HarvardCapstone2Report

webebe

20 4 2020

Introduction

This report describes the analysis steps performed on data containing observations of patients suffering or not suffering chronic kidney desease (CKD). The report consists of a methods section that explains the process and techniques used including data cleaning, data exploration and visualization, insights gained, and modeling approaches, results section that presents the modeling results and discusses the model performance and a conclusion section that gives a brief summary of the report, its limitations and future work.

The used dataset was downloaded from the UCI - Machine Learning Repository - https://archive.ics.uci.edu/ml/datasets/Chronic_Kidney_Disease . Credit goes to: Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

The dataset was obtained as follows:

```r
#Read Dataset
url_file<-"https://raw.githubusercontent.com/webebe/HarvardCapstone2/master/chronic_kidney_disease.txt"
chronic_kidney_disease<-read.csv(url(url_file))
```

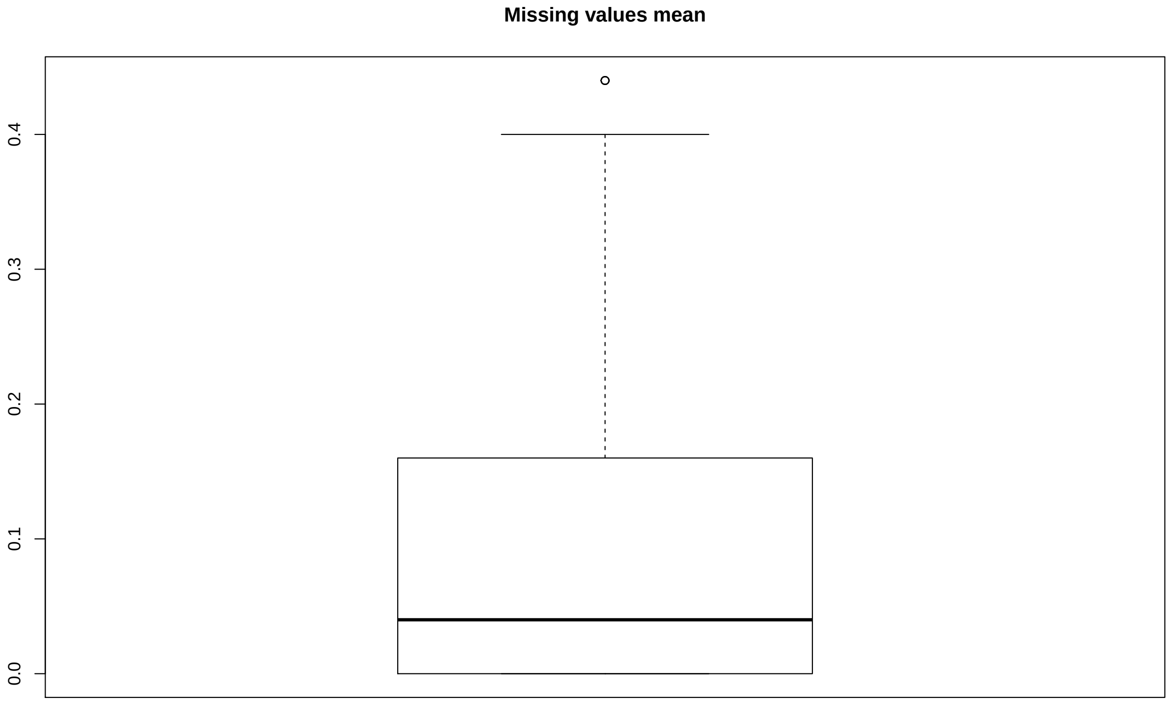
Methods

Data cleaning

Many observations include NA values in several variables and obvious wrong entries, therefor the dataset was cleaned. Variables where factorized where possible in order to ease application of algorithms and enhance interpretability later. It should be noted that data cleaning was mainly done manually - variables where inspected or plotted and subsequently adapted.

