Question 14.1
The breast cancer data set breast-cancer-wisconsin.data.txt from http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/ (description at http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29) has missing values.

1. Use the mean/mode imputation method to impute values for the missing data.

The code for the mean imputation approach can be found in appendix 14.1.1.1. Here, I noticed that column V7 was the only one that had missing values. After that, I calculated how many rows had missing values as a fraction of the total number of rows. The answer was 0.02288984, which meant that less than 5% of the data would need to be imputed (so it was safe to use imputation). I then proceeded to replace the "?" values with NA ones instead. Following this, I performed mean imputation and replaced the NA values with the calculated mean, which was 3.544656. I then converted the V7 column to integer values like the rest of the columns.

The code for the mode imputation approach can be found in appendix 14.1.1.2. Here, I converted the "?" values in V7 to NA like before. Then, I found all unique, non-NA values in V7. Following this, I tabulated the number of times each unique value appeared in the V7 column. I noticed that 1 was the most frequent occurrence and so set that as the mode. As a result, I replaced all NA values with 1 and then converted the column to integer values like the rest of the columns.

2. Use regression to impute values for the missing data.

The code for this approach can be found in appendix 14.1.2. Here, I first prepared my data by temporarily removing the rows with the missing values as well as the identification column (V1) and the response column (V11). With my new data, I then performed a linear regression with the following call: lm(V7~V2+V3+V4+V5+V6+V8+V9+V10, data=reg_imp_data). When I looked at the summary for the data, I noticed not all factors were significant. Therefore, I used stepwise regression to select the best variables. My resulting model was: lm(V7~V2+V4+V5+V8, data = reg_imp_data). I then performed 10-fold cross-validation to check the quality of my model and calculated the R-squared value: 0.6064699. I then used my linear regression to predict the missing values:

```
24        41       140       146       159       165
5.4585352 7.9816106 0.9872832 1.6218560 0.9807851 2.2157441
      236       250       276       293       295       298
2.7152652 1.7634059 2.0741942 6.0866099 0.9872832 2.5265324
      316       322       412       618
5.2438347 1.7634059 0.9872832 0.6634986
```

I then replaced the missing values with their corresponding calculated values after rounding them and converting them to integers. I also checked to see if any of the newfound values were out of bounds (<1 or >10) and none were so I left them alone.

3. Use regression with perturbation to impute values for the missing data.

The code for this approach can be found in appendix 14.1.3. Most of the steps were similar to the normal regression imputation method shown above. However, using the predicted missing values that came from the linear regression (as show in question 14.1.2), I calculated perturbation values with a normal distribution:

```
[1]  5.37954914  6.75067787 -3.66534310 -0.57170564  1.63141669
 [6]  1.34343335  1.77642664 -1.91264829  1.51283458  3.18857649
[11]  1.47167408  1.14374698  9.17948313 -0.46607695  3.78925154
[16] -0.05151801
```

I then replaced the missing data values with these, rounded them, and converted the column to integer values like the other columns. However, since some of the values were out of bounds (<1 and >10), I needed to correct them. For this, I implemented code that would set any value below 1 to 1 and any value above 10 to 10.

4. Compare all three imputation approaches (*this is my own question since I don't want to do the optional one, and I am using this for extended investigation*).

The code for this approach can be found in appendix 14.1.4. I wanted to investigate each imputation approach to see which was the best. For this, I first created training (70%) and testing (30%) data sets. For each of the training and testing data sets, I then changed the missing values from "?" to NA. Following these procedures, I began by first performing the mean imputation on the training and testing data sets. I then performed linear regression imputations on the testing and training data sets using the *mice* library. I also went ahead and performed the random perturbation regression approach on both data sets by using the *missForest* library. For all imputation approaches, I made sure to set the bounds (i.e., all values below 1 were set to 1 and all values above 10 were set to 10). I also fitted all of the imputed data sets to a linear regression model and obtained the mean squared error (MSE) and the R-squared values associated with each approach:

```
Mean imputation error: 0.1448082
Mean imputation R-squared: 0.833089
Linear regression imputation error: 0.1398115
Linear regression imputation R-squared: 0.8343196
Linear regression with perturbation imputation error: 0.141096
Linear regression with perturbation imputation R-squared: 0.8296648
```

