Compulsory exercise 3

TMA4268 Statistical Learning V2020

Solfrid H. Johansen

29 April, 2020

Problem 1

a)

```
# Computing mean and standard deviation, using training data (not outstate)
mean = apply(college.train[, -9], 2, mean)
std = apply(college.train[, -9], 2, sd)

# Applying normalization to both training and test data
college.train[, -9] = scale(college.train[, -9], center = mean, scale = std)
college.test[, -9] = scale(college.test[, -9], center = mean, scale = std)
```

b)

Equation which determines the value of output layer, using a linear function for the activation function for the output layer.

$$\hat{y}(\boldsymbol{x}) = c_0 + \sum_{n=1}^{64} c_n \max\Big(0, \beta_{0n} + \sum_{m=1}^{64} \beta_{mn} \max\Big(0, k_{0m} + \sum_{j=1}^{17} k_{jm} x_j\Big)\Big)$$

As the goal is regression, we only want one output node, which predicts the value of outstate. Thus a linear activation function is chosen, which gives a values, unlike e.g. sigmoid where we get a probability.

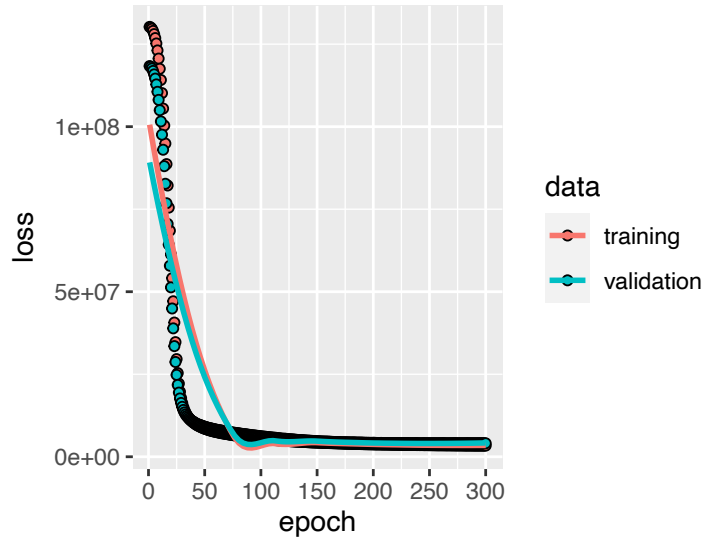
c)

```
college.train.x = model.matrix(Outstate ~ ., college.train)[, -1]
college.train.y = college.train$Outstate
college.test.x = model.matrix(Outstate ~ ., college.test)[, -1]
college.test.y = college.test$Outstate

set.seed(123)
# install.packages('keras') library(keras) install_keras()

# Nr. of predictors
```

```
p = ncol(college.train) - 1
# Define model
network = keras_model_sequential() %>% layer_dense(units = 64, activation = "relu",
    input_shape = c(p)) %>% layer_dense(units = 64, activation = "relu") %>% layer_dense(units = 1,
    activation = "linear")
# Compile
network %>% compile(optimizer = "rmsprop", loss = "mse")
# Train
history = network %>% fit(college.train.x, college.train.y, epochs = 300, batch_size = 8,
    validation_split = 0.2)
plot(history)
```




```
# To report the test MSE
mse = network %>% evaluate(college.test.x, college.test.y)
```

MSE of the network is 3.7158284×10^6.

The errors I obtained in Assignement 2 were Forward selection: 4,2 mill Lasso: 3,7 mill Polynomial regression: approx. 11 mill Smoothing spline: 5,9 mill Random forest: approx. 4 mill

Thus, the error obtained is as the Lasso method, which was the method that gave the smallest error. Therefore from the test MSE the neural network performs well. However interpretation is difficult, which can be an advantage for the Lasso.

d)

