifying various demographic attributes.

In Table 01 we present the different categories with the corresponding metrics mean, median, minimum value and maximum value. We have also added a row for number of NA's for each category. One interesting feature is that the median for the age category is 3, which corresponds to age 25-34. Judging by the histogram in Figure 01, we get an overall picture of the different respondents' answers. For example, we see from the histograms that a large percentage of the respondents is White, that English is by far the most spoken language, that there were mostly female respondents and that a great portion of the respondents have lived in the San Fran./Oakland/San Jose Area for more than ten years (Hastie, Tibshirani, & Friedman, 2009).

Table 01: Descriptive statistics

	Income	Sex	Marital	Age	Edu	Occupation	Lived	Dual_Income	Household	Householdu18	Status	Home_Type	Ethnic	Language	High
Mean	4.9	1.55	3.03	3.42	3.84	3.79	4.2	1.54	2.85	0.67	1.84	1.86	5.96	1.13	0.24
Median	5.0	2.00	3.00	3.00	4.00	4.00	5.0	1.00	3.00	0.00	2.00	1.00	7.00	1.00	0.00
Min	1.0	1.00	1.00	1.00	1.00	1.00	1.0	1.00	1.00	0.00	1.00	1.00	1.00	1.00	0.00
Max	9.0	2.00	5.00	7.00	6.00	9.00	5.0	3.00	9.00	9.00	3.00	5.00	8.00	3.00	1.00
No.NA	0.0	0.00	160.00	0.00	86.00	136.00	913.0	0.00	375.00	0.00	240.00	357.00	68.00	359.00	0.00

Further, we create a dummy variable called *High* which is one for Income equal to $50,000 or more and zero otherwise, and this ends up being our dependent variable. From Figure 01, we see that this variable has around three quarters as zero, and one quarter as one.

