TMA4268 V2025 Exam

TMA4268 Statistical Learning V2025

Stefanie Muff and Sara Martino, Department of Mathematical Sciences, NTNU

May 19, 2025

Check in the end which libraries we need:

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

0.5P per correct answer

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 this course we learned about methods that can be used for statistical learning. We have mainly discussed supervised statistical learning methods, broadly divided into regression and classification problems. We were thereby discriminating between models that we use for prediction versus inference.

The methods and models in the course were broadly divided into supervised an unsupervised approaches. In our course we were mainly discussing supervised statistical learning methods, broadly divided into regression and *classification* (prediction, inference, supervised, unsupervised,...) problems. In supervised learning, there are two main purposes: *inference* (*prediction*, *variance reduction*, *unsupervised learning*, *non-parametric methods*, *over-fitting*) and *prediction* (*inference*, *bias reduction*, *variance reduction*, *supervised learning*, *model selection*, *bias-variance trade-off*). In both cases we want to learn from data and build a model that relates a set of variables to an outcome, but in the latter case we do not care about the actual model parameters, because we do not want to interpret them. When the goal is to interpret and understand causal effects, we prefer to use *parametric* (non-parametric, flexible, classification, least squares) models. In a classification problem, for example, we then would prefer *logistic regression* (hierarchical clustering, linear regression, boosted classification trees, neural networks, KNN classification, K-means clustering, local regression).

An important topic that we were discussing throughout the course was the bias-variance trade-off. The main idea is that the *reducible error* (training error, prediction bias, irrediculbe 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 *more sparse 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, neural networks), 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 (Multiple choice and numeric answer, 8 P)

a) (2P)

In a study it was investigated how weight (kg) of adults changes with age (years) and sex (male=1, female=0). The result from the analysis is given in the following table:

```
##                Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 56.5841248  6.3245948 8.9466799 9.875405e-18
## age          0.1142729  0.1799040 0.6351883 5.256313e-01
## sex          5.0345319  9.4008716 0.5355388 5.925442e-01
## age:sex      0.6750225  0.2667099 2.5309238 1.171900e-02
```

Please select all statements that are true, according to the regression output:

(i) Female weights increased on average by 0.11 kg/year.
(ii) Males are on average 5.03 kg heavier than females.
(iii) Male weights increased on average by 0.79 kg/year.
(iv) The dependency of weight on age appears to be different for males and females.

Solution True - False - True - True
Note that (ii) is false, because the coefficient is only valid as long as age=0.

b) (2P)

Please select all statements that are true about the reationship between Lasso and least squares:

(i) Lasso is more flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance.
(ii) Lasso is more flexible and hence will give improved prediction accuracy when its increase in variance is less than its decrease in bias.
(iii) Lasso is less flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance.
(iv) Lasso is less flexible and hence will give improved prediction accuracy when its increase in variance is less than its decrease in bias.

Solution False - False - True - False

By using shrinkage, lasso can reduce the number of predictors so is less flexible. As a result, it will lead to an increase in bias by approximating the true relationship. We hope that this increase is small but that we dramatically reduce variance (i.e. the difference we would see in the model fit between different sets of training data).

c) (2P)

Suppose you build a neural network with:

- 3 input variables
- 1 hidden layer with 4 neurons
- 1 output neuron

Assume also that all units are fully connected.

(i) How many weights (parameters) are there in total, including biases?
(ii) Suppose a hidden layer neuron has:

- Input values: $x_1 = -1, x_2 = 2, x_3 = 0$

- Weights: $w_1 = 0.5, w_2 = 0, w_3 = 1$
- Bias: $b = 0$

Assume a ReLU activation function. Question: What is the output of the neuron?

Solution (i) [21] Each hidden neuron has 3 input weights + 1 bias = 4 parameters. Total for hidden layer $4 \times 4 = 16$. Output neuron has 4 input weights (from hidden neurons) + 1 bias = 5 parameters.Final total = 16+5=2116+5=21.

(ii) [0] Linear Combination: $-1 \times 0.5 + 2 * 0 + 3 * 1 = -0.5$

ReLU(-0.5) = 0

d) (2P)

This problem has to do with odds.

(i) On average, what fraction of people with an odds of 0.37 of defaulting on their credit card payment will in fact default?
(ii) Suppose that an individual has a 16% chance of defaulting on her credit card payment. What are the odds that she will default?

Solution

(i) 0.27
(ii) 0.19

e) (2P)

Please select all statements that are true.

In smoothing splines, increasing the smoothing parameter λ will:

(i) Make the fitted function more flexible and wiggly.
(ii) Make the fitted function smoother and potentially underfit the data.
(iii) Increase the penalty for wiggliness.
(iv) Decrease the effective degrees of freedom.

