# Compulsory Exercise 1 - Solutions

TMA4268 Statistical Learning V2020

Martina Hall, Michail Spitieris, Stefanie Muff, Department of Mathematical Sciences, NTNU

Hand out date: February 3, 2020

Last changes: 02.02.2020

---

**The submission deadline is Thursday, February 20th 2020, 12:00h using Blackboard**

### R packages

You need to install the following packages in R to run the code in this file.

```
install.packages("knitr")  #probably already installed
install.packages("rmarkdown")  #probably already installed
install.packages("ggplot2")  #plotting with ggplot
install.packages("ggfortify")
install.packages("MASS")
install.packages("dplyr")
```

## Problem 1 - 10P

We have a univariate continuous random variable $Y$ and a covariate $x$. Further, we have observed a training set of independent observation pairs $\{x_i, y_i\}$ for $i = 1, \ldots, n$. Assume a regression model

$$Y_i = f(x_i) + \varepsilon_i \ ,$$

where $f$ is the true regression function, and $\varepsilon_i$ is an unobserved random variable with mean zero and constant variance $\sigma^2$ (not dependent on the covariate). Using the training set we can find an estimate of the regression function $f$, and we denote this by $\hat{f}$. We want to use $\hat{f}$ to make a prediction for a new observation (not dependent on the observations in the training set) at a covariate value $x_0$. The predicted response value is then $\hat{f}(x_0)$. We are interested in the error associated with this prediction.

### a) (1P)

Write down the definition of the expected test mean squared error (MSE) at $x_0$.

**Solution:**

The *expected test mean squared error (MSE) at* $x_0$ is defined as:

$$\mathrm{E}[Y - \hat{f}(x_0)]^2$$

## b) (2P)

Derive the decomposition of the expected test MSE into three terms.

**Solution:**

Useful rule: $\text{Var}[Y] = \text{E}[Y^2] - \text{E}[Y]^2$.

Use that $y_0 = f(x_0) + \varepsilon$:

$$\begin{aligned}\text{E}[y_0 - \hat{f}(x_0)]^2 &= \text{E}[f(x_0) + \varepsilon - \hat{f}(x_0)]^2 \\ &= \text{E}[f(x_0)^2] + \text{E}[\varepsilon^2] + \text{E}[\hat{f}(x_0)^2] - 2\text{E}[f(x_0)\hat{f}(x_0)] - \underbrace{2\text{E}[\varepsilon\hat{f}(x_0)]}_{=0} + \underbrace{2\text{E}[\varepsilon f(x_0)]}_{=0},\end{aligned}$$

where the last two terms are zero due to the independence of $\epsilon$ with anything else, and $\text{E}[\varepsilon] = 0$. Using that $\text{E}[\hat{f}(x_0)^2] = \text{Var}[\hat{f}(x_0)^2] + \text{E}[\hat{f}(x_0)]^2$, $\text{E}[f(x_0)^2] = f(x_0)^2$ and $\text{E}[\varepsilon^2] = \text{Var}[\varepsilon]$, we get

$$\begin{aligned}&= f(x_0)^2 + \text{Var}[\varepsilon] + \text{Var}[\hat{f}(x_0)] + \text{E}[\hat{f}(x_0)]^2 - 2f(x_0)\text{E}[\hat{f}(x_0)] \\ &= \text{Var}[\varepsilon] + \text{Var}[\hat{f}(x_0)] + [\text{Bias}(\hat{f}(x_0))]^2,\end{aligned}$$

where the last equation just rearranges the terms and uses that $[\text{Bias}(\hat{f}(x_0))]^2 = (f(x_0) - \text{E}[\hat{f}(x_0)])^2$.

## c) (1P)

Explain with words how we can interpret the three terms.

**Solution:**

$$\text{E}[(Y - \hat{f}(x_0))^2] = \text{Var}(\varepsilon) + \text{Var}[\hat{f}(x_0)] + [\text{Bias}(\hat{f}(x_0))]^2$$

- First term: irreducible error, $\sigma^2$ and is always present unless we have measurements without error. This term cannot be reduced regardless how well our statistical model fits the data.

- Second term: variance of the prediction at $x_0$ or the expected deviation around the mean at $x_0$. If the variance is high, there is large uncertainty associated with the prediction.
- Third term: squared bias. The bias gives an estimate of how much the prediction differs from the true mean $f(x_0)$. If the bias is low the model gives a prediction which is close to the true value.

## d) (2P) - Multiple choice

Which of the following statements are true and which are false? Say for *each* of them if it is true or false.

(i) If the relationship between the predictors and response is highly non-linear, a flexible method will generally perform **better** than an inflexible method.
(ii) If the sample size $n$ is extremely large and the number of predictors $p$ is small, a flexible method will generally perform **worse** than an inflexible method. method.
(iii) If the number of predictors $p$ is extremely large and the number of observations $n$ is small, a flexible method will generally perform **better** than an inflexible method.
(iv) If the variance of the error terms, $\sigma^2 = Var(\epsilon)$ is extremely high, a flexible method will generally perform **worse** than an inflexible method.

List of answers:

**Solution:**

TRUE, FALSE, FALSE, TRUE

(i) flexible learning method would perform **better** because it is less restrictive on the shape of fit
(ii) flexible learning method would perform **better** because sample size is large enough to fit more parameters and small number of predictors limits model variance.
(iii) flexible learning method would perform **worse** because it would be more likely to overfit
(iv) flexible learning method would perform **worse** because it would be more likely to overfit

## e) (2P) - Multiple choice

Figure 1 shows the squared bias, variance, irreducible error and total error for increasing values of $K$ in KNN. Which of the following statements are true and which are false? Say for *each* of them if it is true or false.

(i) The blue line corresponds to the irriducible error.
(ii) Increased $K$ corresponds to increased flexibility of the model.
(iii) The squared bias increases with increased value of $K$.
(iv) The variance increases with increased value of $K$.

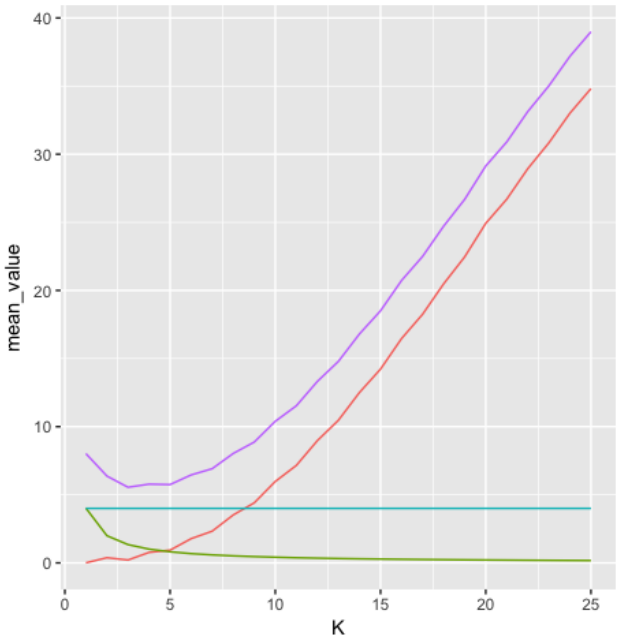



Figure 1: Squared bias, variance, irreducible error and total error for increasing values of $K$ in KNN

List of answers:

**Solution:**

TRUE, FALSE, TRUE, FALSE

Explanation: (i) TRUE: Blue line is the irreducible error, constant for all $K$ (ii) FALSE: Increasing $K$ means that we include more points when classifying, so increased $K$ corresponds to decreased flexibility of the model. (iii) TRUE: Orange curve corresponds to squared bias. (iv) FALSE: The green line corresponds to the variance, which decreases for increased $K$.