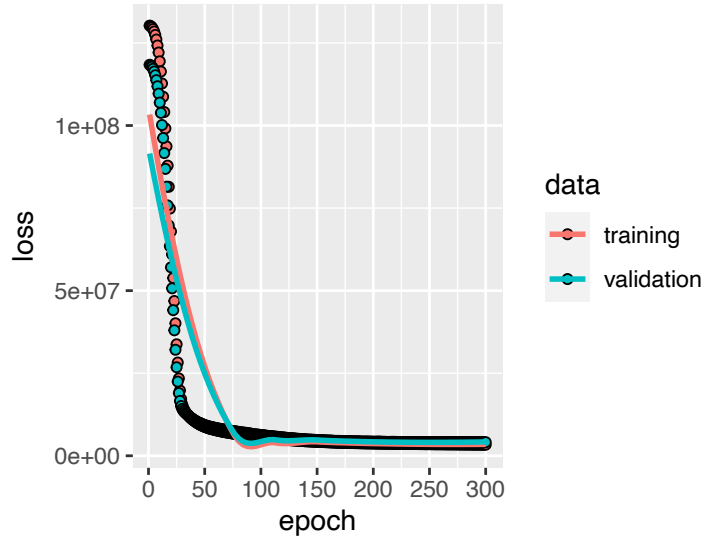
```
set.seed(123)

network = keras_model_sequential() %>% layer_dense(units = 64, activation = "relu",
    input_shape = c(p), kernel_regularizer = regularizer_l2(l = 0.001)) %>% layer_dense(units = 64,
    activation = "relu", kernel_regularizer = regularizer_l2(l = 0.001)) %>% layer_dense(units = 1,
    activation = "linear")
```

```
# Compile
network %>% compile(optimizer = "rmsprop", loss = "mse")

# Train
history = network %>% fit(college.train.x, college.train.y, epochs = 300, batch_size = 8,
    validation_split = 0.2)

plot(history)
```




```
# To report the test MSE
mse_regulized = network %>% evaluate(college.test.x, college.test.y)
```

The test MSE found was 3.8935039×10^6. This is slighlty smaller than the unregulized, as expected, as this avoids overfitting and the original network contained a lot of epochs.

Problem 2

a)

```
# Nr. deceased for each country
france <- subset(d.corona, country == "France", select = c(deceased))
deceased_france = sum(france$deceased)
japan <- subset(d.corona, country == "japan", select = c(deceased))
deceased_japan = sum(japan$deceased)
korea <- subset(d.corona, country == "Korea", select = c(deceased))
deceased_korea = sum(korea$deceased)
indonesia <- subset(d.corona, country == "indonesia", select = c(deceased))
deceased_indonesia = sum(indonesia$deceased)
countries = c("France", "Japan", "Korea", "Indonesia")
deceased = c(deceased_france, deceased_japan, deceased_korea, deceased_indonesia)
deceased_d = data.frame(Country = c("France", "Japan", "Korea", "Indonesia"), Deseased = deceased)
deceased_d
```

```
##     Country Deseased
## 1    France       14
## 2     Japan        3
## 3     Korea       26
## 4 Indonesia        2
```

```r
# Number of deceased for each sex
female <- subset(d.corona, sex == "female", select = c(deceased))
deceased_female = sum(female$deceased)
male <- subset(d.corona, sex == "male", select = c(deceased))
deceased_male = sum(male$deceased)
gender = c("Female", "Male")
deceased_gender = c(deceased_female, deceased_male)
deceased_g = data.frame(Gender = gender, Deseased = deceased_gender)
deceased_g
```

```
##   Gender Deseased
## 1 Female       14
## 2   Male       31
```

```r
# Nr deceased, for each sex pr country
france_f = subset(d.corona, country == "France" & sex == "female", select = c(deceased))
deceased_france_f = sum(france_f$deceased)
nondeceased_france_f = nrow(france_f)
japan_f <- subset(d.corona, country == "japan" & sex == "female", select = c(deceased))
deceased_japan_f = sum(japan_f$deceased)
nondeceased_japan_f = nrow(japan_f)
korea_f <- subset(d.corona, country == "Korea" & sex == "female", select = c(deceased))
deceased_korea_f = sum(korea_f$deceased)
nondeceased_korea_f = nrow(korea_f)
indonesia_f <- subset(d.corona, country == "indonesia" & sex == "female", select = c(deceased))
deceased_indonesia_f = sum(indonesia_f$deceased)
nondeceased_indonesia_f = nrow(indonesia_f)
france_m <- subset(d.corona, country == "France" & sex == "male", select = c(deceased))
deceased_france_m = sum(france_m$deceased)
nondeceased_france_m = nrow(france_m)
japan_m <- subset(d.corona, country == "japan" & sex == "male", select = c(deceased))
deceased_japan_m = sum(japan_m$deceased)
nondeceased_japan_m = nrow(japan_m)
korea_m <- subset(d.corona, country == "Korea" & sex == "male", select = c(deceased))
deceased_korea_m = sum(korea_m$deceased)
nondeceased_korea_m = nrow(korea_m)
indonesia_m <- subset(d.corona, country == "indonesia" & sex == "male", select = c(deceased))
deceased_indonesia_m = sum(indonesia_m$deceased)
nondeceased_indonesia_m = nrow(indonesia_m)
countries = c("France", "Japan", "Korea", "Indonesia")
deceased_f = c(deceased_france_f, deceased_japan_f, deceased_korea_f, deceased_indonesia_f)
deceased_m = c(deceased_france_m, deceased_japan_m, deceased_korea_m, deceased_indonesia_m)
survive_f = c(nondeceased_france_f, nondeceased_japan_f, nondeceased_korea_f, nondeceased_indonesia_f)
survive_m = c(nondeceased_france_m, nondeceased_japan_m, nondeceased_korea_m, nondeceased_indonesia_m)
deceased_g_c = data.frame(Countries = countries, DeseasedFemale = deceased_f, TotalFemale = survive_f,
    DeseasedMale = deceased_m, TotalMale = survive_m)
deceased_g_c
```

```
##   Countries DeseasedFemale TotalFemale DeseasedMale TotalMale
## 1    France              5          60            9        54
## 2     Japan              0         120            3       174
## 3     Korea              8         879           18       654
## 4 Indonesia              1          30            1        39
```