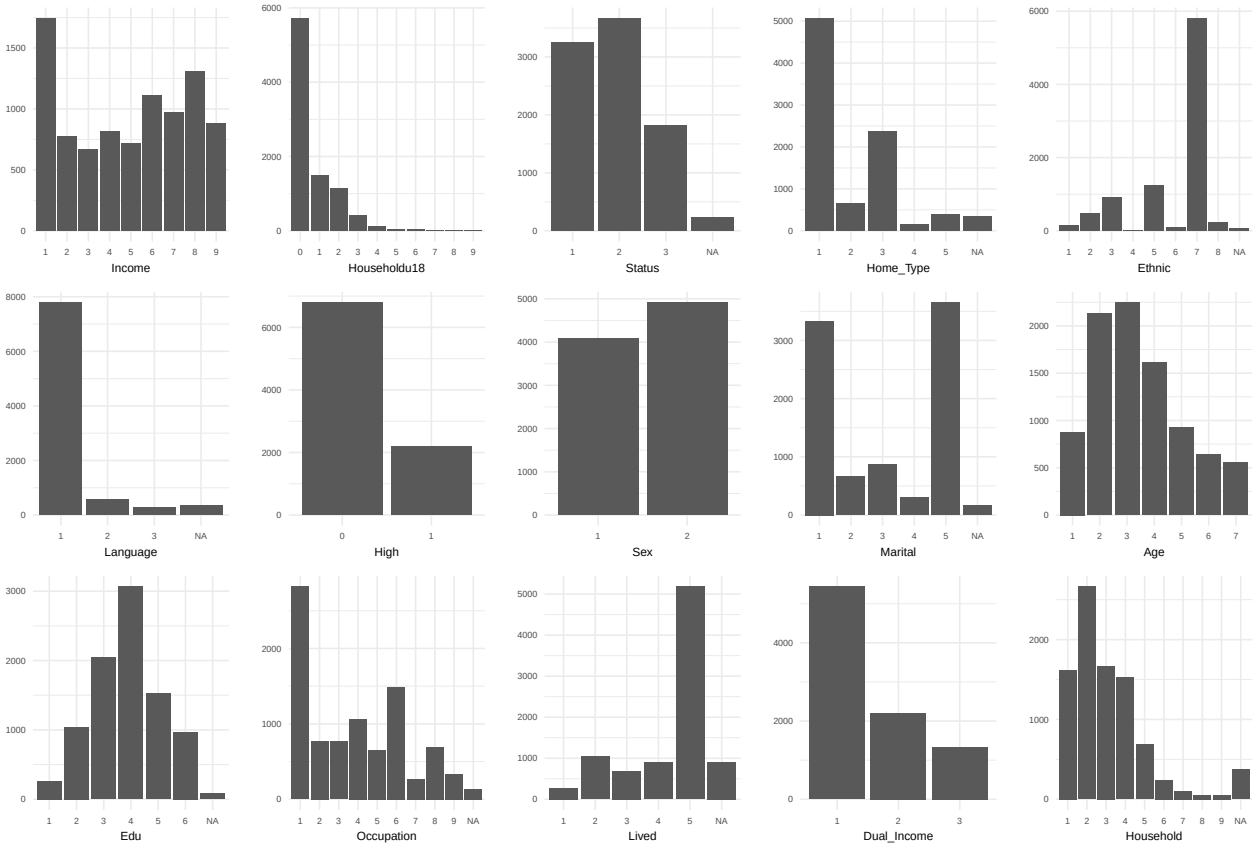

Figure 01: Histogram of variables

There are a number of missing values in the set, which we need to deal with. When we investigate the missing values in the data set further, we deduct from Table 01 that around one third of the missing values are present in the *Lived* valiable, a variable which identifies how long one has lived in the San Fran./Oakland/San Jose area. Since this variable accounts for a large percentage of the missing values in the total data set, we experimented with removing the variable from the data set, before removing the observations with missing values, in order to presevere as many observations as possible. When we remove the variable *Lived* before we remove the observations with missing values, we restore 700 observations, compared to removing all the missing values from the beginning. We recognize that there is a trade off between possible losing information from the *Lived* variable

and losing information from the 700 observations that are being restored when first removing the variable. In either case, we might end up losing valuable data. Another way of dealing with missing values is to use what is called mean imputation, a method where an observation's missing value(s) is being substituted by the mean of the other observations in the given variable. By doing so, the sample size is preserved, but the variability in the data will be reduced. Since there are multiple ways of dealing with the missing values in the data set, we choose to run all the models, using four different approaches in order to see which of the approaches yields the highest accuracy and AUC. The four approaches are as following; removing all observations with missing values, removing *Lived* before removing observations with missing values, mean imputation on the entire data set an mean imputation after removing the variable *Lived*. The results from all four approaches can be found in (Appendix A). It can be seen in the results that the accuracy and AUC is fairly close, regardles of the method used to deal with missing values. However, since the AUC is on average highest for the data set where the *Lived* column is kept, and the NAs are substituted with the mean of each variable, the remainding part of the paper focuses an exclusively on this data set. We are aware that this solution might not apply to all other data sets and that the results might change with a different train/test-split. We are then left with 8993 observations, one response variable and 13 explanatory variables.

Task B & C

Brief Explanation of Prediction Methods

We start by presenting a brief explanation of the prediction methods and the evaluation methods we are using in this paper.

Logistic Regression

When we have a qualitative response variable, it is preferable to use a classification method such as logistic regression. Logstic regression models the probability that the response variable corresponds to the different categories,meaning that for any input value of x, the output is between zero and one (James, Witten, Hastie, & Tibshirani, 2013). In our case, we will calculate the probability that the variable *High* corresponds to either 1 or 0.

Linear discriminant analysis

Linear discriminant analysis (LDA) is, together with principal component analysis (PCA), a linear transformation technique that is being used for dimensionality reduction. PCA is called an unsupervised algorithm, because it ignores the class labels and its aim is to capture the directions which in turn maximizes the variance in the data set. In contrast, the LDA is supervised and computes the directions – called linear discriminants – that will represent the axes that maximizes the separation between multiple classes. Furthermore, LDA is linked to regression analysis and ANOVA (analysis of variance), which is another way of expressing the dependent variable as a linear combination of different features. The difference is that LDA uses continuous independent variables and a categorical dependent variable, whereas the ANOVA (analysis of variance) treats a continuous dependent variable and categorical dependent variable. The logistic regression is also

somewhat similar to LDA, but is a routine that is desirable in cases where we do not assume normally distributed independent variables. This is an assumption of the technique used in LDA.

Classification and Regression Trees (CART)

CART is a machine learning method that is used for both classification and regression. The method, partitions the data into multiple sub-spaces, making the final sub-space as homogenous as possible. The approach is called recursive partitioning, and the produced result consist of a set of rules for predicting the given outcome variable. However, a fully "grown tree" increases the likelihood of overfitting and one possible strategy of avoiding this problem is by pruning the tree. The strategy consists of avoiding splitting a partition unless it does not significantly increase the overall quality of the model (James et al., 2013).

Random forest

Random forest is an ensemble learning method for classification, regression and other tasks that may be constructed by decision trees. The output is a mode of class or prediction of individual trees. Each tree is grown from a random sample of the training data, each node is formed by picking a variable to split on and a value to make the split. Generally, the method builds multiple decision trees and use the built trees to perform more accurate predictions. Using slightly different trees to build each tree adds diversity to the model, and then averaging multiple trees together reduces the risk of overfitting. The randomness of this method results in diversity to the set of trees, such that more subtle patterns from the training data may be picked up (James et al., 2013).

Boosting

Boosting is an approach for improving the predictions resulting from a decision tree. In this method, we are combining a large number of decision trees and the trees are grown sequentially, meaning that each tree uses information from the previously grown trees (James et al., 2013).

Evaluation Methods

We start by presenting a brief explanation of some evaluation methods before we motivate our choice of cross-validation method and our choice of fitting function.

Cross-Validation

Cross-Validation involves different validation techniques used to estimate the test error. This is done by holding out a subset of the training observations from the fitting process, and use this subset to evaluate the fitted model (James et al., 2013, p 176). Cross-Validation outputs the test error estimate for each model so that we can select the model with the estimated lowest test error, hence estimate the accuracy of our prediction model. Further on, we will evaluate the predictions from Best Subset Selection using two different Cross-Validation methods; *Leave-One-Out Cross-Validation* and *K-fold Cross-Validation*.

