Compulsory exercise 3

TMA4268 Statistical Learning V2020

Silje Anfindsen

28 April, 2020

Problem 1: College tuition

a) Preprocessing

Start with applying feature-wise normalization to the predictors

```
#divide data into response and covariates
train_x <- college.train[,-9] #remove Outstate
test_x <- college.test[,-9]
train_y <- college.train[,9] #only incl. Outstate
test_y <- college.test[,9]

#find mean and std of the training data
mean <- apply(train_x , 2, mean)
std <- apply(train_x, 2, sd)

#normalization of the covariates
train_x <- scale(train_x, center = mean, scale = std)
test_x <- scale(test_x, center = mean, scale = std)
```

b) Network equation

Below is the equation describing a network that predicts `Outstate` with 17 predictors (nodes) in the input layer, and 2 hidden layers using `ReLu` activation function. The output layer has 1 node wich is continous numerical (response), implying a regression problem in statistics. We therefore choose the `linear` activation function for this layer.

$$\hat{y_1}(x) = \beta_{01} + \sum_{m=1}^{64} \beta_{m1} \max(\gamma_{0m} + \sum_{l=1}^{64} \gamma_{lm} \max(\alpha_{0l} + \sum_{j=1}^{17} \alpha_{jl} x_j, 0), 0)$$

skal det være med bias term eller ikke for hvert lag?

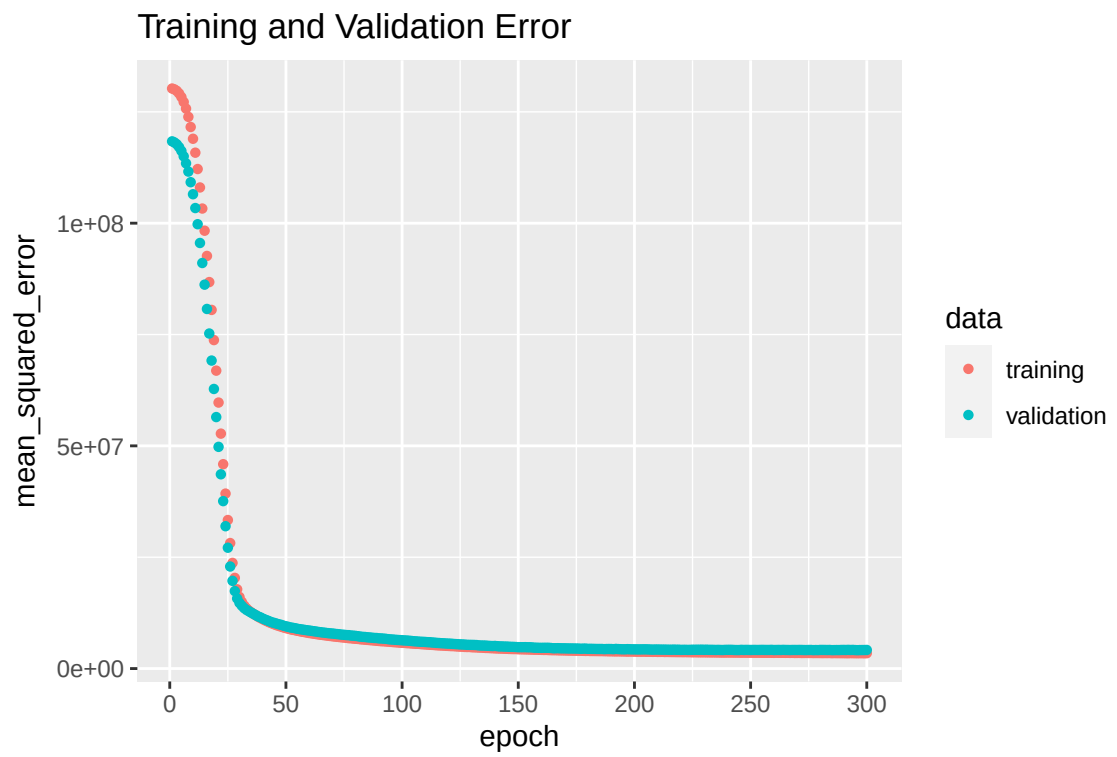
c) Implement network

(i) First, train the network from above using 20% of the training data as validation set.

```r
set.seed(123)

#define the model
model = keras_model_sequential() %>%
  layer_dense(units = 64, activation = 'relu', input_shape = dim(train_x)[2]) %>%
  layer_dense(units = 64, activation = 'relu') %>%
  layer_dense(units = 1 , activation = 'linear')

#compile
model %>% compile(optimizer = "rmsprop", loss = "mse", metrics = "mean_squared_error")

#train
history = model %>% fit(train_x, train_y, epochs = 300, batch_size = 8,
    validation_split = 0.2) #20% of the training set as validation set
```

