Compulsory exercise 1: Group 3

TMA4268 Statistical Learning V2023

Helle Villmones Haug, Hjalmar Jacob Vinje and Sanna Baug Warholm

23 February, 2023

Problem 1

For this problem you will need to include some LaTex code. Please install latex on your computer and then consult Compulsor1.Rmd for hints how to write formulas in LaTex.

a)

b)

c)

d)

e)

Problem 2

a)

i)

ii)

b)

c)

Problem 3

The Bigfoot Field Researchers Organization (BFRO)-problem, using the suggested code:

```
bigfoot_original <- readr::read_csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/maste
```

Plus the data preparation, not included in the pdf (but shown in the Rmarkdown-file)

Next, setting seed and creating training- and test-sets:

```
set.seed(2023)
# 70% of the sample size for training set
training_set_size <- floor(0.7 * nrow(bigfoot))
train_ind <- sample(seq_len(nrow(bigfoot)), size = training_set_size)
train <- bigfoot[train_ind, ]
test <- bigfoot[-train_ind, ]
```

Task a)

(i)

```r
model <- glm(class~longitude+latitude+visibility+fur+howl+saw+heard, family="binomial", data=train)

glm_probabilities <- predict(model, test, type="response")
no_classified = sum(glm_probabilities >= 0.5)
no_classified # Number of reports classified as clear sightings: 441
```

```
## [1] 441
```

Number of clear sightings: 441

(ii)

```r
summary(model)
```

```
##
## Call:
## glm(formula = class ~ longitude + latitude + visibility + fur +
##     howl + saw + heard, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0710  -1.0149  -0.4291   1.0007   2.1469
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.989051   0.422048   2.343 0.019106 *
## longitude   -0.003112   0.003460  -0.900 0.368374
## latitude    -0.036988   0.009849  -3.756 0.000173 ***
## visibility  -0.005681   0.023686  -0.240 0.810449
## furTRUE      0.575172   0.136328   4.219 2.45e-05 ***
## howlTRUE    -0.792152   0.189803  -4.174 3.00e-05 ***
## sawTRUE      1.291894   0.097630  13.233  < 2e-16 ***
## heardTRUE   -1.075540   0.099634 -10.795  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2948.6  on 2126  degrees of freedom
## Residual deviance: 2509.9  on 2119  degrees of freedom
## AIC: 2525.9
##
## Number of Fisher Scoring iterations: 4
```

The coefficients for sawTRUE is 1.29, which means that the average change in log odds with one unit increase of the value.

```r
change <- exp(1.29)
print(change)
```

```
## [1] 3.632787
```

The answer is therefore D) Multiply by 3.64")

Task b)

(i)

```
require(MASS)
qda_model <- qda(class~longitude+latitude+visibility+fur+howl+saw+heard, data=train)
qda_predicted <- predict(qda_model, test)
table(qda_predicted$class)
```

```
##
##   0   1
## 286 626
```

Number of clear sightings: 626

(ii)

1): True, 2): False, 3): False, 4): False

Task c)

(i)

```
require(class)
?knn()
knn_model <- knn(train=train, test=test, cl=train$class, k=25, prob=TRUE)

table(knn_model)
```

```
## knn_model
##   0   1
## 471 441
```

Number of clear sightings: 441

Task c)

(ii)

Trade-off between bias and variance, higher k -> less variance and more bias. How to tune the k-parameter in a better way: I could create plots for different k-values and choose the k-value with the lowest error.

Task d)

(i)

Prediction, because we use existing data for creating a model that will classify a new instance correctly as often as possible. With inference, we are more interested in evaluating the relationship between the response variables and the predictor, i.e. the interepretability of the model. All models are interesting with predicting, but KNN and QDA would not been as relevant for inference.

(ii)

Sensitivity: True positive value, probability of a positive test result, given that instance truly is positive. Specificity: True negative value, probability of a negative test result, given that instance tryly is negative.

For all confusion matrices: rows show prediction values and columns show true values.

```r
# Confusion matrix Glm
glm_predicted <- rep(0, 912)
glm_predicted[glm_probabilities > 0.5] <- 1
table(glm_predicted, test$class)
```

```
##
## glm_predicted   0   1
##             0 323 148
##             1 142 299
```

```r
glm_sensitivity <- 299/(299+148)
glm_specificity <- 323/(323+142)
glm_sensitivity
```

```
## [1] 0.6689038
```

```r
glm_specificity
```

```
## [1] 0.6946237
```

Glm sensitivity is 66,9 % and specificity is 69,5 %

```r
# Confusion matrix QDA
table(qda_predicted$class, test$class)
```

```
##
##       0   1
##   0 228  58
##   1 237 389
```

```r
qda_sensitivity <- 389/(389+58)
qda_specificity <- 228/(228+237)
qda_sensitivity
```

```
## [1] 0.8702461
```

```r
qda_specificity
```

```
## [1] 0.4903226
```

QDA sensitivity is 87,0 % and specificity is 49,0 %

```r
# Confusion matrix KNN
table(knn_model, test$class)
```

```
##
## knn_model   0   1
##         0 386  85
##         1  79 362
```

```r
knn_sensitivity <- 362/(362+85)
knn_specificity <- 386/(386+79)
knn_sensitivity
```

```
## [1] 0.8098434
```