Validation Set Approach

The Validation Set Approach involves "randomly dividing the available set of observations into a training set and a validation set. The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set" (James et al., 2013, p 176).

Leave-One-Out Cross-Validation

Leave-One-Out Cross-Validation (LOOCV) involves splitting the data into two parts with a single observation as the validation set and the remaining observations as the training set. This process is then repeated n times with different a observation as the validation set, every time, resulting in n squared errors. By taking the average of all the errors, we get the LOOCV estimate for the test error (James et al., 2013).

K-fold Cross-Validation

K-fold Cross-Validation involves randomly dividing the set of observations into k-folds, of approximately equal size. One fold is treated as a validation set, and the method is fit on the remaining k - *1* folds. The estimated test error is then computed on the observations in the held-out fold. Further, the procedure is repeated k times with a different group of observations each time. This yields k estimates of the test error and the estimated test error is then calculated by averaging the estimated test errors of all the k-folds (James et al., 2013). This method results in k different test errors, and the lowest one corresponds to the best model.

Choice of method

When training the models, we split the data into a test set and a train set, where the train set consists of 75% of observations in the total data set, while the test set consists of the remaining observations. The 75/25-split is chosen in order to include as many observations in the train set as possible, while still having enough observations in the test set to properly evaluate the fitted models.

In addition to this, we choose the k-fold cross-validation (k-fold CV) method when training the models. K-fold CV is chosen over the *Validation Set Approach* and the *Leave-one-out cross-validation* method for a few different reasons. Firstly, the validation estimate of the test error from the *Validation Set Approach* can vary a lot, depending on which observations that are included in the training and validation set. Furthermore, only a subset of the observations is used to fit the model which may lead an overestimation of the test error (James et al., 2013). When comparing the abovementioned CV methods, we note that the k-fold CV is less computational expensive than LOOCV if $k<n$, making it a better option for our laptops. Secondly, due to a bias variance trade-off, k-fold CV can often give a more accurate estimate of the test error rate than LOOCV. This is true, despite the fact that LOOCV gives a more unbiased estimate of the test error, compared to k-fold CV. Because LOOCV has a higher variance than k-fold CV with $k<n$, the test error estimate from LOOCV often has a higher variance than the test error estimate resulting from k-fold CV. With this in mind, we choose a k-fold CV with $k = 10$. $K = 10$ is chosen considering the fact that

empirical testing has shown that $k = 10$, in addition to $k = 5$, has yielded test error rate estimate that are neither excessively high biased, nor a very high variance (James et al., 2013).

To summarize, when training and testing our models, we implement a 75/25-split and use a *10*-fold CV to estimate the test error and tune the parameters. We also note that some models compute a probability for the categories, rater than a outcome 0 or 1. In these models, we set the threshold for classifying the prediction to to 0.5.

Choice of fitting function

When validating the different prediction models, we measure both accuracy and AUC. When computing the accuracy, we divide the number of correct predictions by the total number of predictions. This gives us a percentage of the correctly classified predictions and can be used when comparing the different models. We note however, that accuracy alone can some times be misleading. When one outcome is very rare, predicting the opposite can result in very high accuracy. In our data set, 25.79% of the observations are classified as 1, meaning that a prediction of all zeros would yield an accuracy of 74.21%. Due to this, we choose to measure the quality of the predictions with the addition of the AUC metric.

The Area Under the Curve (AUC) is the area under the ROC-curve. ROC is an acronym for Receiver Operating Characteristics and the curve shows the relationship between the true positive rate (sensitivity) and the false positive rate (1- specificity) for every possible threshold. Every point on the curve represents a possible threshold. This means that we do not specify a specific threshold when computing AUC. AUC can be defined as "the overall performance of a classifier, summarized over all possible thresholds" (James et al., 2013).

Logistic Regression

When we fit our logistic regression model, we used the **caret** package and the function *glm*. We also set the argument *familiy = binomial* to ensure that the regression model we are fitting, is a logistic regession model.

The logistic regression model yields an accuracy of 80.78% and an AUC of 81.01%, on the test set. From the predictions, we can further calculate the confusion matrix in (Appendix B) and the ROC curve can be seen in Figure 02.