(ii) I have used `mse` as the loss function and metric and `RMSprop` as optimizer in the training phase. The plot below shows the training and validation error as a function of epochs.

```r
plot(history, metrics = "mean_squared_error", smooth = FALSE) +
  ggtitle("Training and Validation Error")
```



(iii)

```r
error<-  model %>% evaluate(test_x, test_y)
nn_mse<- error$mean_squared_error
```

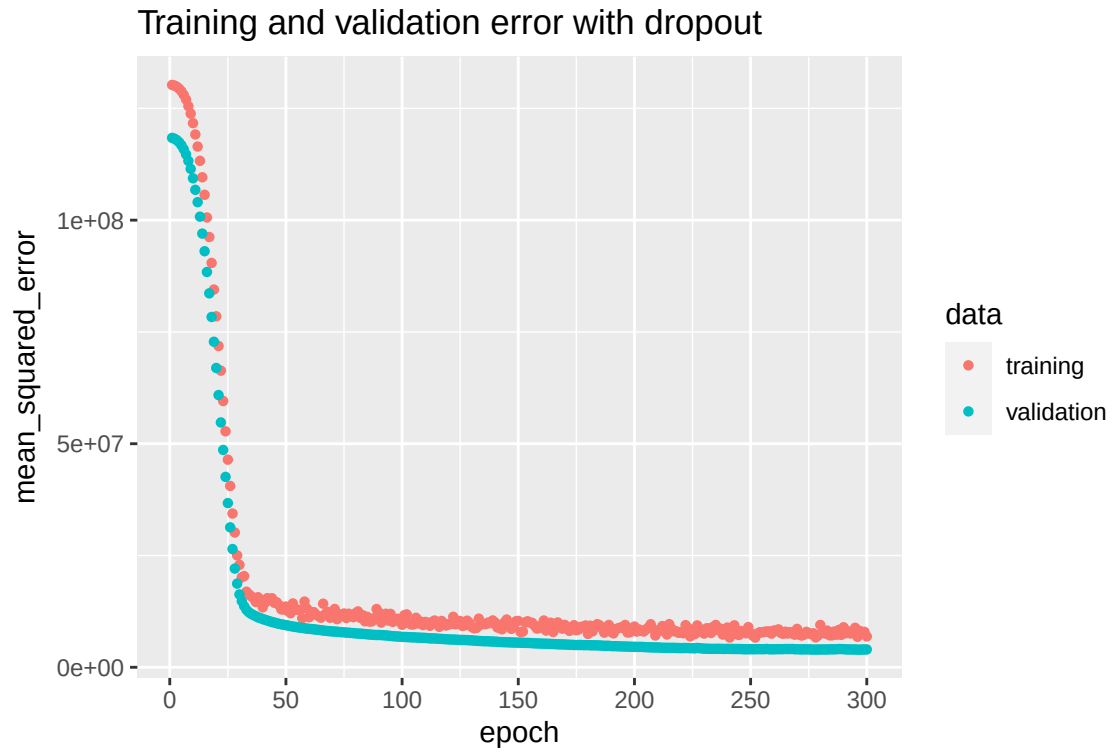
The test MSE for this network is 3.770505×10^6. From Compulsory 2, using the same dataset, the test MSE for forward selection is 4.11268×10^6 , for Lasso, 3.71702×10^6 and 2.607985×10^6 for random forest. We

observe that the test MSE for the network model and Lasso method is very alike. Forward selection has the higest test MSE. Random forests still has the lowest test MSE between the four methods.

