TMA4268 V2022 Exam

TMA4268 Statistical Learning V2022

Stefanie Muff, Department of Mathematical Sciences, NTNU

June 3, 2022

Warming up

Problem 1 (Fill-in-the-blank text, 5P)

Read the whole text and fill in the blanks such that the whole text makes sense (you might only understand which answer is correct after you continued reading):

In our course we were mainly discussing supervised statistical learning methods, broadly divided into regression and *classification* (prediction, inference, supervised, unsupervised,...) problems. We were thereby discriminating between models that we use for prediction versus inference. In the latter case, we prefer to use *parametric* (non-parametric, flexible, classification, least squares) models, for example *logistic regression* (hierarchical clustering, linear regression, support vector machines, neural networks, KNN classification, K-means clustering, local regression) for a classification problem. However, when the aim is pure prediction, we are free to choose any model that is giving good predictions. *Non-parametric* (regression, classification, clustering, supervised) methods are very flexible and thus more complex and *less interpretable* (better interpretable, more appealing, better suited for inference) than linear models.

An important topic that we were discussing throughout the course was the bias-variance trade-off. The main idea is that the *test error/ reducible error* (training error, prediction bias, irreducible error) is smallest for a model that balances bias and variance. With "variance'' we here mean *the variance of the estimated function* (the residual variance, the variance of the covariates, the variance of the response, the reducible error)

In the context of the bias-variance trade-off we were also discussing variable selection and methods based on shrinkage. Lasso and ridge regression were two shrinkage methods we learned about. Both Lasso and ridge regression are *less flexible* (more flexible, more interpretable, less biased, easier to use) compared to least squares. Lasso tends to lead to *simpler models* (more accurate models, models with lower prediction error, better classification models, more flexible models) than ridge regression, thus Lasso is suitable for variable selection. In contrast to *AIC minimization* (linear regression, ridge regression, the boostrap, boosted regression trees), for example, Lasso does not suffer from model selection bias and should thus be preferred over other methods when the aim is to select a subset of variables.

Problem 2 (Dropdown menu, 2P)

Choose from the dropdown menu below the correct optimization method that corresponds to the problem formulation:

a)

Maximise M by choosing $\beta_0, \beta_1, \ldots, \beta_p$ subject to $\sum_{j=1}^{p} \beta_j^2 = 1$, $y_i(\beta_0 + \beta_1 x_{i1} + \ldots \beta_p x_{ip} \geq M(1 - \epsilon_i) \quad \forall i = 1, \ldots, n$, $\epsilon_i > 0, \sum_i \epsilon_i \leq C$ for some $C > 0$. (*Support Vector Classifier*, logistic regression, regression tree, least squares regression, Lasso)

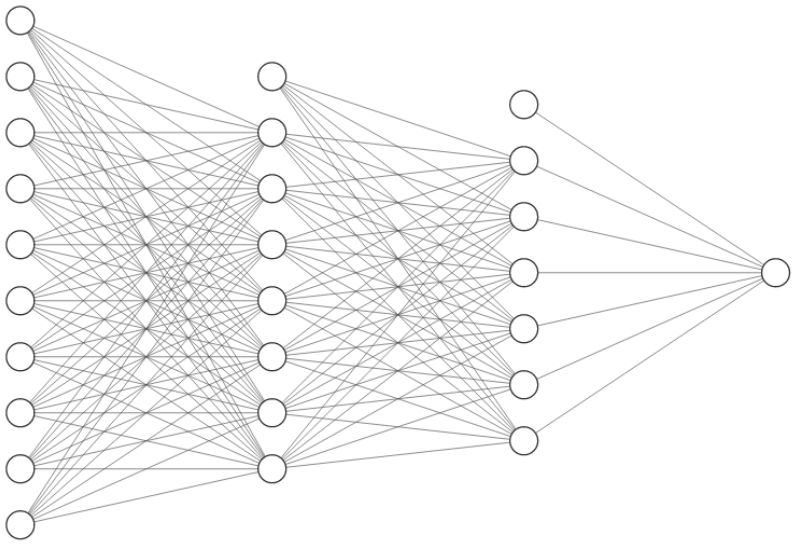
b)

$\text{argmax}_\beta \sum_{i=1}^{n} (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$, where $p_i = \frac{1}{1+\exp(-(\beta_0+\beta_{i1}+\ldots+\beta_p x_{ip}))}$. (Support Vector Classifier, *logistic regression*, classification tree, least squares regression, Lasso, feed-forward neural network)

Problem 3 (6P)

a) (4P)

Look at the following graphical description of a neural network:



(i) (2P) Write down the equation that describes how the input is related to output in this network, using ReLU activation functions in the two hidden layers, and sigmoid activation function in the output layer, assuming that there is one bias node in each of the layers (except the output layer).
(ii) (1P) What kind of task can this network be used for?
(iii) (1P) How many parameters are estimated in this network?

Solution: (i)

$$y_1(\mathbf{x}) = \left(1 + \exp(-\beta_0 - \sum_{m=1}^{6} \beta_m \max(\gamma_{0m} + \sum_{l=1}^{7} \gamma_{lm} \max(\alpha_{0l} + \sum_{j=1}^{9} \alpha_{jl} x_j, 0), 0)) \right)^{-1} .$$

The students might also want to choose other notation, e.g.

$$f(x) = \sigma_3(W_3 \sigma_2(W_2 \sigma_1(W_1 x + \beta_1) + \beta_2) + \beta_3) \ ,$$

but then we need the appropriate dimensions and explanation of the notation (i.e., what is σ_1, W_1 etc).

(ii) Classification into two categories. Note: The point is also given if the student gives a concrete example, e.g. "to classify X-ray images into healthy/unhealthy". However, prediction of a continuous outcome is not correct, because the sigmoid activation function is used in the output layer.

