Compulsory exercise 1: Group 11

TMA4268 Statistical Learning V2023

Torbjørn Vatne, Ludvik Braathen and Johan Bjerkem

13 February, 2023

```
install.packages("knitr")     # probably already installed
install.packages("rmarkdown") # probably already installed
install.packages("ggplot2")   # plotting with ggplot2
install.packages("dplyr")     # for data cleaning and preparation
install.packages("tidyr")     # also data preparation
install.packages("carData")   # dataset
install.packages("class")     # for KNN
install.packages("pROC")      # calculate roc
install.packages("plotROC")   # plot roc
install.packages("ggmosaic")  # mosaic plot
```

Problem 1 (9P)

We consider the following regression problem

$$Y = f(\mathbf{x}) + \varepsilon, \text{ where } \mathrm{E}(\varepsilon) = 0 \text{ and } \mathrm{Var}(\varepsilon) = \sigma^2.$$

Assume now that the true function $f(\mathbf{x})$ is a linear combination of the observed covariates, that is $f(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\beta} = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$, where $\mathbf{x}$ and $\boldsymbol{\beta}$ are both vectors of length $p+1$.

We know that the OLS estimator in this case is $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}$, with design matrix $\mathbf{X}$ and response vector $\mathbf{Y}$. In this task we will look at a competing estimator $\widetilde{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{Y}$ (this is called the ridge regression estimator), where $\lambda \geq 0$ is a constant tuning parameter that controls the bias-variance trade-off. Observe that for $\lambda = 0$ the ridge regression estimator is equivalent to $\hat{\boldsymbol{\beta}}$.

We will first derive mathematical formulas for the bias and variance of $\widetilde{\boldsymbol{\beta}}$ and then we will plot these curves in R.

a) (1P)

Find the expected value and the variance-covariance matrix of $\widetilde{\boldsymbol{\beta}}$.

Here we find the expected value:

$$E(\widetilde{\boldsymbol{\beta}}) = E((\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{Y})$$
$$E(\widetilde{\boldsymbol{\beta}}) = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top E(\mathbf{Y})$$
$$E(\widetilde{\boldsymbol{\beta}}) = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top E(\mathbf{X}\boldsymbol{\beta} + \varepsilon)$$
$$E(\widetilde{\boldsymbol{\beta}}) = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top (\mathbf{X}\boldsymbol{\beta} + \mathbf{0})$$

$$E(\widetilde{\boldsymbol{\beta}}) = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{X} \beta$$

Here we find the variance-covarince matrix:

$$\begin{aligned}
Cov(\widetilde{\boldsymbol{\beta}}) &= Cov((X^T X + \lambda I)^{-1} X^T Y) && (1)\\
&= (X^T X + \lambda I)^{-1} X^T Cov(Y)((X^T X + \lambda I)^{-1} X^T)^T && (2)\\
&= (X^T X + \lambda I)^{-1} X^T \sigma^2 I((X^T X + \lambda I)^{-1} X^T)^T && (3)
\end{aligned}$$

b) (2P)

Let $\widetilde{f}(\mathbf{x}_0) = \mathbf{x}_0^T \widetilde{\boldsymbol{\beta}}$ be the prediction at a new covariate vector $\mathbf{x}_0$. Using a), find the expected value and variance for $\widetilde{f}(\mathbf{x}_0) = \mathbf{x}_0^T \widetilde{\boldsymbol{\beta}}$.

c) (1P)

Explain with words how we can interpret the three terms: bias, variance and irreducible error.