## f) (1P) - Single choice

**X** is a 2-dimensional random vector with covariance matrix

$$\mathbf{\Sigma} = \begin{bmatrix} 3 & 0.6 \\ 0.6 & 4 \end{bmatrix}$$

The correlation between the two elements of **X** is: (i) 0.05 (ii) 0.17 (iii) 0.03 (iv) 0.60 (v) 0.10

Answer:

**Solution** (ii): $\frac{0.6}{\sqrt{3 \cdot 4}} = 0.17$

## g) (1P) - Single choice

Which of the plots (A-D) in Figure 2 corresponds to the following covariance matrix?

$$\mathbf{\Sigma} = \begin{bmatrix} 1 & 0.2 \\ 0.2 & 4 \end{bmatrix}$$

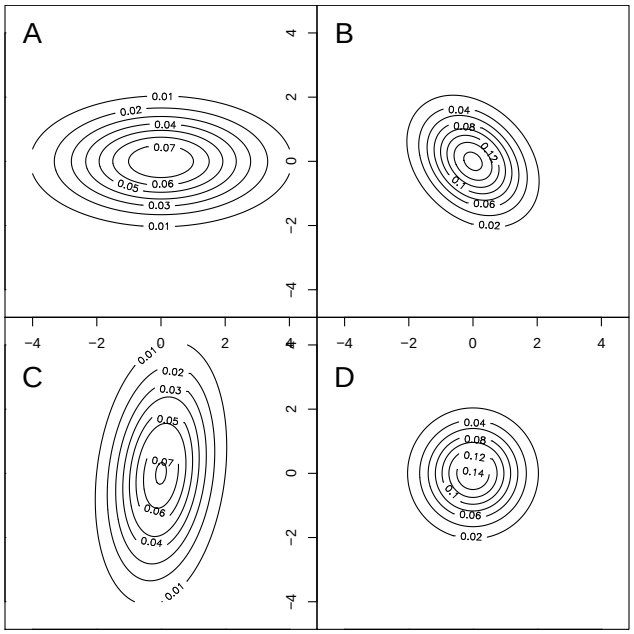



Figure 2: Contour plots

**Solution:** C

# Problem 2 - 15P

We are looking at a linear regression problem. Biologists studied badgers in Switzerland, and these animals mainly eat earthworms. In their exctrements one can find an undigestable part of the earthworm (the muscular stomach), and in order to find out how much energy a badger took in by eating earthworms, the biologists wanted to figure out how the circumference of the muscular stomach can be used to predict to the weight of the earthworm that the badger ate. So they went to collect 143 earthworm, which they weighted and of which they measured the circumference of the muscular stomach.

The earthworm dataset can be loaded as follows:

```
id <- "1nLen1ckdnX4P9n8ShZeU7zbXpLc7qiwt"  # google file ID
d.worm <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id))
```

Look at the data by using both `head(d.worm)` and `str(d.worm)`. The dataset contains the following variables:

- `Gattung`: The species of the worm (L=Lumbricus; N=Nicodrilus; Oc=Octolasion)
- `Nummer`: A worm-specific ID
- `GEWICHT`: The weight of the earthworm
- `FANGDATUM`: The date when the data were collected
- `MAGENUMF`: The circumference of the muscular stomach

## a) (2P)

What is the dimension of the dataset (number of rows and columns)? Which of the variables are qualitative, which are quantitative?

**Solution**:

```
id <- "1nLen1ckdnX4P9n8ShZeU7zbXpLc7qiwt"  # google file ID
d.worm <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id))
head(d.worm)
```

```
##   Gattung Nummer GEWICHT FANGDATUM MAGENUMF
## 1      Oc     32    0.19  23.09.97     1.56
## 2      Oc     34    0.59  23.09.97     1.63
## 3      Oc     48    0.09  23.09.97     1.69
## 4      Oc     55    0.23  23.09.97     1.69
## 5      Oc     41    0.24  23.09.97     1.75
## 6      Oc     24    0.19  23.09.97     1.81
```

```
str(d.worm)
```

```
## 'data.frame':    143 obs. of  5 variables:
##  $ Gattung  : Factor w/ 3 levels "L","N","Oc": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Nummer   : int  32 34 48 55 41 24 39 35 45 27 ...
##  $ GEWICHT  : num  0.19 0.59 0.09 0.23 0.24 0.19 0.26 0.19 0.15 0.34 ...
##  $ FANGDATUM: Factor w/ 3 levels "12.10.97","15.09.97",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ MAGENUMF : num  1.56 1.63 1.69 1.69 1.75 1.81 1.81 1.88 2 2.13 ...
```

The dimension of the dataset is 143 times 5. `Gattung` and `FANGDATUM` are encoded as factor variables, and these are categorical (qualitative). The `Nummer` (ID) is encoded as an integer, but it is actualy also a qualitative varialbe. `Gewicht` and `Magenumf` are quantitative (and one could argue that the date is also quantitative, so this is not counted as wrong here).

## b) (2P)

An important step before fitting a model is to look at the data to understand if the modelling assumptions are reasonable. In a linear regression setup, it is for example recommended to look at the relation of the variables to see if the linearity assumption makes sense. If this is not the case, you can try to transform the variables.

Make a scatterplot of `GEWICHT` (weight) against `MAGENUMF` (circumference of stomach), where you color the points according to the three species (variable `Gattung`).

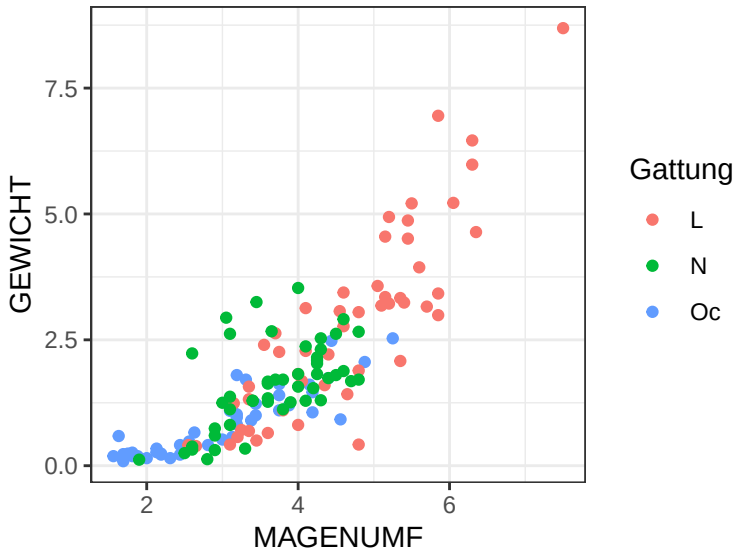
Does this relationship look linear? If not, try out some transformations of `GEWICHT` and `MAGENUMF` until you are happy.