(iii) $10 \cdot 7 + 8 \cdot 6 + 7 = 125$ (only right or wrong).

b) (2P)

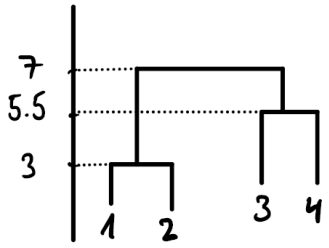
We have four observations for which we know the distance matrix in Euclidean space:

$$\begin{bmatrix} 0 & 3 & 5 & 7 \\ 3 & 0 & 6 & 4 \\ 5 & 6 & 0 & 5.5 \\ 7 & 4 & 5.5 & 0 \end{bmatrix}.$$

Based on this dissimilarity matrix, sketch the dendogram that results from hierarchical clustering using complete linkage. On the plot, indicate the height where each fusion occurs, as well as the observastions that correspond to the leafs in the dendogram (enumerated as 1, 2, 3, 4).

Solution

Complete linkage means that the maximal intercluster dissimilarity between all pairs in a cluster are used. The result therefore looks as follows:



-1P is deducted for each mistake (including errors on the y-axis, or when the axis is missing).

Problem 4 – Data analysis 1 (17P)

In this data analysis problem we are using a dataset that is very similar to the bodyfat data we used in our course. The bodyfat dataset we use here was taken from kaggle https://www.kaggle.com/datasets/fedesoriano/body-fat-prediction-dataset . When clicking on the link you will also find a detailed description of the variables

Load the dataset and create a training and a test set using the following code:

```
id <- "1kGOLsnKAOUq2lWKlMjhAF8h71scOWcLO"  # google file ID
d.bodyfat <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id))[, -c(1)]

set.seed(1234)
training_set_size <- floor(0.8 * nrow(d.bodyfat))

samples <- sample(1:nrow(d.bodyfat), training_set_size, replace = F)
d.body.train <- d.bodyfat[samples, ]
d.body.test <- d.bodyfat[-samples, ]
```

Before you get started, it is smart to make yourself familiar with the data. Use `str(d.body.train)`, `summary(d.body.train)` or other functions or graphical tools you learned about in the course.

a)

(i) (1P) Fit a linear regression model on the training data, with `BodyFat` as the response variable. Use all predictors linearly plus a quadratic term for `Abdomen` (but no interactions)

(ii) (1P) Report and interpret R^2, and relate it to the adjusted R^2. What does the difference tell you?

(iii) (1P) Do a residual analysis using the Tukey-Anscombe plot and the QQ-diagram. Explain which plot addresses which assumption(s) and whether the respective assumptions are met here.

Solution (i)

```
formula = BodyFat ~ . + I(Abdomen^2)
r.lm <- lm(formula, d.body.train)
summary(r.lm)
```

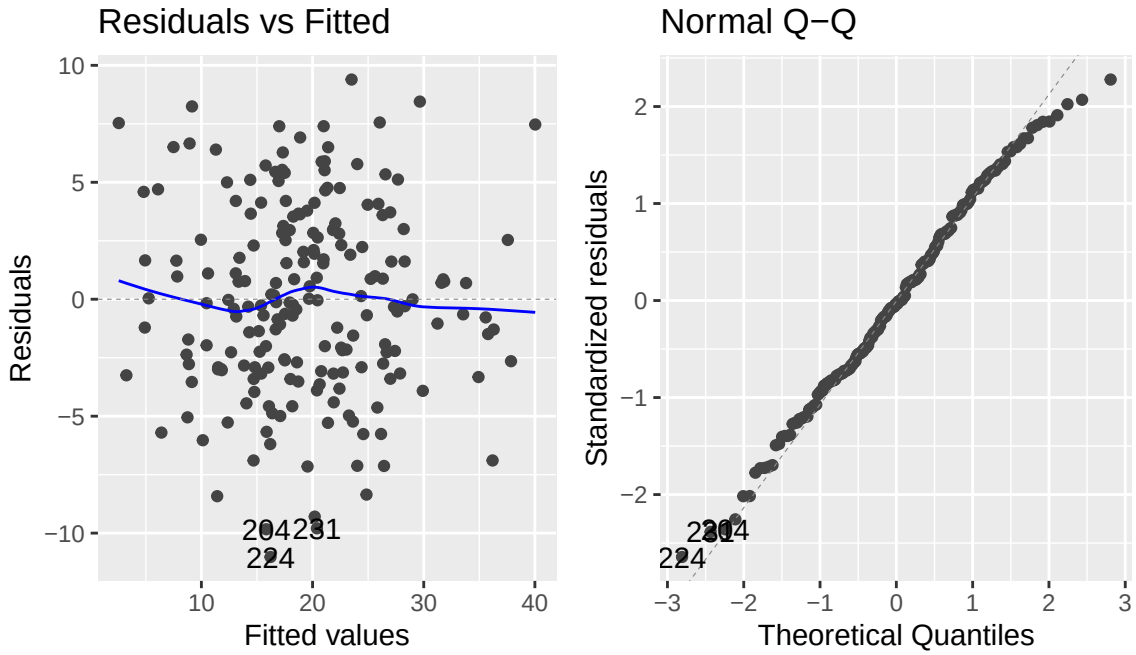
```
##
## Call:
## lm(formula = formula, data = d.body.train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.0198  -2.9100  -0.1409   2.9595   9.3920
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -40.455180  22.053225  -1.834 0.068187 .
## Age            0.067543   0.035660   1.894 0.059765 .
## Weight        -0.041957   0.061638  -0.681 0.496910
## Height        -0.066017   0.100857  -0.655 0.513560
## Neck          -0.551787   0.257295  -2.145 0.033285 *
## Chest         -0.084785   0.107160  -0.791 0.429830
## Abdomen        1.675292   0.300530   5.574 8.63e-08 ***
## Hip           -0.093039   0.167284  -0.556 0.578760
## Thigh          0.087891   0.156380   0.562 0.574768
## Knee          -0.106406   0.276498  -0.385 0.700799
## Ankle          0.112232   0.230627   0.487 0.627087
## Biceps         0.350943   0.201611   1.741 0.083391 .
## Forearm        0.332821   0.215280   1.546 0.123807
## Wrist         -2.084782   0.580179  -3.593 0.000418 ***
## I(Abdomen^2)  -0.003919   0.001571  -2.494 0.013490 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.234 on 186 degrees of freedom
## Multiple R-squared:  0.7559, Adjusted R-squared:  0.7375
## F-statistic: 41.15 on 14 and 186 DF,  p-value: < 2.2e-16
```