Bias is the error that comes by oversimplifying the model. This means underfitting the model, meaning high bias will over simplify the relationships in the data. Variance, on the other hand, is the error that comes by creating a too complex mode. This means overfitting the model, where the model tries to hard to fit the function to the noise of the data. There is a famous plot of the relationship between bias and variance. When we train a model we start with a high bias that decays as we train the model. The variance starts low and increases as we train the model. We want to hit the sweet spot where bias and variance intercepts, as this is the point with the minimal total error. Irreducible error is the error that comes with the data. As the name suggests this error cannot be eliminated by any model, as it is inherit by the data and is not due to bias or variance.

d) (2P)

Find the expected MSE at $\mathbf{x}_0$, $E[(y_0 - \widetilde{f}(\mathbf{x}_0))^2]$.

Hint: Use that the expected MSE can be decomposed into squared bias, variance and irreducible error, and then move on from there.

Plotting the bias-variance trade-off

The estimator $\widetilde{\boldsymbol{\beta}}$ is a function of the tuning parameter λ, which controls the bias-variance trade-off. Using the decomposition derived in c) we will plot the three elements (bias, variance and irreducible error) using the values in the code chunk below. `values` is a list with the design matrix `X`, the vector $\mathbf{x}_0$ as `x0`, the parameters vector `beta` and the irreducible error `sigma`.

```
id <- "1X_8OKcoYbng1XvYFDirxjEWr7LtpNr1m" # google file ID
values <- dget(sprintf("https://docs.google.com/uc?id=%s&export=download", id))
X <- values$X
dim(X)
```

```
## [1] 100  81
```

```
x0 <- values$x0
dim(x0)
```

```
## [1] 81  1
```

```
beta <- values$beta
dim(beta)
```

```
## [1] 81  1
```

```r
sigma <- values$sigma
sigma
```

```
## [1] 0.5
```

e) (1P)

First we will create the squared bias function (`bias`) which takes as inputs the parameter `lambda`, `X`, `x0`, `beta` and returns the squared bias. You have only to fill the `value <- ...` and run the chunk of code to plot the squared bias, where value is the squared bias as derived in d).

```r
library(ggplot2)
bias <- function(lambda, X, x0, beta) {
  p <- ncol(X)
  value <- ...
  return(value)
}
lambdas <- seq(0, 2, length.out = 500)
BIAS <- rep(NA, length(lambdas))
for (i in seq_along(lambdas)) BIAS[i] <- bias(lambdas[i], X, x0, beta)
dfBias <- data.frame(lambdas = lambdas, bias = BIAS)
ggplot(dfBias, aes(x = lambdas, y = bias)) +
  geom_line(color = "hotpink") +
  xlab(expression(lambda)) +
  ylab(expression(bias^2))
```

f) (1P)

Now we will create the variance function which takes the same inputs as the squared bias. As in e) you have to fill only the `value <- ...` and run the code to plot the variance.

```r
variance <- function(lambda, X, x0, sigma) {
  p <- ncol(X)
  inv <- solve(t(X) %*% X + lambda * diag(p))
  value <- ...
  return(value)
}
lambdas <- seq(0, 2, length.out = 500)
VAR <- rep(NA, length(lambdas))
for (i in seq_along(lambdas)) VAR[i] <- variance(lambdas[i], X, x0, sigma)
dfVar <- data.frame(lambdas = lambdas, var = VAR)
ggplot(dfVar, aes(x = lambdas, y = var)) +
  geom_line(color = "gold") +
  xlab(expression(lambda)) +
  ylab("variance")
```

g) (1P)

Fill in the `exp_mse` of the following code to calculate the expected MSE for the same lambda values that you plugged in above. Then plot all the components together and find the value of λ which minimizes the expected MSE.

```r
exp_mse <- ...
lambdas[which.min(exp_mse)]
```

Problem 2 (15P)

Bert-Ernie has his eye on a career in academia, but has not really decided on a field yet. Like most academics, his greatest concern is his future salary as a tenured professor. He comes across the `Salaries` dataset from the `carData` `R` package, and decides to perform a statistical analysis to try to find out how he can maximize his future salary as an academic.

