Teil 5

Classification

Rebecca Karwen

Contents

1. Traineren zwei beliebiger Classifier 1
2. Evaluation der Performance 5
3. Verbesserung der Performance der Classifier 6

1. Traineren zwei beliebiger Classifier

Dafür werden zunächst Test und Trainings Datensätze erstellt, mit denen daraufhin weitergearbeitet wird.

```
seed_val <- 45 # for reproducability
set.seed(seed_val)

#Split data in training and test
sample_size = round(nrow(stroke_data)*.70) # setting sample size to 70% of the data set
index <- sample(seq_len(nrow(stroke_data)), size = sample_size)

train_x <- stroke_data[index, ]
test_x <- stroke_data[-index, ]

#Outcome in extra Vektor
train_y <- train_x$stroke
train_x$stroke = NULL
test_y <- test_x$stroke
test_x$stroke = NULL

test_x$id = NULL
train_x$id = NULL
```

Erster Classifier Decision Tree

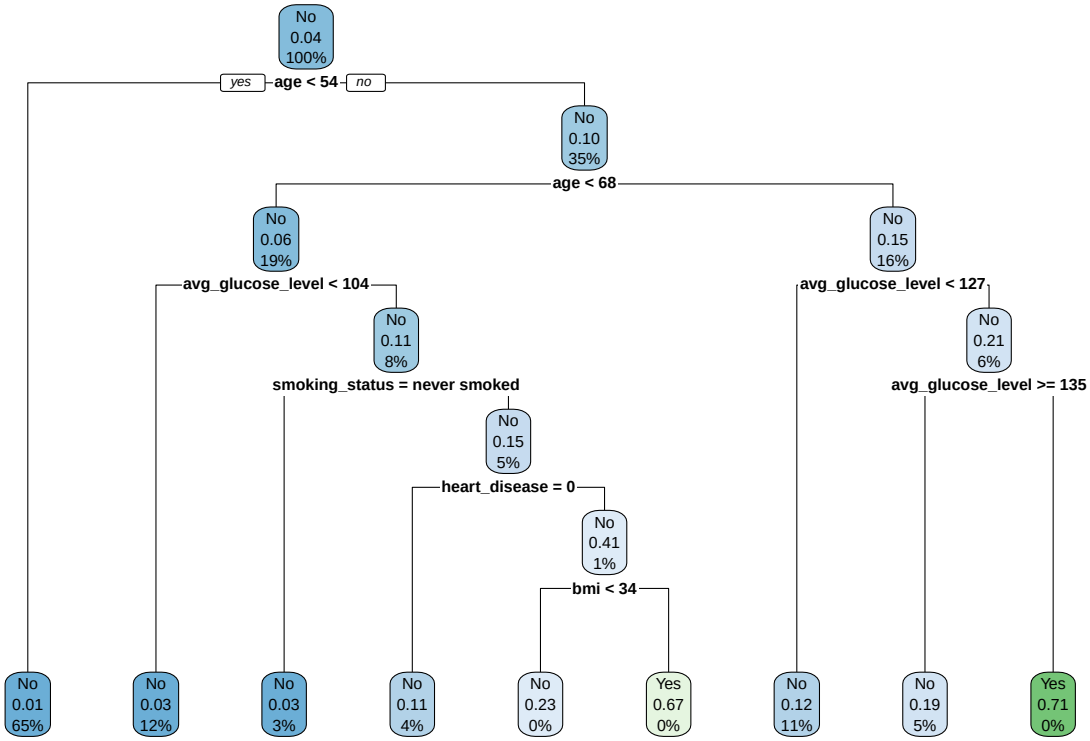
Als ersten Classifier wird der Decision Tree gewählt, da dieser einer der einfacheren Methoden ist, der keine Probleme mit verschieden Datentypen hat und wenig Data Cleaning bedarf. Dabei prunen wir den Decision Tree ,damit er nicht weiter als 8 Ebenen geht. Zuletzt schauen wir uns noch die Werte genauer an, zur späteren Evaluierung der Performance.

```
tree <- rpart(train_y ~ .,data=train_x ,method = "class", parms=list(split="information"), control = rpa

# Evaluate the performance
pred_tree <- predict(tree, test_x,type="class")

confusionMatrix(table(pred_tree, test_y), mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##          test_y
## pred_tree   No  Yes
##       No  1396   65
##       Yes   10    2
##
##                Accuracy : 0.9491
##                  95% CI : (0.9366, 0.9597)
##     No Information Rate : 0.9545
##     P-Value [Acc > NIR] : 0.8557
##
##                   Kappa : 0.0373
##
##  Mcnemar's Test P-Value : 4.507e-10
##
##             Sensitivity : 0.99289
##             Specificity : 0.02985
##          Pos Pred Value : 0.95551
##          Neg Pred Value : 0.16667
##               Precision : 0.95551
##                  Recall : 0.99289
##                      F1 : 0.97384
##              Prevalence : 0.95451
##          Detection Rate : 0.94773
##    Detection Prevalence : 0.99185
##       Balanced Accuracy : 0.51137
##
##        'Positive' Class : No
##
```