d) Regularization

We will now add dropout to each of the hidden layers. The dropout rate is 0.4, indicating what fraction of features that are randomly being zeroed-out.

```
set.seed(123)

#define model, with dropout on each hidden layer
reg_model <- keras_model_sequential() %>%
  layer_dense(units = 64, activation = 'relu', input_shape = dim(train_x)[2]) %>%
              layer_dropout(0.4) %>%
  layer_dense(units = 64, activation = 'relu') %>%
              layer_dropout(0.4) %>%
  layer_dense(units = 1, activation = 'linear')

#compile
reg_model %>% compile(optimizer = "rmsprop", loss = "mse", metrics = "mean_squared_error")

#train
reg_history = reg_model %>% fit(train_x, train_y, epochs = 300, batch_size = 8,
    validation_split = 0.2)

#plot
plot(reg_history, metrics = "mean_squared_error", smooth = FALSE)+
  ggtitle("Training and validation error with dropout")
```



```r
#test mse
reg_error <-  reg_model %>% evaluate(test_x, test_y)
reg_mse <- reg_error$mean_squared_error
```

It is hard to notice any improvements by looking at the two plots with and without regularization. The training and validaiton loss seem to overlap, and the loss decreases rapidly during the first 30 epochs. Thereafter it decreases slowly. We observe that 40% of the features are zeroed out from the dropout. The test MSE for the regularized model is 3.5034693×10^6. Recall, for the non-regularized model the MSE is 3.770505×10^6. This indicates that the regularized model performs a bit better than the non-regularized model, but compared to random forests from compulsory 2 the improvement is not impressing.

Problem 2: Covid-19 infection

a) Inspecting the data

Table 1: The number of deceased for each `country`.

	Died	Survived
France	100	14
indonesia	67	2
japan	291	3
Korea	1507	26

Table 2: The number of deceased for each `sex`.

	Died	Survived
female	1075	14
male	890	31

Table 3: The number of deceased, separate for each `sex`, per `country`.