To get more information about the covariates in the dataset, you should run `?Salaries` after loading the data. It's also a good idea to do a *descriptive analysis* of the data before fitting any models. A good starting point can be the `ggpairs()` function from the `GGally` package.

```
library(carData)
?Salaries
GGally::ggpairs(Salaries)
```

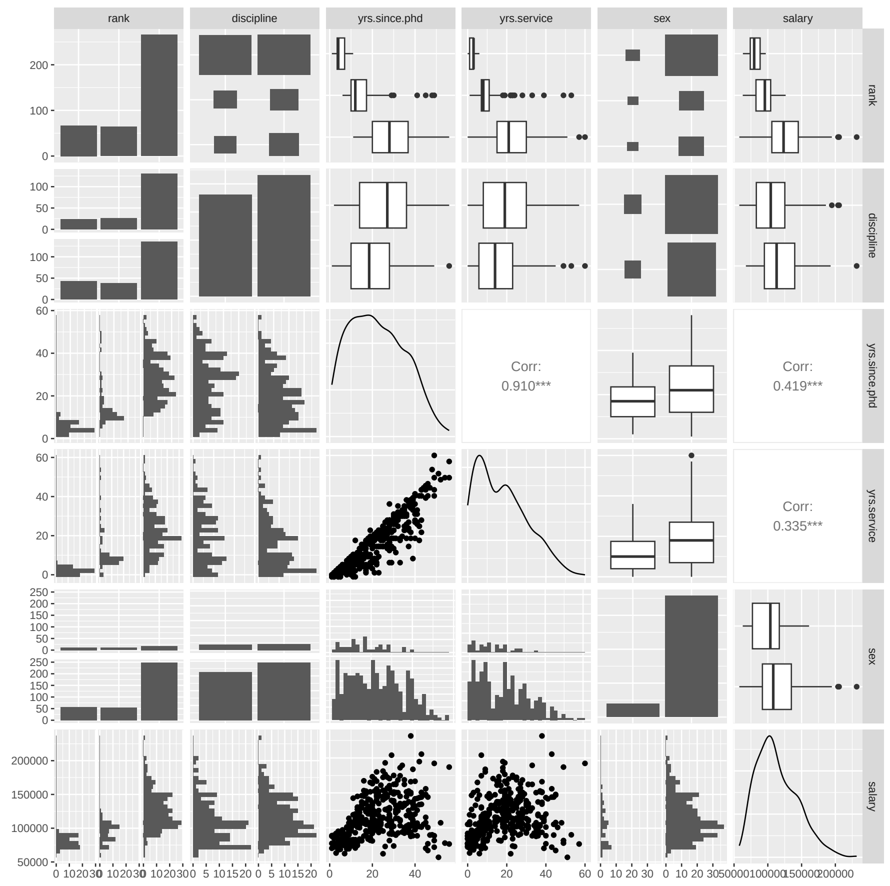

Figure 1: Pairs plot of the academic salary data set.

After doing a descriptive analysis, Bert-Ernie makes a multiple linear regression model with all covariates included, which he calls `model1`.

```
# Fit full model
model1 <- lm(salary ~ ., data = Salaries)
summary(model1)
```

```
##
## Call:
## lm(formula = salary ~ ., data = Salaries)
##
```

```
## Residuals:
##    Min     1Q Median     3Q    Max
## -65248 -13211  -1775  10384  99592
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    65955.2     4588.6  14.374  < 2e-16 ***
## rankAssocProf  12907.6     4145.3   3.114  0.00198 **
## rankProf       45066.0     4237.5  10.635  < 2e-16 ***
## disciplineB    14417.6     2342.9   6.154 1.88e-09 ***
## yrs.since.phd    535.1      241.0   2.220  0.02698 *
## yrs.service     -489.5      211.9  -2.310  0.02143 *
## sexMale         4783.5     3858.7   1.240  0.21584
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22540 on 390 degrees of freedom
## Multiple R-squared:  0.4547, Adjusted R-squared:  0.4463
## F-statistic:  54.2 on 6 and 390 DF,  p-value: < 2.2e-16
```

a) Categorical variables (2P)