(ii) $R^2 = 0.756$ and $R^2_{\text{adj}} = 0.738$. The difference is relatively small, but not completely ingorable. There might be a few redundant variables.

(iii) The TA plot can be used to check that the mean of the error terms is zero, that the error variance is constant and that the errors are (approximately) independent. The QQ plot indicates that the normal distribution assumption is fulfilled. Both plots indicate that all the modeling assumptions are met. (0.5

for the interpretation, 0.5 for the correct plots).

```
library(ggfortify)
autoplot(r.lm, which = 1:2)
```



b)

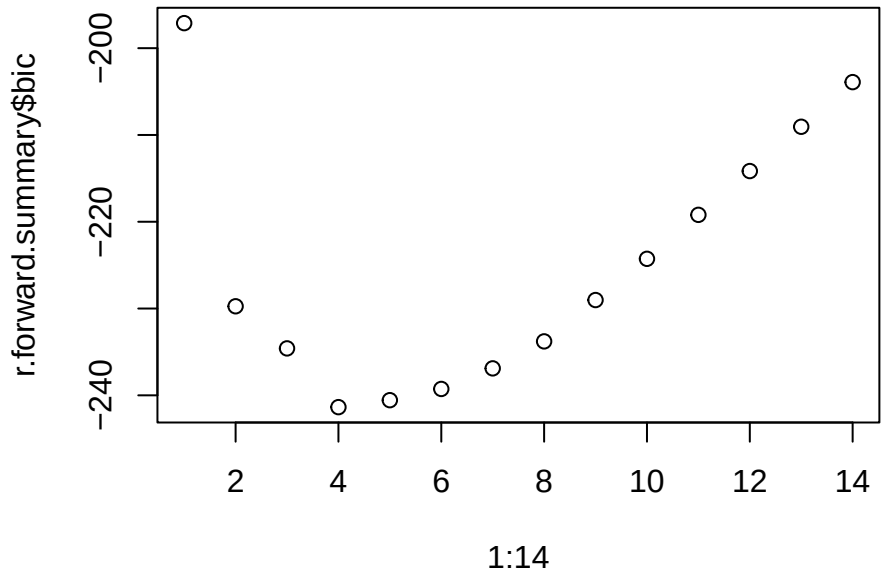
(i) (2P) To make the model efficient and easy to use in a practical application, we are interested in finding a model that has as few variables as possible and still gives good predictions. To this end, perform forward selection on the model you used in a) (including the quadratic term for `Abdomen`). Choose the model with the lowest BIC.

(ii) (1P) Fit the model selected in (i) to the training data and calculate the MSE on the test data.

Solution: (i) 1P for performing the forward selection, 1P for chosing the right model (with 4 predictors).

```
library(leaps)
n_predictors = ncol(d.body.train)
best_subset_forward = regsubsets(formula, d.body.train, nvmax = n_predictors,
    method = "forward")
r.forward.summary <- summary(best_subset_forward)

plot(1:14, r.forward.summary$bic)
```




```
n_selected_predictors <- 4
indices <- summary(best_subset_forward)$which[n_selected_predictors,
    ]
selected_predictors <- colnames(summary(best_subset_forward)$which)[indices]

r.lm.best <- lm(BodyFat ~ Weight + Abdomen + I(Abdomen^2) + Wrist, d.body.train)
```

(ii)

```
(mse.best.lm = mean((d.body.test$BodyFat - predict(r.lm.best, newdata = d.body.test))^2))
```

```
## [1] 18.58279
```

c) Lasso

(i) (2P) Now use Lasso to do model selection. To this end, use the training data and choose the largest λ within 1 standard error from the lambda with the minimal error in a 5-fold cross-validation. As above, include the quadratic term for `Abdomen`. **Requirement**: Use `set.seed(4268)` before running the cross-validation.

(ii) (2P) Fit the Lasso-model with λ_{1se} selected in (i) to the training data and calculate the MSE on the test data.

(iii) (2P) Compare the model and MSE you found in (ii) with a model where you use λ_{min} instead. Is the model with λ_{min} useful for tue purpose of task c)?

