Compulsory Exercise 3

TMA4268 Statistical Learning

Helge Bergo

30 April, 2020

Problem 1

a)

Using the `College` data set, the training and test data was preprocessed, by separating the response and predictors into an x-matrix and y-vector for each set, and then scaling the predictors.

```
y.train = college.train$Outstate
y.test = college.test$Outstate

x.train <- subset(college.train, select = -c(Outstate))
x.test <- subset(college.test, select = -c(Outstate))

mean <- apply(x.train, 2, mean)
std <- apply(x.train, 2, sd)

x.train <- as.array(scale(x.train, center = mean, scale = std))
x.test <- as.array(scale(x.test, center = mean, scale = std))
```

b)

The equation for the network to predict `Outstate`, using an input layer with the 17 predictors and a `relu` activation function for the hidden layers is:

$$\hat{y}_1(\mathbf{x}) = \beta_{01} + \sum_{m=1}^{64} \beta_{m1} \max(\sum_{l=1}^{64} \gamma_{lm} \cdot \max(\sum_{j=1}^{17} \alpha_{jl} x_j, 0), 0)$$

The activation function chosen for the output layer was the `linear` function, since this is a regression problem.

c)

(i)

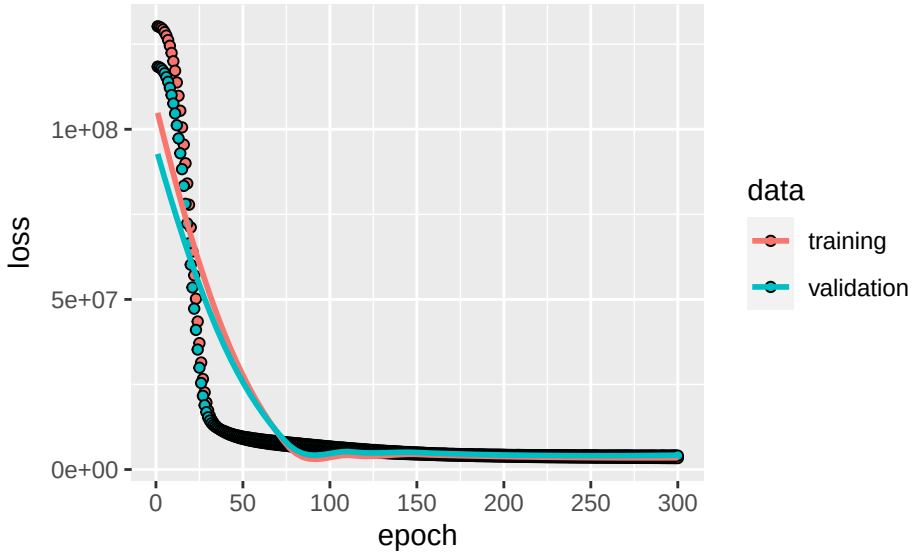
The network was trained using the `keras` library, using the chosen `linear` function for the output layer, and `mse` as the loss function.

```
set.seed(123)
model <- keras_model_sequential() %>%
  layer_dense(units = 64, activation = "relu", input_shape = c(17)) %>%
  layer_dense(units = 64, activation = "relu") %>%
  layer_dense(units = 1, activation = "linear")

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

history <- model %>%
  fit(x.train, y.train, epochs = 300, batch_size = 8, validation_split = 0.2)
```

(ii)

After training for 300 epochs, with 20% of the training data as the validation set, the results are plotted below.

```
plot(history)
```



As can be seen, both the training and validation error falls very quickly the first 30 epochs, and then continue to decrease slowly throughout the training.