Solution FALSE, TRUE, TRUE, TRUE

f) (3P)

You perform PCA on a dataset with 5 **standardized** variables: X_1, X_2, X_3, X_4, X_5.

You obtain the following results:

Principal Component	Eigenvalue	Proportion of Variance Explained (PVE)
PC1	2.7	0.54
PC2	1.5	0.30
PC3	0.5	0.10
PC4	0.2	0.04
PC5	0.1	0.02

You also know the loadings for the first principal component (PC1):

$$\text{PC1} = 0.85X_1 + 0.2X_2 + 0.2X_3 + 0.1X_4 + 0.05X_5$$

$$\begin{pmatrix} 0.85 \\ 0.2 \\ 0.2 \\ 0.1 \\ 0.05 \end{pmatrix}$$

You have an observation with the following standardized variable values:

$$X_1 = 1,\ X_2 = 0.5,\ X_3 = -0.5,\ X_4 = -1,\ X_5 = 0$$

Questions

1. What proportion of the total variance is explained by the first four principal components combined? [0.96]
2. If you want to retain at least 90% of the total variance, how many principal components do you have to keep? [3]
3. Calculate the value of PC1 for the observation above, using the given loadings and standardized values. [0.75]
4. If you plotted the observations using only PC1 and PC2, what percentage of the original dataset's variability would be represented in the plot? [84]
5. Which variable contribute most strongly to PC1, based on the loadings? [1]
6. What is the total variance in the original data? [5. since the variable are standardized the total variance is the sum of the eigenvalues]

g) (1P)

Why are nonlinear activation functions necessary in a neural network? (Please select all statements that are true)

(i) To make training faster
(ii) To allow the network to approximate complex, non-linear functions
(iii) To reduce the number of parameters
(iv) To make the network fully connected

Solution FALSE, TRUE, FALSE, FALSE

Problem 3 (Theory, 4P)

a) (2P)

Assume that you have a dataset with $n = 100$ observations. Each observation contains a quantitative response Y and one continuous predictor X. You then fit the following regression models:

- Model a) $Y = \beta_0 + \beta_1 X + \epsilon$
- Model b) $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^2 + \epsilon$

(i) Suppose that the true relationship between X and Y is linear, that is, Model a) is correct. Consider the *training residual sum of squares* (RSS) for both models. Would we expect one to be lower than the other, would we expect them to be the same, or is there not enough information to tell? Justify your answer.
(ii) Consider now the *test residual sum of squares* (RSS). How would you answer to the previous question?

Solution
(i) we would expect the Model b) to have lower training RSS since it is at least as flexible as the linear regression. (ii) Though we could not be certain, the test RSS would likely be higher due to overfitting.

b) (2P)

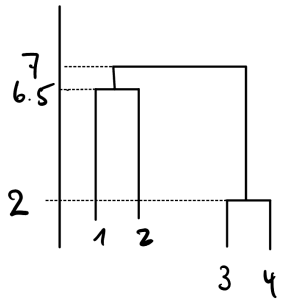
We have four observations for which we know the distance matrix in Euclidean space:

$$\begin{bmatrix} 0 & 6.5 & 5 & 7 \\ 6.5 & 0 & 6 & 4 \\ 5 & 6 & 0 & 2 \\ 7 & 4 & 2 & 0 \end{bmatrix}.$$

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 (16P)

Here we will use the Boston dataset where variables related to median house values (`medv`, in 1000$) in the US are stored. We load and prepare the dataset as follows (copy the code into your Rmd file). Note that we only use 9 covariates and the response, and we convert `rad` (the index of accessibility to radial highways) into a factor variable:

```
library(MASS)
data(Boston)

d.boston <- Boston[,c("crim","indus","chas","rm","age","dis","rad","tax","black","medv")]
d.boston$rad <- as.factor(d.boston$rad)
```

```
# Look at and familiarise yourself with the data, for example using the pairs plot and checking its str
?Boston
pairs(d.boston)
str(d.boston)
summary(d.boston)
```

a) (5P)

We start with a model for inference, that is, we want to understand the relationship between the variables in our dataset and the house value (`medv`).

(i) (1P) Fit a linear regression model on the dataset `d.boston`, with `medv` as the response variable. Use all predictors linearly plus a quadratic term for the number of rooms `rm` by adding it as `I(rm^2)`.

(ii) (1P) How many degrees of freedom does the model consume?

(iii) (2P) Based on the results from (i), report the effect of having a house that bounds the Charles River (`chas` variable, use `?Boston` to find out what the variable means). How much more/less does an average property cost when it faces the river? Quantify both the effect size and its 95% confidence interval.