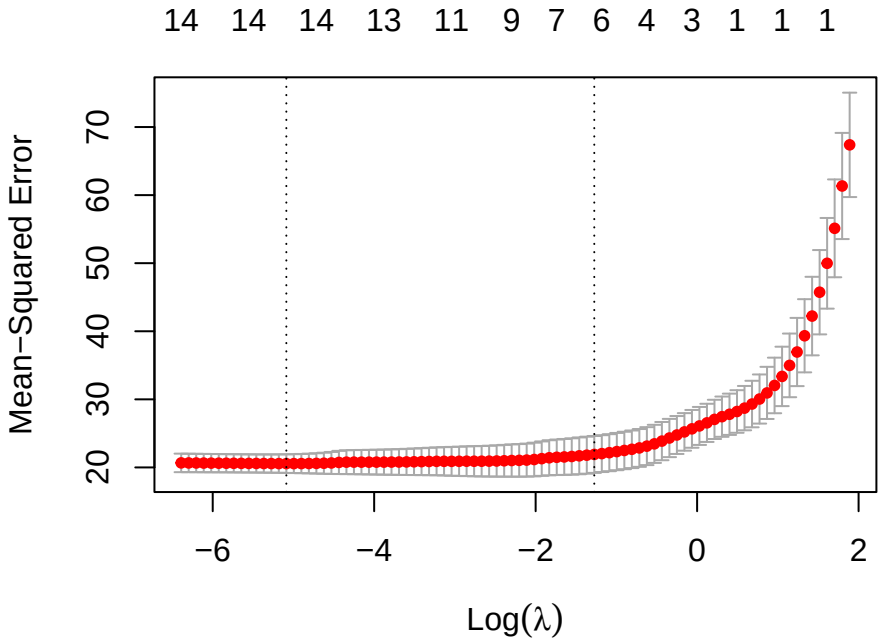
Solution:

(i) (1P) for the CV and 1P for finding the lambda.1se.

```
x.train <- model.matrix(formula, data = d.body.train)[, -c(1)]
y.train <- d.body.train$BodyFat
x.test = model.matrix(formula, data = d.body.test)[, -c(1)]
y.test = d.body.test$BodyFat

library(glmnet)
set.seed(4268)
cv.lasso <- cv.glmnet(x.train, y.train, alpha = 1, nfolds = 5)
```

```
# The plot is not required, but nice to have:
plot(cv.lasso)
```



The 1se lambda is:

```
print(cv.lasso$lambda.1se)
```

```
## [1] 0.2798399
```

(ii)

```
bodyfat.lasso <- glmnet(x.train, y.train, alpha = 1, lambda = cv.lasso$lambda.1se)

coef(bodyfat.lasso)
```

```
## 15 x 1 sparse Matrix of class "dgCMatrix"
##                        s0
## (Intercept)  -8.62360877
## Age           0.06597586
## Weight        .
## Height       -0.10611314
## Neck         -0.13843245
## Chest         .
## Abdomen       0.66800837
## Hip           .
## Thigh         .
## Knee          .
## Ankle         .
## Biceps        .
## Forearm       0.16285878
## Wrist        -1.58509289
## I(Abdomen^2)  .
```

```
mse.lasso = mean((y.test - predict(bodyfat.lasso, newx = x.test))^2)
mse.lasso
```

```
## [1] 20.0992
```

(iii) 1P for the code (-0.5P for each small error), 1P for the interpretation.

```
bodyfat.lasso.min <- glmnet(x.train, y.train, alpha = 1, lambda = cv.lasso$lambda.min)
coef(bodyfat.lasso.min)
```

```
## 15 x 1 sparse Matrix of class "dgCMatrix"
##                              s0
## (Intercept)  -34.710735680
## Age            0.066942633
## Weight        -0.044511133
## Height        -0.062352334
## Neck          -0.553684585
## Chest         -0.071504305
## Abdomen        1.527033803
## Hip           -0.104160210
## Thigh          0.088258777
## Knee          -0.068863591
## Ankle          0.091873633
## Biceps         0.344136589
## Forearm        0.346869214
## Wrist         -2.059362053
## I(Abdomen^2)  -0.003181533
```

```
mse.lasso.min = mean((y.test - predict(bodyfat.lasso.min, newx = x.test))^2)
mse.lasso.min
```

```
## [1] 19.35002
```

Interpretation: When using $\lambda_{\min}$, we get a lower test error. However, none of the βs is then becoming zero, thus no variables are selected. This model is thus not useful when the aim is selecting a simpler model for practical applications.

d)

Now we are looking at principal component analysis (PCA) and principal component regression (PCR).

(i) (2P) First, make a PCA for the training data, including all the regression variables (i.e., all variables except Bodyfat) plus the squared version of `Abdomen`. Display in a scree plot the variance explained by the PCs against the PC number. Looking at this plot, what could be a useful number of PCs to be included in a PCR?

(ii) (2P) Instead of directly using the number of PCs you found in (i), run a cross-validated PCR on the training data. Choose the number of PCs that lead to the smallest CV error. Then use the respective model fitted on the training data to calculate the test MSE.

(iii) (1P) Compare the number of chosen PCs from (i) and (ii) and interpret the difference.

R-Hints:

- To do the PCA with all the variables including `Abdomen^2`, you can add it directly to the training and test sets:

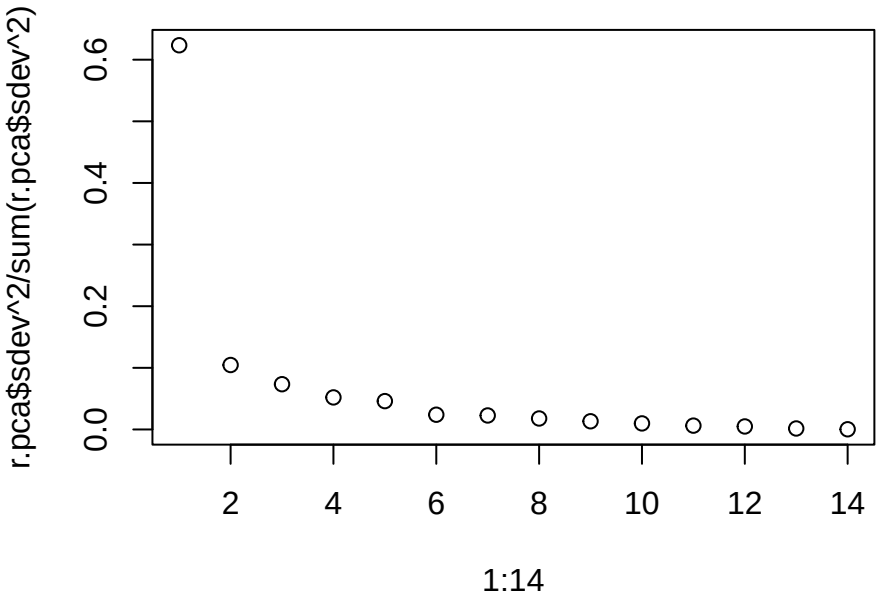
```
d.body.train$Abd2 <- (d.body.train$Abdomen)^2
d.body.test$Abd2 <- (d.body.test$Abdomen)^2
```

