Stroke Prediction

HarvardX PH125.9x Capstone, Choose Your Own Project

Christoph Kuemmerli

30 3 2021

Table of Contents

1 Introduction

Stroke is a neurological disease that occurs when the oxygen supply to the vulnerable brain tissue is diminished. It is often disabilitating because brain tissue barely regenerates and about 3 % of the population is affected over a lifetime. Prevention therefore is of paramount importance. However, preventive measures like medication (primary prevention) may have side effects. Hence, risk-adjusted interventions should be applied. In this study, a machine learning algorithm to predict stroke in an adult population was trained and validated with the goal to predict stroke. The data used is publicly available on https://www.kaggle.com/fedesoriano/stroke-prediction-dataset. Data data will be imported, cleaned, explored and preprocessed. The modeling approach included logistic regression, random forest and k-nearest neighbor techniques. The predicted class was compared between models to find the best fit. As a performance metric, sensitivity will be given priority because we try not to miss a stroke at the expense of false positive results with the consequence of an overtreatment of people who will never have a stroke.

2 Methods

2.1 Relevant packages are installed and loaded.

```
if(!require(tidyverse)) install.packages("tidyverse", repos =
"http://cran.us.r-project.org")
if(!require(knitr)) install.packages("knitr", repos = "http://cran.us.r-
project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-
project.org")
if(!require(e1071)) install.packages("e1071", repos = "http://cran.us.r-
project.org")
if(!require(tableone)) install.packages("tablone", repos = "http://cran.us.r-
project.org")
if(!require(corrplot)) install.packages("corrplot", repos =
"http://cran.us.r-project.org")
if(!require(MLeval)) install.packages("MLeval", repos = "http://cran.us.r-
project.org")
if(!require(zoo)) install.packages("zoo", repos = "http://cran.us.r-
project.org")
if(!require(xts)) install.packages("xts", repos = "http://cran.us.r-
project.org")
if(!require(quantmod)) install.packages("quantmod", repos =
"http://cran.us.r-project.org")
if(!require(DMwR)) install.packages("https://cran.r-
project.org/src/contrib/Archive/DMwR/DMwR_0.4.1.tar.gz", repos=NULL,
type="source" )
if(!require(kernlab)) install.packages("kernlab", repos = "http://cran.us.r-
project.org")
if(!require(markdown)) install.packages("markdown", repos =
"http://cran.us.r-project.org")
if(!require(rmarkdown)) install.packages("rmarkdown", repos =
"http://cran.us.r-project.org")
```

2.2 Import from the github repository

```
data <-
read.csv("https://raw.githubusercontent.com/ckuemmerli/cyo_capstone/main/heal
thcare-dataset-stroke-data.csv")
```

The structure of the data and the first six observations are displayed below.

```
str(data)

## 'data.frame':    5110 obs. of  12 variables:
##  $ id                : int  9046 51676 31112 60182 1665 56669 53882 10434
27419 60491 ...
##  $ gender            : chr  "Male" "Female" "Male" "Female" ...
##  $ age               : num  67 61 80 49 79 81 74 69 59 78 ...
##  $ hypertension     : int  0 0 0 0 1 0 1 0 0 0 ...
##  $ heart_disease    : int  1 0 1 0 0 0 1 0 0 0 ...
##  $ ever_married     : chr  "Yes" "Yes" "Yes" "Yes" ...
##  $ work_type        : chr  "Private" "Self-employed" "Private" "Private"
...
##  $ Residence_type   : chr  "Urban" "Rural" "Rural" "Urban" ...
##  $ avg_glucose_level: num  229 202 106 171 174 ...
##  $ bmi              : chr  "36.6" "N/A" "32.5" "34.4" ...
##  $ smoking_status   : chr  "formerly smoked" "never smoked" "never smoked"
"smokes" ...
##  $ stroke           : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
head(data)
```

```
##      id gender age hypertension heart_disease ever_married     work_type
## 1  9046   Male  67            0             1          Yes       Private
## 2 51676 Female  61            0             0          Yes Self-employed
## 3 31112   Male  80            0             1          Yes       Private
## 4 60182 Female  49            0             0          Yes       Private
## 5  1665 Female  79            1             0          Yes Self-employed
## 6 56669   Male  81            0             0          Yes       Private
##   Residence_type avg_glucose_level  bmi  smoking_status stroke
## 1          Urban            228.69 36.6 formerly smoked      1
## 2          Rural            202.21  N/A    never smoked      1
## 3          Rural            105.92 32.5    never smoked      1
## 4          Urban            171.23 34.4          smokes      1
## 5          Rural            174.12   24    never smoked      1
## 6          Urban            186.21   29 formerly smoked      1
```

The codebook looks like this:
1) id: unique identifier
2) gender: “Male”, “Female” or “Other”
3) age: age of the patient
4) hypertension: 0 if the patient doesn’t have hypertension, 1 if the patient has hypertension
5) heart_disease: 0 if the patient doesn’t have any heart diseases, 1 if the patient has a heart disease
6) ever_married: “No” or “Yes”
7) work_type: “children”, “Govt_jov”, “Never_worked”, “Private” or “Self-employed”
8) Residence_type: “Rural” or “Urban”
9) avg_glucose_level: average glucose level in blood
10) bmi: body mass index
11) smoking_status: “formerly smoked”, “never smoked”, “smokes” or “Unknown”*
12) stroke: 1 if the patient had a stroke or 0 if not

The dataset contains 5110 observations with 12 variables. *Stroke* is the outcome or dependent variable.

2.3 Data cleaning

First, non-adults are excluded.

```
data <- filter(data, age >= 18)
```

Class and levels of variables are changed as appropriate.

2.3.1 Missing values

All variables are assessed for the percentage of missing values.

key	num.missing	percent_missing
bmi	181	4.254819

Only 201 values are missing for *bmi*.
For data exploration only, *bmi* is categorized according to the WHO definition. (1)

	normal	obese	overweight
stroke	36	97	75
no_stroke	867	1754	1244

The structure of the data now looks like the following.

```
str(data)

## 'data.frame':    4254 obs. of  13 variables:
##  $ id               : int  9046 51676 31112 60182 1665 56669 53882 10434
27419 60491 ...
##  $ gender           : Factor w/ 3 levels "Female","Male",..: 2 1 2 1 1 2 2
1 1 1 ...
##  $ age              : num  67 61 80 49 79 81 74 69 59 78 ...
##  $ hypertension     : Factor w/ 2 levels "hypertension",..: 2 2 2 2 1 2 1
2 2 2 ...
##  $ heart_disease    : Factor w/ 2 levels "heart_disease",..: 1 2 1 2 2 2 1
2 2 2 ...
##  $ ever_married     : Factor w/ 2 levels "married","not_married": 1 1 1 1
1 1 1 2 1 1 ...
##  $ work_type        : Factor w/ 4 levels "Govt_job","Never_worked",..: 3 4
3 3 4 3 3 3 3 3 ...
##  $ Residence_type   : Factor w/ 2 levels "Rural","Urban": 2 1 1 2 1 2 1 2
1 2 ...
##  $ avg_glucose_level: num  229 202 106 171 174 ...
##  $ bmi              : num  36.6 NA 32.5 34.4 24 29 27.4 22.8 NA 24.2 ...
##  $ smoking_status   : Factor w/ 4 levels "formerly smoked",..: 1 2 2 3 2 1
2 2 4 4 ...
```

```
##  $ stroke           : Factor w/ 2 levels "stroke","no_stroke": 1 1 1 1 1 1
1 1 1 1 ...
##  $ bmi_class        : Factor w/ 3 levels "normal","obese",..: 2 NA 2 2 1 3
3 1 NA 1 ...
```

2.4 Split the dataset

First create the data partition based on the dependent variable *stroke* and 80% of observations will be used to train the algorithm. Due to the number of observations, a ratio of 80:20 for train and test dataset shouldn't cause problems.

```
set.seed(123, sample.kind="Rounding") # if using R 3.5 or earlier, use
`set.seed(1)`

test_index <- createDataPartition(data$stroke,
                                  times = 1,
                                  p = 0.2,
                                  list = FALSE)
```

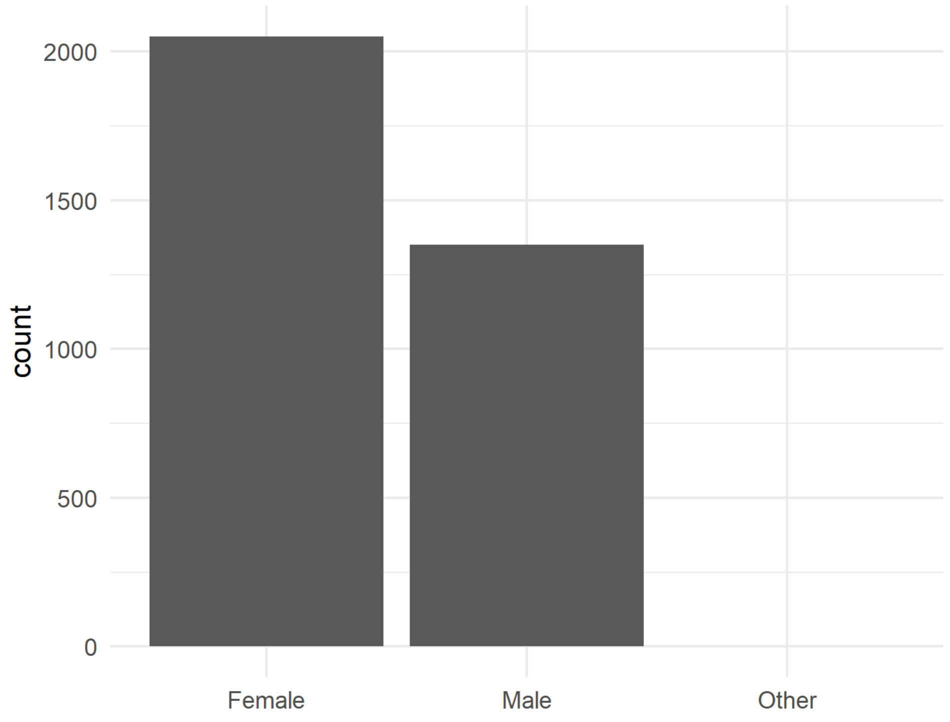
Split the datasets into train and test set.

```
trainset <- data[-test_index,]

testset <- data[test_index,]
```