(iv) (1P) Is there evidence that `rad` is a relevant variable in the model? Calculate and report the p-value that corresponds to an appropriate test, and interpret the result.

Solution (i)

```
formula <- medv~ . + I(rm^2)
r.boston<-lm(formula, data=d.boston)
summary(r.boston)
```

```
##
## Call:
## lm(formula = formula, data = d.boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -33.757  -2.466  -0.432   1.695  35.263
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  95.968166  10.461201   9.174  < 2e-16 ***
## crim         -0.162895   0.034163  -4.768 2.46e-06 ***
## indus        -0.095789   0.064012  -1.496 0.135190
## chas          3.364226   0.904835   3.718 0.000224 ***
## rm          -28.155183   3.143924  -8.955  < 2e-16 ***
## age          -0.080829   0.012422  -6.507 1.90e-10 ***
## dis          -0.727248   0.188516  -3.858 0.000130 ***
## rad2          1.715927   1.549921   1.107 0.268794
## rad3          3.049979   1.392722   2.190 0.029001 *
## rad4          0.949135   1.240096   0.765 0.444420
## rad5          3.084790   1.245091   2.478 0.013566 *
## rad6          0.904843   1.535958   0.589 0.556062
## rad7          1.903045   1.644701   1.157 0.247807
## rad8          1.519508   1.544154   0.984 0.325584
## rad24         3.098845   1.849487   1.676 0.094474 .
## tax          -0.008093   0.004063  -1.992 0.046971 *
## black         0.012593   0.002775   4.537 7.18e-06 ***
## I(rm^2)       2.746695   0.243639  11.274  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.973 on 488 degrees of freedom
## Multiple R-squared:  0.7175, Adjusted R-squared:  0.7076
## F-statistic: 72.89 on 17 and 488 DF,  p-value: < 2.2e-16
```

(ii) The degrees of freedom is the number of regression parameters, including the intercept. Here, they are 18.

(iii) A property on the river has, on average, a value that is $3.36 \cdot 1000$ \$ higher. The 95% CI is from 1.59 to 5.14 times 1000 dollars. (1P for the correct effect size and 1P for the correct CI).

(iv) Since we have to test whether all the 8 slope variables (for all the 8 dummy variables) are =0, we need

an F-test, which we do via the anova function:

```
anova(r.boston)
```

```
## Analysis of Variance Table
##
## Response: medv
##            Df  Sum Sq Mean Sq  F value    Pr(>F)
## crim        1  6440.8  6440.8 260.4238 < 2.2e-16 ***
## indus       1  5433.7  5433.7 219.7039 < 2.2e-16 ***
## chas        1  1511.7  1511.7  61.1247 3.328e-14 ***
## rm          1 11431.2 11431.2 462.2033 < 2.2e-16 ***
## age         1   285.9   285.9  11.5619 0.0007284 ***
## dis         1  1025.8  1025.8  41.4772 2.860e-10 ***
## rad         8   648.0    81.0   3.2754 0.0011914 **
## tax         1    76.1    76.1   3.0755 0.0801069 .
## black       1   650.6   650.6  26.3042 4.214e-07 ***
## I(rm^2)     1  3143.3  3143.3 127.0946 < 2.2e-16 ***
## Residuals 488 12069.2    24.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value for the categorical `rad` variable is about 0.001, so there is strong evidence that the variable is associated with the response.

b) (2P)

Now we change gears and try to find a model that is good in predicting the value of a new house. To this end, we split the data into a training and a test set:

```
set.seed(4268)
samples <- sample(1:506, 354, replace = F)
d.boston.train <- d.boston[samples, ]
d.boston.test <- d.boston[-samples, ]
```

We use the same model as in a). Fit a linear regression model on the training data and calculate the mean squared error for the test data.

Solution:

```
formula2 <- medv~ . +I(rm^2)
r.boston2<-lm(formula2, data=d.boston.train)
summary(r.boston2)$coef
```

```
##                  Estimate   Std. Error    t value     Pr(>|t|)
## (Intercept)  92.707125726 12.019633929  7.7129741 1.409464e-13
## crim         -0.133426413  0.027692405 -4.8181591 2.196949e-06
## indus        -0.031164342  0.057225766 -0.5445858 5.863996e-01
## chas          2.757910454  0.815187597  3.3831605 8.011989e-04
## rm          -28.863144154  3.642811616 -7.9233151 3.429817e-14
## age          -0.072377193  0.011368852 -6.3662713 6.355860e-10
## dis          -0.430788609  0.164870329 -2.6128935 9.381374e-03
## rad2          2.009345569  1.429863711  1.4052707 1.608647e-01
## rad3          2.725012390  1.287518657  2.1164838 3.503852e-02
## rad4          1.562572676  1.158081451  1.3492770 1.781565e-01
## rad5          3.563339063  1.177565434  3.0260221 2.669165e-03
## rad6          1.307260898  1.390094371  0.9404116 3.476820e-01
```