- use `scale=TRUE` in the PCA and PCR.
- Use `set.seed(4268)` before running the cross-validation in task (ii).

Solution: (i)

```
# d.body.train <- scale(d.body.train)
d.body.train$Abd2 <- (d.body.train$Abdomen)^2
d.body.test$Abd2 <- (d.body.test$Abdomen)^2

r.pca <- prcomp(d.body.train[, -c(1)], scale = TRUE)
plot(1:14, r.pca$sdev^2/sum(r.pca$sdev^2))
```

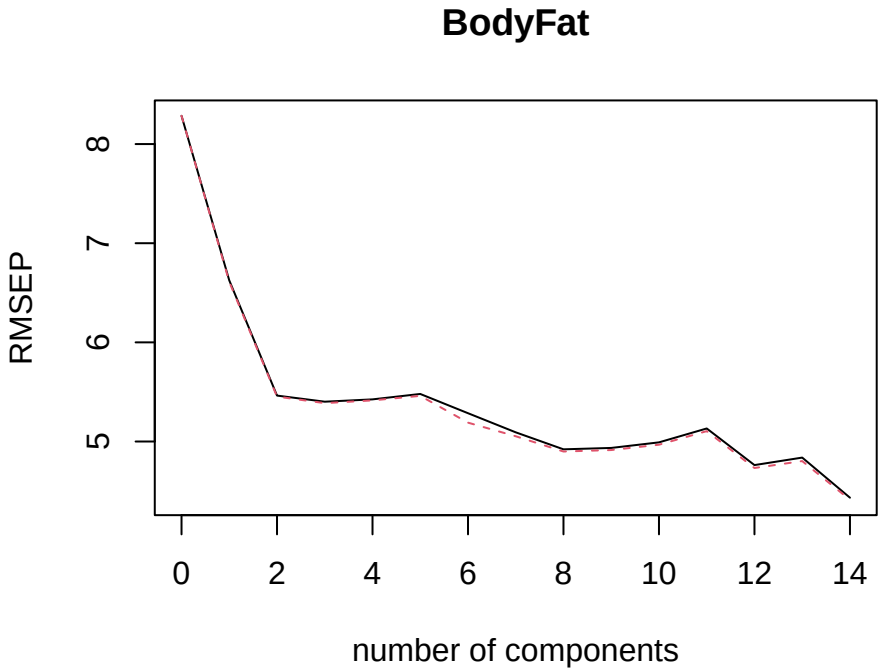


According to this plot, it looks like 2 to 6 PCs would be enough. (Some students might, however, correctly point out that the variance explained in the covariates is not necessarily related to the variance explained in the response, but that's not expected here).

Note: -1P when scaling is not done.

(ii) The PCR is given below. Interestingly, it turns out that the CV error is smallest for 14 PCs (check either the `summary()` or the `validationplot()` to see this).

```
set.seed(4268)
pcr.fit = pcr(BodyFat ~ ., data = d.body.train, scale = TRUE, validation = "CV")
validationplot(pcr.fit)
```




```
summary(pcr.fit)
```

```
## Data:    X dimension: 201 14
##  Y dimension: 201 1
## Fit method: svdpc
## Number of components considered: 14
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           8.285    6.625    5.463    5.402    5.425     5.48    5.285
## adjCV        8.285    6.611    5.453    5.387    5.415     5.46    5.190
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV       5.092    4.921    4.935     4.992     5.132     4.762     4.838
## adjCV    5.053    4.899    4.914     4.968     5.105     4.733     4.803
##        14 comps
## CV        4.433
## adjCV     4.412
##
## TRAINING: % variance explained
##          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X          62.34     72.8    80.14    85.34    89.94    92.34    94.61    96.39
## BodyFat    39.68     59.2    60.71    60.74    61.88    67.66    68.03    69.16
##          9 comps  10 comps  11 comps  12 comps  13 comps  14 comps
## X          97.72     98.70     99.32     99.81     99.97    100.00
## BodyFat    69.43     69.68     69.79     73.41     73.55     75.59
```

```
pcr.pred = predict(pcr.fit, x.test, ncomp = 14)
mean((y.test - pcr.pred)^2)
```

```
## [1] 19.30452
```

Since we used all the 14 PCs, this model is actually identical to the full regression model from a) (but we do not expect students to make this point).

(iii) In contrast to (i), the PCR done in (ii) indicates that we need to include almost all the PCs to get reasonably good predictions. This tells us two things (the point is given if one of the points is made):

- The first PCs might explain a lot of variance among the covariates, but they do not correlate very well with the response variable, so we need to include many (or actually: all).
- `BodyFat` is a rather complex trait, and we need a lot of PCs/variables to predict it well. (This is also reflected in the Lasso, where λ_{min} did not manage to kick any variables out of the model – this point is not expected to be made in the exam, though.)