(iii)

```
score <- model %>%
  evaluate(x.test, y.test)
```

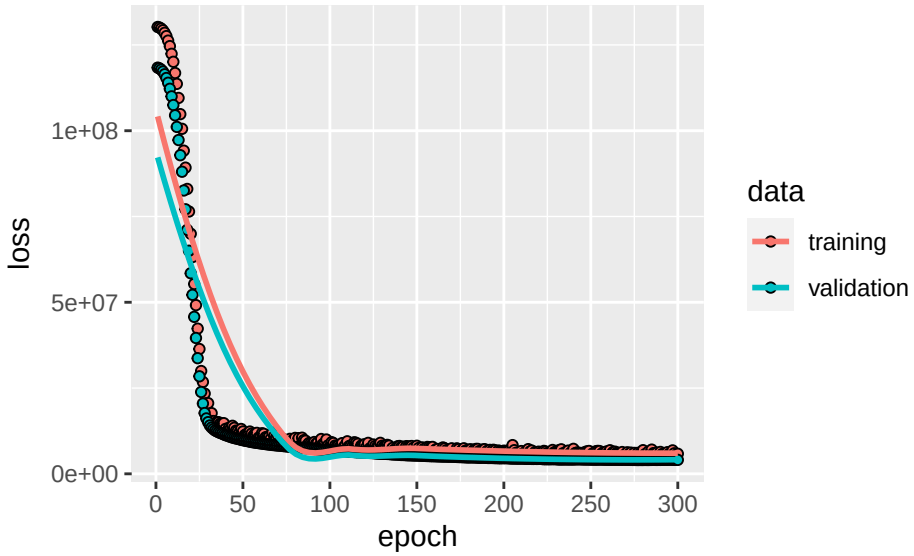
The final MSE after training the model for 300 epochs was 3.7×10^6. Compared to the MSE of the methods from Compulsory 2, this is a relatively good MSE score, and compares to both lasso and forward selection. It is better than polynomial regression and smoothing splines, but both bagging and random forest beat it, scoring 3.3×10^6 and 2.6×10^6 respectively.

d)

Both dropout and weight decay was tried out for improving the performance of the network.

```
set.seed(123)
model_reg <- keras_model_sequential() %>%
  layer_dense(units = 64, activation = "relu", input_shape = c(17),
              kernel_regularizer = regularizer_l2(l = 0.001)) %>%
  layer_dropout(0.3) %>%
  layer_dense(units = 64, activation = "relu") %>%
  layer_dropout(0.3) %>%
  layer_dense(units = 1, activation = "linear")

model_reg %>%
  compile(optimizer = "rmsprop", loss = "mse")

history_reg <- model_reg %>%
  fit(x.train, y.train, epochs = 300, batch_size = 8, validation_split = 0.2)
```

```
plot(history_reg)
```




```
score_reg <- model_reg %>% evaluate(x.test, y.test)
```

After implementing 30% dropout for the two hidden layers, and L2 regularization for the first hidden layer, the final MSE after training was 3.6×10^6, so lower than the unimproved network, but still not better than random forest, for example.

Problem 2

a) Inspecting your data

Table 1: Number of deceased per country.

country	n
France	14
indonesia	2
japan	3
Korea	26

Table 2: Number of deceased per sex.

sex	n
female	14
male	31

Table 3: Number of deceased per country, separated by gender.