Dealing with missing values

The following decission was made: Observations with more than 16% of missing values were removed. It was assumed that those were observations where either not enough information as available or something went wrong when inputting the data. Further on mean substitution was chosen to handle missing values in continous variables, median substitution was applied to categorical variables. Mean and median values from the original dataset were used in order to avoid unwanted variations in the resulting analysis dataset. 3 variables were completely removed because of numerous missing values and no way of knowing how to submit them in a meaningful way.



```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0400  0.1009  0.1600  0.4400
```

One prediction variable caught special attention that is ‘rbc’ containing information about Red blood cells. Missing values were calculated from a different variable ‘rbcc’ the count of red blood cells. NA values in ‘rbc’ were filled with ‘normal’ if corresponding entry in ‘rbcc’ is closer to the mean of other ‘normal’ rbcc counts, otherwise it was categorized ‘abnormal’. Analysis of mean values of ‘rbc’ by group clearly shows that there is a difference of ‘normal’ and ‘abnormal’ cases.

```
#rbc ->calcualte from rbcc
range(which(dat$rbc=="normal"))
```

```
## [1]   3 400
```

```
range(which(dat$rbc=="abnormal"))
```

```
## [1]  10 249
```

```
mean(which(dat$rbc=="normal"))
```

```
## [1] 265.8159
```

```r
mean(which(dat$rbc=="abnormal"))
```

```
## [1] 127.2553
```

```r
#fill NA values in rbc with 'normal' if closer to the mean of other normal rbcc values
#and with 'abnormal'
dat_clean$rbc<-ifelse(is.na(dat_clean$rbc),
                ifelse(abs(dat_clean$rbcc-mean(which(dat$rbc=="normal")))>
                       abs(dat_clean$rbcc-mean(which(dat$rbc=="abnormal"))),
                       "2","3"),
                dat_clean$rbc)

#dat_clean$rbc
dat_clean$rbc<-ifelse(dat_clean$rbc=="2","abnormal","normal")%>%factor()
```

After manually investigating the dataset, some observations where removed due to lack of input in more than 2 variables. Refactorising variables after cleaning gets rid of unused factors.

After cleaning the original dataset the resulting analysis dataset consists of 300 observations of 21 variables. The ‘class’ variable serves as the outcome attribute and can form values of ‘ckd’ for patients actually suffering from CKD and ‘notcdk’ for patients who don’t.

```r
#After cleaning the dataset we end up with a set of 300 Observations of 21 variables.
abbreviation<-c("age","bp","sg","su","rbc","bgr","bu","sc","sod","pot","hemo",
                "pcv","wbcc","rbcc","htn","dm",
                "cad","appet","pe","ane","class")

vars<-c("age","blood pressure","specific gravity","sugar","red blood cells",
        "blood glucose random","blood urea","serum creatinine","sodium","potassium",
        "hemoglobin","packed cell volume",
        "white blood cell count","red blood cell count","hypertension","diabetes mellitus",
        "coronary artery disease","appetite","pedal edema","anemia","class")

tabeldat <- data_frame(abreviation=abbreviation, description = vars )
tabeldat %>% knitr::kable()
```

abreviation	description
age	age
bp	blood pressure
sg	specific gravity
su	sugar
rbc	red blood cells
bgr	blood glucose random
bu	blood urea
sc	serum creatinine
sod	sodium
pot	potassium
hemo	hemoglobin
pcv	packed cell volume
wbcc	white blood cell count
rbcc	red blood cell count
htn	hypertension

abreviation	description
dm	diabetes mellitus
cad	coronary artery disease
appet	appetite
pe	pedal edema
ane	anemia
class	class