France

	Died	Survived
female	55	5
male	45	9

Indonesia

	Died	Survived
female	21	0
male	47	2

Japan

	Died	Survived
female	130	1
male	162	2

Korea

	Died	Survived
female	843	7
male	590	12

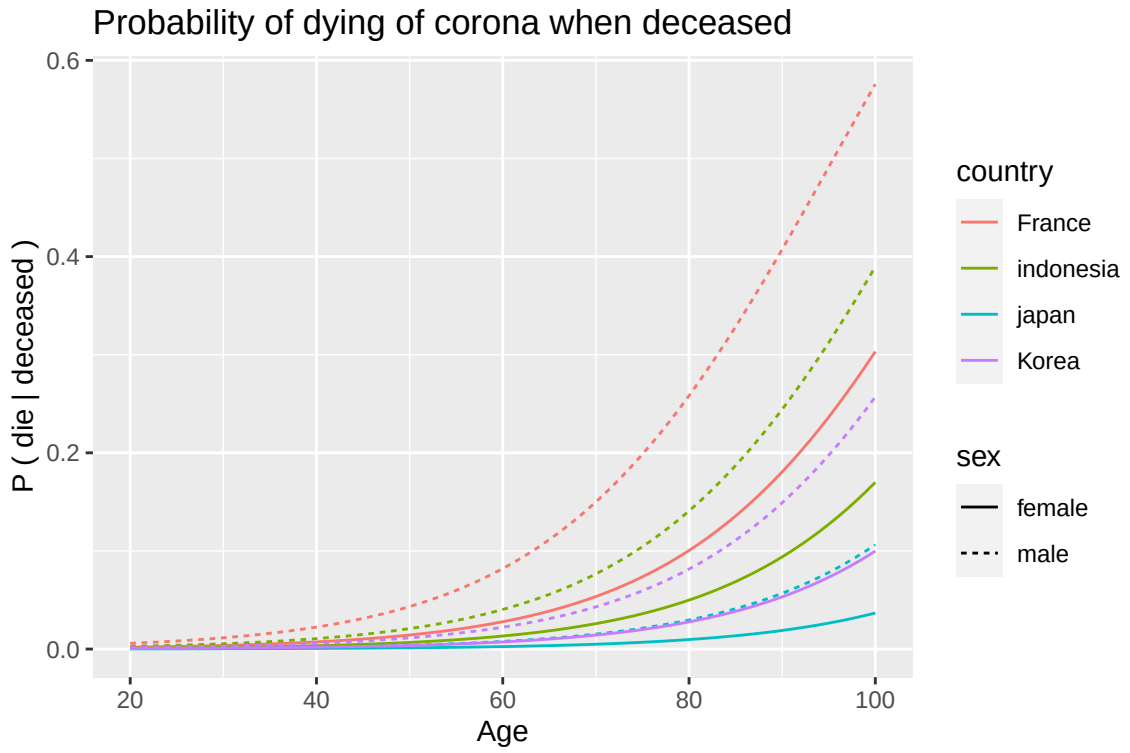
b) Multiple choice

FALSE, FALSE, TRUE, FALSE

c) Plot

The plot displays the probabilities to die of coronavirus given that the patient is deceased, as a function of `age`, coloured by `country` and linetype by `sex`.

```
#make list of age, country and sex
a <- seq(20,100,1)
s = unique(d.corona$sex)
c = unique(d.corona$country)

#generate gridded data and predict
newdata = expand.grid(age=a,country=c, sex=s)
p <- predict(model, newdata, type="response")

#plot
ggplot(d.corona, aes(x = age, y = deceased)) +
  geom_line(data = newdata, aes(x=age, y=p, col=country, linetype=sex)) +
  labs(title="Probability of dying of corona when deceased",
       y="P ( die | deceased )", x="Age")
```



d) Questions

(i) It seems to be a trend that males have higher probability of dying of corona compared to women from the plot in c). To verify this assumption we make a model with `sex` as the only covariate and look at the

coefficient estimate. Here we observe that its sign is positive, meaning males are more likely to die of corona virus than females.

```
mod1 <- glm(deceased ~ sex, data = d.corona, family = "binomial")
summary(mod1)$coefficients[2,]
```

```
##    Estimate  Std. Error     z value    Pr(>|z|)
## 0.983784353 0.325177412 3.025377274 0.002483232
```

(ii) In order to check if age is a greater risk factor for males than for females, we need to fit a model to see if the interaction between `age` and `sex:male` is significant. Here, we notice that the estimate for the interaction is small and negative which indicates that age actually is a greater risk factor for females than for males In addition, we also notice that the p-value is greater than the usual significance level indicating that the interaction is not significant.

```
mod2 <- glm(deceased ~ sex*age, data=d.corona, family = "binomial")
summary(mod2)$coefficients[4,]
```

```
##     Estimate   Std. Error      z value     Pr(>|z|)
## -0.004067455  0.020485348 -0.198554340  0.842611374
```

(iii) In order to check if age is a greater risk factor for the French populaiton compared to the Korean, we do the same procedure as in (ii). We fit a model with an interaction term between `country` and `age`. Let´s take a look at the coefficient estimate for `age:countryKorea` which gives the difference with respect to the reference category, France. It´s negative which means that age is a greater risk factor for the French population compared to the Korean. Again, the p-value is greater than the usual significance level which indicates that the interaction is not significant.

```
mod3 <- glm(deceased ~ age*country, data=d.corona, family = "binomial")
summary(mod3)$coefficients[8,]
```

```
##    Estimate  Std. Error     z value    Pr(>|z|)
## -0.02660045  0.03034971 -0.87646476  0.38077743
```

e) Interpret the model

When reporting the tables in a) one easily notice the inequality between the total number of observations per country. For example, there are about ten times more observations from Korea, and two times more from Japan compared to from France. Usually, more data implies a more accurate model. It is also important to research how the data was collected. For example, if the observations are collected from a certain group and not a random part of the population, the data and the following results will have less scientifical value. So, even though the French population seem to have a higher risk of dying from Covid-19, it does not necessarily represent the reality.