The covariate `rank` is categorical, with the three possible values `AsstProf` (assistant professor), `AssocProf` (associate professor) and `Prof` (full professor).

i) How does the `lm()` function encode for this type of covariate, and what are the interpretations of the `rankAssocProf` and `rankProf` estimates, respectively? (1P)

The output from `summary()` tells us the p-values of `rankAssocProf` and `rankProf`, but not `rank` as a whole.

ii) Perform an appropriate test to check whether there is evidence that `rank` as a whole has an impact on `salary`. The output from `summary()` is not enough to do this. (1P)

b) Conflicting model results? (2P)

Having heard of the gender wage-gap, Bert-Ernie is surprised to see that `model1` finds no evidence that `sex` has an effect on `salary`. To make sure nothing has gone wrong, he fits a new model with `sex` as the only covariate (along with an intercept).

```r
sex_model <- lm(salary ~ sex, data = Salaries)
summary(sex_model)
```

```
##
## Call:
## lm(formula = salary ~ sex, data = Salaries)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -57290 -23502  -6828  19710 116455
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   101002       4809  21.001  < 2e-16 ***
## sexMale        14088       5065   2.782  0.00567 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 30030 on 395 degrees of freedom
## Multiple R-squared:  0.01921,	Adjusted R-squared:  0.01673
## F-statistic: 7.738 on 1 and 395 DF,  p-value: 0.005667
```

The new model finds strong evidence that `sex` has an effect on `salary`. How would you explain the fact that `sex_model` finds evidence of `sex` impacting `salary`, but the full model `model1` does not? Make one or two figures to demonstrate your explanation, with proper figure captions explaining the figure(s). (2P)

Hint: The descriptive analysis might be useful to identify what is going on.

c) Model assumptions and transformation (2P)

Checking the assumptions of `model1`, Bert-Ernie observes that the assumptions of the linear regression model are not fulfilled in `model1`.

```r
library(ggplot2)
library(ggfortify)
autoplot(model1, smooth.colour = NA)
```

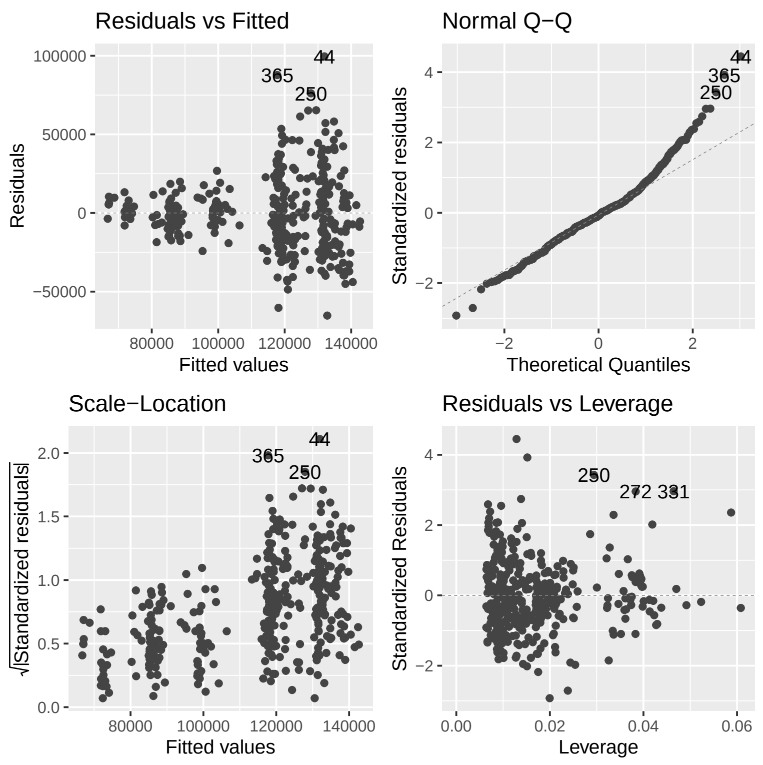

Figure 2: Diagnostic plots for ‘model1‘.