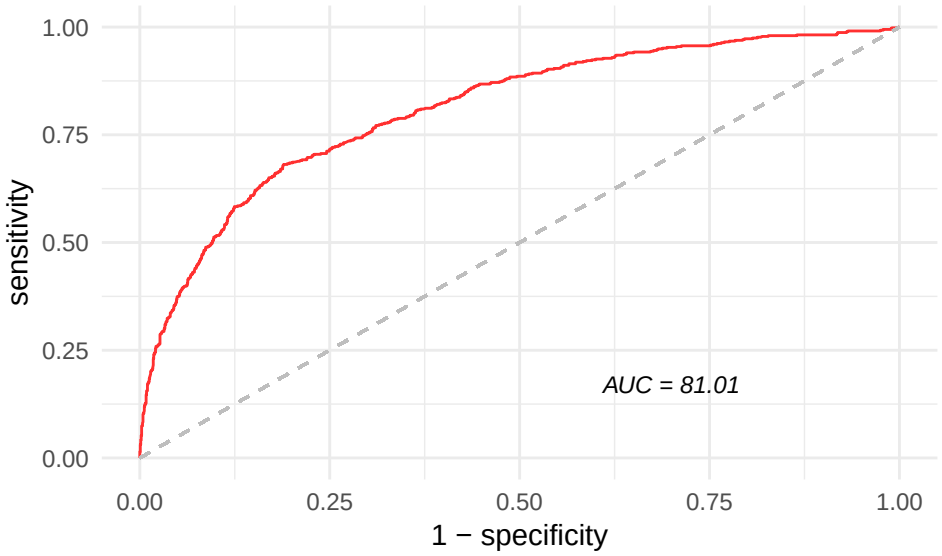

Figure 02: Logistic regression ROC curve

Linear Disriminant Analysis

When we fit our LDA model, we used the **caret** package and the function *lda*.

The LDA model yields an accuracy of 82.87% and an AUC of 80.92%, on the test set. From the predictions, we can further calculate the confusion matrix in (Appendix B) and the ROC curve can be seen in Figure 03.

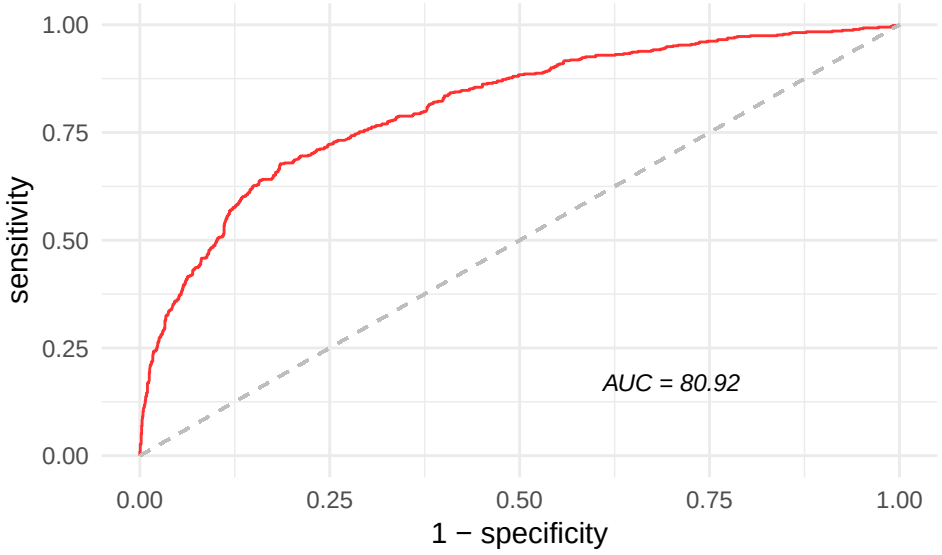

Figure 03: LDA ROC curve

Classification tree with pruning

For this particular method, we will use the **caret** package and the *rpart* function. First, the model is fitted using a fully-grown tree. However, the full tree includes all predictors and is also difficult to interpret as we have a large dataset with multiple predictors. In addition, as we briefly looked into previously a fully-grown tree is more likely to overfit the training and thereby result in poor test set performance. One way of limit the problem of overfitting, is to prune back the tree resulting in a simpler tree with fewer splits and better interpretation at the cost of little bias (James et al., 2013).

The general idea with pruning is to investigate whether a smaller subtree can give us a comparable result to the fully-grown tree. To achieve a smaller subtree, we can use the strategy of split a partition only if that improves the overall quality of the model. Within the rpart package, this is controlled by the complexity parameter (cp). A low cp might result in overfitting, and a high cp might result in a very small subtree, both yielding poor predictions of the test data. The optimal cp is found by minimizing the cross-validation error, as shown in Figure 06.

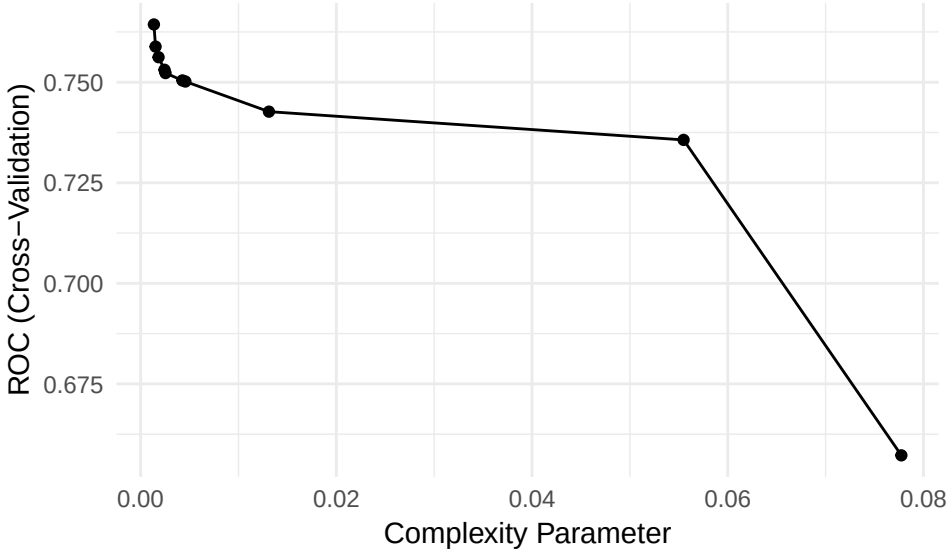

Figure 04: Optimal complexity parameter

The pruned tree yields an accuracy of 82.65% and AUC of 76.49%. From the predictions, we can further calculate the confusion matrix in (Appendix B) and the ROC curve can be seen in Figure 05.