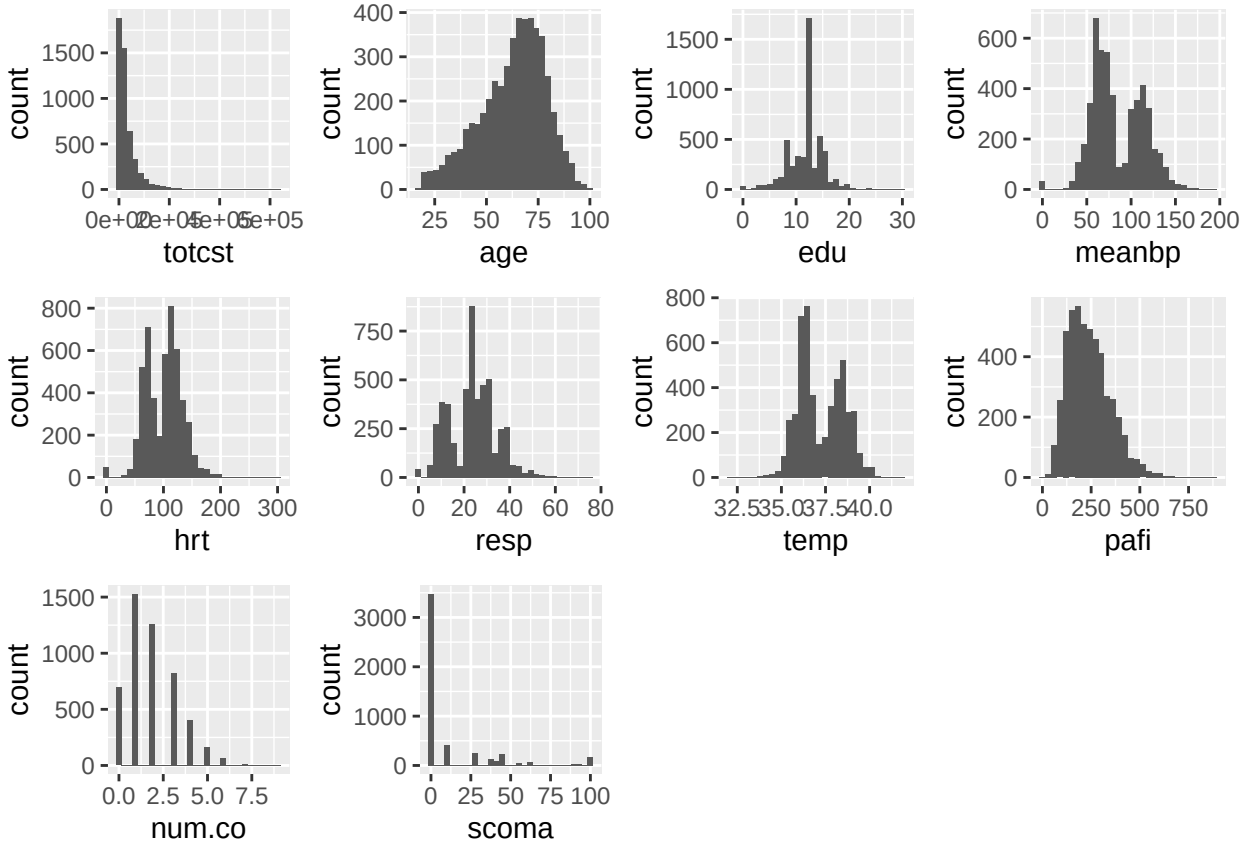
f) Multiple choice

TRUE, TRUE, FALSE, TRUE

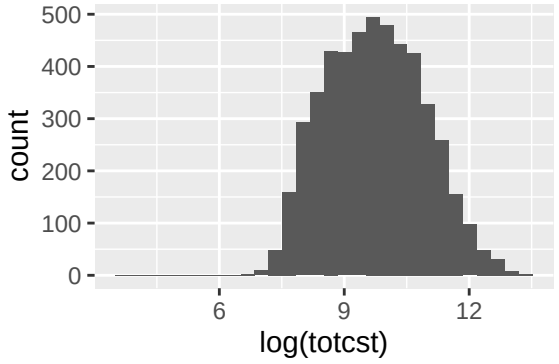
Problem 3: Hospital costs

a) Histograms

Below is a visualization of the distributions of all the continous and integer variables.



We notice that the distribution of the response `totcst` is right-skewed, so we try the transformation log(`totcst`) and notice that the distribution now looks normal.



b) Multiple linear regression model

Now fit is a multiple linear regression model with the transformed hospital costs log(`totcst`) as response and the following covariates: `age`,`temp`,`edu`,`resp`,`num.co` and `dzgroup`.