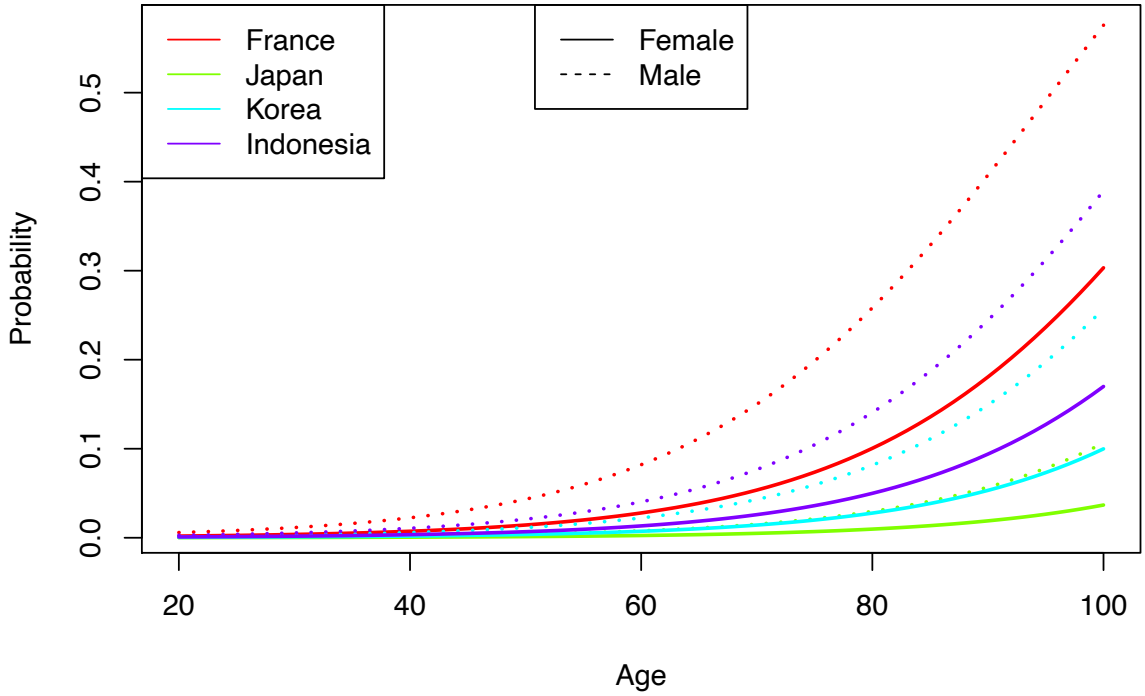
b)

Answers: **FALSE, FALSE, TRUE, FALSE**

c)