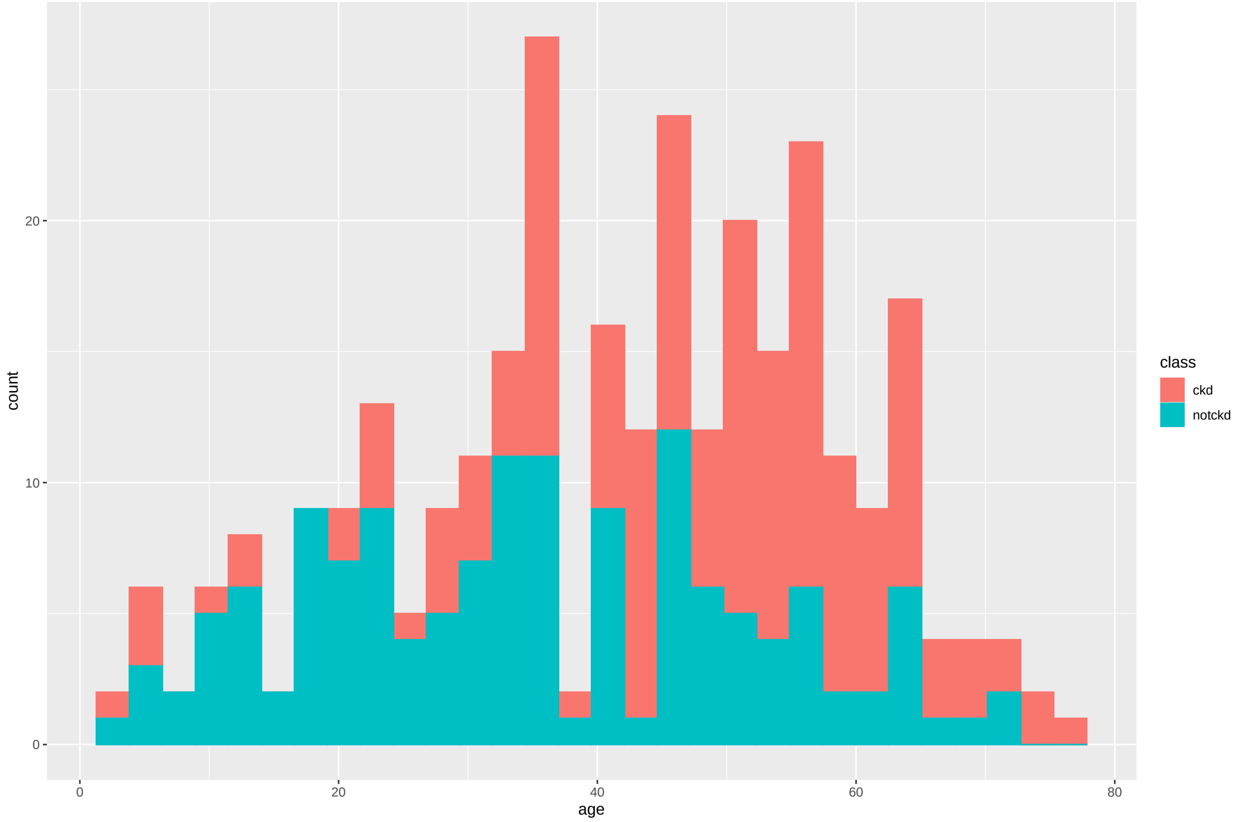
Data visualization

```
#summarize and visualize data
head(dat_clean)
```