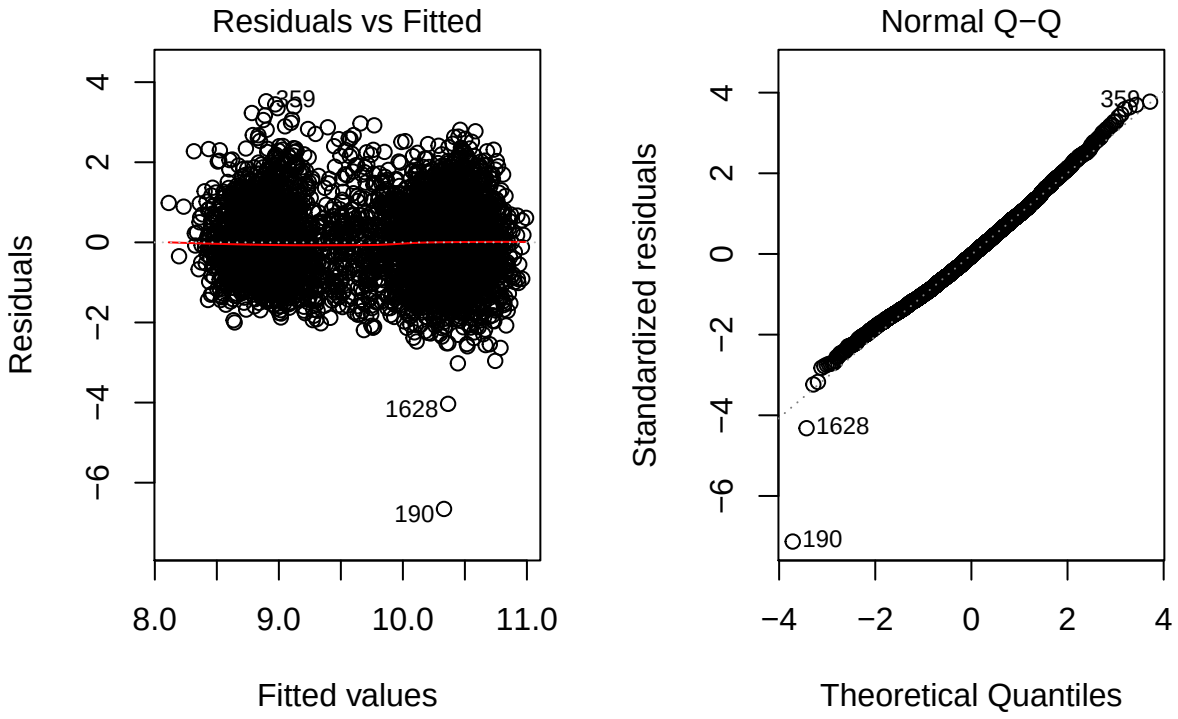
```
mod <- lm(log(totcst) ~ age+temp+edu+resp+num.co+dzgroup, data=d.support)
```

(i) In order to find how the response is affected when the patient's age increase by 10 years we look at the coefficient estimate for `age` and multiply it with 10 (1 year = 1 unit). The estimate which representes the difference in the predicted response for one unit change in the covariate is transformed back (since we have transformed y).

```
age10 <- 10^(coef(mod)[2]*10)
```

The total costs is expected to increase with a factor of **0.8512356** as the patient's age increases by 10 years.

(ii) We will now check if the model assumptions for a linear regression model are fulfilled.



Turkey-Anscombe diagram *(Residual vs. Fitted)*: there are two clear outliers out of the 4960 observations. The linearity seems to hold since the mean of the residuals (red line) lays exactly on the null-line (dashed line). The spread of the residuals seems to be constant along the line, which indicate homoscedasticity of residuals.
QQ-diagram: the points lay on a straight line and therefore strengthen the assumption about normallity.

(iii) We will now do a hypothesis test between the original model and a new model including the interaction between `age` and `dzgroup` (disease group).

```
## Analysis of Variance Table
##
## Model 1: log(totcst) ~ age + temp + edu + resp + num.co + dzgroup
## Model 2: log(totcst) ~ age + dzgroup + num.co + edu + income + scoma +
##     race + meanbp + hrt + resp + temp + pafi + age * dzgroup
```

```
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1   4947 4312.9
## 2   4927 4135.7 20    177.19 10.554 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value of the test is 2.2×10^{-16}, which is smaller than the usual significane level. It therefore suggests that the null hypothesis is false *(H0: interaction-term is zero)*. So the effect of age depend on the disease group.

c)

Now, use ridge regression to build a more robust model. First seperate 80% of the data into a training set, and the remaining 20% into a test set.