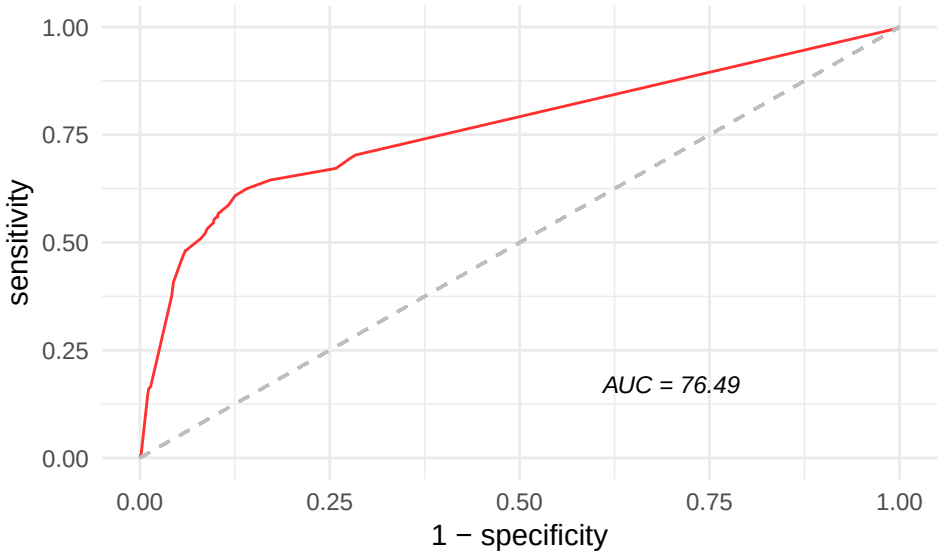

Figure 05: Pruned tree ROC curve

Random Forest

For this particular method, we will use the **caret** package and specify the method as *Random Forest*. This method automatically selects the optimal number of predictor variables randomly sampled as candidates at each split, such that the optimal random forest model is fitted. Furthermore, it is specified within the model that we wanted to maximize the AUC-value that can be derived from the ROC-curve, as we want to optimize the prediction accuracy. By default, 500 trees are trained, and the optimal number of variables sampled at each split is equal to 2, as shown in the plot below.

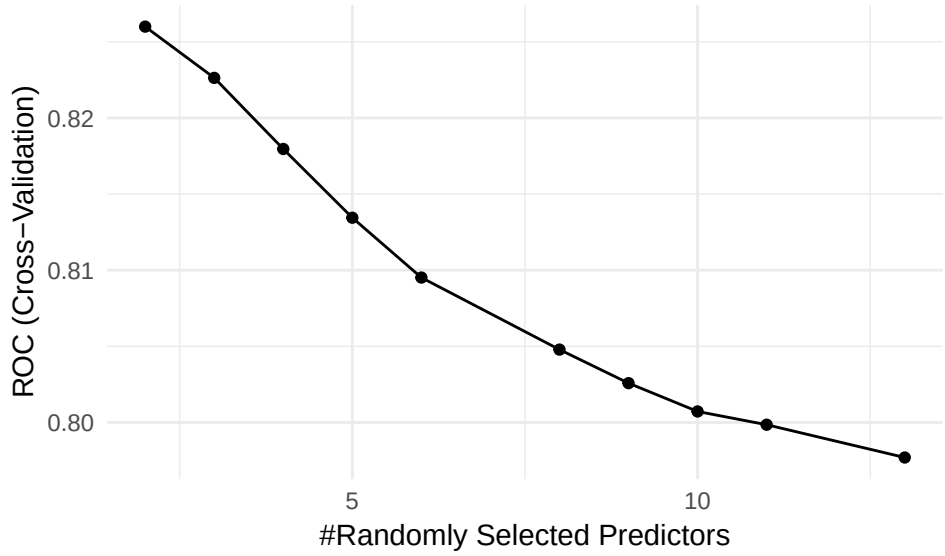

Figure 06: Optimal complexity parameter

The fitted model yields an accuracy of 82.61% and AUC of 84.08%. From the predictions, we can

further calculate the confusion matrix in (Appendix B) and the ROC curve can be seen in Figure 07.