country	male	female
France	9	5
japan	3	0
indonesia	1	1
Korea	18	8

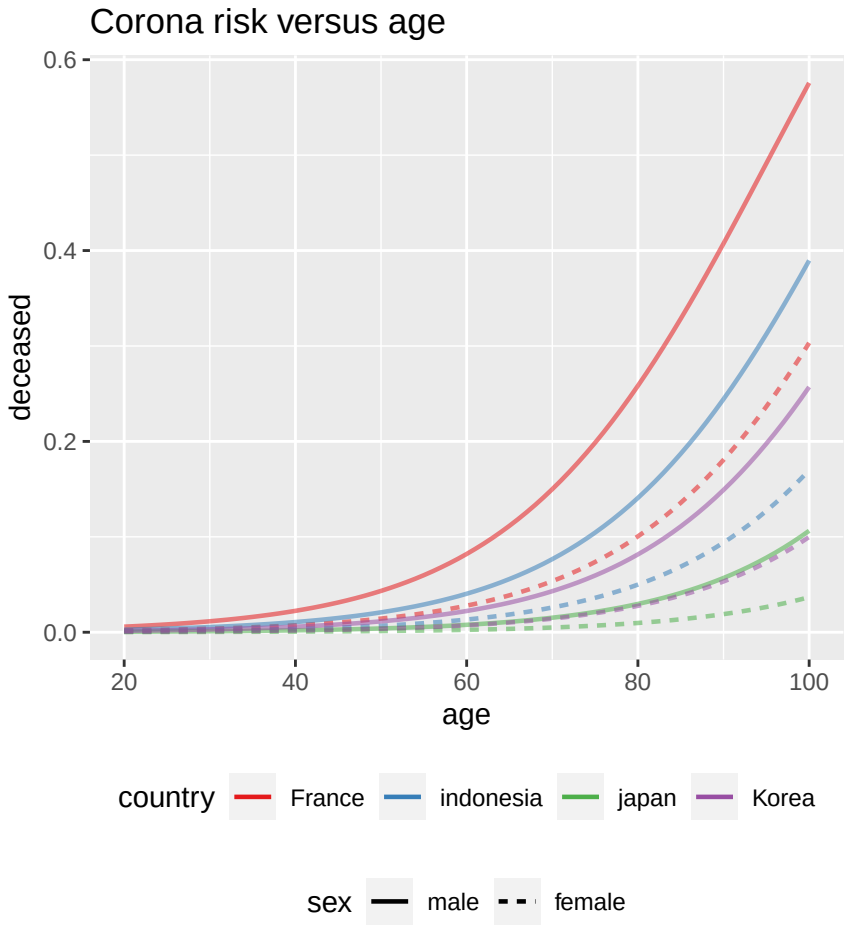
b) Multiple choice

FALSE, FALSE, TRUE, TRUE

c)

Using `ggplot`, the probability to die of coronavirus was plotted for each sex in each country.

```
newdat <- expand.grid(age = seq(20,100,1), sex = c('male','female'), country =
                      unique(d.corona$country))
newdat$pred <- glm_model %>%
  predict(newdata = newdat, type = 'response')

d.corona %>%
  ggplot(aes(x = age, y = deceased, colour = country, lty = sex, alpha = 0.8)) +
  geom_line(data = newdat, aes(y = pred), size = 0.8) +
  labs(title = "Corona risk versus age")  +
  theme(legend.position = "bottom", legend.box = 'vertical') +
  scale_alpha(guide = 'none') +
  scale_color_brewer(palette = "Set1")
```



d)

i)

```
d.corona %>%
  glm(deceased ~ sex, data = ., family = 'binomial') %>%
  summary() %>%
  coefficients()
```

```
##               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
```

Yes, it seems like males have a higher probability of dying from the coronavirus than females.

ii)

```
d.corona %>%
  glm(deceased ~ sex * age, data = ., family = 'binomial') %>%
  summary() %>%
  coefficients()
```

```
##                 Estimate Std. Error    z value     Pr(>|z|)
```

```
## (Intercept) -9.280110711 1.36542855 -6.7964821 1.072046e-11
## sexmale      1.386685990 1.64881103  0.8410218 4.003357e-01
## age          0.073876575 0.01674458  4.4119690 1.024348e-05
## sexmale:age -0.004067455 0.02048535 -0.1985543 8.426114e-01
```

Looking at the coefficients for the `sexmale:age` interaction, this is in fact negative, so this means age is not a greater risk factor for males than females.

iii)

```
d.corona %>%
  glm(deceased ~ country * age, data = ., family = 'binomial') %>%
  summary() %>%
  coefficients()
```

```
##                           Estimate Std. Error    z value     Pr(>|z|)
## (Intercept)          -9.22100450 2.35913551 -3.9086371 9.281828e-05
## countryindonesia      5.29256248 3.14368336  1.6835546 9.226779e-02
## countryjapan          2.91047778 3.21279035  0.9059034 3.649870e-01
## countryKorea          0.73700107 2.53247234  0.2910204 7.710357e-01
## age                   0.09553120 0.02804439  3.4064283 6.581883e-04
## countryindonesia:age -0.08735423 0.04659026 -1.8749461 6.080013e-02
## countryjapan:age     -0.06736478 0.04211855 -1.5994086 1.097298e-01
## countryKorea:age     -0.02660045 0.03034971 -0.8764648 3.807774e-01
```

No, the coefficient for the Korean population interacting with age is negative, so lower than the one for France. However, the p-value is quite high, so it is not a clear trend in either direction.

e) Interpret your model

Covid-19 is a complicated disease that without a doubt has hit the elderly population the hardest. Still, there are many differences in how different countries collect data, and especially in how a "Covid-19-death" is registered. Some countries report it a Covid-related death if the patient had Covid, regardless of what the final cause of death was, while others only count it if the actual cause of death was Covid-19. In addition, the dataset is not the biggest, and the differences between number of tested vs. number of deceased is very different from country to country. For example Korea has about twice the number of deaths, but 15 times as many healthy individuals tested, that didn't die, compared to France.