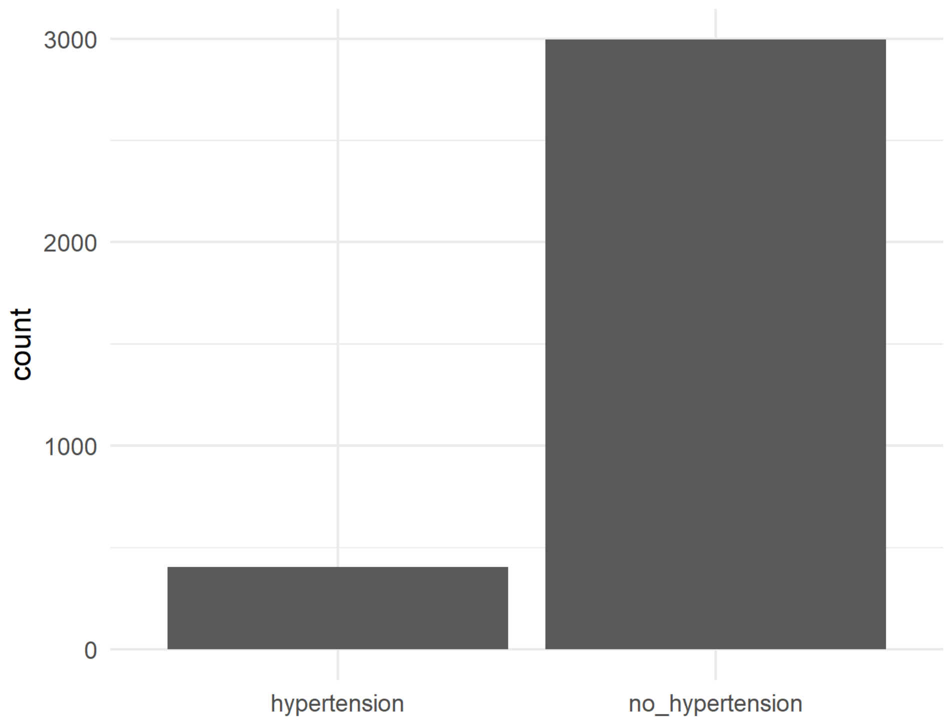
2.5 Data exploration

Using the appropriate class of the objects, a summary is displayed. In addition, barplots for all factor variables are created.

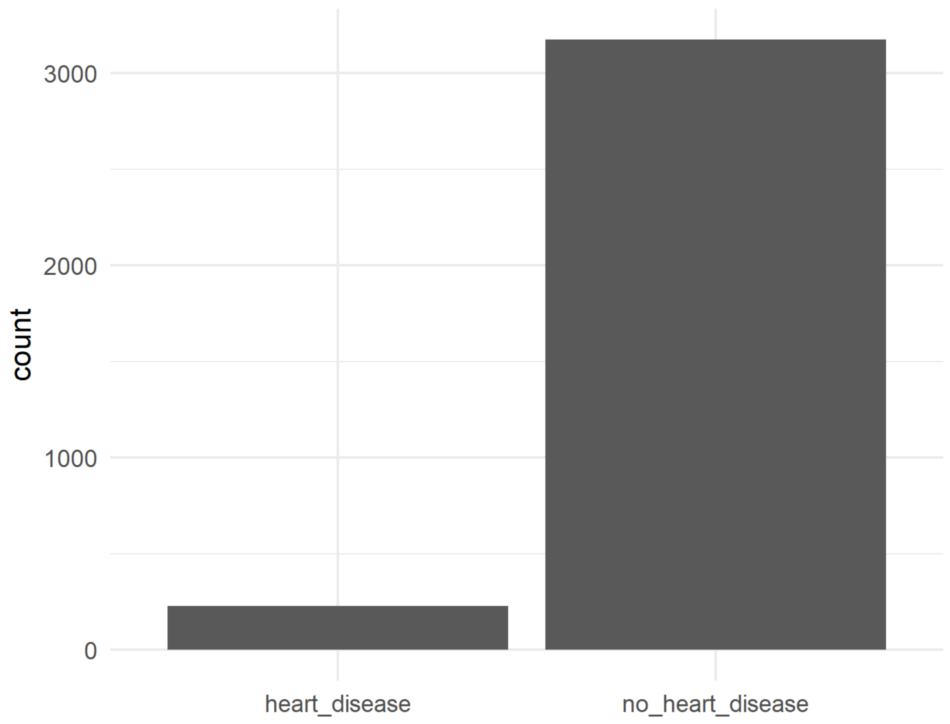
```
##     gender          age                  hypertension
heart_disease
##  Female:2051   Min.   :18.0   hypertension   : 406   heart_disease   : 227
##  Male  :1350   1st Qu.:36.0   no_hypertension:2996   no_heart_disease:3175
##  Other :   1   Median :51.0
##                Mean   :50.3
##                3rd Qu.:64.0
##                Max.   :82.0
##
##       ever_married          work_type    Residence_type avg_glucose_level
##  married    :2706   Govt_job     : 523   Rural:1665     Min.   : 55.12
##  not_married: 696   Never_worked :   5   Urban:1737     1st Qu.: 77.93
##                     Private      :2232                  Median : 92.89
##                     Self-employed: 642                  Mean   :108.53
##                                                         3rd Qu.:116.42
##                                                         Max.   :271.74
##
##       bmi                  smoking_status       stroke          bmi_class
##  Min.   :11.30   formerly smoked: 674   stroke   : 197   normal    : 718
##  1st Qu.:25.50   never smoked   :1404   no_stroke:3205   obese     :1476
##  Median :29.20   smokes         : 625                    overweight:1066
##  Mean   :30.45   Unknown        : 699                    NA's      : 142
##  3rd Qu.:34.20
##  Max.   :92.00
##  NA's   :142

## $gender
```