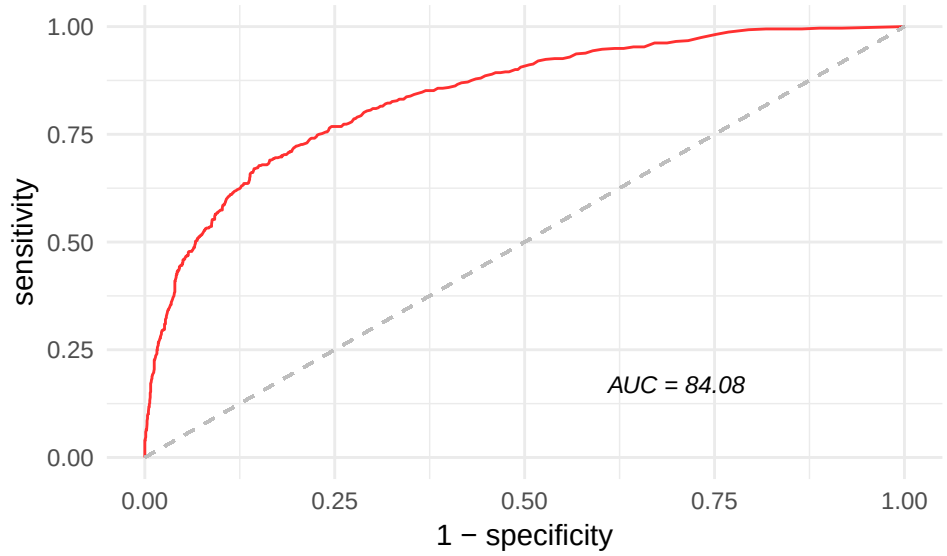

Figure 07: Random forest ROC curve

Boosting

When fitting our boosted decision tree, we used the *gbm* function from the **gbm** package.

When we use boosting, we mainly focus on three tuning parameters; the number of trees B, the shrinkage parameter λ, also known as the learning rate, and the number of d splits in each tree.

In our model, we use cross-validation to select the optimal number of trees from a list of 50 trees, ranging from 20:1000. Further, we find the optimal λ by fitting the model two times with a λ of 0.1 and 0.01 respectively. We note that the optimal λ might be different if we were to test other λs, but this would require more processing poweer. Lastly, we look at five different options when it comes to the number of d splits in each tree; 1, 3, 5, 7, 9.

After we run the model with the abovementioned tuning grid, we choose the model with the highest ROC and use the tuning parameters from this model when we predict on our test set. As can be seen in Figure 08, the highest ROC is achieved when we fit the model, using **1000** trees, a shrinkage of **0.01** and **9** d splits.

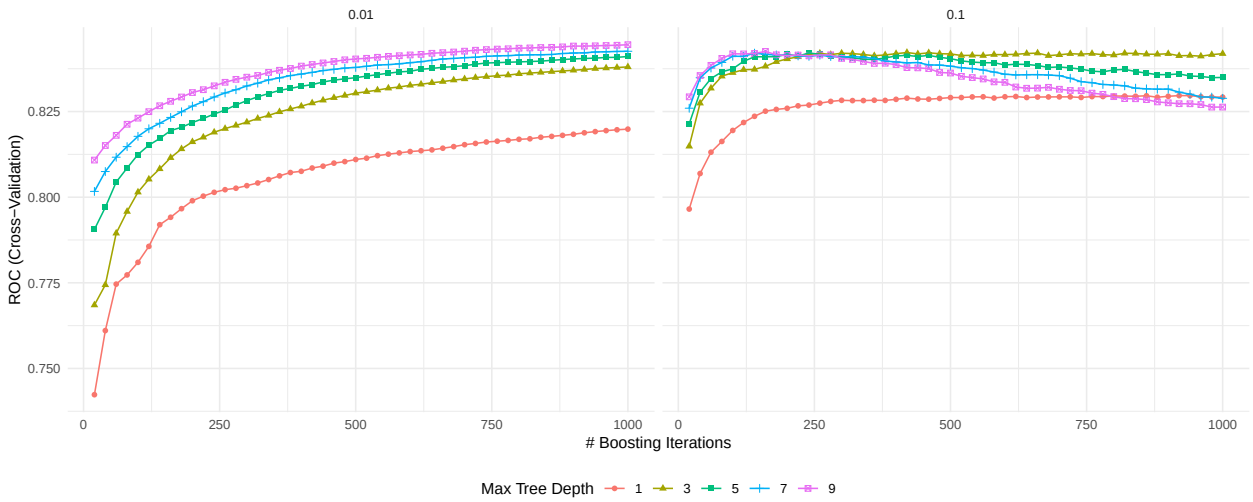

Figure 08: Fitting of GBM trees

When we fit the model on the test set, we get an accuracy of 82.87% and an AUC of 85.63%. From the predictions, we can further calculate the confusion matrix in (Appendix B) and the ROC curve can be seen in Figure 09.