```
set.seed(12345)
train.ind = sample(1:nrow(d.support), 0.8 * nrow(d.support))
d.support.train = d.support[train.ind, ] #80% data
d.support.test = d.support[-train.ind, ] #20% data

#Make a x matrix and y vector for both the training and testing set
x_train<-model.matrix(log(totcst)~.,d.support.train)[,-1]
y_train<-log(d.support.train$totcst)
x_test<-model.matrix(log(totcst)~.,d.support.test)[,-1]
y_test<-log(d.support.test$totcst)
```

Then, for the training set, run cross validation and find the largest value of λ such that the error is within 1 std. error of the λ corresponding to the lowest MSE.

```
set.seed(555)
cv.ridge = cv.glmnet(x_train,y_train,alpha=0) #CV on train set
best.lambda = cv.ridge$lambda.1se
```

Now use $\lambda = 0.4526981$ to calculate the test MSE of the ridge regression.

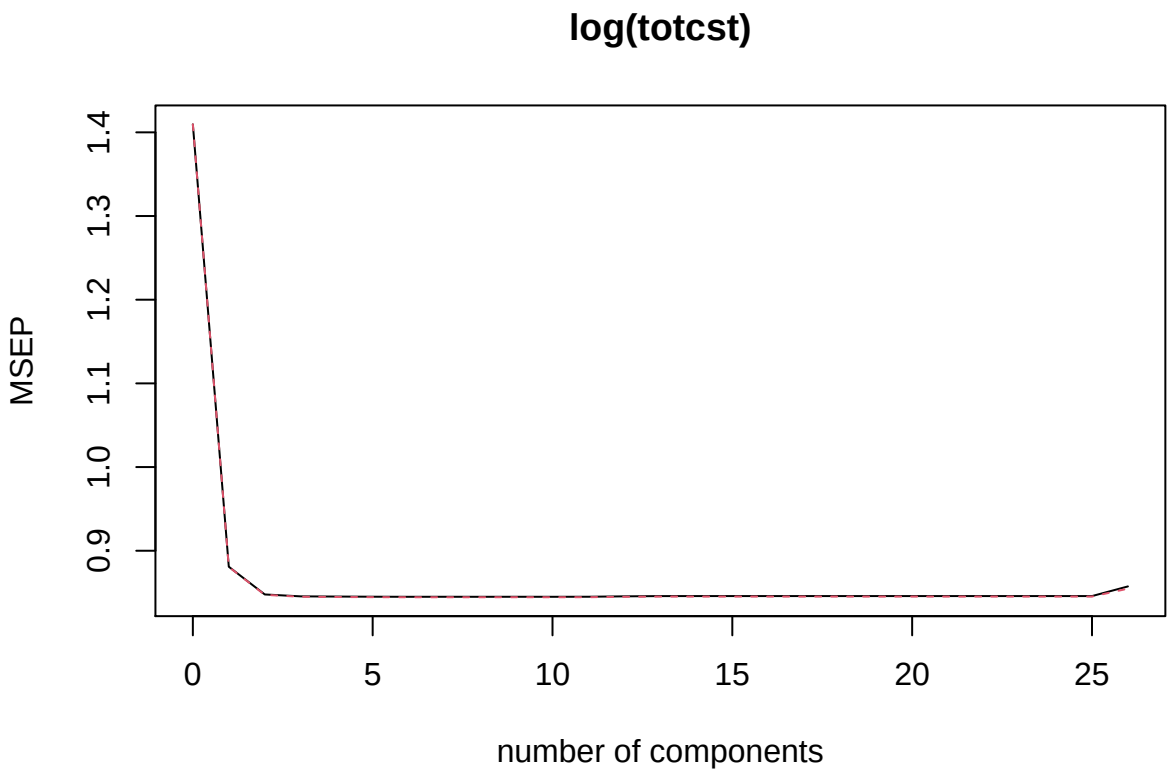
```
ridge.pred <- predict(cv.ridge, s=best.lambda, newx=x_test)
MSE_ridge <- mean((ridge.pred-y_test)^2)
```

The test MSE for ridge regression is **0.8809278**.

d)

(i) Now, run a partial least squares (PLS) regression.