Problem 5 – Data analysis 2 (17P)

In this example with look at a dataset taken from https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease . The data stem from a large health survey in the US and contain information about factors that are related to heart disease status of individuals. When clicking on the link you will also find a detailed description of the variables.

The entire datasets contains almost 400'000 rows. For convenience, we are looking at a portion of the data with only 20'000 instances. Load the dataset and create a training and a test set using the following code:

```
id <- "1HM1ytt-x9QkTHQu7bMvhBJSJWihzpZJ2"  # google file ID
d.heart <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id))
d.heart$HeartDisease <- as.factor(d.heart$HeartDisease)

# 70% of the sample size for training set
training_set_size <- floor(0.7 * nrow(d.heart))

set.seed(4268)
train_ind <- sample(seq_len(nrow(d.heart)), size = training_set_size)

train <- d.heart[train_ind, ]
test <- d.heart[-train_ind, ]
```

Before you get started, it is smart to make yourself familiar with the data. Use `str(train)`, `summary(train)` or other functions or graphical tools you learned about in the course.

a) (3P)

(i) (2P) Fit a logistic regression model using the training data with `HeartDisease` as the response and `BMI`, `Smoking`, `AlcoholDrinking`, `Sex` and `AgeCategory` as covariates, including interactions between `Smoking` and `Sex` and between `AlcoholDrinking` and `Sex`. Print the `summary()` table.

(ii) (1P) How many age categories are in this dataset? How many regression parameters are estimated for `AgeCategory`?

Solution: We deduct -1P for each error (e.g., when an interaction is missing).

(i)

```
##
## Call:
## glm(formula = formula, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6768  -0.4691  -0.3061  -0.1491   3.5543
```

```
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -7.035768   0.523698 -13.435  < 2e-16 ***
## BMI                          0.042950   0.004948   8.681  < 2e-16 ***
## SmokingYes                   0.574912   0.097360   5.905 3.53e-09 ***
## SexMale                      0.563997   0.097182   5.803 6.49e-09 ***
## AlcoholDrinkingYes          -0.314465   0.234249  -1.342  0.17945
## AgeCategory25-29            -0.766922   0.868092  -0.883  0.37699
## AgeCategory30-34             1.161927   0.565991   2.053  0.04008 *
## AgeCategory35-39             0.995359   0.570593   1.744  0.08108 .
## AgeCategory40-44             1.669477   0.537230   3.108  0.00189 **
## AgeCategory45-49             1.651726   0.537050   3.076  0.00210 **
## AgeCategory50-54             2.409280   0.517575   4.655 3.24e-06 ***
## AgeCategory55-59             2.626340   0.513470   5.115 3.14e-07 ***
## AgeCategory60-64             2.880835   0.510277   5.646 1.65e-08 ***
## AgeCategory65-69             3.034768   0.509109   5.961 2.51e-09 ***
## AgeCategory70-74             3.532333   0.507697   6.958 3.46e-12 ***
## AgeCategory75-79             3.788152   0.508763   7.446 9.64e-14 ***
## AgeCategory80 or older       4.185637   0.507548   8.247  < 2e-16 ***
## SmokingYes:SexMale           0.141130   0.130081   1.085  0.27795
## SexMale:AlcoholDrinkingYes   0.001067   0.300323   0.004  0.99717
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8265.8  on 13999  degrees of freedom
## Residual deviance: 7024.4  on 13981  degrees of freedom
## AIC: 7062.4
##
## Number of Fisher Scoring iterations: 8
```

(ii) 13 age categories, but 12 regression parameters. (0.5P for each correct answer)

b) (4P)

(i) (2P) Use the fitted model from a) to write down the estimated regression model of `HeartDisease` vs. `BMI` for a non-smoking male age 20 that does not drink alcohol.

(ii) (1P) Is there evidence that the effect of drinking alcohol differs between males and females? Give a reason.

(iii) (1P) Do you think the purpose of this model was inference or prediction? Give a reason.

Solution:

(i) In this case we have `sex=1`, `smoking=0`, `AlcoholDrinking=0`, `AgeCategory=18-24` (which is the reference category) thus the equation is given as $\log\left(\frac{P(y_i=1)}{1-P(y_i=1)}\right)$ = -7.04 + 0.56 + 0.04·BMI_i = -6.47 + 0.04·BMI_i. -1P for any mistake.

(ii) No, there is no evidence (0.5P). The p-value is very large (0.5P for the reasoning).

(iii) Inference (0.5). Why? We used logistic regression and only a few variables (0.5P for the reasoning).

c) (6P)

From now one we are using *all* the covariates in the dataset (but no interaction terms).

(i) (1P) Carry out a linear and a quadratic discriminant analysis (LDA, QDA) on the training data, again using `HeartDisease` as the response.
(ii) (2P) Using the fitted models from (i), calculate the test error for both the LDA and QDA.
(iii) (2P) Calculate the AUC using the test data for both models (still using the fitted model from (i)).
(iv) (1P) Why would k-nearest-neighbor (KNN) classification probably not work very well for this task?