As can be seen, the linear regression imputation seems to have performed the best with the lowest MSE and the highest R-squared. This is in line with what we learnt in lecture, especially seeing how the error in the linear regression with perturbation approach is a bit higher than just in the linear regression approach alone. However, in this situation, all R-squared values and MSE values are relatively close, so I can't quite say if there's a significant difference in any one approach over the others.

Question 15.1
Describe a situation or problem from your job, everyday life, current events, etc., for which optimization would be appropriate. What data would you need?

In the past, I ran a business. In running the business, there was a situation where I needed to change my production process to maximize profit. In such a situation, I could have used an

optimization model. For instance, the variables could have been cost, production time, profit margin, availability, cost of resources, etc. The constraints would have been things like the maximum number of employees we had for work, the amount of raw material available, etc. Our objective function could have been one that aimed to maximize profit. Taking all of these different aspects into account, we could have generated an optimization model to determine the production plan we should have chosen as a company.

Appendix

Question 14.1.1.1

```
#This is the mean imputation
#Load Data
data <- read.table("C:\\Users\\User\\OneDrive\\Desktop\\Data 14.1\\breast-can
cer-wisconsin.data.txt", stringsAsFactors = F, header = F, sep=",")

#Upon analysis of the table, only V7 has missing values
missing_val <- data[which(data$V7 == "?"), ]
missing_val
```

```
##          V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11
## 24  1057013  8  4  5  1  2  ?  7  3   1   4
## 41  1096800  6  6  6  9  6  ?  7  8   1   2
## 140 1183246  1  1  1  1  1  ?  2  1   1   2
## 146 1184840  1  1  3  1  2  ?  2  1   1   2
## 159 1193683  1  1  2  1  3  ?  1  1   1   2
## 165 1197510  5  1  1  1  2  ?  3  1   1   2
## 236 1241232  3  1  4  1  2  ?  3  1   1   2
## 250  169356  3  1  1  1  2  ?  3  1   1   2
## 276  432809  3  1  3  1  2  ?  2  1   1   2
## 293  563649  8  8  8  1  2  ?  6 10   1   4
## 295  606140  1  1  1  1  2  ?  2  1   1   2
## 298   61634  5  4  3  1  2  ?  2  3   1   2
## 316  704168  4  6  5  6  7  ?  4  9   1   2
## 322  733639  3  1  1  1  2  ?  3  1   1   2
## 412 1238464  1  1  1  1  1  ?  2  1   1   2
## 618 1057067  1  1  1  1  1  ?  1  1   1   2
```

```
#Checking the 5% rule for imputation
five_rule <- nrow(data[which(data$V7 == "?"), ])/nrow(data)
five_rule
```

```
## [1] 0.02288984
```

```
#Replace ? values with NA
mean_imp_data <- data
mean_imp_data["V7"][mean_imp_data["V7"] == "?"] <- NA

#Convert column to numeric
mean_imp_data$V7 <- as.numeric(mean_imp_data$V7)

#Perform mean imputation
mean_imp_data$V7[is.na(mean_imp_data$V7)] <- mean(mean_imp_data$V7, na.rm = T
)
mean_imp_data$V7[is.na(mean_imp_data$V7)] <- mean(mean_imp_data$V7, na.rm = T
)
```

```
#Convert column to integer
mean_imp_data$V7 <- as.integer(mean_imp_data$V7)
```