**R-hint**:

```
ggplot(d.worm, aes(x = ..., y = ..., colour = ...)) + geom_point() + theme_bw()
```

**Solution**:

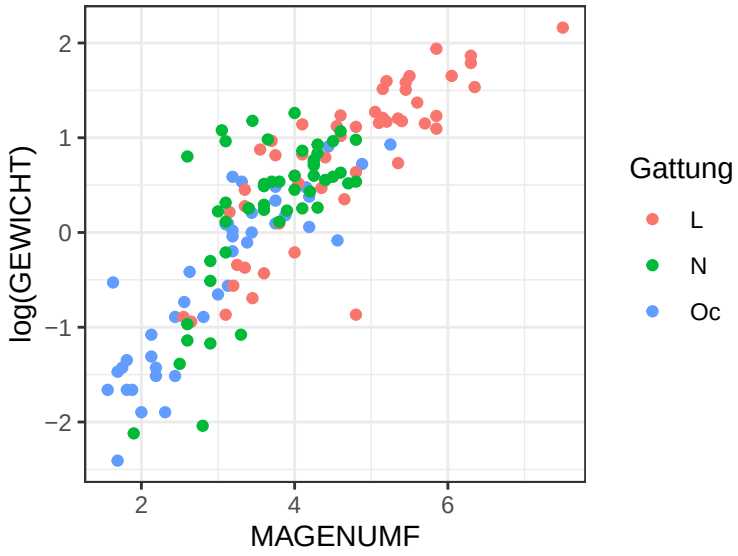
```
library(ggplot2)
ggplot(d.worm, aes(x = MAGENUMF, y = GEWICHT, colour = Gattung)) + geom_point() +
    theme_bw()
```




It is clearly visible that the relation between weight and the stomach circumference is not linear. So it is a good idea to look for some transformations.

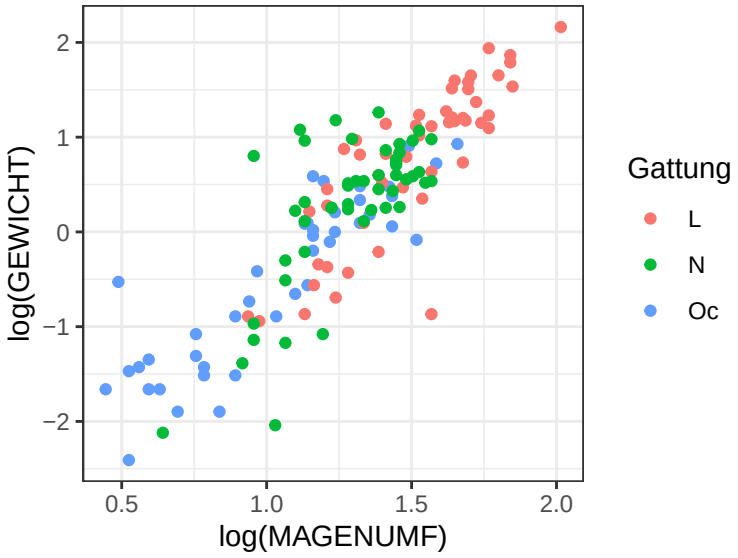
Given that the relation looks somewhat exponential, a reasonable starting point is to use a log-transformation for the response variable and then continue like this for the rest of the exercise:

```
ggplot(d.worm, aes(x = MAGENUMF, y = log(GEWICHT), colour = Gattung)) + geom_point() +
    theme_bw()
```



However, it seems also good (or even better?) to also use the log of `MAGENUMF`:

```
ggplot(d.worm, aes(x = log(MAGENUMF), y = log(GEWICHT), colour = Gattung)) +
    geom_point() + theme_bw()
```




Students came up with other ideas during the exercise session. You can continue with the solution you find here, given that it in fact renders the relation kind of linear.

## c) (3P)

Fit a regression model that predicts the weight (`GEWICHT`) given the circumference of the stomach (`MAGENUMF`) and the species (`Gattung`). **Use the transformed version of the variable(s) from b)**. Use only linear terms that you combine with `+` (no interactions) (1P). After you fitted the models, write down the separate regression models with the estimated parameters for the three species as three separate equations (1P). Is `Gattung` a relevant predictor? (1P)

**R-hints** : `lm()`, `summary()`, `anova()`

**Solution:** (Students that go in with another transformation are given the full points here if they otherwise solve the question right.)

```
r.worm <- lm(log(GEWICHT) ~ MAGENUMF + Gattung, d.worm)
summary(r.worm)
```

```
##
## Call:
## lm(formula = log(GEWICHT) ~ MAGENUMF + Gattung, data = d.worm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.74894 -0.27575  0.02197  0.28513  1.30867
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.53555    0.22147 -11.449   <2e-16 ***
## MAGENUMF     0.71187    0.04529  15.719   <2e-16 ***
## GattungN     0.17801    0.11009   1.617    0.108
## GattungOc   -0.09073    0.12791  -0.709    0.479
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5059 on 139 degrees of freedom
## Multiple R-squared:  0.7367, Adjusted R-squared:  0.731
## F-statistic: 129.6 on 3 and 139 DF,  p-value: < 2.2e-16
```

```
anova(r.worm)
```

```
## Analysis of Variance Table
##
## Response: log(GEWICHT)
##            Df Sum Sq Mean Sq  F value  Pr(>F)
## MAGENUMF    1 97.802  97.802 382.0947 < 2e-16 ***
## Gattung     2  1.725   0.862   3.3691 0.03725 *
## Residuals 139 35.579   0.256
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The anova table shows some evidence that `Gattung` could be a relevant predictor, that is, the intercept is dependent on the species ($p = 0.037$). Note, however, that other transformations may lead to somewhat different conclusions. The three separate models are given as

- Lumbricus: $\log(weight)_i$ = -2.54 + -0.09 $\cdot$MAGENUMF$_i + \varepsilon_i$
- Nicodrilus: $\log(weight)_i$ = -2.54 + 0.71 + -0.09 $\cdot$MAGENUMF$_i + \varepsilon_i$
- Octolasion: $\log(weight)_i$ = -2.54 + 0.18 + -0.09 $\cdot$MAGENUMF$_i + \varepsilon_i$

So the three models differ in their intercept value. However, we enforce the slope for `GEWICHT` (weight) to be the same for all three species.

## d) (2P)

In question c) it was assumed that there is no interaction between the species and `MAGENUMF` to predict the weight of a worm. Test whether an interaction term would be relevant by fitting an appropriate model.

**Solution:**

```
r.worm2 <- lm(log(GEWICHT) ~ MAGENUMF * Gattung, d.worm)
anova(r.worm2)
```

```
## Analysis of Variance Table
##
## Response: log(GEWICHT)
##                   Df Sum Sq Mean Sq  F value  Pr(>F)
## MAGENUMF           1 97.802  97.802 386.5547 < 2e-16 ***
## Gattung            2  1.725   0.862   3.4084 0.03592 *
## MAGENUMF:Gattung   2  0.917   0.458   1.8112 0.16735
## Residuals        137 34.662   0.253
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The $p$-value for the interaction term is not very small ($p = 0.167$), so there is no evidence that we need it. Although remember that a large $p$-value does not mean that the effect does not exist (we cannot prove the null hypothesis) - the only thing we can say is that we cannot reject it.