i) Based on the figure below, which assumptions are not fulfilled? (1P)
ii) Bert-Ernie has heard that sometimes such issues may be solved by transforming the response or covariates of the model, so he decides to try treating the log-transform of `salary` as the response in a new model. Implement the new model, and call it `model2`. Have the model assumptions improved in `model2` compared to `model1`? (1P)

d) Interactions (2P)

Bert-Ernie hypothesizes that the impact of `sex` on salary in academia might be stronger in older generations. While keeping the log-transformed response and all the other covariates in the model he therefore decides to design a model with an interaction term between `sex` and `yrs.since.phd`.

i) Implement the model with the interaction term, and call it `model3` (1P).

ii) Interpret the results from `model3`. Is Bert-Ernie's hypothesis correct? (1P)

e) Bootstrap (4P)

You might have noticed that the R^2 we get from `summary()` is just a point estimate, without any uncertainty associated to it. We can use the bootstrap to estimate the uncertainty of the R^2 in our models. To this end:

(i) Generate 1000 bootstrap samples of the R^2 in `model1`. Use `set.seed(4268)` at the beginning of the bootstrap iterations to ensure that your results are reproducible (2P).
(ii) Plot the respective distribution of the values (0.5P).
(iii) Find the standard error and the 95% quantile interval (1P).
(iv) Interpret what you see (0.5P).

Throughout, show your R code and printouts.

f) Picking a field (3P)

We (and Bert-Ernie) now assume that `model1` is the correct (that is, the data-generating) model. After some character development (namely, all this R coding is giving him a headache) Bert-Ernie has figured out that he would rather work in a theoretical field than an applied field. But, money-conscious as always, he decides that he will follow this dream if and only if his model predicts that 20 years after finishing his PhD he will be earning *at least* 75 000$ every nine months, with 95% certainty. He does not care how much he earns, as long as it is more than 75 000$. He is sure that he will get a permanent position immediately upon finishing his PhD (so his `yrs.since.phd` will equal his `yrs.service`), that he will have become full professor within the 20 years and that his sex will still be male.

```r
# Make a data frame containing two new observations, corresponding to
# Bert-Ernie's two possible futures
bert_ernie <- data.frame(rank = c("Prof", "Prof"),
                         discipline = c("A", "B"), # Theoretical, applied
                         yrs.since.phd = c(20, 20),
                         yrs.service = c(20, 20),
                         sex = c("Male", "Male"))
# Use the full model to predict his salary
preds <- predict(object = model1,
                 newdata = bert_ernie,
                 interval = "confidence",
                 level = 0.975) # Not 0.95 since we don't care about upper limit
# Check predictions
preds
```

```
##        fit      lwr      upr
## 1 116715.6 110985.0 122446.2
## 2 131133.2 126145.6 136120.8
```

```r
# Check if lower limit for salary in a theoretical field is large enough
preds[1, 2] > 75000
```

```
## [1] TRUE
```

He sees that the lower limit of the computed interval beats 75 000$ by a fair amount, so he breathes a sigh of relief. However, he has made two errors in the call to `predict()`.

i) Correct his mistakes, and make the calculation he was actually interested in, still using `predict()`. Will he still be able to follow his dream after making the correction, or are many sleepless nights of debugging R code looming in his future? (1P)

ii) Write down an analytic expression for the correct lower limit, and implement the analytic expression in R to make sure you get the same result as from `predict()`. (2P)

Hint: see Recommended Exercises Week 3.

R-hint: In the implementation you might have use for the functions `model.matrix()`, `coef()`, `sigma()`, `df.residual()`, `qt()`, `sqrt()`, `solve()` and `t()`, and the matrix multiplication operator `%*%`.