```
## rad7          1.902614155  1.450211973  1.3119559 1.904306e-01
## rad8          1.480875367  1.584820796  0.9344119 3.507625e-01
## rad24         2.935349564  1.700588314  1.7260789 8.525250e-02
## tax          -0.009486162  0.003637219 -2.6080809 9.511590e-03
## black         0.011529370  0.002468634  4.6703447 4.354524e-06
## I(rm^2)       2.892246996  0.281331095 10.2805806 9.776576e-22
```

```
(mse.lm = mean((d.boston.test$medv - predict(r.boston2,newdata=d.boston.test))^2))
```

```
## [1] 50.7865
```

c) (4P)

(i) (2P) Now use Lasso to do model selection and at the same time to find a (possibly simpler) model that (hopefully) is better at predicting than the linear model fitted in b). To this end, use the training data and choose the λ with the minimal error in a 10-fold cross-validation. As above, include the quadratic term for `age`. **Requirement**: Use `set.seed(4268)` before running the cross-validation.

(ii) (2P) Fit the Lasso-model with λ_{min} selected in (i) to the training data, and calculate the MSE on the test data. Compare to the linear regression situation and interpret.

R-hints:

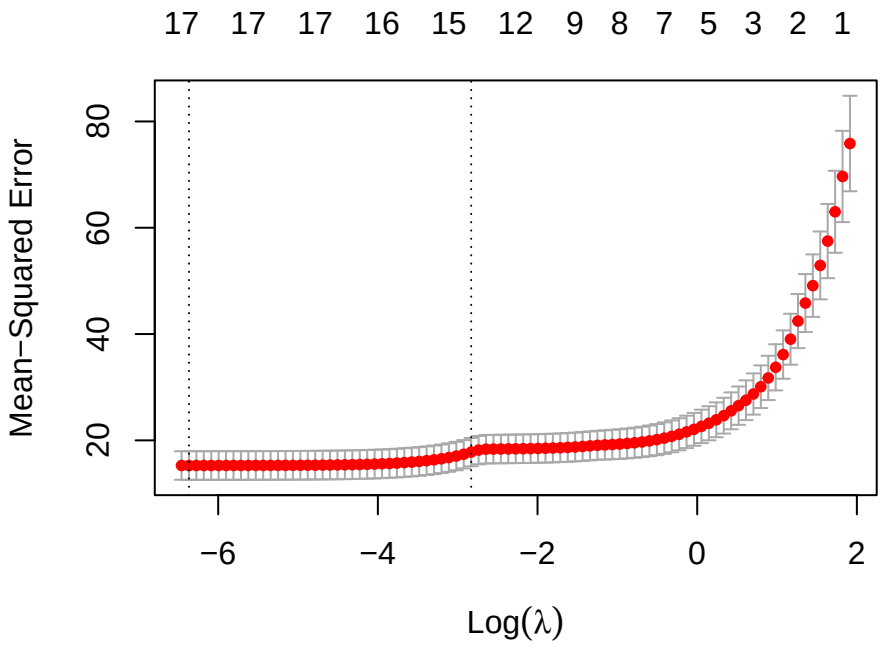
```
x.train <- model.matrix(medv ~ ..., data = d.boston.train)
y.train <- d.boston.train$...

x.test <- ...
y.test <- ...

# Lasso
cv.lasso <- cv.glmnet(x.train,  y.train, alpha = ...)
plot(cv.lasso)
cv.lasso$...
fit.lasso = glmnet(..., ... , alpha=..., lambda = ...)
```

Solution:

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

```
x.train <- model.matrix(medv ~ . + I(rm^2), data = d.boston.train)[, -1]
y.train <- d.boston.train$medv
x.test = model.matrix(medv ~ . + I(rm^2), data = d.boston.test)[, -1]
y.test = d.boston.test$medv
```