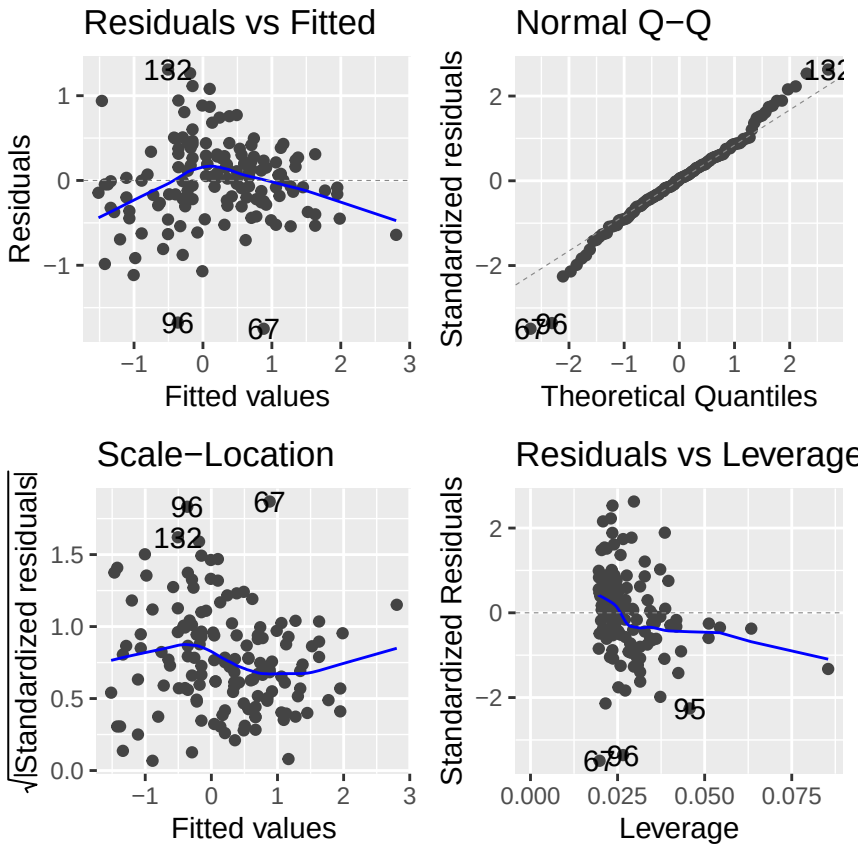
## e) (2P)

Carry out a residual analysis using the `autoplot()` function from the `ggfortify` package. Use the model without interaction term.

- Do you think the assumptions are fulfilled? Explain why or why not.
- Compare to the residual plot that you would obtain when you would not use any variable transformations to fit the regression model.

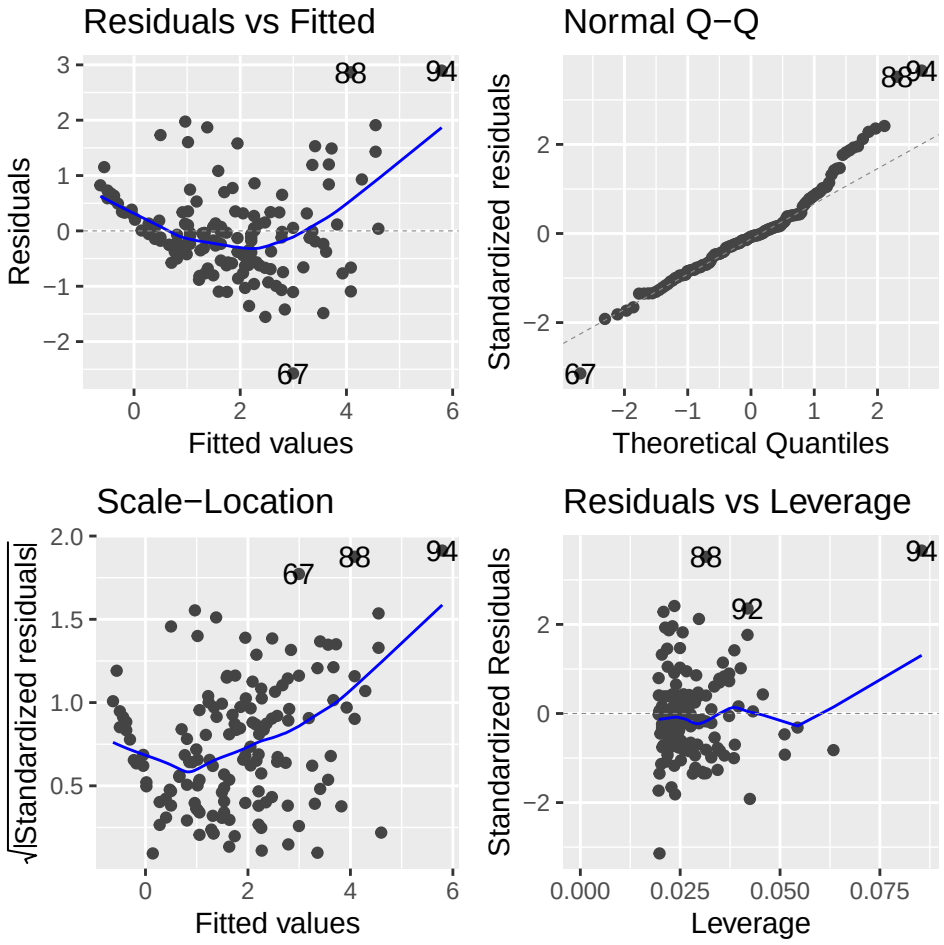
**Solution:** In the residual plot below we see that the assumptions seem more or less ok, although there is some kind of pattern in the Tukey-Anscombe plot (upper left). Perhaps it would be wise to continue looking for better transformations.

```
library(ggfortify)
autoplot(r.worm)
```




If we would use untransformed variables, we would see some pattern in the residuals. In particular the Tukey-Anscombe plot shows a curvature:

```
r.worm3 <- lm(GEWICHT ~ Gattung + MAGENUMF, d.worm)
autoplot(r.worm3)
```




## f) (2P)

- Why is it important that we carry out a residual analysis, i.e., what happens if our assumptions are not fulfilled?
- Mention at least one thing that you could do with your data / model in case of violated assumptions.

**Solution:**

- To check assumptions that we put into the model. If these are violated, we may draw invalid conclusions. This is particularly problematic if we want to do inference.
- Ideas are: Use another model, transform variables, be careful with your conclusions, check if there are outliers,...

## g) (2P) - Multiple choice

Given a null hypothesis ($H_0$), an alternative hypothesis ($H_1$) and an observed result with an associated $p$-value. Think of the example where $H_0$ is that a slope parameter in a regression model is $\beta = 0$. Which of the following statements is correct?