Solution:

(i) 0.5P for the lda and the qda. Important that the formula is adjusted – we need to include all the variables now:

```
library(MASS)
formula <- HeartDisease ~ .

ldaMod <- lda(formula, data = train)
qdaMod <- qda(formula, data = train)
```

(ii) To calculate the test error we first need the predictions and the confusion tables:

```
postLDA <- predict(ldaMod, newdata = test)$posterior
predLDA <- predict(ldaMod, newdata = test)$class

postQDA <- predict(qdaMod, newdata = test)$posterior
predQDA <- predict(qdaMod, newdata = test)$class

(conf_LDA <- table(true = test$HeartDisease, predicted.lda = predLDA))
```

```
##      predicted.lda
## true    No  Yes
##   No  5311  161
##   Yes  417  111
```

```
(conf_QDA <- table(true = test$HeartDisease, predicted.qda = predQDA))
```

```
##      predicted.qda
## true    No  Yes
##   No  3651 1821
##   Yes  105  423
```

```
1 - (sum(diag(conf_LDA))/sum(conf_LDA[1:2, 1:2]))
```

```
## [1] 0.09633333
```

```
1 - (sum(diag(conf_QDA))/sum(conf_QDA[1:2, 1:2]))
```

```
## [1] 0.321
```

(iii)

```
library(pROC)

LDA.ROC <- roc(response = test$HeartDisease, predictor = postLDA[, 2])
QDA.ROC <- roc(response = test$HeartDisease, predictor = postQDA[, 2])

auc(LDA.ROC)
```

```
## Area under the curve: 0.8186
```

```
auc(QDA.ROC)
```

```
## Area under the curve: 0.7971
```

(iv) Curse of dimensionality / too many regression variables (only correct when either of the two is mentioned).

d) (4P)

(i) (3P) The aim of this task is to find a tree-based method that gives a lower test error than both LDA and QDA above. To this end, choose a tree-based method and fit it using the training data, then predict the response using the test data. Justify the choice of any parameters you use.

(ii) (1P) Based on the model you chose in (i), which three variables are most important to predict heart disease, according to an importance measure based on node purity?

Solution: (i) Ideally the students should choose a random forest approach, but they might end up with good results for another approach too. As we only discussed boosting for regression trees, we do not expect anyone to use it. If somebody does it anyway, and does it right, the points are of course given.

```
library(randomForest)
set.seed(4268)
rf.peng = randomForest(HeartDisease ~ ., data = train, mtry = 4, ntree = 500,
    importance = TRUE)

t.pred.rf <- predict(rf.peng, test, type = "class")
(confMat <- confusionMatrix(t.pred.rf, test$HeartDisease)$table)
```

```
##            Reference
## Prediction   No  Yes
##        No  5441  497
##        Yes   31   31
```

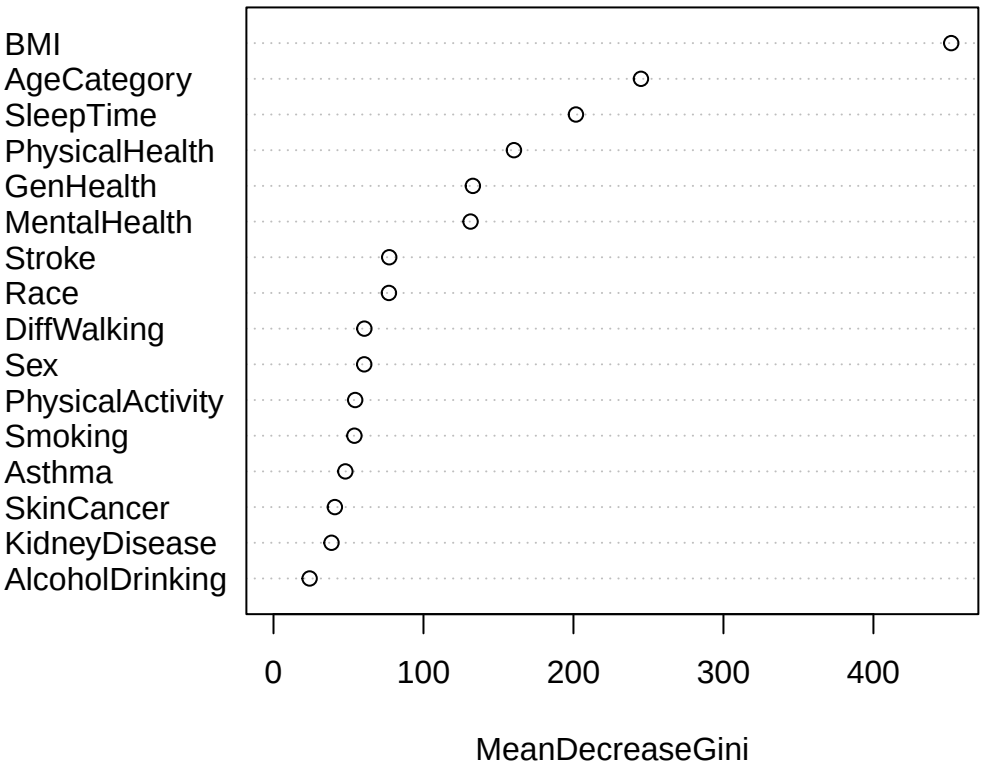
```
1 - (sum(diag(confMat))/sum(confMat[1:2, 1:2]))
```

```
## [1] 0.088
```

Here we have to choos `mtry`, which should be $\sqrt{p}$, with $p = 17$ (number of regression variables). So we can use 4, perhaps 5. The number of trees is not a tuning parameter, but the students should mention that it should be chosen "large enough".

(ii)

```
varImpPlot(rf.peng, type = 2)
```



The most important predictors are BMI, age (or age category) and sleep time.

Multiple and single choice questions

Problem 6 (6P, single choice, 2P each)

a)

We are referring to the logistic regression model you fitted in Problem 5a). What is the probability that a smoking and alcohol drinking male age 77 with a BMI of 25 suffers from heart disease?