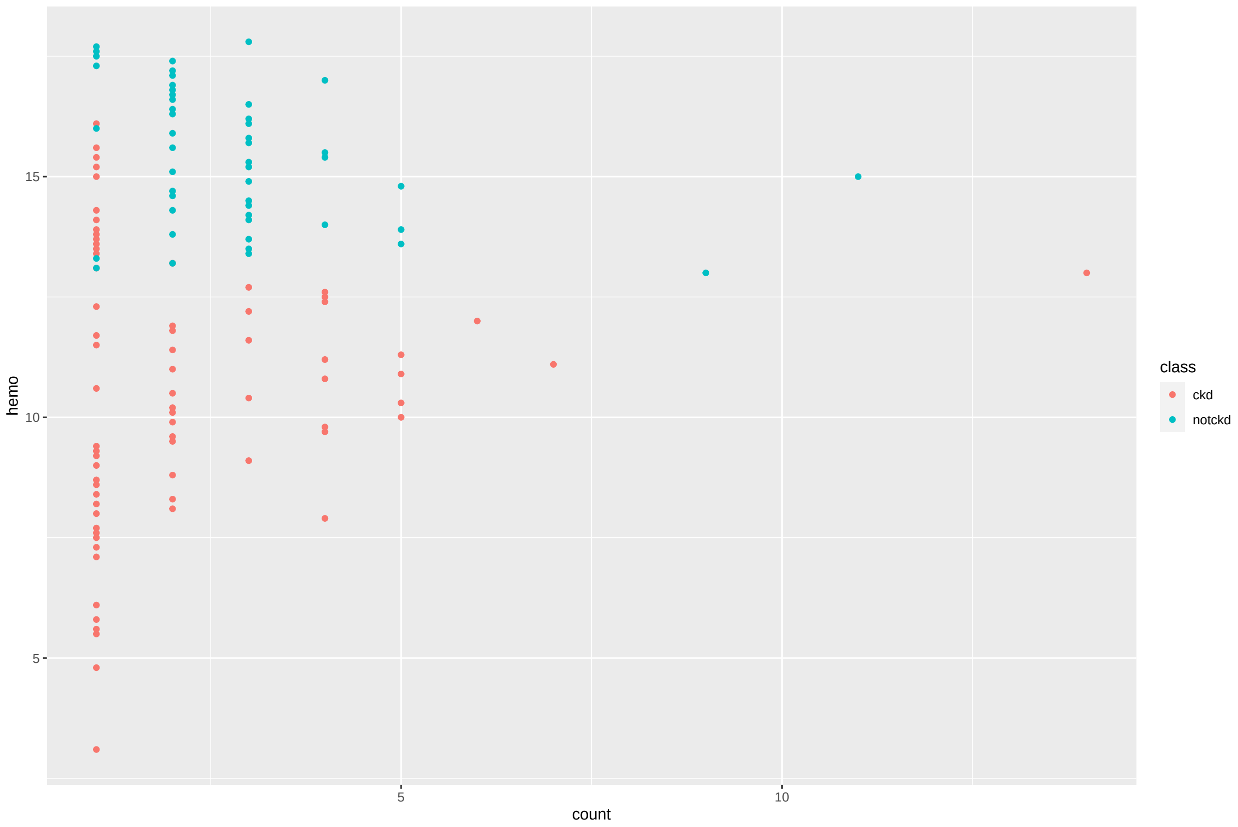
```
##   age bp    sg su      rbc bgr bu   sc sod pot hemo pcv wbcc rbcc htn  dm cad
## 1  37 80  1.02  0 abnormal 121 36  1.2 138 5.0 15.4  34   74   36 yes yes  no
## 3  53 80  1.01  3   normal 423 53  1.8 138 5.0  9.6  21   72   30  no yes  no
## 4  37 70 1.005  0   normal 117 56  3.8 111 2.5 11.2  22   64   21 yes  no  no
## 5  41 80  1.01  0   normal 106 26  1.4 138 5.0 11.6  25   70   29  no  no  no
## 6  51 90 1.015  0 abnormal  74 25  1.1 142 3.2 12.2  29   74   27 yes yes  no
## 7  59 70  1.01  0 abnormal 100 54 24.0 104 4.0 12.4  26   57   30  no  no  no
##   appet  pe ane class
## 1  good  no  no   ckd
## 3  poor  no yes   ckd
## 4  poor yes yes   ckd
## 5  good  no  no   ckd
## 6  good yes  no   ckd
## 7  good  no  no   ckd
```

Visual inspection and plotting of variables grouped by the class variable shows that some of the predictor variables might have higher prediction power than others. ‘Age’ for example doesn’t seem to be a very good predictor but ‘Hemo’ might very well be.

```
#age and CKD
dat_clean%>%ggplot(aes(x=age))+geom_histogram(aes(color=class,fill=class))
```



```
#hemo and CKD
dat_clean%>%ggplot(aes(y=hemo))+geom_point(aes(color=class,fill=class),stat="count")
```



Machine learning algorithms and prediction

The dataset will be split into a test and a training dataset. Algorithms will be trained on the train dataset and then applied to the test set. Performance of diffent model will be compared by accuracy, specificity and sensitivity.

```
#Create Test and training dataset
test_index<-createDataPartition(dat_clean$class,times=1,p=0.2,list = FALSE)
test_set<-slice(dat_clean[test_index,])
train_set<-slice(dat_clean[-test_index,])
train_x<-train_set[,-21]
test_x<-test_set[,-21]
train_y<-train_set[,21]
test_y<-test_set[,21]
#Check if Class distribution if even in the two subsets
mean(test_set$class=="ckd")
```

```
## [1] 0.5333333
```

```
mean(train_set$class=="ckd")
```

```
## [1] 0.5333333
```