(i) The $p$-value is the probability that $H_0$ is true.
(ii) The $p$-value is the probability that $H_1$ is true.
(iii) $(1 - p)$ is the probability that $H_1$ is true.

(iv) $p$ is the probability to observe a data symmary under the null hypothesis ($H_0$) that is at least as extreme as the one observed.

List of answers:

**Solution:**

FALSE - FALSE - FALSE - TRUE

# Problem 3 - 16P

In this problem, we will use a dataset from the Wimbledon tennis tournament for Women in 2013. We will predict the result for player 1 (win=1 or loose=0) based on the number of aces won by each player and the number of unforced errors commited by both players. The data set is a subset of a data set from https://archive.ics.uci.edu/ml/datasets/Tennis+Major+Tournament+Match+Statistics, see that page for information of the source of the data.

The files can be read using the following code.

```
# read file
id <- "1GNbIhjdhuwPOBr0Qz82JMkdjUVBuSoZd"
tennis <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id), header = T)
```

We will first create a logistic regression model where the probability to win for player 1 has the form

$$P(Y_i = 1|\mathbf{X} = \boldsymbol{x}_i) = p_i = \frac{e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}}{1 + e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\beta_3 x_{i3}+\beta_4 x_{i4}}} ,$$

where $x_{i1}$ is the number of aces for player 1 in match $i$, $x_{i2}$ is the number of aces for player 2 in match $i$, and $x_{i3}$ and $x_{i4}$ are the number of unforced errors commited by player 1 and 2 in match $i$.

## a) (1P)

Use the above expression to show that $\text{logit}(p_i) = \log(\frac{p_1}{1-p_i})$ is a linear function of the covariates.

## b) (1P)

The model above has been fitted and gives the following output. Interpret the effect of $\beta_1$, i.e. how will one more ace for player 1 affect the result of the tennis match?

```
r.tennis = glm(Result ~ ACE.1 + ACE.2 + UFE.1 + UFE.2, data = tennis, family = "binomial")
summary(r.tennis)
```

```
##
## Call:
## glm(formula = Result ~ ACE.1 + ACE.2 + UFE.1 + UFE.2, family = "binomial",
##     data = tennis)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0517  -0.8454   0.3725   0.8773   2.0959
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept) -0.02438    0.59302  -0.041 0.967211
## ACE.1        0.36338    0.10136   3.585 0.000337 ***
## ACE.2       -0.22388    0.07369  -3.038 0.002381 **
## UFE.1       -0.09847    0.02840  -3.467 0.000527 ***
## UFE.2        0.09010    0.02479   3.635 0.000278 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 163.04  on 117  degrees of freedom
## Residual deviance: 124.96  on 113  degrees of freedom
## AIC: 134.96
##
## Number of Fisher Scoring iterations: 4
```

**Solution: a-b**

- Let us write $\eta = \beta_0 + \boldsymbol{x_i}^T\boldsymbol{\beta}$

$$\begin{aligned}\text{logit}(p_i) &= \log(\frac{p_i}{1-p_i}) = \log(\frac{\frac{exp(\eta)}{1+exp(\eta)}}{1-\frac{exp(\eta)}{1+exp(\eta)}}) = \log(\frac{\frac{exp(\eta)}{1+exp(\eta)}}{\frac{1+exp(\eta)-exp(\eta)}{1+exp(\eta)}}) \\ &= \log(\frac{exp(\eta)}{1}) = \eta \\ &= \eta\end{aligned}$$

- The logit function is the log odds, that is, the log of the probability of sucess divided by probability of failure. So increasing the number of aces by 1, we see that the odds will increase by

$$\frac{odds(x_1+1)}{odds(x_1)} = \frac{exp(\beta_0+\beta_1(x_1+1)+\beta_2x_2+\beta_3x_3+\beta_4x_4)}{exp(\beta_0+\beta_1x_1+\beta_2x_2+\beta_3x_3+\beta_4x_4)} = exp(\beta_1) = exp(0.36) = 1.43.$$

This means that for each ace player 1 manages, she will increase her odds of winning with 1.43.

The same interpretation goes for the other coefficients, where the negative coefficients (`ACE.2` and `UFE.1`) gives a reduced odds for player 1 winning, while the positive coefficients gives an increased odds.

## c) (4P)

We will now reduce the number of covariates in our model by looking at the difference between aces performed by player 1 and 2 and the difference in unforced errors made by the players. Use the following code to create these variables and divide the data into a train set and a test set.

```
# make variables for difference
tennis$ACEdiff = tennis$ACE.1 - tennis$ACE.2
tennis$UFEdiff = tennis$UFE.1 - tennis$UFE.2

# divide into test and train set
n = dim(tennis)[1]
n2 = n/2
set.seed(1234)  # to reproduce the same test and train sets each time you run the code
train = sample(c(1:n), replace = F)[1:n2]
tennisTest = tennis[-train, ]
tennisTrain = tennis[train, ]
```

- Use these variables to fit a logistic regression model on the form `Result` $\sim$ `ACEdiff` + `UFEdiff` on your training set.
- Using a 0.5 cutoff as decison rule, we classify an observation with covariates $\boldsymbol{x}$ as "Player 1 wins" if $\hat{P}(Y = 1|\boldsymbol{x}) > 0.5$. Write down the formula for the class boundary between the classes (results) using this decision rule. The boundary should be of the form $x_2 = bx_1 + a$.
- Make a plot with the training observations and and then add a line that represents the class boundary. Hint: in `ggplot` points are added with `geom_point` and a line with `geom_abline(slope=b, intercept=a)`, where $a$ and $b$ comes from your class boundary.
- Use your fitted model to predict the result of the test set. Make a confusion table using a 0.5 cutoff and calculate the sensitivity and specificity.

**Solution:**

```
# fit model
r.tennis2 = glm(Result ~ ACEdiff + UFEdiff, data = tennisTrain, family = "binomial")
summary(r.tennis2)
```

```
##
## Call:
## glm(formula = Result ~ ACEdiff + UFEdiff, family = "binomial",
##     data = tennisTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8546  -0.8968   0.4204   0.8247   1.9382
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.28272    0.31175   0.907  0.36447
## ACEdiff      0.22355    0.07959   2.809  0.00497 **
## UFEdiff     -0.08607    0.02832  -3.039  0.00237 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 80.959  on 58  degrees of freedom
## Residual deviance: 63.476  on 56  degrees of freedom
## AIC: 69.476
##
## Number of Fisher Scoring iterations: 4
```