```
library(glmnet)
set.seed(4268)
cv.lasso <- cv.glmnet(x.train, y.train, alpha=1,nfolds=10)
# The plot is not required, but nice to have:
plot(cv.lasso)
```



(ii)

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

coef(boston.lasso)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
## (Intercept)  88.625800051
## crim         -0.131707487
## indus        -0.031262490
## chas          2.764831501
## rm          -27.574753179
## age          -0.072081799
## dis          -0.436333926
## rad2          1.897548130
## rad3          2.627538786
## rad4          1.451449353
## rad5          3.461712415
## rad6          1.173145823
## rad7          1.791093327
## rad8          1.424513907
## rad24         2.759777694
## tax          -0.009372247
## black         0.011546449
## I(rm^2)       2.793184512
```

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

```
## [1] 50.81502
```

MSE does not get better than with simple linear regression. Also, none of the coefficients have become zero. Probably this is a situation where most coefficients are having an effect - that is probably also why they are in the dataset.

d) (3P)

It is time to use a more advanced prediction method with lover MSE. We therefore turn to a tree-boosting method using the `gbm()` function from the `gbm` package in R.

(i) (1P) Use the same input variables as above (you may drop the quadratic term for `rm`) and fit the model on the training data. Choose reasonable parameters and explain all your choices carefully.

(ii) Predict on the test set and calculate the MSE. Interpret the result in comparison to what you found in a) and c).

(iii) Look at variable importances using the `summary()` function.

Solution: (i)

```
library(gbm)
r.gbm  <- gbm(medv ~ . , data=d.boston.train,distribution="gaussian",n.trees=1000,interaction.depth=2,sh
```

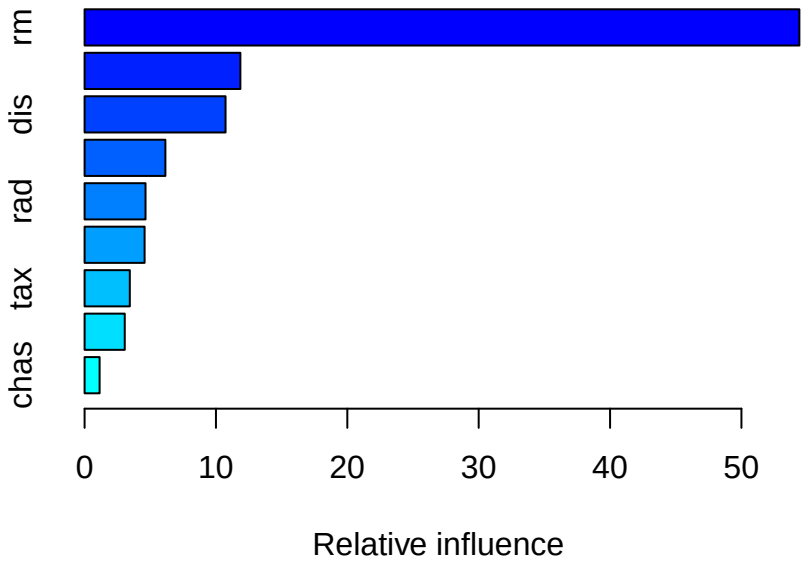
(ii)

```
mean((d.boston.test$medv -predict(r.gbm,newdata=d.boston.test))^2)
```

```
## [1] 44.02472
```

(iii)

```
summary(r.gbm)
```




```
##         var   rel.inf
## rm       rm 54.411892
## crim   crim 11.859304
## dis     dis 10.729720
## age     age  6.150306
## rad     rad  4.638064
## black black  4.568154
## tax     tax  3.441922
## indus indus  3.058132
## chas   chas  1.142507
```

e) (2P)

Finally, we are fitting a GAM using the R-code below:

```
library(gam)
r.gam <- gam(medv ~ bs(rm,df=5) +  crim + dis + bs(age,knots=quantile(age,c(0.2,0.4,0.6,0.8))) + black
```

(i) How many degrees of freedom does the spline for `age` in the model consume?
(ii) Fit the above model and calculate the MSE for the test dataset. Compare to the results from the other methods.

Solution:

(i) 7: 5 for the knots + 2

(ii)

```
mean((d.boston.test$medv -predict(r.gam,newdata=d.boston.test))^2)
```

```
## [1] 47.18571
```

So even though we do not use all the regression variables, we get a lower test error for the GAM compared to linear regression or the Lasso. However, the test error is a bit higher than for the boosted trees, making tree boosting the winner in this case.

Problem 5 – Data analysis 1 (xx P)

We will look at a classification problem using the data set `default.csv` that you can find here.

Download the .Rmd and the data in the same folder.