Create a plot of probabilities to die of coronavirus as a function of age, separately for the two sexes and each country.

```
# Lines for each country, and each sex.
nr_lines_color = 4
co = rainbow(4)
# Logistic regression
fit = glm(deceased ~ ., data = d.corona, family = binomial)
contries = c("France", "japan", "Korea", "indonesia")
for (i in 1:4) {
    male = expand.grid(sex = "male", age = seq(20, 100, 1), country = contries[i])
    glm.male = predict(fit, newdata = male, type = "response")
    female = expand.grid(sex = "female", age = seq(20, 100, 1), country = contries[i])
    glm.female = predict(fit, newdata = female, type = "response")
    if (i == 1) {
        plot(glm.male, x = seq(20, 100, 1), type = "l", ylab = "Probability", xlab = "Age",
            lty = "dotted", col = "white")
    }
    lines(glm.female, x = seq(20, 100, 1), col = co[i], lwd = 2)
    lines(glm.male, x = seq(20, 100, 1), col = co[i], lwd = 2, lty = "dotted")
}
legend("topleft", legend = c("France", "Japan", "Korea", "Indonesia"), col = co,
    lty = 1)
legend("top", legend = c("Female", "Male"), lty = 1:2)
```



d)

i)

```
fit_sex = glm(deceased ~ sex, data = d.corona, family = binomial)
summary(fit_sex)$coef
```

```
##               Estimate Std. Error    z value     Pr(>|z|)
## (Intercept) -4.3410186  0.2689957 -16.137873 1.382224e-58
## sexmale      0.9837844  0.3251774   3.025377 2.483232e-03
```

As $\beta_{male} = 0.9837$ with a small p-value, the **the probability of dying as a male is greater than dying as a female.**

ii)

```
fit = glm(deceased ~ country + age + sex, data = d.corona, family = binomial)
# Including an interaction term between sex and age
fit_interaction = glm(deceased ~ country + age + sex + sex * age, data = d.corona,
    family = binomial)
anova(fit_interaction, fit, test = "F")
```

```
## Analysis of Deviance Table
##
## Model 1: deceased ~ country + age + sex + sex * age
## Model 2: deceased ~ country + age + sex
##   Resid. Df Resid. Dev Df  Deviance      F Pr(>F)
## 1      2003     321.05
## 2      2004     321.07 -1 -0.018751 0.0188 0.8911
```

The p-value of a model including interactions between `age` and `sex` is large, so age is not a higher risk factor for males.

iii)

```
france = subset(d.corona, country == "France")
fit = glm(deceased ~ age, data = france, family = binomial)
summary(fit)$coef
```

```
##               Estimate Std. Error   z value     Pr(>|z|)
## (Intercept) -9.2210045 2.35913502 -3.908638 9.281797e-05
## age          0.0955312 0.02804438  3.406429 6.581868e-04
```

```
korea = subset(d.corona, country == "Korea")
fit = glm(deceased ~ age, data = korea, family = binomial)
summary(fit)$coef
```

```
##                Estimate Std. Error   z value     Pr(>|z|)
## (Intercept) -8.48400343 0.92081259 -9.213605 3.153521e-20
## age          0.06893075 0.01160246  5.941046 2.832090e-09
```

```
fit_no_age = glm(deceased ~ country + age + sex, data = d.corona, family = binomial)
fit_inter = glm(deceased ~ country + age + sex + country * age, data = d.corona,
    family = binomial)
anova(fit_no_age, fit_inter, test = "F")
```

```
## Analysis of Deviance Table
##
## Model 1: deceased ~ country + age + sex
## Model 2: deceased ~ country + age + sex + country * age
##   Resid. Df Resid. Dev Df Deviance      F Pr(>F)
## 1      2004     321.07
## 2      2001     315.74  3   5.3273 1.7758 0.1493
```

The p-value of including a `country` and `age` interaction term is large, so age is not a greater risk factor for the French population than for the Korean.

e)

As there is not enough information about how the data was collected, and we do not know who is tested (and thus included in the data) we cannot conclude that France has a higher risk of dying. It could for instance

be that France is only testing those who get very sick, while other countries test those who bare have any symptoms. Therefore, many people in France can have the virus, so that the probability is in fact lower. However, it could be that France has an overall older population, but again we see from the plot in c) that the probability as a function of age is (for ages above approximately 40) higher than for other countries.