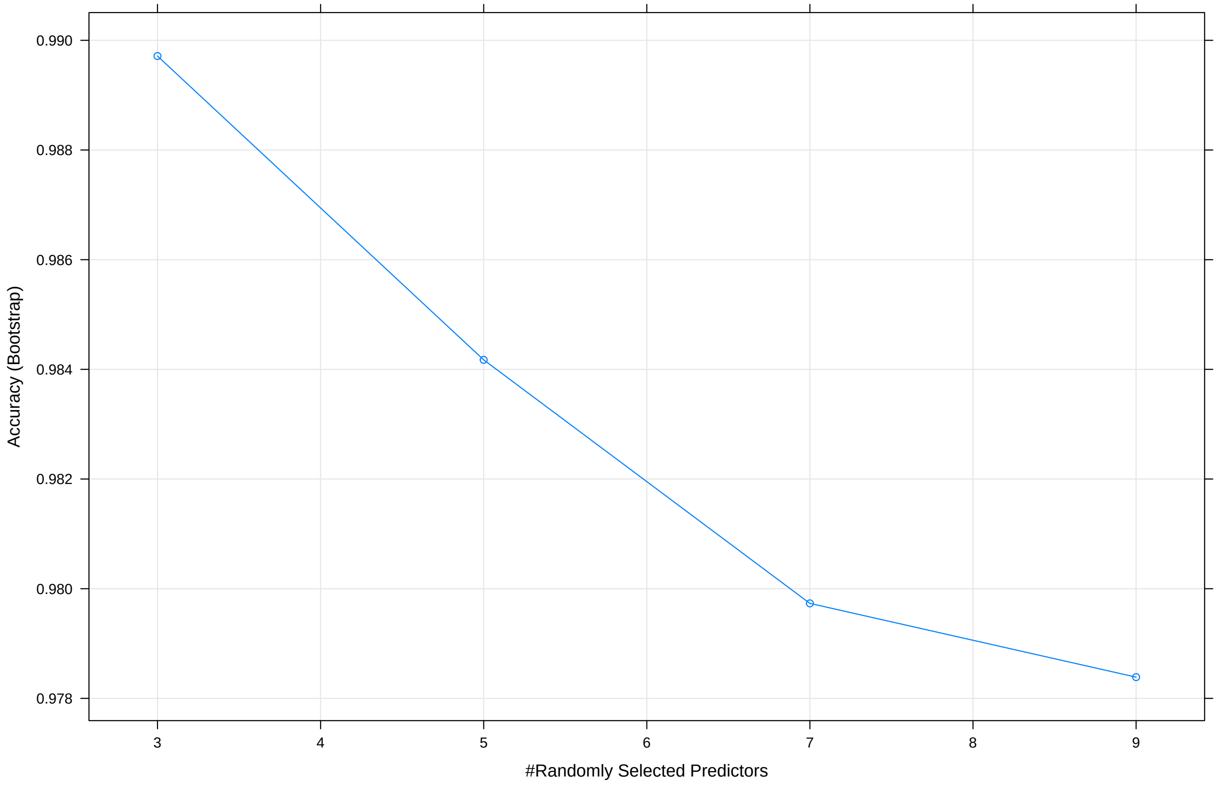
```r
##Logistic Regression
fit_logreg<-glm(as.numeric(train_set$class=="notckd")~.,data = train_set, family="binomial")
y_hat_logreg<-predict(fit_logreg,newdata = test_set,type="response")
y_hat_logreg<-ifelse(y_hat_logreg>0.5,"notckd","ckd")%>%factor()
acc_logreg<-confusionMatrix(y_hat_logreg,test_set$class)$overall["Accuracy"]
```

```r
##KNN
fit_knn <- knn3(train_y~.,data=train_set,k=5)
y_hat_knn<-predict(fit_knn, newdata = test_set, type="class")
acc_Knn<-confusionMatrix(y_hat_knn,reference = test_y)$overall["Accuracy"]
```

```r
##LDA
fit_lda<-train(class~.,method= "lda", data= train_set)
y_hat_lda<-predict(fit_lda,newdata = test_set)
acc_lda<-confusionMatrix(y_hat_lda,test_set$class)$overall["Accuracy"]
```

```r
##LOESS
fit_loess <- train(train_x,train_y,method="gamLoess")
y_hat_loess<-predict(fit_loess,test_x)
acc_loess<-confusionMatrix(y_hat_loess,test_y)$overall["Accuracy"]
```