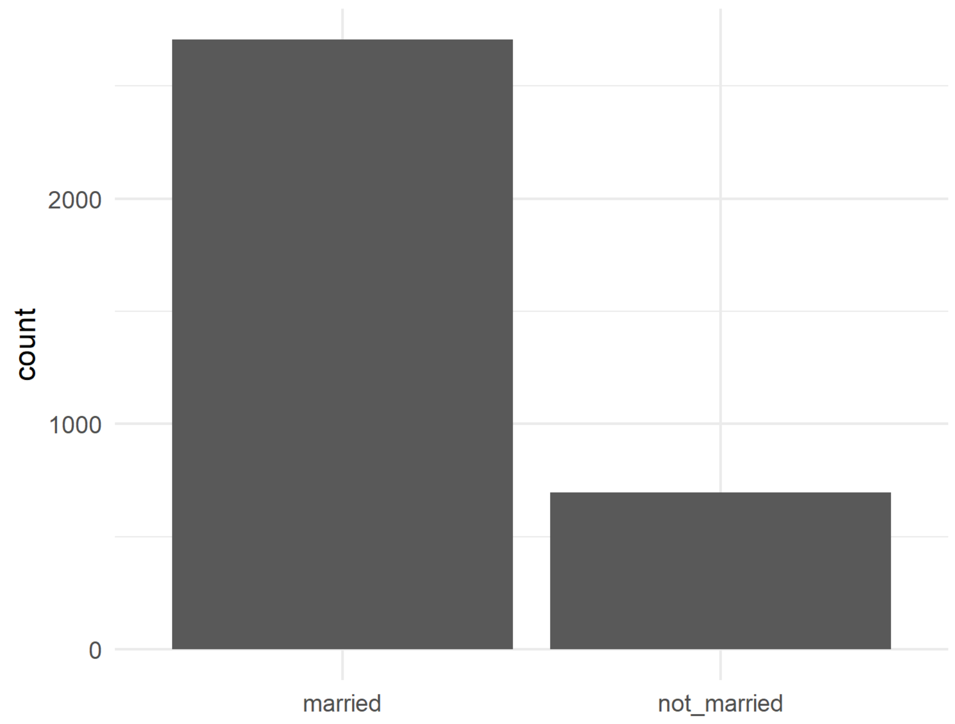
```
##
## $hypertension
```



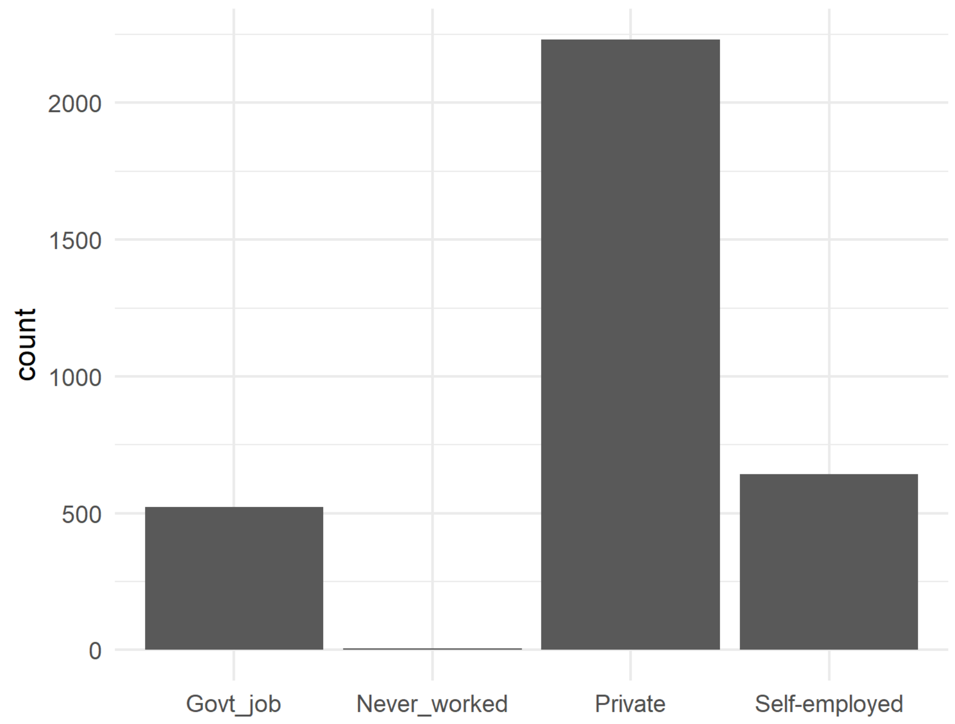
```
##
## $heart_disease
```



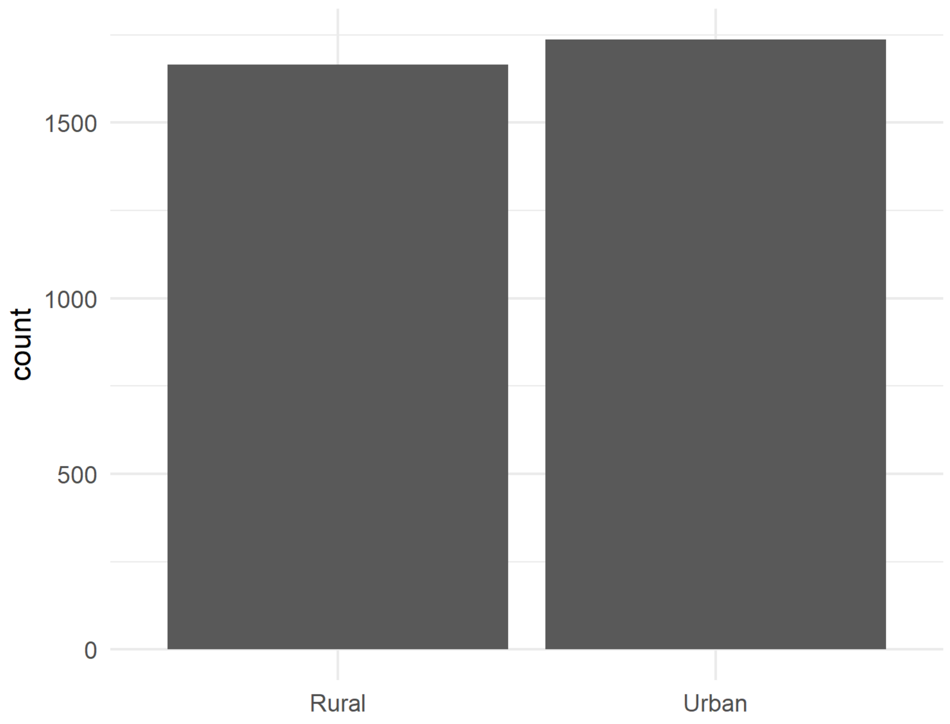
```
##
## $ever_married
```



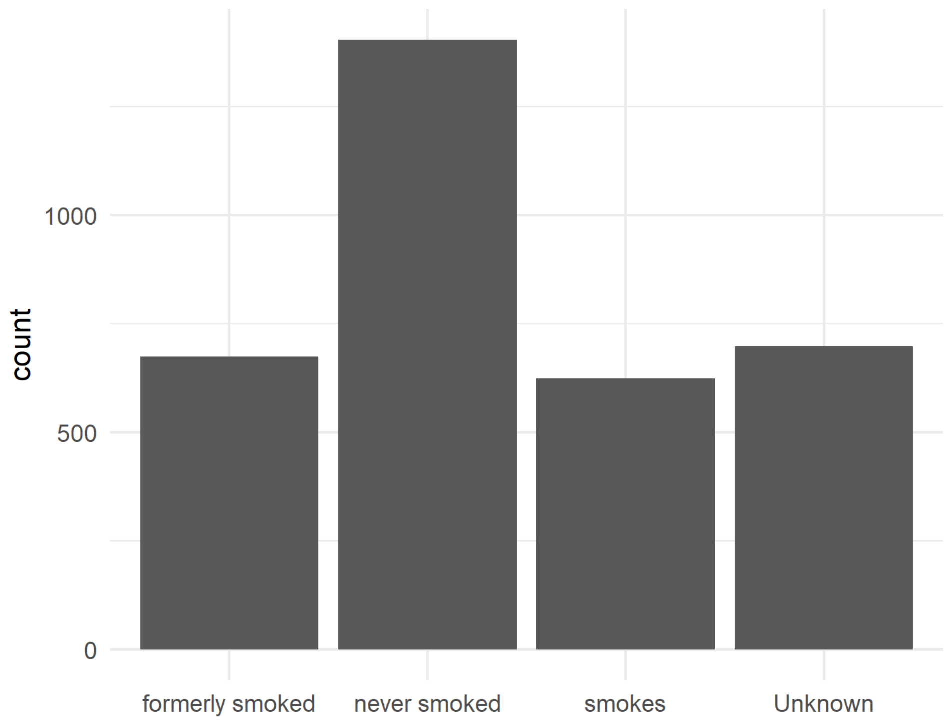
```
##
## $work_type
```



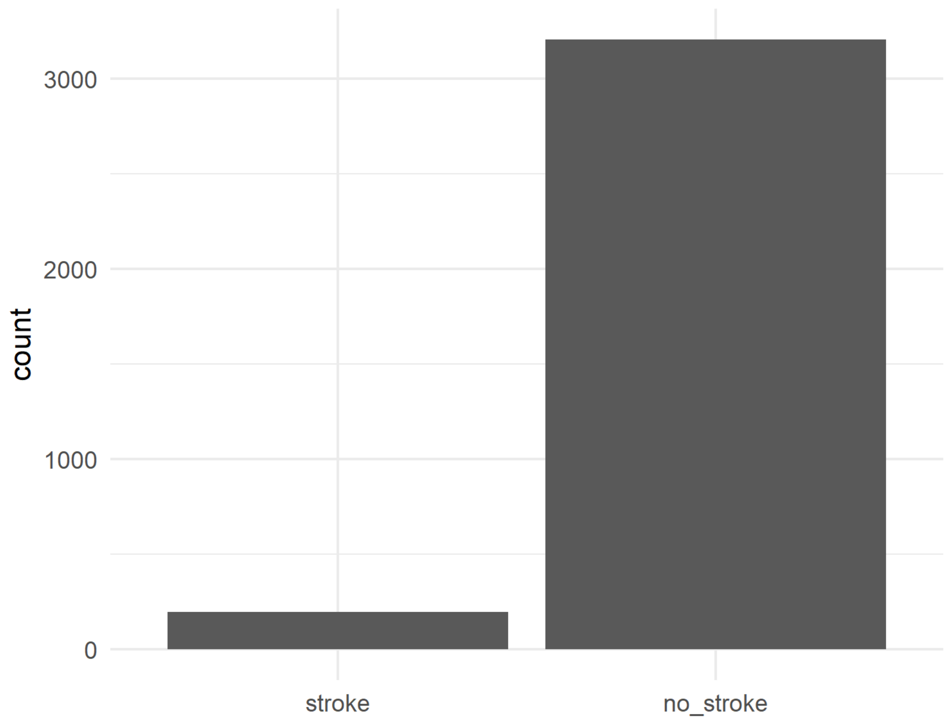
```
##
## $Residence_type
```



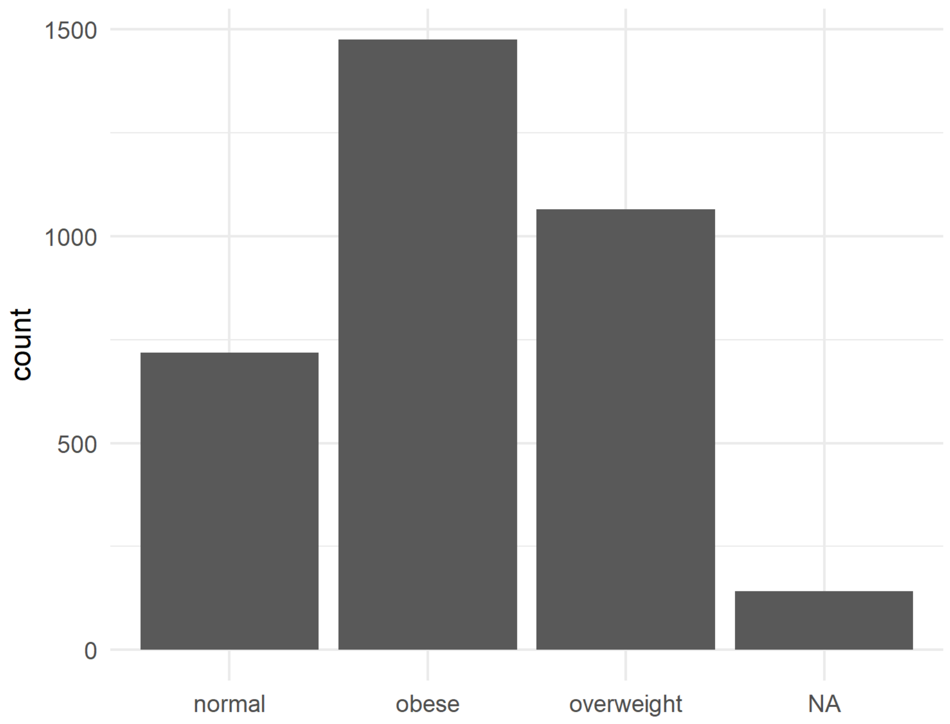
```
##
## $smoking_status
```



```
##
## $stroke
```



```
##
## $bmi_class
```



The dependent variable *stroke* is imbalanced, meaning that the two possible outcome (developing a *stroke* or not) have a different prevalence in this dataset. This will cause problems later during model building and has to be addressed. Apart from the imbalanced dependent variable, also heart disease has a low prevalence. To assess the necessity of imputation, the prevalence of *stroke* in the incomplete cases (with missing *bmi*) is checked.

```
##     n
## 1  39
```

These observations have a high prevalence of *stroke*. Hence, due to the low prevalence of *stroke* in the entire dataset, we want to keep these observations. Therefore, imputation is carried out after splitting the dataset into train and test set.

2.6 SMOTE (Synthetic Minority Oversampling Technique)

To address the issue of imbalance with “no stroke” being much more prevalent and *stroke* a rare event, synthetic minority oversampling technique (SMOTE) is introduced. SMOTE is an approach to the construction of classifiers from imbalanced datasets. A dataset is imbalanced if the classification categories are not equally represented (e.g. 50 % of observations have a stroke and the others don’t). Data sets are often composed of “normal” examples with only a small percentage of “abnormal” or “interesting” examples, e.g. having a stroke. It is also the case that the cost of misclassifying an “interesting” example as a normal example is often much higher than the cost of the reverse error, i.e. to classify a normal example as an “interesting” one. Often, one speaks of a “minority” and “majority” class. The minority class is underrepresented (stroke in this dataset). To balance the data, several options are available. The first is gather more data, surely the preferable approach, but often not possible. Resampling is another option. In this case, under-sampling of the majority class or over-sampling of the minority class is carried out. SMOTE generates new minority instances between existing instances. The new instances created are not just a copy of existing minority cases. The algorithm takes sample of feature space for each target class and its neighbors and then generates new instances that combine the features of the target cases with features of its neighbors.

SMOTE takes the entire dataset as an input, but it increases the percentage of only the minority cases.

Other approaches are up or downsampling. We could also use subsampling inside the train function and like below, outside the train function.

2.7 Confounders

A confounder is a variable that influences both the dependent variable or outcome variable and the independent variable or feature. Based on the exploratory analysis, age may influence som features.

work_type	mean
Govt_job	51.21989

Never_worked	19.20000
Private	47.23073
Self-employed	60.43925

hypertension	mean
hypertension	62.54926
no_hypertension	48.63485

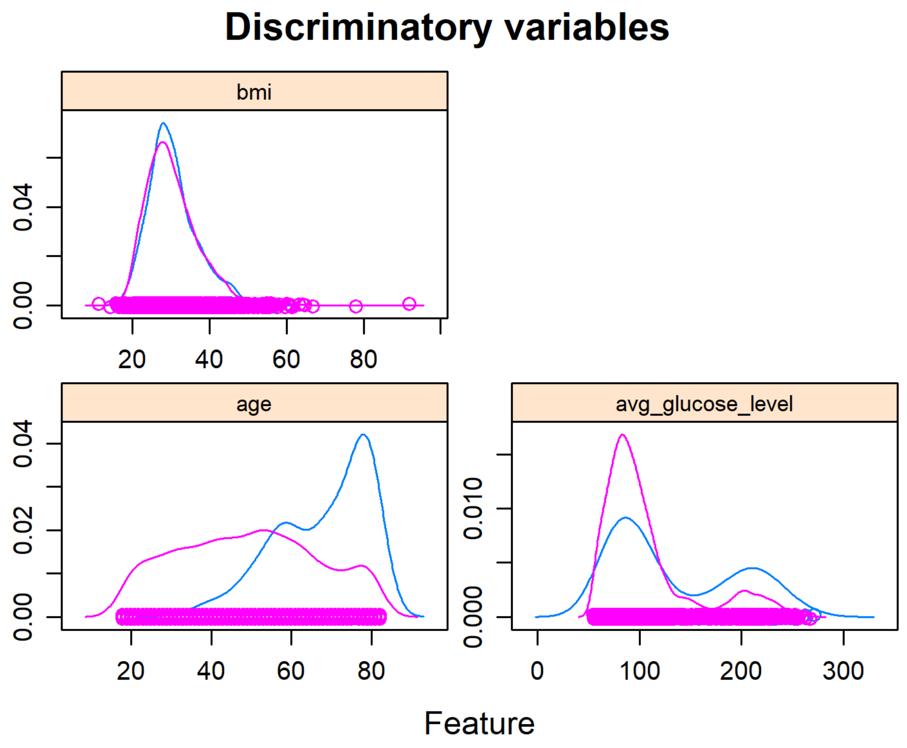
Residence_type	mean
Rural	49.82042
Urban	50.75072

smoking_status	mean
formerly smoked	56.74481
never smoked	49.37607
smokes	47.15840
Unknown	48.72818

As suspected, age influences features. Older people are more often self-employed, have hypertension and quit smoking. Our model will perhaps perform better if these variables will be excluded later.

2.8 Discrimination

The discriminatory ability of numeric predictor variables is assessed with feature plots.



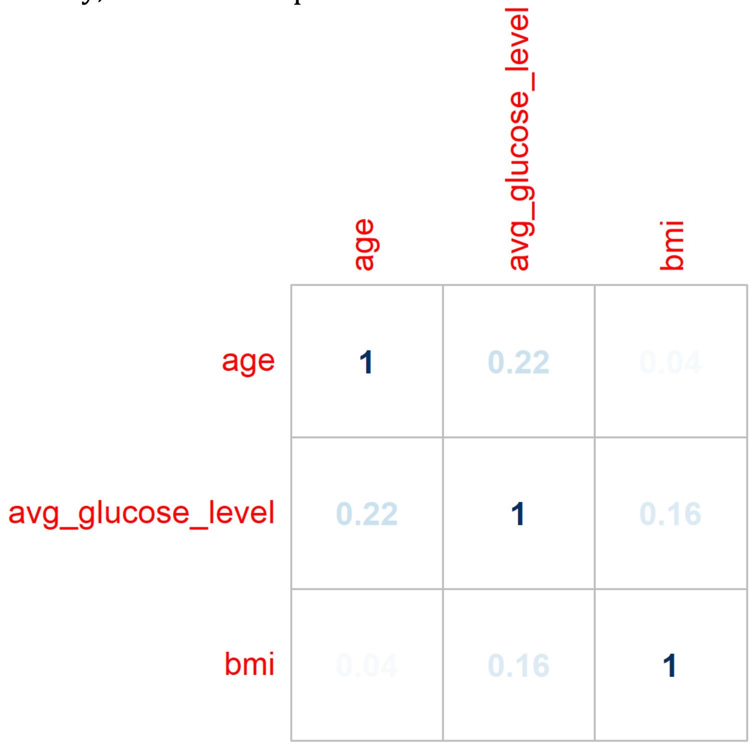
Age seems to have the best discriminatory ability to distinguish between patient who develop a stroke and people who don't. We would expect a higher prevalence of stroke in older people due to the higher prevalence of cardiovascular risk factors in this population (see also confounding above).