Nun wird der Decision Tree graphisch dargestellt, um einen besseren Überblick zu bekommen.

```
#Plot the tree
rpart.plot(tree)
```



Zweiter Classifier - Naive Bayes

Als zweiter Classifier wird Naive Bayes gewählt, weil dieser sehr schnell ist, andere Methoden übertreffen soll und auch mit kleinen Datasets einfach zu trainieren ist. Dabei wird auch wieder die Confusion Matrix erstellt - zu besseren Auswertung der Performance.

```
model_naive_bayes <- naive_bayes(train_y ~ ., data = train_x, laplace = T)
pred_naive_bayes <- predict(model_naive_bayes, test_x)

confusionMatrix(table(pred_naive_bayes, test_y), mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##                 test_y
## pred_naive_bayes   No  Yes
##              No  1264   41
##              Yes  142   26
##
##                Accuracy : 0.8758
##                  95% CI : (0.8578, 0.8922)
##     No Information Rate : 0.9545
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1671
##
##  Mcnemar's Test P-Value : 1.444e-13
##
##             Sensitivity : 0.8990
##             Specificity : 0.3881
##          Pos Pred Value : 0.9686
##          Neg Pred Value : 0.1548
##               Precision : 0.9686
##                  Recall : 0.8990
##                      F1 : 0.9325
##              Prevalence : 0.9545
##          Detection Rate : 0.8581
##    Detection Prevalence : 0.8859
##       Balanced Accuracy : 0.6435
##
##        'Positive' Class : No
##
```

2. Evaluation der Performance

Bei Naive Bayes ist in der Confusion Matrix sichtbar, dass 1264 der Vorhersagen für keinen Schlaganfall meist zutreffen. Es gibt 142 Werte des Error Typs I, d.h. jemand, der eigentlich keinen Schlaganfall hatte, wird als jemand, der einen Schlaganfall hatte, vorhergesagt. Andersherum gibt es 41 Personen, für die kein Schlaganfall ermittelt wurde, die aber einen hatten - also ein Type II Error.

Beim weiteren Output der Confusion Matrix fällt die Specificity, also die Fähigkeit negative Werte zu bestimmen, auf, da diese nur 39 % beträgt, und die Negative Predictive Value (Neg Prev Value), da diese auch nur bei 15% liegt. Die Kappa Value ist mit 0.17 auch relativ niedrig mit. Die Kappa Statistik misst die Übereinstimmung mit dem, was zufällig erwartet wird. Ein Wert von 1 würde völlige Übereinstimmung bedeuten.

Beim Decision Tree ist in der Confusion Matrix sichtbar, dass 1396 der Vorhersagen für keinen Schlaganfall meist zutreffen, es gibt 10 Werte des Error Typs I, d.h. jemand, der eigentlich keinen Schlaganfall hatte, wird als jemand, der einen Schlaganfall hatte, vorhergesagt. Andersherum gibt es 65 Personen für die kein Schlaganfall ermittelt wurde, die aber einen hatten, also ein Type II Error - welcher im Gegensatz zu Naive Bayer relativ hoch ist.

Beim weiteren Output der Confusion Matrix fällt auch die Specificity auf, da diese nur 3 % beträgt und die Negative Predictive Value (Neg Prev Value), da diese auch nur bei 16% liegt. Die Kappa Value ist mit 0.04 auch relativ niedrig.

Für diesen Datensatz würde ich Naive Bayes nutzen, da diese Methode nach der Auswertung die besseren Vorhersagen zu treffen scheint. Besonders, da es bei Krankheiten wichtiger ist, einen geringen Error des Typs II zu haben.