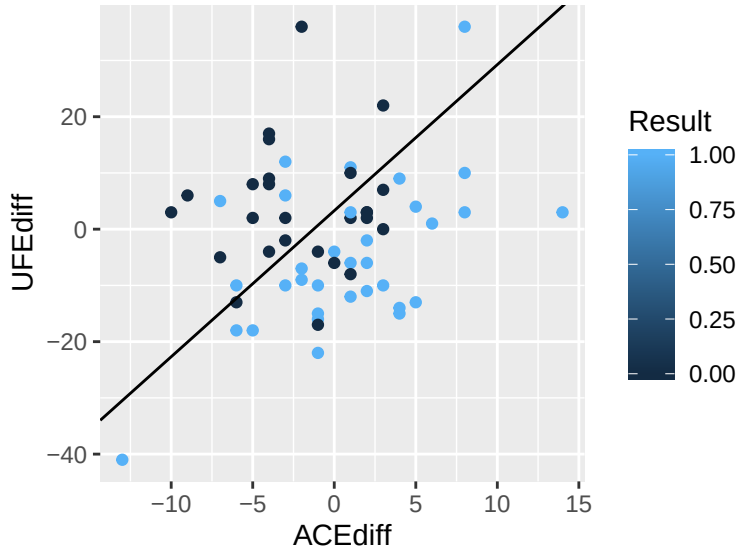
- The decision boundary with decision rule 0.5 are found using the following

$$\hat{P}(Y = 1|X) = \frac{exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2)}{1 + exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2)} = 0.5$$
$$exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2) = 0.5 + 0.5 exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2)$$
$$(1 - 0.5) exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2) = 0.5$$
$$exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2) = 1$$
$$\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 = \log(1) = 0$$
$$x_2 = \frac{-\hat{\beta}_0 - \hat{\beta}_1 x_1}{\hat{\beta}_2} = \frac{-0.336 - 0.314 x_1}{-0.090} = 3.73 + 3.49 x_1$$

- Plot decision boundary and predict test set

```r
slopeGLM = -summary(r.tennis2)$coefficients[2, 1]/summary(r.tennis2)$coefficients[3,
    1]
interceptGLM = -summary(r.tennis2)$coefficients[1, 1]/summary(r.tennis2)$coefficients[3,
    1]

ggplot(tennisTrain, aes(x = ACEdiff, y = UFEdiff, color = Result)) + geom_point() +
    geom_abline(intercept = interceptGLM, slope = slopeGLM)
```




```r
# predict
pred = predict(r.tennis2, newdata = tennisTest, type = "response")

# confusion table - columns represents predictions
t = table(tennisTest$Result, round(pred))
t
```

```
##
##      0  1
##   0 22  7
##   1  6 24
```

```r
# Sens + spes
sens = t[2, 2]/(t[2, 1] + t[2, 2])
spes = t[1, 1]/(t[1, 1] + t[1, 2])

c(sens, spes)
```

```
## [1] 0.8000000 0.7586207
```

## d) (1P)

Next, we will use LDA and QDA to classify the result using the same covariates (`ACEdiff` and `UFEdiff`) from the tennis data. In linear discriminant analysis with $K$ classes, we assign a class to a new observation based on the posterior probability

$$P(Y = k|\mathbf{X} = \boldsymbol{x}) = \frac{\pi_k f_k(\boldsymbol{x})}{\sum_{l=1}^{K} \pi_l f_l(\boldsymbol{x})},$$

where

$$f_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_k})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_k})}.$$

- Explain what $\pi_k$, $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}$ and $f_k(x)$ are in our `tennis` problem with the two covariates (no calculations, only explanations).

## e) (3P)

In a two class problem ($K = 2$) the decision boundary for LDA between class 0 and class 1 is where $x$ satisfies

$$P(Y = 0|\mathbf{X} = \boldsymbol{x}) = P(Y = 1|\mathbf{X} = \boldsymbol{x}).$$

- (1P) Show that we can express this as

$$\delta_0(\boldsymbol{x}) = \delta_1(\boldsymbol{x}), \tag{1}$$

  where

$$\delta_k(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \log\pi_k; \quad k \in \{0, 1\}. \tag{2}$$

- (1P) We use the rule to classify to class 1 for an observation with covariates $\boldsymbol{x}$ if $\hat{P}(Y = 1 \mid \boldsymbol{x}) > 0.5$. Write down the formula for the class boundary between the classes. Hint: formulate it as $ax_1+bx_2+c = 0$ and solve for $x_2$. Use R for the calculations.
- (1P) Make a plot with the training observations and the class boundary. Add the test observations to the plot (different markings). Hint: in `ggplot` points are added with `geom_points` and a line with `geom_abline(slope=b, intercept=a)` where $a$ and $b$ comes from your class boundary.

## f) (3P)

- (1P) Perform LDA on the training data (using the `lda()` function in R).
- (1P) Use your model to classify the results of the test set. Make the confusion table for the test set when using 0.5 as cut-off.
- (1P) Calculate the sensitivity and specificity on the test set.

**Solution: d-f**

- Here $\pi_k$ is the prior probability that a randomly chosen observation comes from the $k$th class. We assume that the observations of class $k$ comes from a multivariate normal distribution $f_k(x)$, where $\mu_k$ is the mean of the $k$th class and $\boldsymbol{\Sigma}$ is the variance.
- Showing that $P(Y = 0|\mathbf{X} = \mathbf{x}) = P(Y = 1|\mathbf{X} = \mathbf{x})$ can be expressed as $\delta_0(\boldsymbol{x}) = \delta_1(\boldsymbol{x})$.

$$\begin{aligned}
\frac{\pi_0 f_0(x)}{\pi_0 f_0(x) + \pi_1 f_1(x)} &= \frac{\pi_1 f_1(x)}{\pi_0 f_0(x) + \pi_1 f_1(x)} \\
\pi_0 f_0(x) &= p_1 f_1(x) \\
\pi_0 \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}} exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_0})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_0})) &= \pi_1 \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}} exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_1})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_1})) \\
\log(\pi_0) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_0})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_0}) &= \log(\pi_1) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_1})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_1}) \\
\log(\pi_0) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_0} - \frac{1}{2}\boldsymbol{\mu_0}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_0} &= \log(\pi_1) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} - \frac{1}{2}\boldsymbol{\mu_1}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} \\
\delta_0(\boldsymbol{x}) &= \delta_1(\boldsymbol{x})
\end{aligned}$$

- LDA on training set

```
library(MASS)
ldaMod = lda(Result ~ ACEdiff + UFEdiff, data = tennisTrain)
```

- Class boundary and plot

$$\delta_0(\boldsymbol{x}) - \delta_1(\boldsymbol{x}) = 0$$
$$\log(\pi_0) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_0} - \frac{1}{2}\boldsymbol{\mu_0}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_0} - \log(\pi_1) + \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} + \frac{1}{2}\boldsymbol{\mu_1}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} = 0$$
$$\log(\pi_0) + \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_0} - \frac{1}{2}\boldsymbol{\mu_0}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_0} - \log(\pi_1) - \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} + \frac{1}{2}\boldsymbol{\mu_1}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} = 0$$
$$\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_0} - \boldsymbol{\mu_1}) - \frac{1}{2}\boldsymbol{\mu_0}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_0} + \frac{1}{2}\boldsymbol{\mu_1}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu_1} + \log(\frac{\pi_0}{\pi_1}) = 0$$
$$ax_1 + bx_2 + c = 0$$
$$x_2 = -c/b - a/bx_1$$