```
set.seed(122)
mod.pls <- plsr(log(totcst)~.,data=d.support.train, validation="CV", scale=TRUE)
validationplot(mod.pls, val.type="MSEP")
```



(ii) We can read from the plot and that the lowest cross-validation error occurs for models using $M = 4$ or more partial least squares directions. As we want a simple model we choose the lowest number giving a small enough error.

```
pls.pred <- predict((mod.pls), x_test, ncomp=4)
MSE_pls <- mean((pls.pred-y_test)^2)
```

(iii) The test MSE using $M = 4$ PCs is **0.8638031**. Compared to the test MSE for ridge regression **0.8809278**, PLS regression seems to be even better model choice for prediction as its test MSE is a bit lower.

e)

Now, we will try to build models with even lower test MSE.

(i) First, try a non-linear transformation of the covariates combined to a GAM. The optimal model is investigated by trying out different combinations of which variables to included in the model and whether to these are linear, smoothing splines or polynomials of different degrees.

```
gam.mod <- gam(log(totcst) ~ s(age,5)+s(temp,6)+s(resp,5)+poly(edu,4)+ s(num.co,5)+dzgroup+poly(meanbp,3
gam.pred <- predict(gam.mod, newdata=d.support.test)
MSE_gam <- mean((gam.pred-y_test)^2)
```

The test MSE for the generalizes additive model is **0.8423729**.

(ii) For the second tree-based model I will try to fit a random forest method. This method which uses bootstrapping to build several trees is known for being quite accurate and flexible injecting more randomness in order to avoid strong predictors dominating the decision trees. For the tuning parameter (nr. of tree in each split) we choose $p/3 = 12/3 = 4$ variables.

```
set.seed(2)
rf.tree <- randomForest(log(totcst) ~ .,data=d.support.train, mtry=4, importance=TRUE)
rf.pred <- predict(rf.tree, newdata=d.support.test)
MSE_rf <- mean((rf.pred-y_test)^2)
```

The test MSE for random forest regression tree is **0.8243286**. Both random forest and GAM give a smaller test MSE compared to the regularized and dimension reduction methods used in c) and d). Random forests is clearly the best methods out of these four.

Problem 4: Mixed Questions

a) Cubic regression spline model

Basis functions: There are $k = 5$ basis functions for the given cubic regression spline model. We have two knots q_1, q_2 respectivelly at $x = 1$ and $x = 2$.

$$\begin{aligned} b_1(x_i) &= x_i & b_4(x_i, q_j) &= (x_i - q_j)_+^3 \\ b_2(x_i) &= x_i^2 & b_5(x_i, q_j) &= (x_i - q_j)_+^3 \\ b_3(x_i) &= x_i^3 \end{aligned}$$

Design matrix: The design matrix will have dimensions $n \times (k+1) = n \times 6$, where n is the number of observations.

$$X = \begin{pmatrix} 1 & b_1(x_1) & b_2(x_1) & b_3(x_1) & b_4(x_1, q_1) & b_5(x_1, q_2) \\ 1 & b_1(x_2) & b_2(x_2) & b_3(x_2) & b_4(x_2, q_1) & b_5(x_2, q_2) \\ 1 & b_1(x_3) & b_2(x_3) & b_3(x_3) & b_4(x_3, q_1) & b_5(x_3, q_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & b_1(x_n) & b_2(x_n) & b_3(x_m) & b_4(x_n, q_1) & b_5(x_n, q_2) \end{pmatrix}$$

b) Inference vs. prediction

TRUE, TRUE, TRUE, FALSE

c) Covid-19

FALSE?, TRUE?, TRUE, FALSE

Problem 5: Multiple Choice and single choice questions

a) Regularization

TRUE, TRUE, FALSE, FALSE?

b) PCR and PLS

FALSE, TRUE, TRUE?, TRUE

c) Ridge regression

(iv)

d) Curse of dimensionality

(ii)

e) KNN

(i)

f) Clinical study

TRUE, TRUE, FALSE, TRUE

g) Athletes' performance

TRUE, FALSE, TRUE, TRUE

References

James, G., D. Witten, T. Hastie, and R. Tibshirani. 2013. An Introduction to Statistical Learning with Applications in R. New York: Springer.