When comparing all features stratified by developing the outcome *stroke*, this table results.

```
##                                       Stratified by stroke
##                                        Overall       stroke
##   n                                     3402          197
##   gender (%)
##      Female                             2051 (60.3)   104 (52.8)
##      Male                               1350 (39.7)    93 (47.2)
##      Other                                 1 ( 0.0)     0 ( 0.0)
##   hypertension = no_hypertension (%)    2996 (88.1)   146 (74.1)
##   heart_disease = no_heart_disease (%)  3175 (93.3)   161 (81.7)
##   ever_married = not_married (%)         696 (20.5)    21 (10.7)
##   work_type (%)
##      Govt_job                            523 (15.4)    27 (13.7)
##      Never_worked                          5 ( 0.1)     0 ( 0.0)
##      Private                            2232 (65.6)   120 (60.9)
##      Self-employed                       642 (18.9)    50 (25.4)
##   Residence_type = Urban (%)            1737 (51.1)   109 (55.3)
##   smoking_status (%)
##      formerly smoked                      674 (19.8)      57 (28.9)
##      never smoked                        1404 (41.3)      71 (36.0)
##      smokes                               625 (18.4)      32 (16.2)
##      Unknown                              699 (20.5)      37 (18.8)
##   bmi_class (%)
##      normal                               718 (22.0)      29 (17.3)
##      obese                               1476 (45.3)      76 (45.2)
##      overweight                          1066 (32.7)      63 (37.5)
##   age (mean (SD))                       50.30 (17.67)  67.79 (11.64)
##   bmi (mean (SD))                       30.45 (7.22)   30.46 (6.46)
##   avg_glucose_level (mean (SD))        108.53 (47.15) 134.74 (61.76)
##                                        Stratified by stroke
##                                         no_stroke      p      test
##   n                                      3205
##   gender (%)                                            0.082
##      Female                              1947 (60.7)
##      Male                                1257 (39.2)
##      Other                                  1 ( 0.0)
##   hypertension = no_hypertension (%)     2850 (88.9)  <0.001
##   heart_disease = no_heart_disease (%)   3014 (94.0)  <0.001
##   ever_married = not_married (%)          675 (21.1)   0.001
##   work_type (%)                                         0.108
##      Govt_job                             496 (15.5)
##      Never_worked                           5 ( 0.2)
##      Private                             2112 (65.9)
##      Self-employed                        592 (18.5)
##   Residence_type = Urban (%)             1628 (50.8)   0.245
##   smoking_status (%)                                    0.012
##      formerly smoked                      617 (19.3)
##      never smoked                        1333 (41.6)
##      smokes                               593 (18.5)
##      Unknown                              662 (20.7)
##   bmi_class (%)                                         0.215
##      normal                               689 (22.3)
##      obese                               1400 (45.3)
##      overweight                          1003 (32.4)
##   age (mean (SD))                       49.22 (17.41) <0.001
##   bmi (mean (SD))                       30.45 (7.26)   0.990
##   avg_glucose_level (mean (SD))        106.92 (45.62) <0.001
```

2.9 Collinearity

To assess collinearity, a correlation plot for numeric variables is used.



There is no relevant correlation between the features.

2.10 Coding

Now, categories with few subjects are removed and manual dummy coding is carried out. We do this separately with the train and test datasets.

2.11 Feature selection

Based on exploration and clinical knowledge, the features are selected. The class is changed to numeric.

```
trainset <- select(trainset,
                   gender,
                   age,
                   hypertension,
                   work_type,
                   avg_glucose_level,
                   bmi,
                   smoking_status,
                   stroke)

trainset[, 1:7] <- sapply(trainset[, 1:7], as.numeric)
```

```
testset <- select(testset,
                  gender,
                  age,
                  hypertension,
                  work_type,
                  avg_glucose_level,
                  bmi,
                  smoking_status,
                  stroke)

testset[, 1:7] <- sapply(testset[, 1:7], as.numeric)
```

2.12 Imputation

Because we want to keep the many events in the group with missing bmi values, mean imputation, separately for train and test dataset, is performed.

```
trainset$bmi[is.na(trainset$bmi)] <- mean(trainset$bmi, na.rm = T)
testset$bmi[is.na(testset$bmi)] <- mean(testset$bmi, na.rm = T)
```

Remove objects no longer needed.

```
rm(data, test_index, missing_values, test_index, corr)
```

The machine learning algorithms used are logistic regression, random forest, k-nearest neighbor and support vector machine (SVM). The first three have been introduced and discussed in earlier courses. The SVM however has not been used in this programme so far and that why an explanation of the concept of this algorithm is provided.
For a dataset consisting of features set and labels set, an SVM classifier builds a model to predict classes for new examples. It assigns new data points to one of the classes. If there are only 2 classes then it can be called as a binary SVM classifier. There are 2 kinds of SVM classifiers, linear SVM classifier and non-linear SVM classifier. In the linear classifier model, training examples are assumed to be plotted in space. These data points are expected to be separated by an apparent gap. It predicts a straight hyperplane dividing the two classes. The primary focus of the hyperplane is to maximize the distance from hyperplane to the nearest data point of either class. When a straight line is not considered appropriate to separate the two classes, a kernel can be applied (e.g. polynomial or radial basis function) and the hyperplane is no longer linear (non-linear SVM).

Performance measures have been discussed in previous courses and the following are used for this two class outcome: Sensitivity or Recall, Specificity, Precision, Receiver Operating characteristics (ROC) curve and the Area under the curve (AUC)

3 Results

3.1 Building models

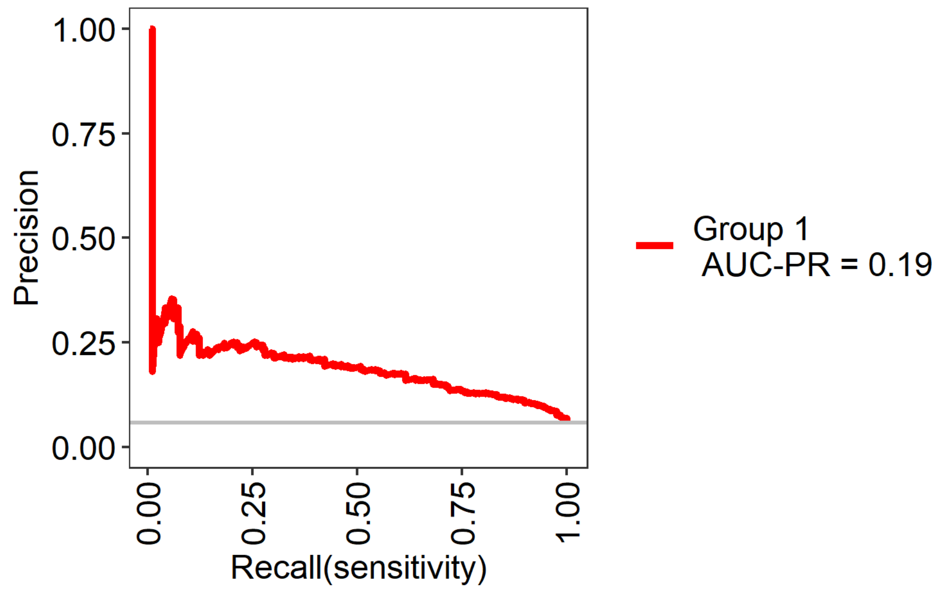
After the trainset is ready for use, the models can be built. First, 10-fold cross validation is applied and because we want to use sensitivity, specificity and ROC with AUC to assess performance, class probabilities and the twoClassSummary is added.

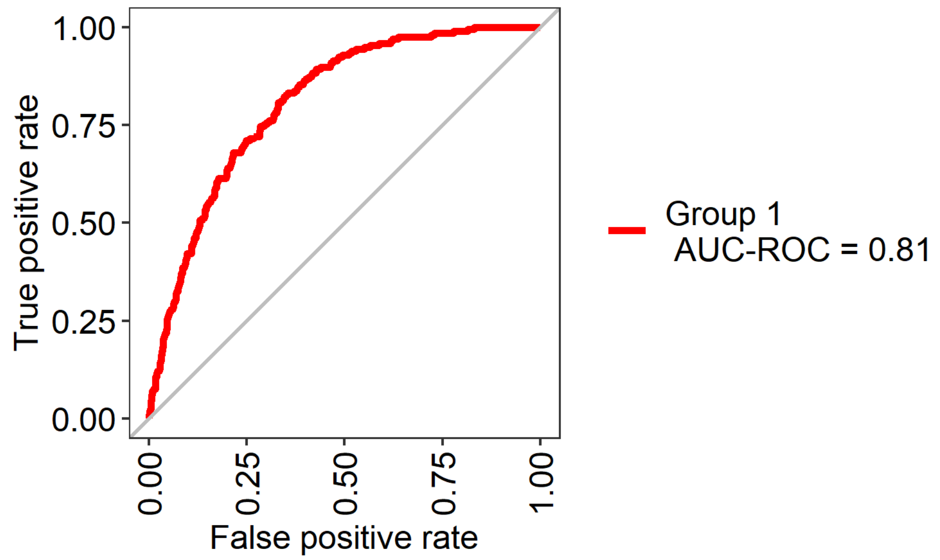
```
ctrl <- trainControl(
  method = "cv",
  number = 10,
  savePredictions = "final",
  classProbs = T,
  summaryFunction = twoClassSummary)
```

3.1.1 Model 1

Logistic regression using the caret package and the train function. This algorithm is chosen to start with because it is well-known and often used to solve classification problems. Due to different ranges of age, glucose and bmi, data is centered and scaled for this and all other models. Because we will later use the AUC to assess model performance, the metric *ROC* is added. The data is preprocessed due to the different range of *age*, *glucose* and *bmi*.

```
set.seed(123, sample.kind="Rounding")
glmFit <- train(stroke ~ ., data = trainset,
                method = "glm",
                family = "binomial",
                trControl = ctrl,
                preProcess = (c("center", "scale")),
                metric = "ROC")
evalm(glmFit, positive = "stroke", plots = c("r", "pr"))$pr
```





```
## NULL
```

The accuracy of the model is high but because the predictive accuracy is solely based on the majority class (*no stroke*) we miss all or almost all patients with a *stroke* and the sensitivity is zero or near zero.

SMOTE is used to balance the outcome prevalence for training.

Var1	Freq
stroke	591
no_stroke	788

Now, prevalence of the positive outcome *stroke* is higher and more balanced.