Problem 3 (13P)

The Bigfoot Field Researchers Organization (BFRO) is an organization dedicated to investigating the bigfoot mystery, and for years they have been collecting reported sightings in a database. They manually classify their reports into

- Class A: Clear sightings in circumstances where misinterpretation or misidentification of other animals can be ruled out with greater confidence
- Class B: Incidents where a possible bigfoot was observed at a great distance or in poor lighting conditions and incidents in any other circumstance that did not afford a clear view of the subject.

However, they wonder if this can be automated and done by a classification algorithm instead. So in this task, you will set up a few different classification algorithms for this aim, and evaluate their performance.

Feel free to look at the original data, `bigfoot_original` below, however in this task we will be using a slightly simplified version of the data set where we have extracted some variables of interest and deleted observations that are missing any of these variables of interest.

Download the data as below, and run through the provided cleaning/preparation steps.

```
bigfoot_original <- readr::read_csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/maste
library(dplyr)
# Prepare the data:
bigfoot <- bigfoot_original %>%
  # Select the relevant covariates:
  dplyr::select(classification, observed, longitude, latitude, visibility) %>%
  # Remove observations of class C (these are second- or third hand accounts):
  dplyr::filter(classification != "Class C") %>%
  # Turn into 0/1, 1 = Class A, 0 = Class B:
  dplyr::mutate(class = ifelse(classification == "Class A", 1, 0)) %>%
  # Create new indicator variables for some words from the description:
  dplyr::mutate(fur = grepl("fur", observed),
                howl = grepl("howl", observed),
                saw = grepl("saw", observed),
                heard = grepl("heard", observed)) %>%
  # Remove unnecessary variables:
  dplyr::select(-c("classification", "observed")) %>%
  # Remove any rows that contain missing values:
  tidyr::drop_na()
```

The data we will use for this task includes the following variables:

- `class`: the assigned class of the observation, coded as 1 = Class A, 0 = Class B
- `longitude`: longitude of the observation

- `latitude`: latitude of the observation
- `visibility`: estimated visibility at the time and place of observation (higher value means better visibility)
- `fur`: does the report contain the word "fur"? `(TRUE/FALSE)`
- `howl`: does the report contain the word "howl"? `(TRUE/FALSE)`
- `saw`: does the report contain the word "saw"? `(TRUE/FALSE)`
- `heard`: does the report contain the word "heard"? `(TRUE/FALSE)`

For the following tasks, we will be using all of these variables.

The data is split into test and training sets as follows:

Please remember to use the same seed when you split the data into training and test set.

```r
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, ]
```

a) (2P)

(i) (1P) Fit a **logistic regression** model using the training set, and perform the classification on the test set, using a 0.5 cutoff. How many of the reports were classified as clear sightings (class A, category 1)?

(ii) (1P) Single choice: According to this model, how would the odds that an observation is classified as Class A change if the report contains the word "saw", compared to if it does not? (all other covariates stay the same)

1) We multiply by 1.292.
2) We multiply it with -3.641.
3) We multiply by 0.275.
4) We multiply by 3.641.
5) We add 3.641.
6) We add 1.292.

b) (3P)

(i) (1P) Fit a **QDA** model using the training set, and perform the classification on the test set, using a 0.5 cutoff. How many of the reports were classified as class A?

(ii) (2P) Which statements about linear discriminant analysis and quadratic discriminant analysis are true and which are false? Say for *each* of them if it is true or false.

1) QDA differs from LDA in that it assumes that observations of different classes come from Gaussian distributions with different covariance matrices.
2) LDA is a better choice in cases where there are a lot of observations and so reducing bias is important.
3) Even if the Bayes decision boundary is linear, QDA will be a better choice.
4) QDA assumes that the observations are drawn from a multivariate Gaussian distribution with common mean vector and class-specific covariance matrix

c) (2P)

(i) (1P) Fit a **KNN** model using the training set, and perform the classification on the test set, with $k = 25$ (use the `knn` function from the `class` package).