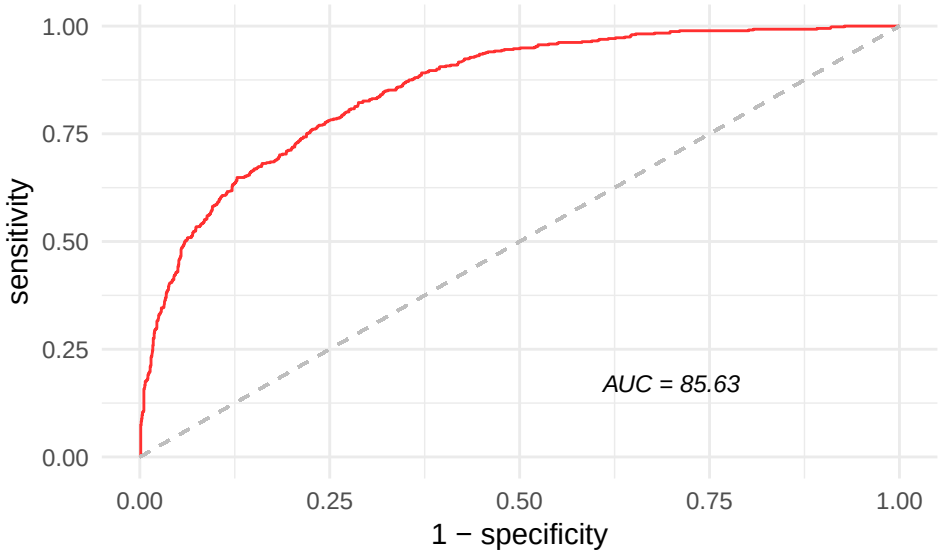

Figure 09: GBM ROC curve

Task D

In this section we will investigate what the different predictors say about the predictor's association with *High*, and whether a high-income individual has any particular features. To investigate this association, we will look at three of the most important variables, yielded from each model.

The methods seem to emphasize the importance of each variable differently, as shown in Figure 010. For instance, the Logistic regression analysis indicates that the *Education* variable is the most

important predictor while both the Linear discriminant analysis model and the Boosting method indicate that an individual's status is the most important. The most important predictor for each method gives intuitive sense at some level. For instance, it is reasonable that the level of one's education is an important predictor for the variable *High*. It is reasonable to assume that with higher level of education comes higher income. Consequently, with lower level of education comes lower level of income. In addition, education is included as top three most important predictor for every method, indicating its importance for predicting whether one has low or high level of income. Furthermore, the variable *Status* indicates whether a house is owned, rented or if the individual is living with parents/family. If one owns a house, it is reasonable to assume that the individual has a higher level of income compared to a person that rents a house or lives with parents/family.

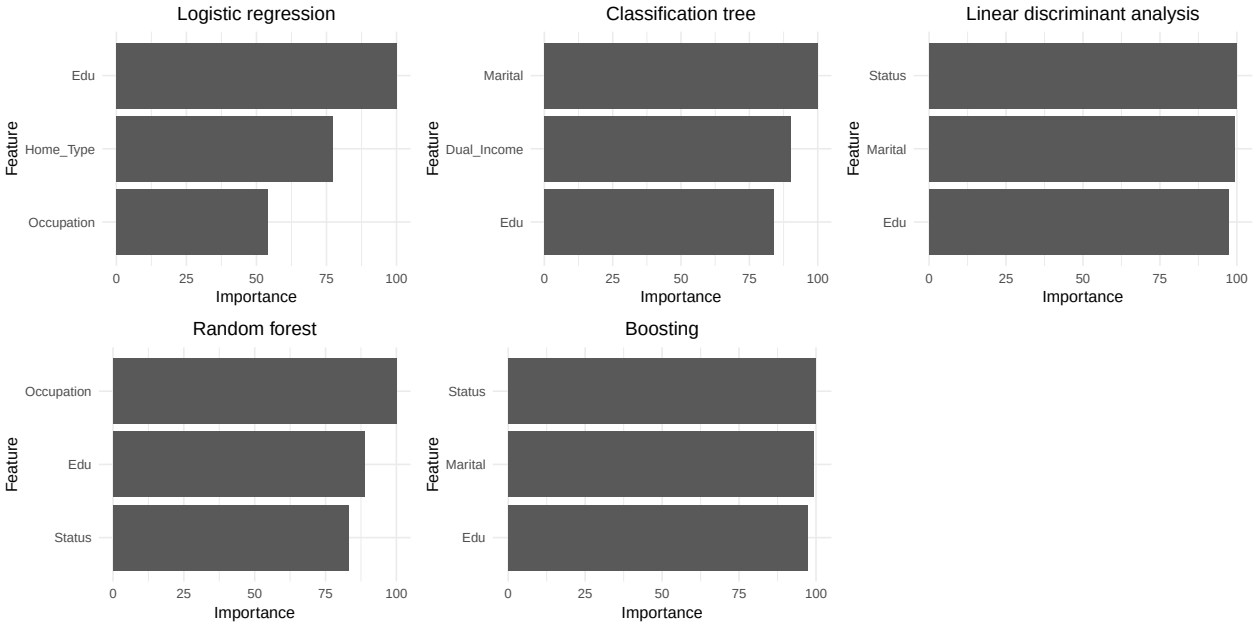

Figure 010: Variable importance

At first glance, the variable *Marital* seem to be less intuitive than *Education.* However, by calculating the correlation between the *Marital*- and *Income* variable, we see a negative and significant correlation. This indicates that when the levels for *Marital* classifications is high, a lower level of income is to be expected. A possible explanation might be that a person who is not married might be a young professionals or students and therefore has a lower income.