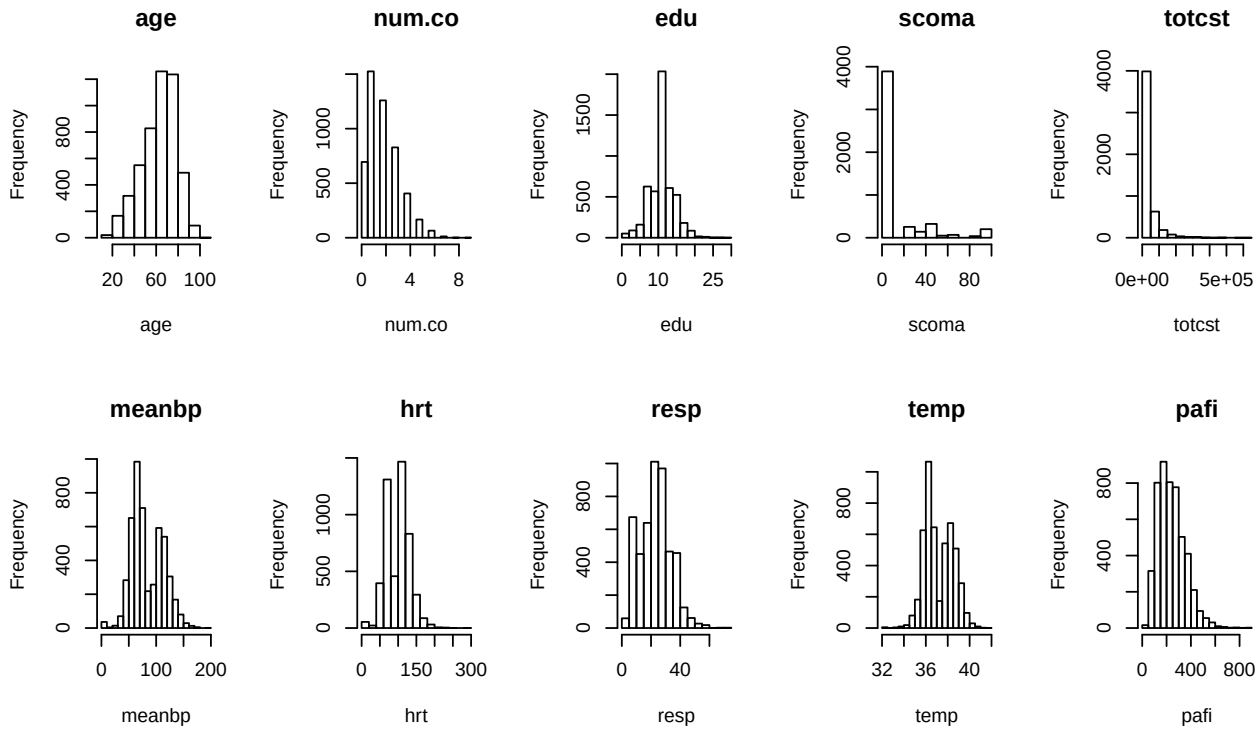
f) Multiple choice

TRUE, TRUE, FALSE, TRUE

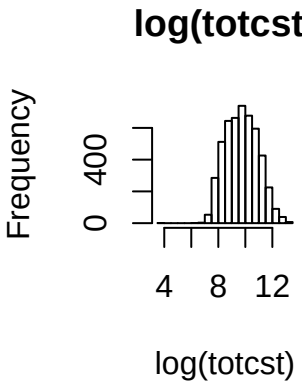
Problem 3

a)

Histograms of all integer and continuous variables are shown below.



A fitting transformation of the `totcst` variable is `log(totcst)`, as shown below.



b)

Fit a multiple linear regression model with the six covariates `age`, `temp`, `edu`, `resp`, `num.co` and `dzgroup` and the (transformed version of the) response `totcst`.