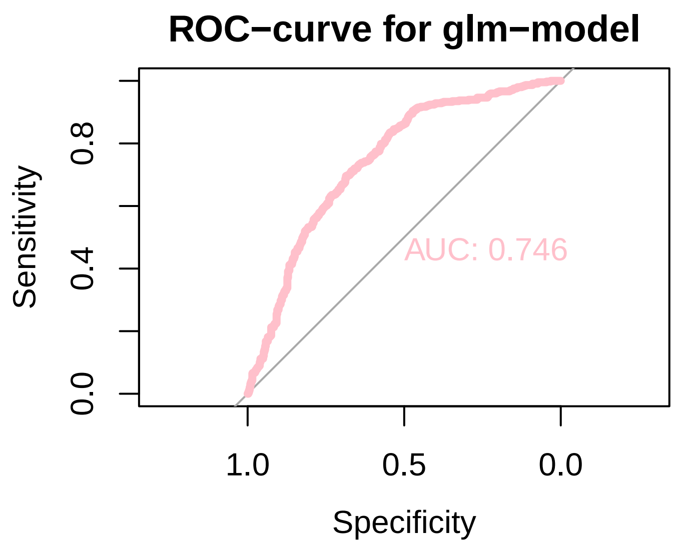
```r
knn_specificity
```

```
## [1] 0.8301075
```

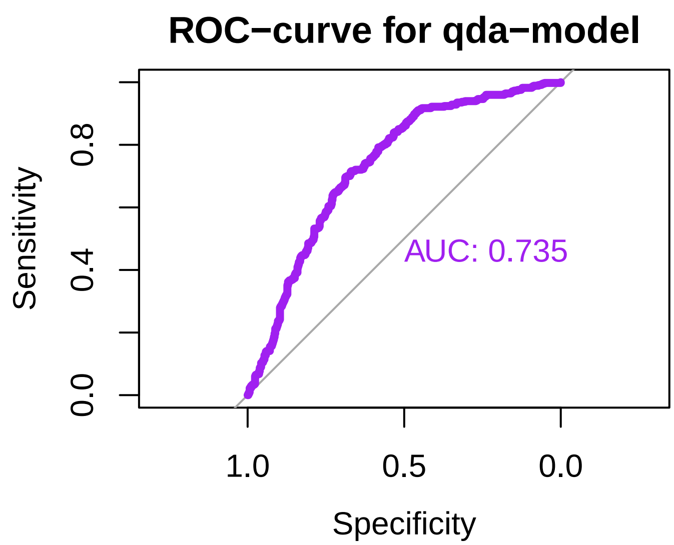
KNN sensitivity is 81,0 % and specificity is 83,0 %

(iii)

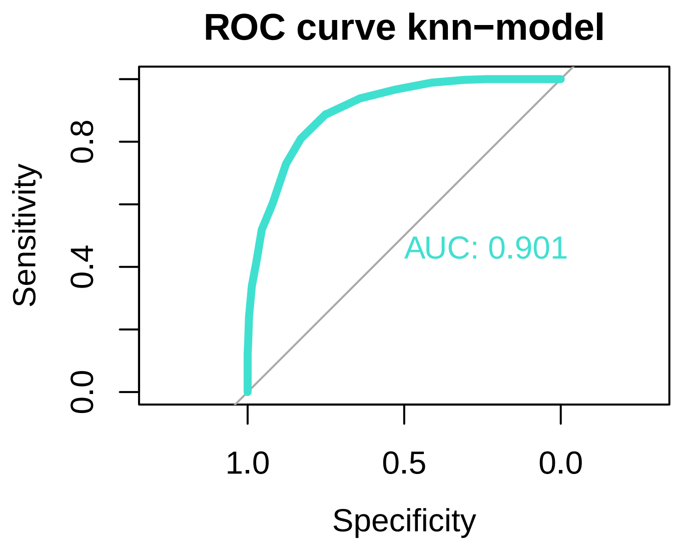
```
library(pROC)

glm_roc <- roc(response = test$class, predictor = glm_probabilities)
plot(glm_roc, col="pink", lwd=4, print.auc=TRUE, main="ROC-curve for glm-model")
```




```
qda_roc <- roc(response = test$class, predictor = qda_predicted$posterior[,"1"])
plot(qda_roc, col="purple", lwd=4, print.auc=TRUE, main="ROC-curve for qda-model")
```




```
knn_probabilities <- ifelse(knn_model == 0, 1 - attributes(knn_model)$prob,attributes(knn_model)$prob)
knn_roc <- roc(response = test$class, predictor = knn_probabilities)
plot(knn_roc, col="turquoise", lwd=4, print.auc=TRUE, main="ROC curve knn-model")
```



(iv)

Glm and QDA performs similar for ROC, while KKN performs significantly better. Would therefore choose the KNN-classifier for this problem.

Problem 4

a)

b)

(i): False, (ii): False, (iii): True, (iv): False