To investigate whether a high-income individual has any particular features, we will see the Figure 010 in relation to Figure 011. For Figure 011 we have only included the variables that ocurred two or more time in total. First, we see from the figure that when it comes to *Education*, the individuals with an education level of 5 and 6 seems to be highly represented in the high income level and individuals with a lower level of education seems to have a lower income. We deduct from Figure 011 that despite the fact that a high level of income accounts for 24.37% of the observations, more than 50% of the individuals that reported an education level of 6 have a higher income. Second, we see that the majority of high income individuals owns a house, as 1 in the *Status* variable indicates ownership of a house, while 2 and 3 represents renting and living with parents/family respectively. Thirdly, Figure 011 shows that the majority of indivduals with a high income tends to be married as 1 indicates marriage. Lastly, individuals that reported with a high level of income also tend to

report to having an occupation as a Professional/Managerial worker. Generally this kind of job occupation requires more coordination of the employees, and consequently these individual are often compensated accordingly. In conlusion, individual with high income tend to have education beyond high school, own a house, be married and have a job occupation as a Professional/Managerial.

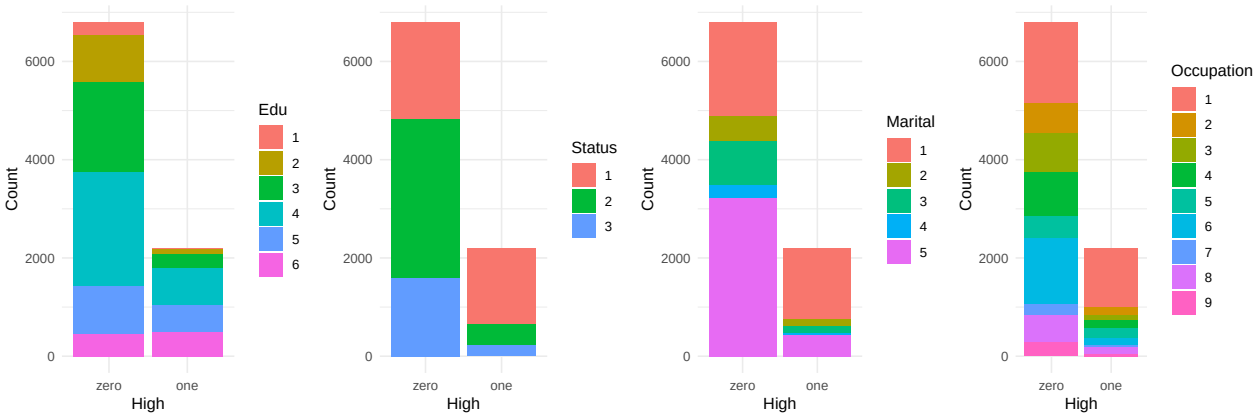

Figure 011: Features of the variable High

Task E

Conclusion

Throughout this paper, we have examined different prediction methods in order to predict the qualitative variable *High*. The methods have included logistic regression, linear discriminant analysis (LDA), classification trees with pruning, generalized boosted model (GBM) and random forest. We have used *10*-fold cross-validation when fitting the models and accuracy and AUC to evaluate the predictions. The different prediction methods have yielded different results, as shown in Table 02.

Table 02: Summary of metrics

	Accuracy	AUC
LDA	80.38	80.92
GBM	82.87	85.63
Logreg	80.78	81.01
Tree	82.65	76.49
Random Forest	82.61	84.08

From Table 02 we can see that the GBM model has both the highest accuracy and the highest AUC. However, the test errors from the random forest model, come fairly close to the test errors from the GBM model. We therefore note that a different train/test split might lead to a different conclusion.

Finally, care should be taken in interpreting the results in this paper. The training and test data are taken as samples from the same dataset, which may be subject to concern.

References

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *Datasets for "The Elements of Statistical Learning"*. Retrieved from https://web.stanford.edu/~hastie/ElemStatLearn/

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning – with applications in r* (Vol. 103). New York: Springer. https://doi.org/10.1007/DOI

Appendices

Appendix A: Dealing with missing values

Table 03: AUC and accuracy

	Include Lived, mean imputation		Remove Lived, na.omit		Include Lived, na.omit		Remove Lived, mean imputation	
	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC
LDA	80.38	80.92	80.41	80.93	79.17	80.05	80.29	80.98
GBM	82.87	85.63	82.37	84.87	80.51	84.55	82.78	85.68
Logreg	80.78	81.01	80.68	80.82	79.52	80.42	80.74	81.06
Tree	82.65	76.49	82.65	76.49	82.65	76.49	82.65	76.49
Random Forest	82.61	84.08	82.63	82.56	80.34	82.43	82.92	83.71

Appendix B: Confusion Matrix

Table 04: Confusion matrix

	LDA		GBM		Logreg		Tree		Random forrest	
	zero	one	zero	one	zero	one	zero	one	zero	one
one	134	245	114	281	128	248	104	266	113	274
zero	1562	307	1582	271	1568	304	1592	286	1583	278