The dataset contains 30000 records of a bank's costumemers default payment. In addition to the response variable `DEFAULT` with values 0/1 (No deafult/Default), the dataset contains 22 features defined as follows:

- `LIMIT_BAL`: Amount of the given credit (in 1000 dollar). It includes both the individual consumer credit and his/her family (supplementary) credit.
- `SEX`: Gender (1 = male; 2 = female).
- `EDUCATION`: Education (1 = graduate school; 2 = university; 3 = high school; 4 = others).
- `MARRIAGE`: Marital status (1 = married; 2 = single; 3 = others).
- `AGE`: Age (year).
- `PAY_0` - `PAY_6`: History of past payment. Monthly payment status from April to September 2005. The measurement scale for the repayment status is: 0 = pay duly; 1 = payment delay for one month; 2 = payment delay for two months; . . .; 8 = payment delay for eight months; 9 = payment delay for nine months and above.
- `BILL_AMT1-BILL_AMT6`: Amount of bill statement (in 1000 dollar) from April to September 2005.
- `PAY_AMT1-PAY_AMT6`: Amount of previous payment (in 1000 dollar) from April to September 2005.

Note that `SEX` and `EDUCATION` are discrete variable and we define them as factors. You can use the following code the read the data:

```
default = read.table("default.csv")
default$MARRIAGE = as.factor(default$MARRIAGE)
default$SEX = as.factor(default$SEX)
```

Before starting, it is smart to investigate the data a little bit, for example by making pairs `plots` or looking at the structure using `str(data)` etc.

We are interested in predicting whether clients default or not.

a) (7P)

(i) (1P) Fit a logistic regression model on the full data set with `DEFAULT` as response variable. Use all the covariates and include the interaction between `SEX` and `PAY_0`

(ii) (2P) Look at the output from (i). How does the feature `PAY_0` influences the odds to default?

(iii) (2P) Explain what sensitivity and specificity are for this specific problem. Then compute them using as threshold value for the predicted probability $p = 0.5$.

(iv) (2P) Figure 1 shows the ROC curve for the model in point (i). Explain what the ROC curve illustrates, in particular explain what the two axes of the plot represent.

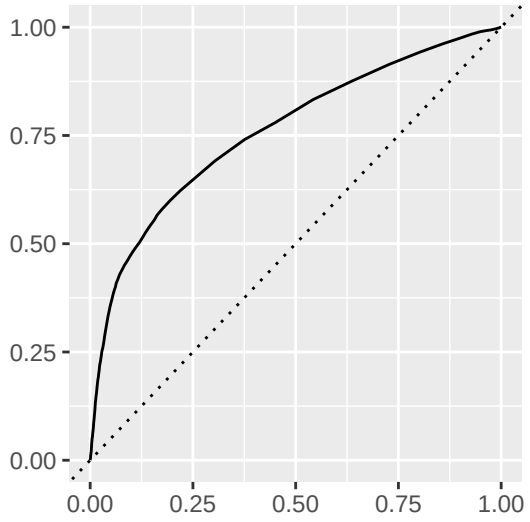


Figure 1: ROC curve

R-hints:

```
mod = glm(..., family = "binomial")
pred = predict(mod, ...)
```

Solution

(i)

```
mod.glm = glm(DEFAULT~ . + SEX:PAY_0, data = default, family = "binomial")
summary(mod.glm)
```

```
##
## Call:
## glm(formula = DEFAULT ~ . + SEX:PAY_0, family = "binomial", data = default)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.961e+00  5.517e-01  -5.368 7.98e-08 ***
## LIMIT_BAL   -1.349e-03  1.487e-04  -9.068  < 2e-16 ***
## SEX2        -2.047e-01  3.750e-02  -5.458 4.81e-08 ***
## MARRIAGE1    1.512e+00  5.460e-01   2.769  0.00562 **
## MARRIAGE2    1.352e+00  5.461e-01   2.475  0.01333 *
## MARRIAGE3    1.511e+00  5.633e-01   2.683  0.00730 **
## AGE          3.681e-03  1.881e-03   1.957  0.05032 .
```

```
## PAY_0        7.965e-01  3.378e-02  23.577  < 2e-16 ***
## PAY_2        4.587e-02  2.652e-02   1.730  0.08365 .
## PAY_3        1.260e-01  2.841e-02   4.435 9.19e-06 ***
## PAY_4        7.509e-02  3.166e-02   2.371  0.01772 *
## PAY_5        9.312e-02  3.420e-02   2.723  0.00647 **
## PAY_6        1.623e-01  2.903e-02   5.590 2.26e-08 ***
## BILL_AMT1   -2.427e-03  1.067e-03  -2.275  0.02292 *
## BILL_AMT2    2.394e-03  1.429e-03   1.675  0.09392 .
## BILL_AMT3    1.640e-03  1.279e-03   1.282  0.19998
## BILL_AMT4   -1.338e-04  1.304e-03  -0.103  0.91823
## BILL_AMT5   -4.839e-04  1.498e-03  -0.323  0.74672
## BILL_AMT6   -9.542e-05  1.194e-03  -0.080  0.93630
## PAY_AMT1    -1.179e-02  2.241e-03  -5.260 1.44e-07 ***
## PAY_AMT2    -8.689e-03  2.041e-03  -4.257 2.07e-05 ***
## PAY_AMT3    -1.780e-03  1.662e-03  -1.071  0.28423
## PAY_AMT4    -2.953e-03  1.757e-03  -1.681  0.09283 .
## PAY_AMT5    -2.956e-03  1.779e-03  -1.661  0.09663 .
## PAY_AMT6    -2.700e-03  1.333e-03  -2.025  0.04284 *
## SEX2:PAY_0   1.497e-01  3.998e-02   3.745  0.00018 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 31705  on 29999  degrees of freedom
## Residual deviance: 26691  on 29974  degrees of freedom
## AIC: 26743
##
## Number of Fisher Scoring iterations: 6
```