R-hints: In the `knn()` function set `prob=T` to ensure you get the class probabilities that you then need in d) (change the "`...`" to the correct input):

```r
knnMod <- knn(train = ..., test = ..., cl = ..., k = 25, prob = TRUE)
```

(ii) (1P) Explain how you could choose the tuning parameter k in a better way. How does k relate to the bias-variance trade-off in this context?

d) (6P)

We now wish to compare the performance of the three models (logistic regression, QDA and KNN) on this dataset, in order to report back to BFRO on our results.

i) (2P) In this case, are we interested in prediction or inference? What implications would it have for the model choice if we wanted to do prediction, and what implications would it have if we wanted to do inference? In conclusion, does the question of whether our aim is prediction or inference exclude any of the three candidate models in this case? (comment briefly)

ii) (3P) Make confusion matrices for the three predictions performed on the test set in a) - c), and report the confusion matrices, along with sensitivity and specificity. Explain briefly: What does sensitivity and specificity mean? Feel free to explain using the bigfoot example. *Make sure it is clear in the confusion matrix which numbers show the predictions and which show the true values.*

iii) (1P) Present a plot of the ROC curves and calculate the area under the curve (AUC) for each of the classifiers.

iv) (1P) Summarize the performance of the three classifiers with words, based on the evaluations above. Which one would you choose? Justify briefly.

R-hints:

- To obtain $P(y = 1)$ from the `knn()` output you have to be aware that the respective probabilities `attributes(knnMod)$prob` are the success probability for the actual class where the categorization was made. So if you want to get a vector for $P(y = 1)$, you have to use $1 - P(y = 0)$ for the cases where the categorization was 0:

```r
probKNN <- ifelse(knnMod == 0,
                  1 - attributes(knnMod)$prob,
                  attributes(knnMod)$prob)
```

- You might find the functions `roc()` and `ggroc()` from the package `pROC` useful, but there are many ways to plot ROC curves.

Problem 4 (6P)

a) (4P)

To calculate the LOOCV statistic when we have N training data points we typically have to fit N models and evaluate the error of each fit on its left out observation. This task can be computationally intensive when N is large. Fortunately, for linear models there is formula for doing this with fitting just the full data model.

Show that for the linear regression model $Y = \mathbf{X}\beta + \varepsilon$ the LOOCV statistic can be computed by the following formula

$$\text{CV} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2 ,$$

where $h_i = \mathbf{x}_i^\top \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{x}_i$ and $\mathbf{x}_i^\top$ is the ith row of $\mathbf{X}$.

Hints:

1. You need to calculate $\hat{y}_{(-i)}$, which is the predicted value of y when the ith observation is kept out. This can be written as $\hat{y}_{(-i)} = \mathbf{x}_i^\top \hat{\boldsymbol{\beta}}_{(-i)}$.
2. $\mathbf{X}_{(-i)}^\top \mathbf{X}_{(-i)} = \mathbf{X}^\top \mathbf{X} - \mathbf{x}_i \mathbf{x}_i^\top$, where $\mathbf{X}_{(-i)}$ is the $\mathbf{X}$ matrix without the ith row.
3. Analogously $\mathbf{X}_{(-i)}^\top \mathbf{y}_{(-i)} = \mathbf{X}^\top \mathbf{y} - \mathbf{x}_i y_i$.
4. The Sherman–Morrison formula will be useful.

b) Multiple choice (2P)

Say for each statement below whether it is true or false.

(i) The LOOCV will lead to more bias, but less variance than 10-fold CV in the estimated prediction error.
(ii) The formula from **a)** is not valid for polynomial regression.
(iii) The formula from **a)** is valid when we use the log transform of the response in linear regression.
(iv) The validation set-approach is the same as 2-fold CV.