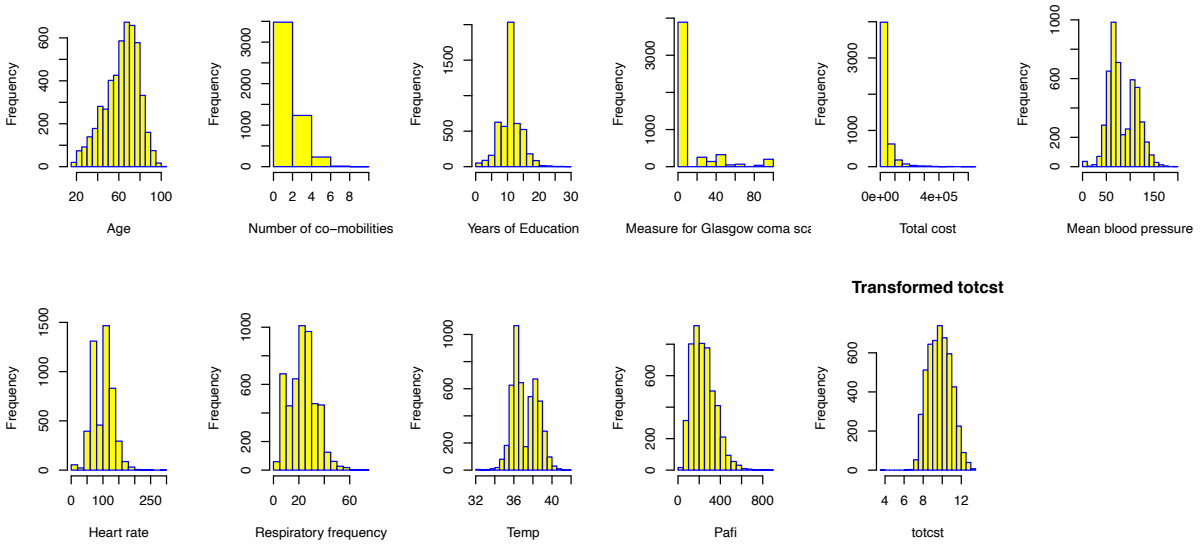
f)

Answers: **TRUE, TRUE, FAlSE, FALSE**

Problem 3

a)

```
par(mfrow = c(2, 6))
hist(as.numeric(as.character(d.support$age)), xlab = "Age", col = "yellow", border = "blue",
    main = "")
hist(as.numeric(as.character(d.support$num.co)), xlab = "Number of co-mobilities",
    col = "yellow", border = "blue", breaks = 5, main = "")
hist(d.support$edu, xlab = "Years of Education", col = "yellow", border = "blue",
    main = "")
hist(d.support$scoma, xlab = "Measure for Glasgow coma scale", col = "yellow", border = "blue",
    main = "")
hist(d.support$totcst, xlab = "Total cost", col = "yellow", border = "blue", main = "")
hist(d.support$meanbp, xlab = "Mean blood pressure", col = "yellow", border = "blue",
    main = "")
hist(as.numeric(as.character(d.support$hrt)), xlab = "Heart rate", col = "yellow",
    border = "blue", main = "")
hist(d.support$resp, xlab = "Respiratory frequency", col = "yellow", border = "blue",
    main = "")
hist(d.support$temp, xlab = "Temp", col = "yellow", border = "blue", main = "")
hist(d.support$pafi, xlab = "Pafi", col = "yellow", border = "blue", main = "")
hist(log(d.support$totcst), xlab = "totcst", col = "yellow", border = "blue", main = "Transformed totcs
```



Transforming `totcst` by log, which improves the distribution, as seen in the plot below. Also from the plots in b) ii) this seems like a good transformation of the data (fulfills assumption).

b)

i)