(ii)

For males (the reference category) we have that

```
exp(coef(mod.glm)["PAY_0"])
```

```
##    PAY_0
## 2.217823
```

Interpretation: For males, the odds for default increases by a factor of 2.217 for every additional month that the payment for April is delayed

For females (the reference category) we have that

```
exp(coef(mod.glm)["PAY_0"] + coef(mod.glm)["SEX2:PAY_0"])
```

```
##    PAY_0
## 2.576059
```

Interpretation: For females, the odds for default increases by a factor of 2.57 for every additional month that the payment for April is delayed

(iii)

```
preds = round(predict(mod.glm, newdata = default, type = "response"))
sensitivity = sum(preds==1 & default$DEFAULT==1)/sum(default$DEFAULT==1)
specificity = sum(preds==0 & default$DEFAULT==0)/sum(default$DEFAULT==0)
```

Sensitivity is the ability of a model to correctly identify clients that default. It can be computed as the

fraction of true defaulters that are identified. In our case it is 0.3339361.

Specificity is the ability of a model to correctly identify clients that do not default. It can be computed as the fraction of true non-defaulters that are identified. In our case it is 0.9563003.

(iv) The ROC curve is a visual representation of model performance across all thresholds. The ROC curve is drawn by calculating the true positive rate (TPR) and false positive rate (FPR) at every possible threshold (in practice, at selected intervals), then plotting TPR (on the y axes) over FPR (on the y axes). The area under the curve (AUC) can be used to summarize the overall performance of a classifier over all possible thresholds, the larger the AUC the better the classifier.

b) (3P)

Split the data into a training and a test set as follows:

```
set.seed(1234)
samples <- sample(1:dim(default)[1], 0.7 *  dim(default)[1], replace = F)
train <- default[samples, ]
test <- default[-samples, ]
```

(i) (1P) Use a K-Nearest Neighbors classifier (using $K = 35$) to predict default on the training set.
(ii) (2P) Generate the confusion table and calculate the error rate, sensitivity and specificity for the prediction on the test set. Compute also the test error rate.

R-hints:

```
pred.knn = knn(..., ..., ...)
```

Solution

(i)

```
pred.knn = knn(train[,-23], test[,-23], cl = (train$DEFAULT), k = i, l = 0)
```

(ii)

```
confusionKNN = table(true=test$DEFAULT, predict=pred.knn)
confusionKNN
```

```
##     predict
## true    0    1
##    0 7002   34
##    1 1917   47
```

```
sensKNN = confusionKNN[2, 2]/(sum(confusionKNN[2,]))
spesKNN = confusionKNN[1, 1]/(sum(confusionKNN[1,]))
c(sensitivity = sensKNN, specificity = spesKNN)
```

```
## sensitivity specificity
##  0.02393075  0.99516771
```

```
test.errorKNN = mean(test$DEFAULT != pred.knn)
c(test_error= test.errorKNN)
```

```
## test_error
##  0.2167778
```

c) (4P)

(i) (3P) Now, choose a tree-based method and fit it using the training data, then predict the response using the test data and compute sensitivity, specificity and error rate. 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 default, according to an importance measure based on node purity?