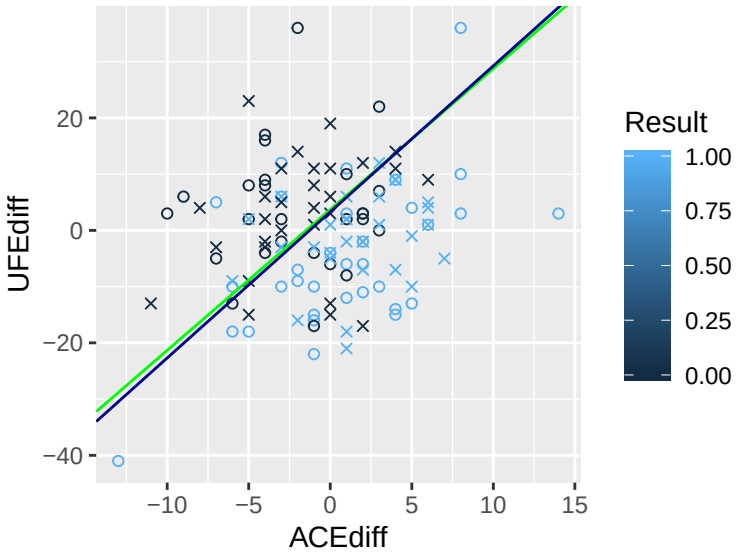
where $c$ is all terms not involving $\boldsymbol{x}$, and $a$ and $b$ are found from solving $\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu_0} - \boldsymbol{\mu_1})$.

```
pi0 = ldaMod$prior[1]
pi1 = ldaMod$prior[2]
mu0 = ldaMod$means[1, ]
mu1 = ldaMod$means[2, ]
sigma0 = cov(tennisTrain[which(tennisTrain$Result == 0), c(8, 9)])
sigma1 = cov(tennisTrain[which(tennisTrain$Result == 1), c(8, 9)])
n0 = dim(tennisTrain[which(tennisTrain$Result == 0), ])[1]
n1 = dim(tennisTrain[which(tennisTrain$Result == 1), ])[1]
sigma = ((n0 - 1) * sigma0 + (n1 - 1) * sigma1)/(n0 + n1 - 2)

ab = solve(sigma) %*% (mu0 - mu1)
a = ab[1]
b = ab[2]
c = -0.5 * t(mu0) %*% solve(sigma) %*% mu0 + 0.5 * t(mu1) %*% solve(sigma) %*%
    mu1 + log(pi0/pi1)

slopeLDA = -a/b
interceptLDA = -c/b

# make plot - add glm-boundary to compare. Not required by the students.
LDAplot = ggplot(data = tennisTrain, aes(x = ACEdiff, y = UFEdiff, colour = Result)) +
    geom_point(pch = 1)
LDAplot + geom_point(data = tennisTest, pch = 4) + geom_abline(slope = slopeLDA,
    intercept = interceptLDA, colour = "green") + geom_abline(slope = slopeGLM,
    intercept = interceptGLM, colour = "darkblue")
```




The green line shows the decision boundary for LDA while the dark blue line shows the decision boundary for GLM. They are very similar. Circles represents the training observations while crosses represents the test observations. We see that the lines separates well for both train and test set.

- Predicting classes of test set and make confusion table with sensitivity and specificity. Columns shows predictions.

```
LDApred = predict(ldaMod, newdata = tennisTest)$class
tLDA = table(tennisTest$Result, LDApred)
tLDA
```

```
##    LDApred
##       0  1
##   0 20  9
##   1  5 25
```

```
sensLDA = tLDA[2, 2]/(tLDA[2, 1] + tLDA[2, 2])
spesLDA = tLDA[1, 1]/(tLDA[1, 1] + tLDA[1, 2])
c(sensLDA, spesLDA)
```

```
## [1] 0.8333333 0.6896552
```

Both sensitivity and specificity are high and we see from the confusion table that most observations are classified correctly. The confunsion table with our data is exactly the same, which makes sense since the decision boundary is so similar for glm and LDA.

## g) (2P)

- Perform QDA on the training set. What is the difference between LDA and QDA?
- Make the confusion table for the test set when using 0.5 as cut-off. Calculate the sensitivity and specificity on the test set.

**Solution:**

QDA assumes that the classes have different covariance matrices and gives a quadratic decision boundary (as we see below).

```
qdaMod = qda(Result ~ ACEdiff + UFEdiff, data = tennisTrain)
QDApred = predict(qdaMod, newdata = tennisTest)$class
```

```
tQDA = table(tennisTest$Result, QDApred)
tQDA
```

```
##    QDApred
##       0  1
##   0 20  9
##   1  6 24
```

```
sensQDA = tQDA[2, 2]/(tQDA[2, 1] + tQDA[2, 2])
spesQDA = tQDA[1, 1]/(tQDA[1, 1] + tQDA[1, 2])
c(sensQDA, spesQDA)
```

```
## [1] 0.8000000 0.6896552
```

## h) (1P)

Figure 3 shows the decision boundary for QDA, where observations falling into the red area will be classified as 0 (loose), and observations in the blue area will be classified as 1 (win). Circles represents observations from the train set, while crosses represents observations from the test set.

- Compare this plot with your corresponding plots for glm and LDA. Would you prefer glm, LDA or QDA for these data? Justify your answer this based on the results from the confusion matrices and in light of the decision boundaries meaning for your tennis-covariates.

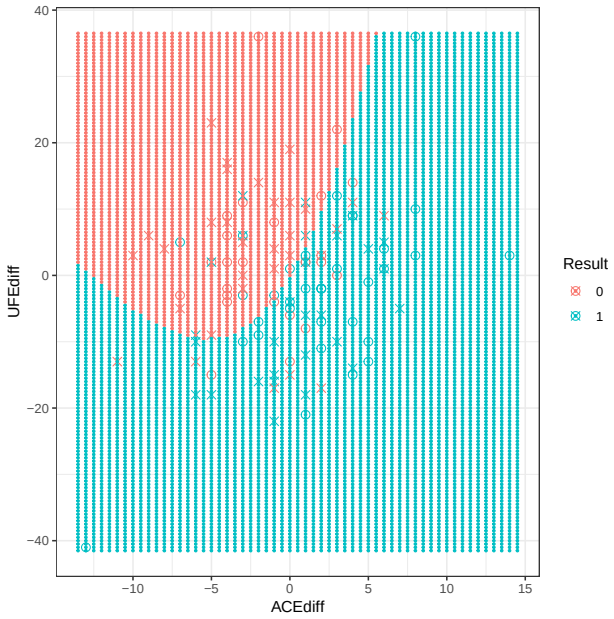



Figure 3: Figure 3

**Solutions**

From the confusion table with corresponding sensitivity and specificity, we see that also QDA performs very similar to the methods above. It has only one more false prediction for the true 0 results, giving it a slightly poorer specificity. The decision boundary shows that only observations in the upper left half are classified as 0, which could make much sense since this corresponds to player 1 having many unforced errors and few aces and is hence loosing. However, the lower left corner and upper right corner will always be classified as a win for player 1 with QDA, but these regions corresponds to player 1 having few aces and less unforced

errors than player 2 and vise verca. One could argue that these regions are problematic for both players, and the result will vary. For this reason, perhaps LDA and glm are better models, as they have linear decision boundaries in these regions.