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

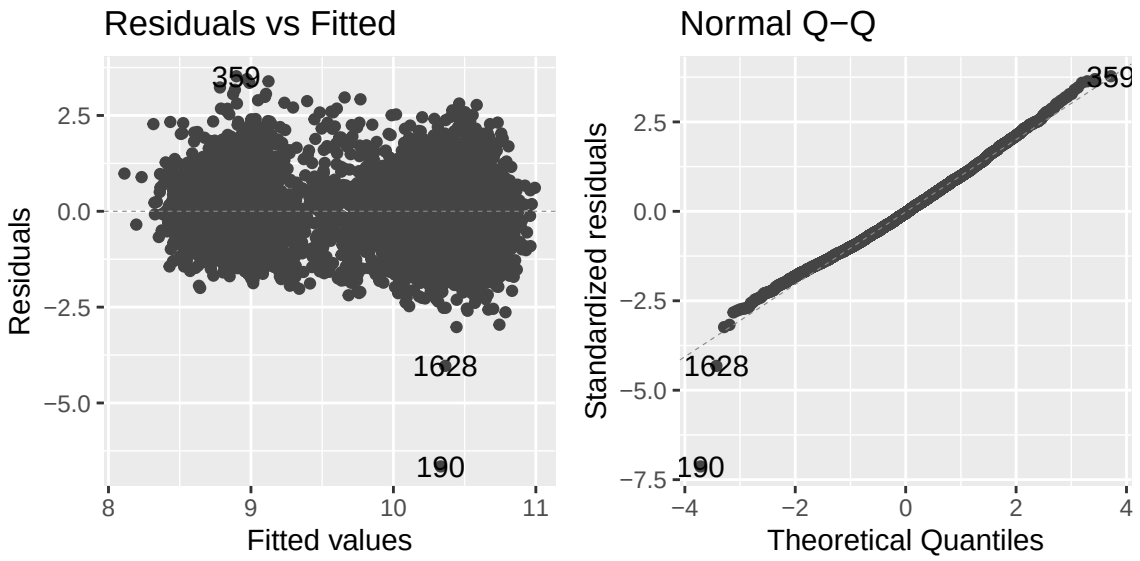
(i)

The change in total costs when a patient's age is increased by 10 years is given by

$$\exp(\beta_{\text{age}} \cdot 10) = \exp(-0.007 \cdot 10) = 0.932$$

(ii)

```
autoplot(mlr_model, which = 1:2, smooth.colour = NA)
```



First, looking at the Tukey-Anscombe plot, it seems like the fitted data follows a linear trend, with a $\mathrm{E}(\varepsilon_i) = 0$. The variance in the residuals also seems to follow the linear trend, with only a few outliers. Looking at the QQ-plot, the assumption of a normal distribution seems to be fulfilled, as nearly all points lie on the straight line of the theoretical normal distribution.

First, it is clear from the left plot that $\mathrm{E}(\varepsilon_i) = 0$ and that the variance of ε_i is within the same area for most of the data points, except a few outliers.

However, for the assumption that all ε_is are independent of each other, there is signs of clustering in the fitted values, where they seem to be in two main groups, with fewer data points in the middle of the left plot. Still, it seems like all of the modelling assumptions in linear regression are fulfilled.

(iii)

To see if the effect of age depends on the disease groups, a null hypothesis test was performed, with the following hypotheses:

$$H_0 : \text{Effect of age does not depend on the disease group}$$
$$H_1 : \text{Effect of age depends on disease group}$$

```
d.support %>%
  lm(log(totcst) ~ temp + edu + resp + num.co + age * dzgroup, data = .) %>%
  anova
```

```
## Analysis of Variance Table
##
## Response: log(totcst)
##               Df Sum Sq Mean Sq  F value    Pr(>F)
## temp           1  238.6  238.59 274.8470 < 2.2e-16 ***
## edu            1  105.2  105.17 121.1507 < 2.2e-16 ***
## resp           1    4.0    3.98   4.5799 0.0323984 *
```

```
## num.co          1  321.4  321.45 370.2935 < 2.2e-16 ***
## age             1  149.1  149.09 171.7433 < 2.2e-16 ***
## dzgroup         7 1844.0  263.43 303.4637 < 2.2e-16 ***
## age:dzgroup     7   24.5    3.51   4.0387 0.0002019 ***
## Residuals    4940 4288.3    0.87
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Using the `anova` function, the `age:dzgroup`, it is clear the interaction has a significant effect, with a p-value of 2.02×10^{-4}. This means that the null hypothesis is forfeited, and H_1 is correct.

c)

The training and test set was created, and made into a data matrix, to use the `glmnet` package.

```
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, ]
x.train <- model.matrix(log(totcst) ~ ., data = d.support.train)[,-1]
y.train <- log(d.support.train$totcst)
x.test <- model.matrix(log(totcst) ~ ., data = d.support.test)[,-1]
y.test <- log(d.support.test$totcst)
```

Cross-validation was run, to find the largest λ within 1 standard error of the smallest λ.

```
ridge_model <- cv.glmnet(x.train, y.train, alpha = 0)
best_lambda <- ridge_model$lambda.1se
```

The value of λ was 0.142, which was then used to find the MSE of the ridge regression.