Logistic regression is again fitted using the the smote dataset.

```
set.seed(123, sample.kind="Rounding")
glmFit2 <- train(stroke ~ ., data = smote_train,
                 method = "glm",
                 family = "binomial",
                 trControl = ctrl,
                 preProcess = (c("center", "scale")),
                 metric = "ROC")
```

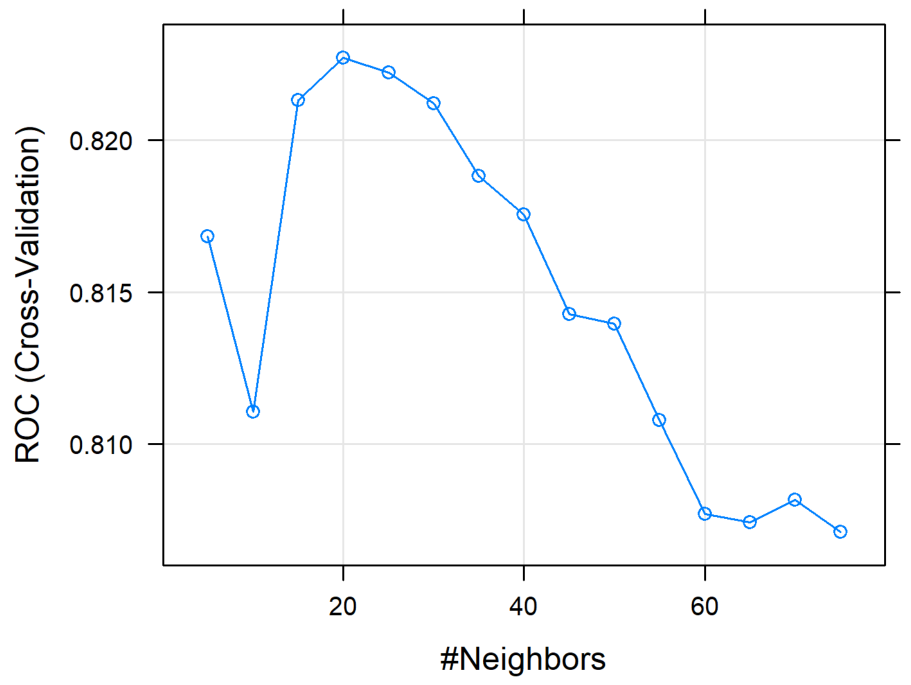
3.1.2 Model 2

Random forest with the tuning parameter mtry set from 1 to 20.

```
##   mtry
## 3    3
```

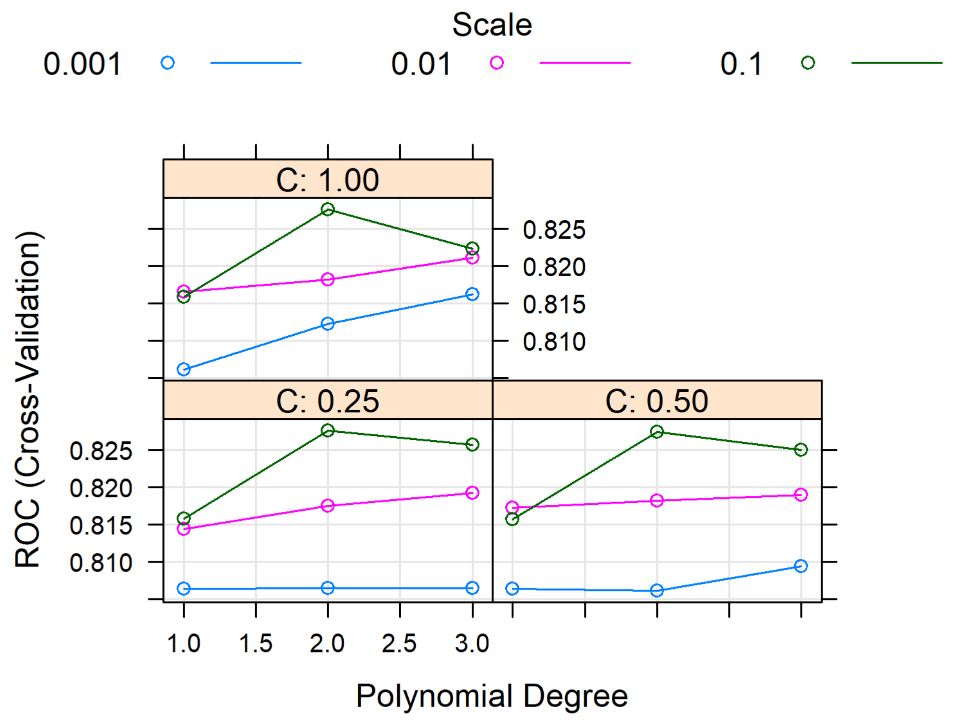
3.1.3 Model 3

k-nearest neighbour, the tuning parameter k is set from 5 to 75 with an increment of 5.



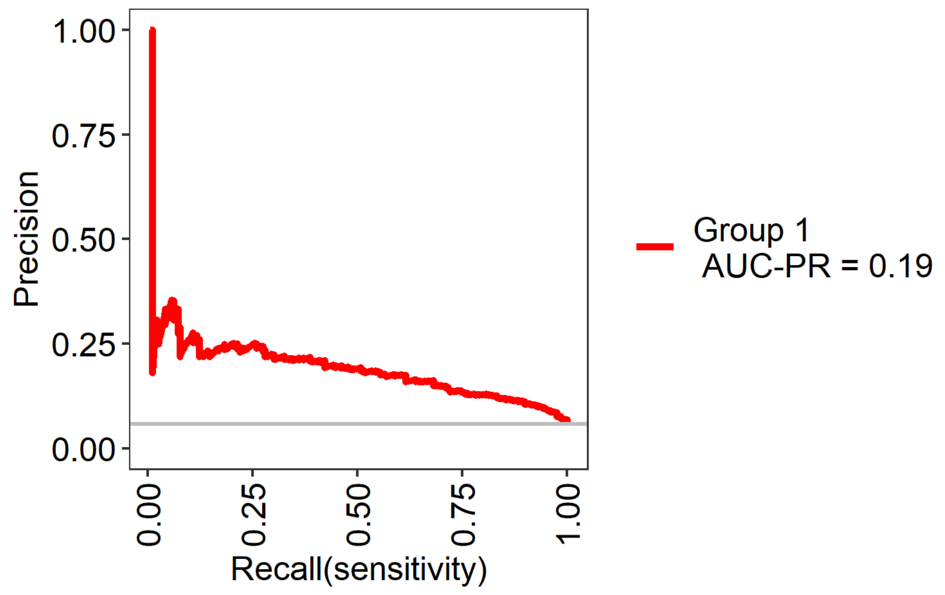
3.1.4 Model 4

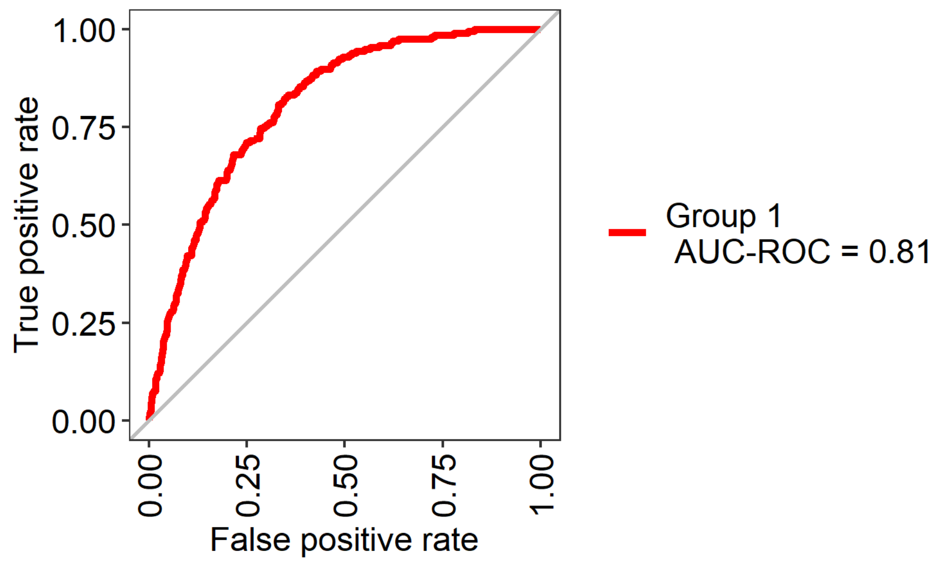
support vector machine (non-linear) without tuning.