Eine weitere Untersuchung ergibt jedoch, dass die Daten sehr unbalanciert sind. Die folgende Tabelle zeigt, dass ohne ein Modell die Wahrscheinlichkeit mit 95% für keinen Schlaganfall vorhersagen würde, besser als beide Verfahren wäre.

```
prop.table(table(stroke_data$stroke))
```

```
##
##         No        Yes
## 0.95742514 0.04257486
```

```
glimpse(pred_naive_bayes)
```

```
##  Factor w/ 2 levels "No","Yes": 1 1 2 1 2 2 2 1 2 1 ...
```

3. Verbesserung der Performance der Classifier

Um die Klasse besser zu balancieren, wird Oversampling angewendet. Dies bedeutet, dass Daten von unterrepräsentierten Klassen geklont werden.

```
set.seed(2)

oversampling_result <- ovun.sample(stroke ~ ., data = stroke_data, method = "both")
oversampled_recession <- oversampling_result$data

prop.table(table(oversampled_recession$stroke))
```

```
##
##        No       Yes
## 0.4937869 0.5062131
```

Nun ist die Verteilung fast gleich. Danach erstellen wir neue Test Datensätze.

```
seed_val <- 45 # for reproducability
set.seed(seed_val)

#Splitt data in training and test
sample_size_2 = round(nrow(oversampled_recession)*.70) # setting sample size to 70% of the data set
index_2 <- sample(seq_len(nrow(oversampled_recession)), size = sample_size)

train_x_2 <- oversampled_recession[index_2, ]
test_x_2 <- oversampled_recession[-index_2, ]

#Outcome in extra Vektor
train_y_2 <- train_x_2$stroke
train_x_2$stroke = NULL
test_y_2 <- test_x_2$stroke
test_x_2$stroke = NULL

test_x_2$id = NULL
train_x_2$id = NULL
```

Dann wird wieder zuerst der neue Entscheidungsbaum berechnet.

```
tree_2 <- rpart(train_y_2 ~ .,data=train_x_2 ,method = "class", parms=list(split="information"), control

# Evaluate the performance
pred_tree_2 <- predict(tree_2, test_x_2,type="class")

confusionMatrix(table(pred_tree_2, test_y_2), mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##            test_y_2
## pred_tree_2  No Yes
##         No  562  27
##         Yes 179 705
##
##                Accuracy : 0.8601
##                  95% CI : (0.8414, 0.8775)
##     No Information Rate : 0.5031
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7206
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7584
##             Specificity : 0.9631
##          Pos Pred Value : 0.9542
##          Neg Pred Value : 0.7975
##               Precision : 0.9542
##                  Recall : 0.7584
##                      F1 : 0.8451
##              Prevalence : 0.5031
##          Detection Rate : 0.3815
##    Detection Prevalence : 0.3999
##       Balanced Accuracy : 0.8608
##
##        'Positive' Class : No
##
```

Danach wieder Naive Bayes angewandt mit den neuen Daten.

```
model_naive_bayes_2 <- naive_bayes(train_y_2 ~ ., data = train_x_2, laplace = T)
pred_naive_bayes_2 <- predict(model_naive_bayes_2, test_x_2)

confusionMatrix(table(pred_naive_bayes_2, test_y_2), mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##                   test_y_2
## pred_naive_bayes_2  No Yes
##                No  570 213
##                Yes 171 519
##
##                Accuracy : 0.7393
##                  95% CI : (0.7161, 0.7616)
##     No Information Rate : 0.5031
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.4784
##
##  Mcnemar's Test P-Value : 0.03641
##
##             Sensitivity : 0.7692
##             Specificity : 0.7090
##          Pos Pred Value : 0.7280
##          Neg Pred Value : 0.7522
##               Precision : 0.7280
##                  Recall : 0.7692
##                      F1 : 0.7480
##              Prevalence : 0.5031
##          Detection Rate : 0.3870
##    Detection Prevalence : 0.5316
##       Balanced Accuracy : 0.7391
##
##        'Positive' Class : No
##
```

Generell haben sich einige oben beschriebenen Werte der Classifier nun verbessert. Bei Naive Bayes hat sich jedoch die Sensitivity verschlechtert, von 90 % auf 76 %, was bedeutet, dass das Modell nun weniger gut True Positive Werte erkennen kann - in unserem Fall also erkennt das Modell weniger gut jemand einen Schlaganfall hat. Die Specificity, die Fähigkeit negative Werte zu erfassen, ist dagegen besser geworden. Leider ist auch der Wert er Leute die laut Modell keinen Schlaganfall haben, im Modell jedoch auch gestiegen.

Beim Entscheidungsbaum haben sich auch nur teilweise die Werte verbessert. Auch hier hat sie Sensitivity abgenommen, die Specificity ist aber wesentlich besser geworden und auch die Neg Pred Value ist jetzt besser. Generell hat sich aber auch die Anzahl der Personen, die mit Ja für Schlaganfall erkannt werden, obwohl sie keinen bekommen, erhöht.

Nach Auswertung der Modelle, würde die Entscheidung welches Modell gewählt wird, diesmal auf den Entscheidungsbaum fallen, da sich die Werte stärker verbesser haben und dort der Error Typ 2 am geringsten ist d.h. es werden weniger Leute die eigentlich einen Schlaganfall haben nicht erkannt.