Solution

(i) Different solutions are possible, here we describe a random forest.

```
mod.rf = randomForest(as.factor(DEFAULT) ~ ., data = train, mtry = 5, ntree = 500,
    importance = TRUE)

pred.rf <- predict(mod.rf, test, type = "class")
```

Here we have to choos `mtry`, which should be ca $\sqrt{p}$, with $p = 22$ (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".

```
confusionRF = table(true=test$DEFAULT, predict=pred.rf)
confusionRF
```

```
##     predict
## true    0    1
##    0 6670  366
##    1 1260  704
```

```
sensRF = confusionRF[2, 2]/(sum(confusionRF[2,]))
spesRF = confusionRF[1, 1]/(sum(confusionRF[1,]))
c(sensitivity = sensRF, specificity = spesRF)
```

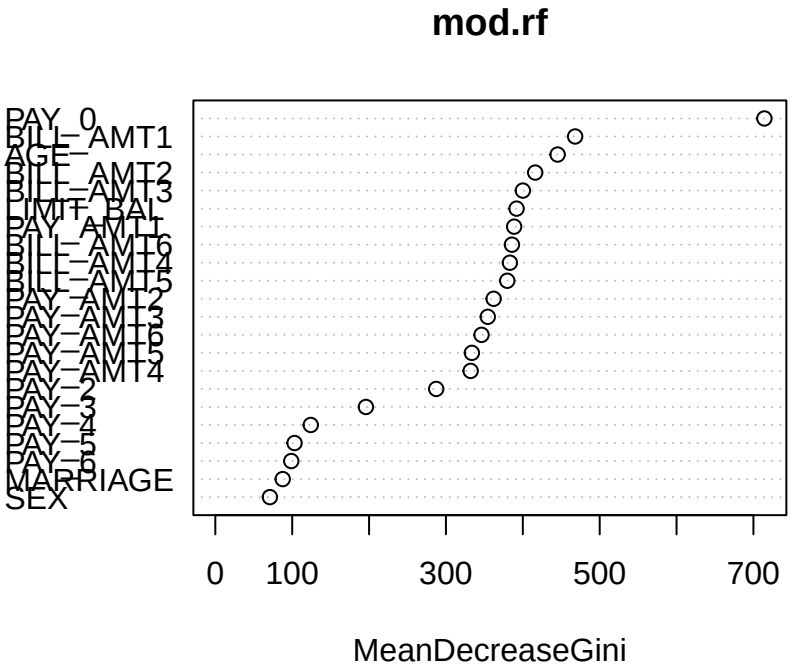
```
## sensitivity specificity
##   0.3584521   0.9479818
```

```
test.errorRF = mean(test$DEFAULT != pred.rf)
c(test_error= test.errorRF)
```

```
## test_error
##  0.1806667
```

(ii)

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



The most important predictors are `PAY_0` and `BILL_AMT1`

d) (3P)

(i) (2P) Use the logistic regression model in a), fitted to the training to predict the DEFAULT in the test set. Compute sensitivity, specificity and error rate. Compare the models you have fitted, which one would you choose?

(ii) (1P) Finally, look at the distribution of the DEFAULT variable, reflect on how this is related to the values of sensitivity, specificity and error rate you have computed.

Solution

(i)

```
mod.glm = glm(DEFAULT~ . + SEX:PAY_0, data = train, family = "binomial")

pred.log = round(predict(mod.glm, newdata = test, type = "response"))

confusionLOG = table(true=test$DEFAULT, predict=pred.log)
confusionLOG
```

```
##     predict
## true    0    1
##    0 6750  286
##    1 1365  599
```

```
sensLOG = confusionLOG[2, 2]/(sum(confusionLOG[2,]))
spesLOG = confusionLOG[1, 1]/(sum(confusionLOG[1,]))
c(sensitivity = sensLOG, specificity = spesLOG)
```

```
## sensitivity specificity
##   0.3049898   0.9593519
```

```
test.errorLOG = mean(test$DEFAULT != pred.log)
c(test_error= test.errorLOG)
```

```
## test_error
##  0.1834444
```

```
data.frame(
  mod = c("logistic_reg", "KNN", "RF"),
  sens = c(sensLOG, sensKNN, sensRF),
  spec = c(spesLOG, spesKNN, spesRF),
  error = c(test.errorLOG, test.errorKNN, test.errorRF))
```

```
##            mod       sens      spec     error
## 1 logistic_reg 0.30498982 0.9593519 0.1834444
## 2          KNN 0.02393075 0.9951677 0.2167778
## 3           RF 0.35845214 0.9479818 0.1806667
```

The RF model seems to have the best performance. It has the lowest error and the highest sensitivity.

(ii)

```
table(default$DEFAULT)/length(default$DEFAULT) * 100
```

```
##
##     0     1
## 77.88 22.12
```

The dataset is unbalanced with a lot more 0s (ca 78%) than 1s (ca 22%). In this case, the model learns to favor the majority class (0) because it can get high overall accuracy just by predicting 0 most of the time. As a result, it correctly predicts 0 very often (high specificity) and it it misses the rare 1s (low sensitivity).