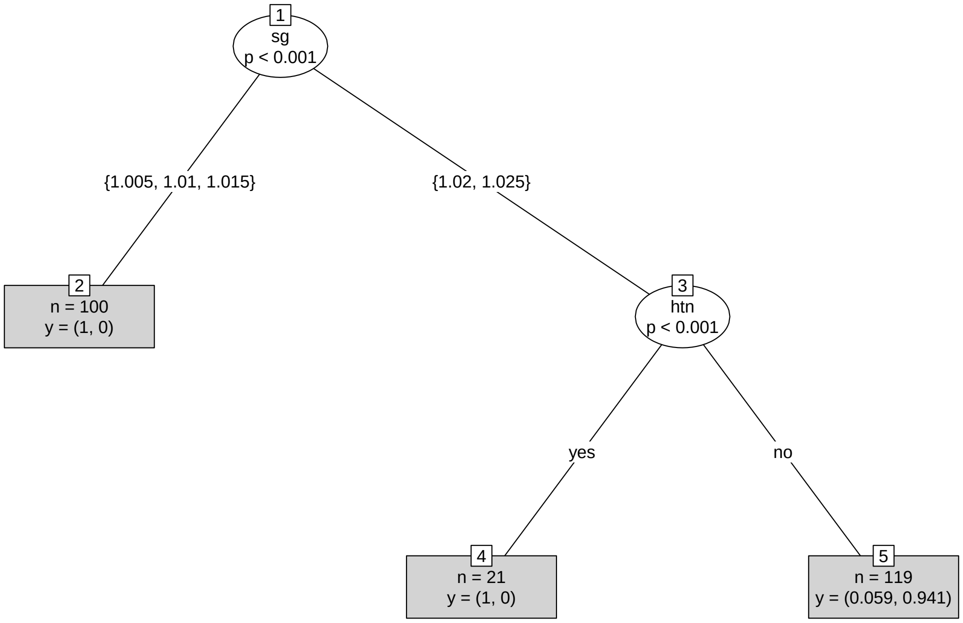
```r
##Random Forest
fit_rf<-train(train_x,train_y,method="rf",tuneGrid = data.frame(mtry = seq(3,9,2)),
                importance=TRUE)
plot(fit_rf)
```



```
y_hat_rf<-predict(fit_rf,test_x)
acc_rf<-confusionMatrix(y_hat_rf,test_y)$overall["Accuracy"]
#Variable importance helps with interpretation of archived results
varImp(fit_rf)
```

```
## rf variable importance
##
##       Importance
## hemo     100.000
## sg        94.511
## pcv       80.927
## rbc       70.338
## sc        67.649
## rbcc      58.641
## htn       56.840
## bgr       56.121
## dm        55.084
## appet     46.674
## sod       40.994
## pe        31.918
## su        29.377
## bu        27.784
## bp        27.444
## age       13.674
## ane       13.051
## wbcc       5.777
## cad        5.014
## pot        0.000
```

Visualisation of a random forest shows that the algorithm acctually only needs very few variables to correctly classify the data.



```r
##Random forest with cross validation to choose tuning parameter
fit_rf2<-train(class~.,method="Rborist",tuneGrid=data.frame(predFixed=2,minNode=c(3,21))
,data=train_set)
y_hat_rf2<-predict(fit_rf2,test_set)
acc_rf2<-confusionMatrix(y_hat_rf2,test_y)$overall["Accuracy"]
```

Results overview

Method	Overall_Accuracy
Logistic Regression	0.9833333
KNN	0.8500000
LDA	1.0000000
Loess	0.9833333
Random Forest	1.0000000
Random Forest Cross Validated	1.0000000

Conclusion

The chosen dataset might not be big enough to evaluate the prediction power of the applied algorithms but it is very convinient to work with on a computer with limited processing power. Additionally it was necessay to clean the dataset quite a bit in order to apply certain machine learning algorithms - that led to a even smaller dataset. However it should be noted that small datasets and data with missing values and wrong entries often are reality in data science. Therefor the analysis in this report form a quite realistic scenario. As far as the limitations mentioned above allow for a comparision of the applied models several machine learning algorithms perfom very well on the given task of predicting CKD. Logistic regression, KNN, LDA and Loess models achive high overall accuracy on the test data, methods like and Random forest even predict the correct class of all observations in the test dataset. This analysis could be extended by other classification algorithms and a combined ensemble method of all applied algorithms could be formed in order to further increase prediction power. E.g. the final classification of an observation could be formed by choosing the class which the majority of models predicted.