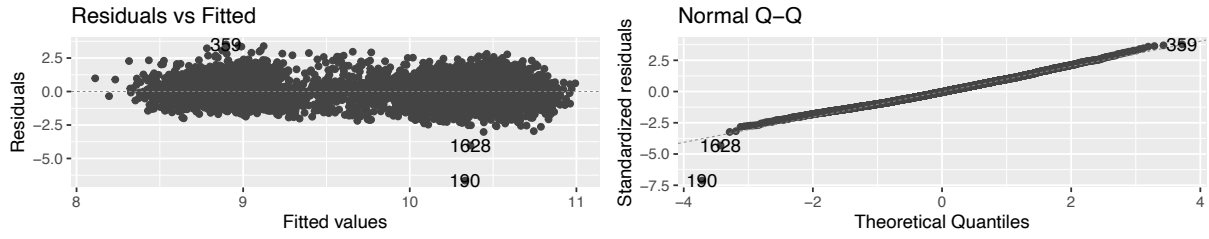
```
# Making numeric, instead of factor
d.support$age = as.numeric(as.character(d.support$age))
d.support$num.co = as.integer(as.character(d.support$num.co))
lm.fit = lm(log(totcst) ~ age + temp + edu + resp + num.co + dzgroup, data = d.support)
beta_age = coefficients(lm.fit)[2]
exp(beta_age * 10)
```

```
##       age
## 0.9324402
```

I.e. `totcst` changes by a factor 0.93 when `age` increases by 10.

ii)

```
autoplot(lm.fit, smooth.colour = NA)[1:2]
```



In the plot to the left (Tukey-Anscombe) the values should be centered around 0.0, so that there are approximately as many values above and below. In the plot, the values appear to have approximately equal variance and the expectation value is around 0. The QQ-diagram to the rigth checks if the assumption on normally distributed residuals is fulfilled. The plot should be approximately a straight line, which it is in this case. The assumptions that the residuals follow $N(0, \sigma^2)$ seems to be fulfilled from these plots.

iii)

Fitting a model with interaction between `age` and `dzgroup` and performing hypthesis test with `anova` function.

```
set.seed(10)
lm.fit = lm(log(totcst) ~ age + temp + edu + resp + num.co + dzgroup, data = d.support)
lm.fit_interaction = lm(log(totcst) ~ age + temp + edu + resp + num.co + dzgroup +
    age * dzgroup, data = d.support)
anova(lm.fit, lm.fit_interaction)
```

```
## Analysis of Variance Table
##
## Model 1: log(totcst) ~ age + temp + edu + resp + num.co + dzgroup
## Model 2: log(totcst) ~ age + temp + edu + resp + num.co + dzgroup + age *
##     dzgroup
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1   4947 4312.9
## 2   4940 4288.3  7    24.541 4.0387 0.0002019 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the table, the p-value is very small, so that the effect of age does depend on the disease group.

c

```
library(glmnet)
set.seed(12345)
train.ind = sample(1:nrow(d.support), 0.8 * nrow(d.support))
d.support.train = d.support[train.ind, ]
d.support.test = d.support[-train.ind, ]
# CV to find largest lamda, so error within one standard error of the smallest
# lamda
x = model.matrix(log(totcst) ~ age + temp + edu + resp + num.co + dzgroup, d.support.train)[,
    -1]
y = log(d.support.train$totcst)
x_1 = model.matrix(log(totcst) ~ age + temp + edu + resp + num.co + dzgroup, d.support.test)[,
    -1]
y_1 = log(d.support.test$totcst)
x_all = model.matrix(log(totcst) ~ age + temp + edu + resp + num.co + dzgroup, d.support)[,
    -1]
y_all = log(d.support$totcst)

# Find lambda
```

```
cv.out = cv.glmnet(x, y, alpha = 0)
lamb = cv.out$lambda.1se
ridge.mod = glmnet(x, y, alpha = 0, lambda = lamb)
ridge.predict = predict(ridge.mod, s = lamb, newx = x_1)
lamb
```

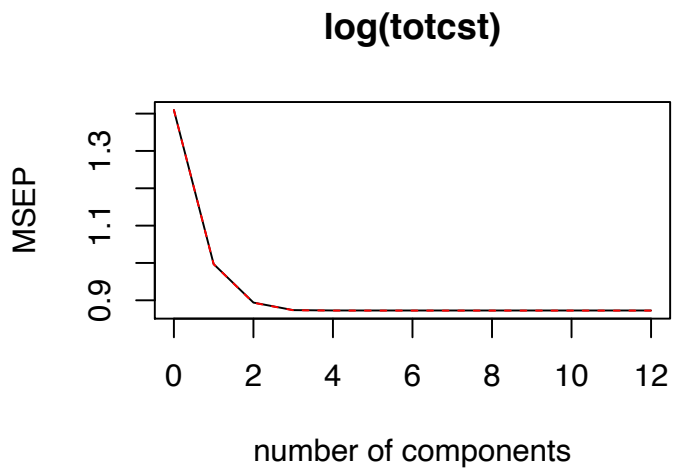
```
## [1] 0.1553992
```