The models perform as shown below:
Model 1: Logistic regression

```
evalm(glmFit, positive = "stroke", plots = c("r", "pr"))$pr
```

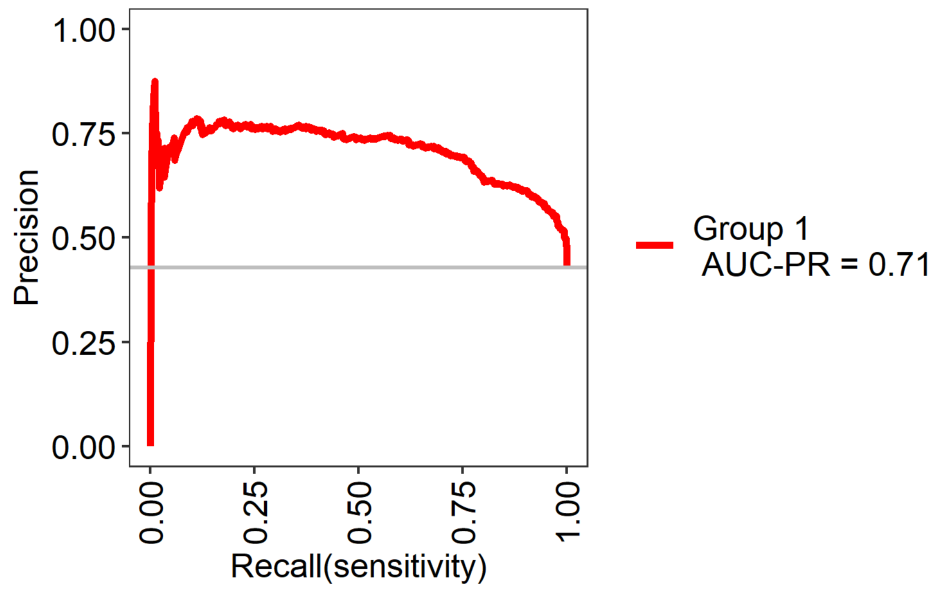


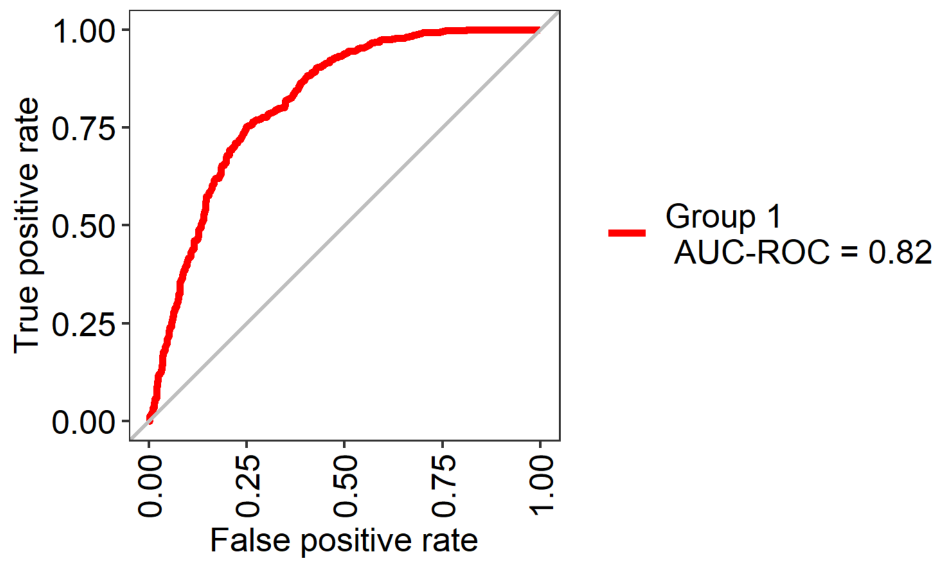


```
## NULL
```

Model 1 with SMOTE

```
evalm(glmFit2, positive = "stroke", plots = c("r", "pr"))$pr
```

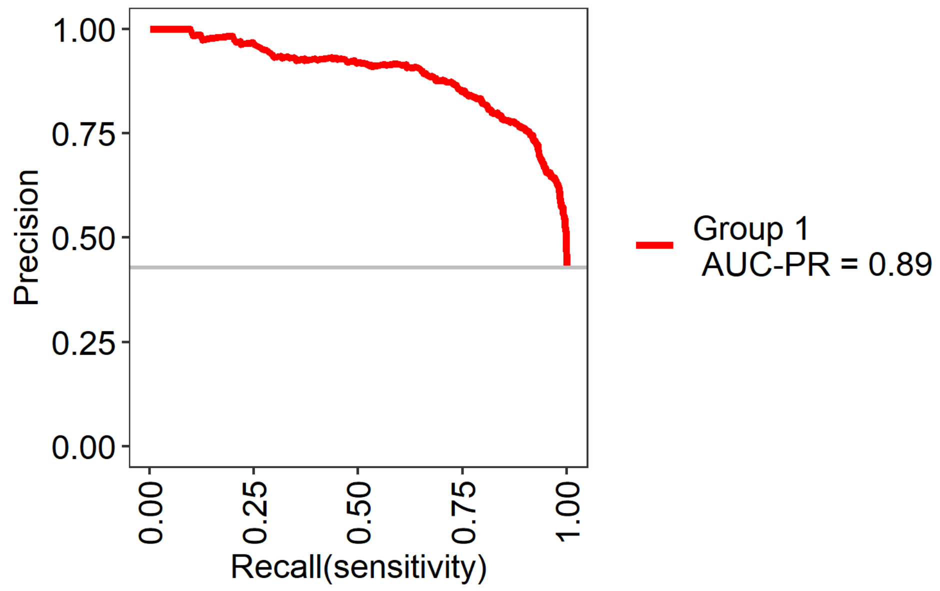


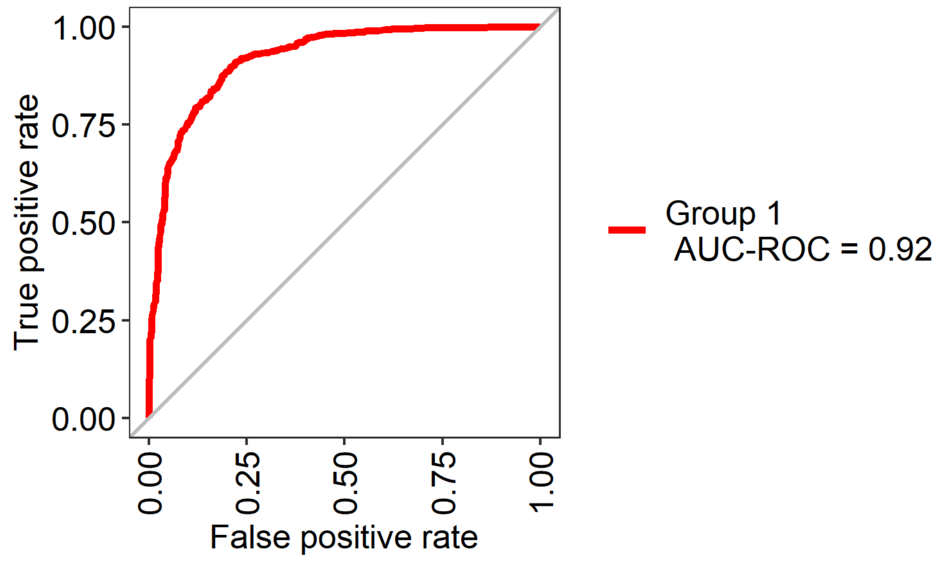


```
## NULL
```

Model 2: Random forest with SMOTE

```r
evalm(rfFit, positive = "stroke", plots = c("r", "pr"))$pr
```

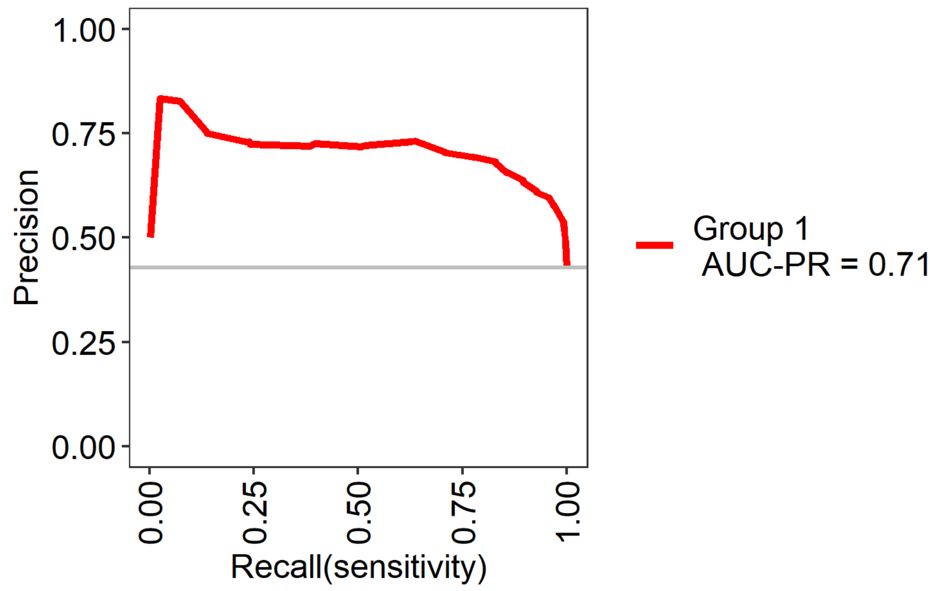


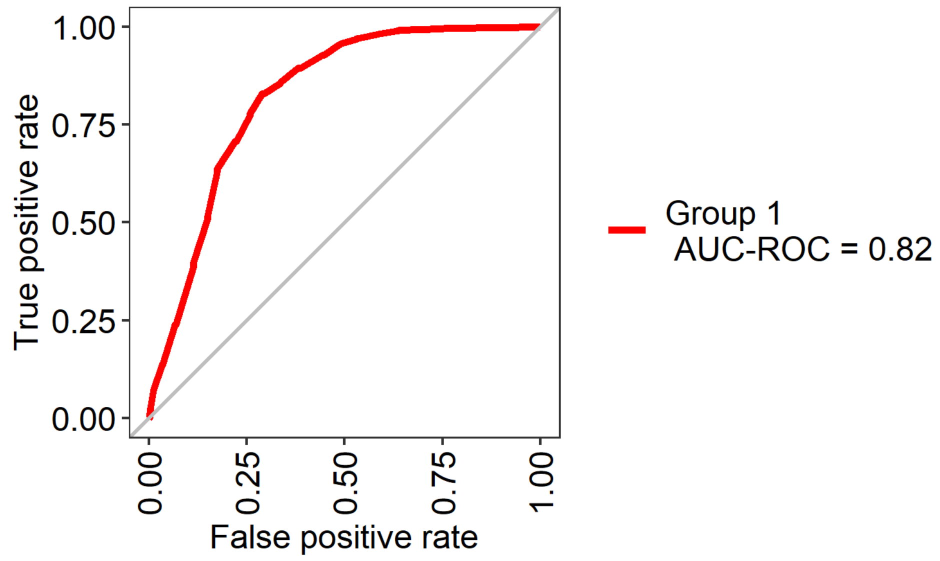


```
## NULL
```

Model 3: K-nearest neighbour with SMOTE

```
evalm(knnFit, positive = "stroke", plots = c("r", "pr"))$pr
```