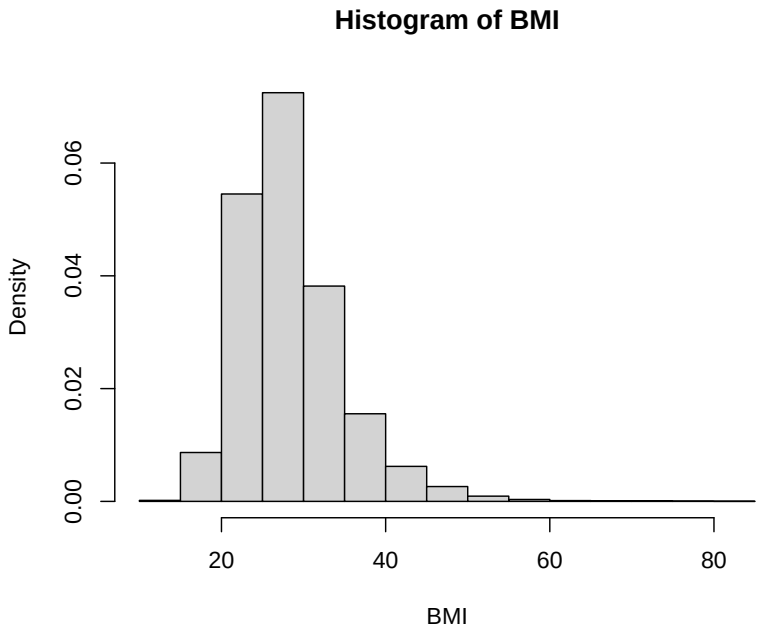
(i) 0.127
(ii) 0.003
(iii) 0.299
(iv) 0.077
(v) 0.23
(vi) 0.007

Solution: (v) is correct

b)

Below you find the histogram for the distribution of BMI in our heart disease dataset from Problem 5.

```
d.bmi <- d.heart$BMI
hist(d.bmi, main = "Histogram of BMI", xlab = "BMI", freq = F)
```



You can see that the distribution is a bit skewed, which implies that the estimated mean $\hat{\mu}$ and median $\hat{\mu}_{\text{med}}$ are unequal. In fact, the *difference* between the mean and the median is 1.005, calculated as

```
mean(d.bmi) - median(d.bmi)
```

Run a bootstrap and find the (approximately) correct 95% confidence interval for the difference betwen the mean and the median. Round to three digits after the comma. Use at least 1000 bootstrap samples. Your answer might differ by a little bit due to sampling error – choose the closest anwer:

(i) 0.945, 1.066
(ii) -2.836, 4.847
(iii) 0.974, 1.036
(iv) 1.004, 1.007
(v) -0.955, 2.965

Solution: (i) is correct. The alternatives should be sufficiently different to not be mixed up ()

```
round(mean(d.bmi) - median(d.bmi) + c(-1.96, 1.96) * sd(d.results), 3)
```

```
## [1] 0.945 1.066
```

c)

$\mathbf{x} = [x_1, x_2, x_3]^T$ is a 3-dimensional random vector with covariance matrix

$$\mathbf{\Sigma} = \begin{bmatrix} 5 & 2 & 4 \\ 2 & 7 & -3 \\ 4 & -3 & 8 \end{bmatrix}$$

The correlation between element x_2 and x_3 of the vector $\mathbf{x}$ is:

(i) -3
(ii) -0.40

(iii) 3
(iv) -0.23
(v) -0.23
(vi) 0.40
(vii) It is not possible to calculate the correlation, because the matrix is not positive definite.

Solution:

(ii) is correct.

```
aa <- matrix(c(5, 2, 4, 2, 7, -3, 4, -3, 8), ncol = 3)
eigen(aa)$values
```

```
## [1] 11.3387678  8.1986824  0.4625498
```

```
-3/(sqrt(7 * 8))
```

```
## [1] -0.4008919
```

Problem 7 (6P, multiple choice, 2P each)

a)

Which of the following statements about clustering methods are true, which false?

(i) The results of hierarchical clustering may differ depending on which datapoints are grouped first.
(ii) The results in K-means clustering may differ for different initial cluster assignments.
(iii) Hierarchical clustering works well when the underlying data have a hierarchical structure.
(iv) A drawback of hierarchical clustering is that we have to decide the number of clusters in advance.

Solution FALSE - TRUE - TRUE - FALSE

(i) No, the method is deterministic (iv) the statement is only true for K-means, but not for hierarchical clustering.

b)

Which of the following statements are true, which false?

(i) A natural cubic spline is linear beyond the boundary knots.
(ii) A regression spline of order 3 with 5 knots has 9 basis functions.
(iii) A regression spline with polynomials of degree M-1 has continous derivatives up to order M-2, but not at the knots.
(iv) Regression splines generally produce more stable estimates than polynomial regression, and are therefore preferable.

Solution TRUE - FALSE - FALSE - TRUE

(ii) We need 8 basis functions, but estimate 9 parameters (including the intercept, which is not a basis function). (iii) The derivative at the knots is also continuous.

c)

When talking about support vector classifiers in $p = 2$ dimensions, we were looking at hyperplanes of the form $\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$. We are now looking at the following non-linear decision boundary

$$(1 + X_1)^2 + (X_2 + 2)^3 - X_2^3 = 2 \ .$$

We assume that class 1 fulfils $(1+X_1)^2+(X_2+2)^3-X_2^3 > 2$ and class 2 fulfils $(1+X_1)^2+(X_2+2)^3-X_2^3 < 2$.

Which of the following statements are true?

(i) This decision boundary is linear in terms of X_1, X_2, X_1^2 and X_2^2.
(ii) The decision boundary has the shape of a circle.
(iii) The point $(x_1, x_2) = (1, -1)$ belongs to class 1.
(iv) The point $(x_1, x_2) = (1, 1)$ belongs to class 2.

Solution TRUE - FALSE - TRUE - FALSE

(i) is true because X_2^3 cancels out. (ii) is false (as some algebra shows). (iii) Plug $(X_1, X_2) = (1, -1)$ to see that this gives 30, which is > 2, thus the class is 1. (iv) Plug again to see that the result is $6 > 2$, thus the class is also 1.