```
mean((ridge.predict - y_1)^2)
```

```
## [1] 0.896075
```

d)

```
set.seed(1234)
library(pls)
pls.fit = plsr(log(totcst) ~ age + temp + edu + resp + num.co + dzgroup, data = d.support.train,
    scale = TRUE, validation = "CV")
# pls.fit = plsr(log(totcst)~., data = d.support.train, scale=TRUE,
# validation='CV')
validationplot(pls.fit, val.type = "MSEP")
```



The CV-error stops decreasing at 3 components, so this is chosen as the number of PCs.

```
pls.pred = predict(pls.fit, d.support.test, ncomp = 4)
mean((pls.pred - y_1)^2)
```

```
## [1] 0.8838701
```

The test MSE is smaller than for ridge regression, thus from the test MSE, PLS is a better choice.

e)

i)

Using smoothing spline

```
library(gam)

fit = gam(log(totcst) ~ s(age, 1) + s(edu, 1) + s(resp, 8) + s(num.co, 4) + s(temp,
    3) + dzgroup, data = d.support.train)
preds = predict(fit, newdata = d.support.test)
testError = mean((preds - log(d.support.test$totcst))^2)
testError
```

```
## [1] 0.8605738
```

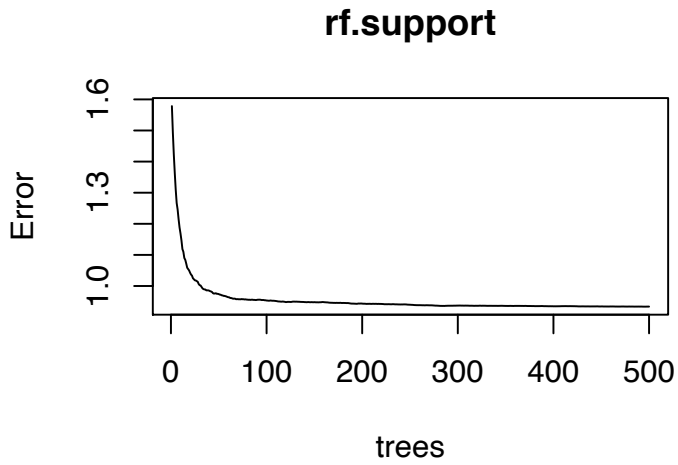
ii)

Choosing a random forest, with $m = 7$.

```
library(randomForest)

d.support.train$temp = as.numeric(d.support.train$temp)

rf.support = randomForest(log(totcst) ~ age + edu + resp + num.co + dzgroup, data = d.support.train,
    importance = TRUE, mtry = 7)
pred = predict(rf.support, newdata = d.support.test)
testError = mean((pred - log(d.support.test$totcst))^2)
# rf.support = randomForest(log(totcst) ~., data = d.support.train, importance =
# TRUE, mtry = 7)
plot(rf.support)
```



```r
yhat.rf = predict(rf.support, newdata = d.support.test)
# Computing MSE, mean difference between predicted value and actual value
testError = mean((yhat.rf - log(d.support.test$totcst))^2)
testError
```

```
## [1] 0.9172537
```

Problem 4

a)

The basis functions are $b_1(x) = x, b_2(x) = x^2, b_3(x) = x^3$, and (trucated power basis functions)

$$b_4(x,1) = \begin{cases} (x-1)^3 & \text{if } x > 1 \\ 0 & \text{otherwise} \end{cases}$$

and

$$b_5(x,2) = \begin{cases} (x-2)^3 & \text{if } x > 2 \\ 0 & \text{otherwise} \end{cases}$$

The design matrix is

$X_{ij} = b_j(x_i)$

b)

(By TRUE I mean that it is suitable)

Answers: **TRUE, TRUE, TRUE, FALSE**

c)

Answers: **FALSE, FALSE, FALSE, FALSE**

Problem 5

a) Answers: **TRUE, TRUE, FALSE, FALSE**

b) Answers: **FALSE, TRUE, FALSE, TRUE**

c) Answer: **iv)**

d) Answer: **ii)**

e) Answer: **i)**

f) Answers: **TRUE, TRUE, FALSE, TRUE**

g) Answers: **FALSE, TRUE, TRUE, TRUE**