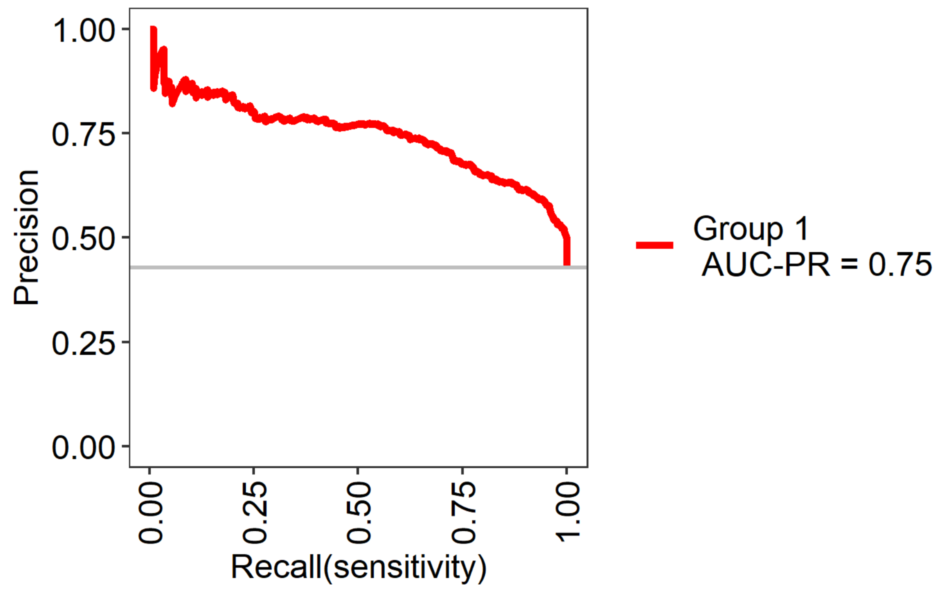


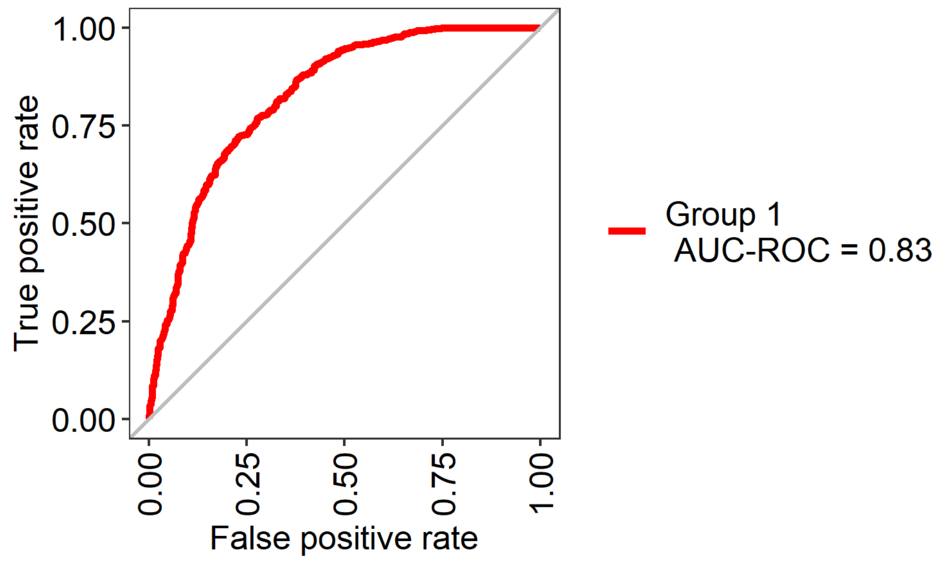


```
## NULL
```

Model 4: Support vector machine with SMOTE

```
evalm(svmFit, positive = "stroke", plots = c("r", "pr"))$pr
```





```
## NULL
```

The random forest model outperforms all others with a AUC-ROC of 92% and a AUC-PR of 82%. For educational purposes only, the training data of all models is summarised.

```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: GLM, GLM_smote, RF, KNN, SVM
## Number of resamples: 10
##
## ROC
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## GLM       0.7403125 0.7834761 0.8109211 0.8097227 0.8431641 0.8798438    0
## GLM_smote 0.7901738 0.8033147 0.8122157 0.8168210 0.8343703 0.8398522    0
## RF        0.9028106 0.9148251 0.9231182 0.9218211 0.9332640 0.9342416    0
## KNN       0.7944647 0.8081420 0.8310684 0.8226985 0.8342238 0.8399485    0
## SVM       0.8062648 0.8157048 0.8282557 0.8276147 0.8403247 0.8494133    0
##
## Sens
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## GLM       0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000    0
## GLM_smote 0.6779661 0.6904661 0.7288136 0.7174859 0.7415254 0.7457627    0
## RF        0.7627119 0.8305085 0.8474576 0.8375141 0.8644068 0.8813559    0
## KNN       0.7666667 0.7796610 0.7966102 0.8037853 0.8262712 0.8644068    0
## SVM       0.6610169 0.7029661 0.7118644 0.7208475 0.7457627 0.7796610    0
##
## Spec
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## GLM       1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    0
## GLM_smote 0.7088608 0.7341772 0.7771016 0.7716488 0.8069620 0.8227848    0
## RF        0.8101266 0.8159283 0.8343882 0.8350211 0.8481013 0.8607595    0
## KNN       0.6835443 0.7088608 0.7260630 0.7272152 0.7436709 0.7820513    0
## SVM       0.7215190 0.7492291 0.7707725 0.7715677 0.8037975 0.8227848    0
```

Sensitivity and Specifity is highest for the random forest model.

The fitted models are now used with the test set which has never been used before, neither to train the model nor to evaluate it.

```
glm_test <- confusionMatrix(predict(glmFit, testset), testset$stroke)
glm2_test <- confusionMatrix(predict(glmFit2, testset), testset$stroke)
rf_test <- confusionMatrix(data = predict(rfFit, testset), reference =
testset$stroke)
knn_test <- confusionMatrix(data = predict(knnFit, testset), reference =
testset$stroke)
svm_test <- confusionMatrix(data = predict(svmFit, testset), reference =
testset$stroke)
```

As the output of the confusion matrix is confusing, below is a summary./

Model	Accuracy	Sensitivity	Specificity
Logistic regression	0.9413146	0.00	1.0000000
Logistic regression with SMOTE	0.7582160	0.74	0.7593516
Random forest with SMOTE	0.7992958	0.58	0.8129676
K-nearest neighbour with SMOTE	0.7230047	0.72	0.7231920
Support vector machine with SMOTE	0.7570423	0.68	0.7618454

Compared to the train dataset, the random forest and k-nearest neighbour show overfitting which results in an inferior performance on the test dataset compared to train set. The logistic regression model with SMOTE shows the best performance on the test set with a sensitivity of 70 % and a specifity of 72.3 %.

4 Conclusion

The machine learning algorithm to predict stroke reached a sensitivity of 72 % using a logistic regression model. The AUC-ROC and AUC PR are 82 % and 71 %, respectively. The specificity of 76.2 % indicates that there is a high false positive rate. However, considering the often disabling condition after a stroke, we would rather treat some individuals (for primary prevention) who never experience a stroke than not treat individuals who will develop this severe disease. Especially because the side effects of the primary prevention with a drug treatment are minor. From a methodological point of view, we have encountered the frequent problem of imbalanced outcome. SMOTE was used to subsample the underrepresented minority class and the models trained on the SMOTE-data showed a much better performance. In addition, better a better validation strategy could also result in a better performance on the test set.
Limitations are the imputation methods used on the entire dataset, overtraining of the random forest and k-nearest neighbour model and the relatively few positive events (= stroke) that required smote that could have been carried out within the train function. Future research should be carried out with bigger datasets and more positive outcomes and should also focus on the increase of sensitivity and specificity of the prediction model to correctly identify the people at risk of a stroke.

5 Acknowledgement

I thank the provider of the stroke prediction dataset, fedesoriano.

6 References

(1) https://www.who.int/news-room/fact-sheets/detail/obesity-and-overweight, accessed May 31, 2021