# Problem 4 (9P)

## a) (2P)

Remember the formula for the $K$-nearest neighbour regression curve to estimate at a covariate value $x_0$,

$$\hat{f}(x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} y_i \ ,$$

where for a given integer $K$, the KNN regression first identifies the $K$ points in the training data that are closest (Euclidean distance) to $x_0$, represented by $\mathcal{N}_0$. It then estimates the regression curve at $x_0$ as the average of the response values for the training observations in $\mathcal{N}_0$.

Given the set of possible values for $K$ in the KNN regression problem specified in a), explain how 10-fold cross validation is performed, and specify which error measure you would use. Your answer should include a formula to specify how the validation error is calculated.

**Solution:**

We look at a set of possible values for $K$ in the KNN, for example $K = (1, 2, 3, 5, 7, 9, 15, 25, 50, 100)$. First we devide the data into a training set and a test set - and lock away the test set for model evaluation.

We work now with the training set.

10-fold CV: we divide the training data randomly into 10 folds of size $n/10$ each and call the folds $j = 1$, to $j = 10$.

For each value of $K$ we do the following.

For $j = 1, \ldots, 10$:

- use the $9n/10$ observations from the folds except fold $j$ to define the $K$-neighbourhood $\mathcal{N}_0$ for each of the observations in the $j$th fold
- the observations in the $j$th fold is left out and is the validation set, there are $n/10$ observations - and we denote them $(x_{0jl}, y_{0jl})$,
- we then estimate $\hat{f}(x_{0jl}) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} y_i$.
- We calculate the error in the $j$th fold of the validation set as $\sum_l (y_{0jl} - \hat{f}(x_{0jl}))^2$ where the $j$ is for the validation fold

The total error on the validation set is thus the validation MSE$=\frac{1}{n} \sum_{j=1}^{10} \sum_l (y_{0jl} - \hat{f}(x_{0jl}))^2$.

So, for each value of $K$ we get an estimate of the validation MSE. Finally, we choose the value of $K$ that gives the lowest validation MSE.

## b) (2P) - Multiple choice

Which statements about validation set approach, $k$-fold cross-validation (CV) and leave-one-out cross validation (LOOCV) are true and which are false? Say for *each* of them if it is true or false.

(i) 5-fold CV will generally lead to more bias, but less variance than LOOCV in the estimated prediction error.

(ii) The validation set-approach is computationally cheaper than 10-fold CV.

(iii) The validation set-approach is the same as 2-fold CV.
(iv) LOOCV is always the cheapest way to do cross-validation.

**Solution:**

TRUE, TRUE, FALSE, FALSE

(i) and (ii) are correct. (iii) is wrong, because in 2-fold CV we would fit the model twice (once with each half of the data), while the validation set approach uses only one half of the data to fit the model. (iv) is wrong, because LOOCV is actually very expensive - expect for linear regression, where a formula exists.

## c) (1P)

We are now looking at a bootstrap example. Assume you want to fit a model that predicts the probability for coronary heart disease (`chd`) from systolic blood pressure (`sbp`) and sex (0=female, 1=male). Load the data in R as follows

```
id <- "1I6dk1fA4ujBjZPo3Xj8pIfnzIa94WKcy"  # google file ID
d.chd <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id))
```

and perform a logistic regression with `chd` as outcome and `sbp` and `sex` as covariates. What is the probability of chd for a male with a sbp=140 in the given dataset?

**Solution:** Doing the regression

```
id <- "1I6dk1fA4ujBjZPo3Xj8pIfnzIa94WKcy"  # google file ID
d.chd <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id))
r.glm <- glm(chd ~ sbp + sex, d.chd, family = "binomial")
summary(r.glm)
```

```
##
## Call:
## glm(formula = chd ~ sbp + sex, family = "binomial", data = d.chd)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0647  -0.8697  -0.7749   1.4191   1.7794
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.386252   0.790657  -3.018  0.00254 **
## sbp          0.011337   0.006273   1.807  0.07075 .
## sex          0.322764   0.235786   1.369  0.17103
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 427.61  on 349  degrees of freedom
## Residual deviance: 422.63  on 347  degrees of freedom
## AIC: 428.63
##
## Number of Fisher Scoring iterations: 4
```

Deriving the probability:

To get the probability you can now either plug in the estimates and calculate $p = exp(\eta)/(1 + exp(\eta))$ with $\eta = \beta_0 + \beta_1 \cdot 140 + \beta_2 \cdot 1$, or directly use the predict function in R:

```
newdata <- data.frame(sbp = 140, sex = 1)
(pred.p <- predict(r.glm, newdata, type = "response"))
```

```
##         1
## 0.3831131
```

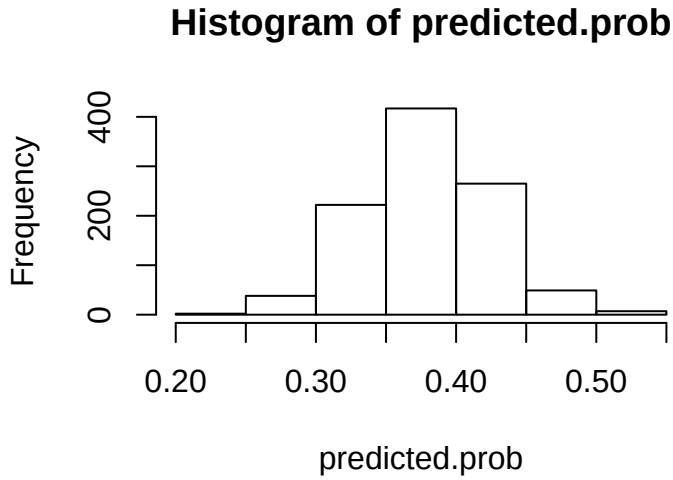
## d) (4P)

We now use the bootstrap to estimate the uncertainty of the probability derived in b). Proceed as follows:

- Use $B = 1000$ bootstrap samples.
- In each iteration, derive and store the estimated probability for `chd`, given `sbp`=140 and `sex`=male.
- From the set of estimated probabilities, derive the standard error.
- Also derive the 95% confidence interval for the estimates, using the bootstrap samples.
- Interpret what you see.

**Solution:**

```
B <- 1000
predicted.prob <- rep(NA, B)
for (ii in 1:B) {
    d.boot <- d.chd[sample(1:nrow(d.chd), size = nrow(d.chd), replace = TRUE),
        ]
    r.glm.boot <- glm(chd ~ sbp + sex, data = d.boot)
    newdata <- data.frame(sbp = 140, sex = 1)
    predicted.prob[ii] <- predict(r.glm.boot, newdata, type = "response")
}
hist(predicted.prob)
```




The standard error is just the standard deviation of the sample. The 95% CI can be obtained by using the interval ranging from the 2.5% to the 97.5% quantile of the samples.

```
sd(predicted.prob)
```

```
## [1] 0.04521481
```

```r
quantile(predicted.prob, probs = c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.2882243 0.4686617
```

Interpretation: The predicted probability for chd for a male with sbp=140 is 0.383, the respective standard error is 0.045 and the 95% CI is ranging from 0.288 to 0.469