```
ridge_pred <- ridge_model %>%
  predict(s = best_lambda, newx = x.test)
ridge_MSE <- mean((ridge_pred - y.test)^2)
```

The final calculated MSE is 0.874.

d)

(i)

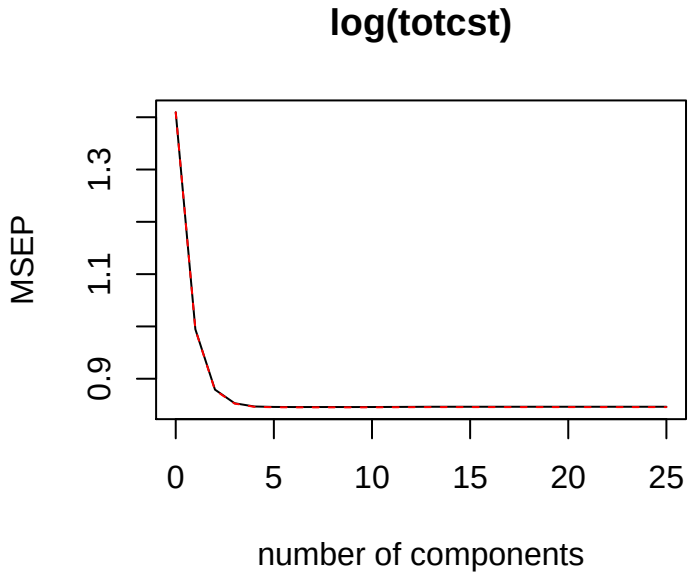
PLS regression was run, using cross-validation.

```
library(pls)
plsr_model <- d.support.train %>%
  plsr(log(totcst) ~ ., data = ., scale = TRUE, validation = "CV")
```

(ii)

Then the validation plot was produced, to see the optimal number of principal components.

```
validationplot(plsr_model, val.type = "MSEP")
```



The number of principal components was chosen to be 4, as this is where the curve clearly starts flattening out, and the decrease in MSE if one were to use more components is not that big. In addition, the model is simpler if we only use 4 components, instead of a higher number.

(ii)

```
plsr_predictions <- plsr_model %>%
  predict(d.support.test, ncomp = 4)
plsr_MSE <- mean((plsr_predictions - log(d.support.test$totcst))^2)
```

The final calculated MSE for PLS was 0.864. This is just slightly lower than the ridgre regression.

e)

(i)

```
gam_model <- d.support.train %>%
  gam(log(totcst) ~ s(age, 2) + s(temp, 6) + edu + s(resp, 7) + s(num.co,6)
                + dzgroup, data = .)
gam_pred <- gam_model %>%
  predict(newdata = d.support.test)
gam_MSE <- mean((gam_pred - y.test)^2)
```

The GAM model was fitted using different combinations of smoothing splines for the different variables, and the MSE was 0.86. This is not that impressive, but is comparable to PLS.

(i)

```
randomForest <- d.support.train %>%
  randomForest(log(totcst) ~., data = ., mtry = ncol(d.support.train)/3,
               ntree = 500, importance = TRUE)
randForest_pred <- randomForest %>%
```

```
  predict(newdata = d.support.test)
randForest_MSE <- mean((randForest_pred - y.test)^2)
```

Random forest was used because it generally performs well. The MSE for the random forest was 0.824, which is by far the best MSE compared to all the other methods tested.

Problem 4 (Mixed questions)

a)

The basis functions for the cubic regression spline model is

$$b_1 = X, \quad b_2 = X^2, \quad b_3 = X^3,$$
$$b_4 = (X-1)^3_+, \quad b_5 = (X-2)^3_+,$$

and the design matrix is given below.

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 & (x_1-1)^3_+ & (x_1-2)^3_+ \\ 1 & x_2 & x_2 & x_2 & (x_2-1)^3_+ & (x_2-2)^3_+ \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n & x_n & (x_n-1)^3_+ & (x_n-2)^3_+ \end{bmatrix}$$

b) Multiple choice

TRUE, TRUE, TRUE, FALSE

c) Multiple choice

FALSE, TRUE, TRUE, FALSE

Problem 5 (Multiple and single choice questions)

a) Multiple choice

TRUE, TRUE, FALSE, TRUE

b) Multiple choice

FALSE, TRUE, FALSE, TRUE

c) Single choice

(iv)

d) Single choice

(ii)

e) Single choice

(iii)

f) Multiple choice

TRUE, TRUE, FALSE, TRUE

g) Multiple choice

TRUE, FALSE, TRUE, TRUE