Question 14.1.1.2

```
#This is mode imputation
#Load Data
data <- read.table("C:\\Users\\User\\OneDrive\\Desktop\\Data 14.1\\breast-can
cer-wisconsin.data.txt", stringsAsFactors = F, header = F, sep=",")

#Replace ? values with NA
mode_imp_data <- data
mode_imp_data["V7"][mode_imp_data["V7"] == "?"] <- NA

#Get values in df
vals <- unique(mode_imp_data$V7[!is.na(mode_imp_data$V7)])
mode <- vals[which.max(tabulate(match(mode_imp_data$V7, vals)))]

#Replace NA values with the mode
mode_imp_data$V7[is.na(mode_imp_data$V7)] <- mode

#Convert column to integer
mode_imp_data$V7 <- as.integer(mode_imp_data$V7)
```

Question 14.1.2

```
#This is regression imputation
#Load Libraries
library("DAAG")

#Load Data
data <- read.table("C:\\Users\\User\\OneDrive\\Desktop\\Data 14.1\\breast-can
cer-wisconsin.data.txt", stringsAsFactors = F, header = F, sep=",")

#Get the missing indices
missing <- which(data$V7 == "?", arr.ind = T)
missing
```

```
##  [1]  24  41 140 146 159 165 236 250 276 293 295 298 316 322 412 618
```

```
#Prepare the needed data
reg_imp_data <- data[-missing, 2:10]
reg_imp_data$V7 <- as.integer(reg_imp_data$V7)

#Create a linear model
reg_m <- lm(V7~V2+V3+V4+V5+V6+V8+V9+V10, data=reg_imp_data)
summary(reg_m)
```

```
##
## Call:
## lm(formula = V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 + V10, data = reg_imp_d
```

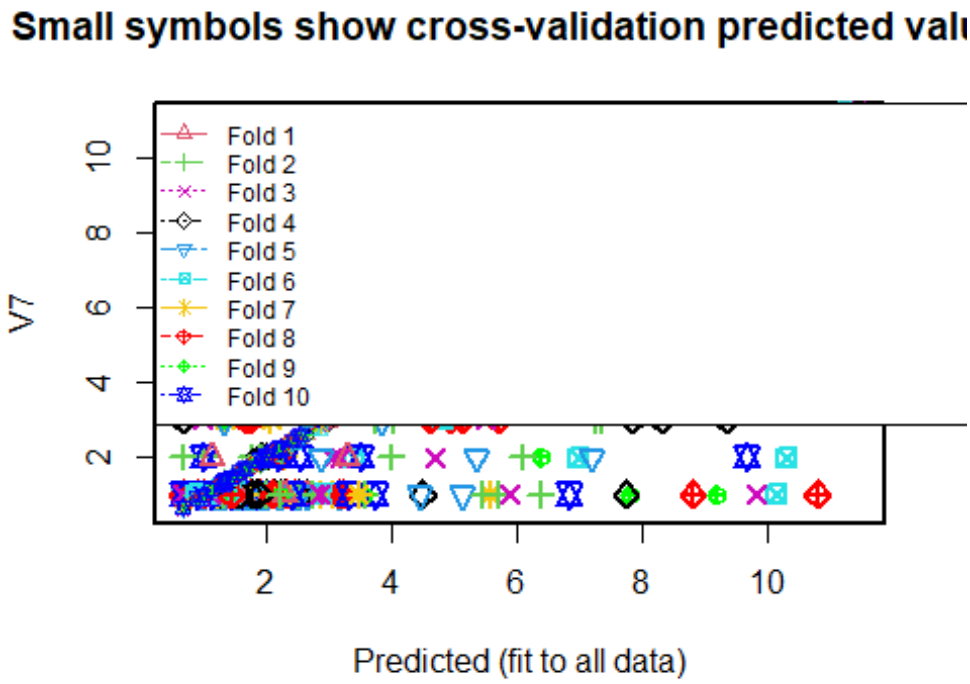
```
ata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7316 -0.9426 -0.3002  0.6725  8.6998
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.616652   0.194975  -3.163  0.00163 **
## V2           0.230156   0.041691   5.521 4.83e-08 ***
## V3          -0.067980   0.076170  -0.892  0.37246
## V4           0.340442   0.073420   4.637 4.25e-06 ***
## V5           0.339705   0.045919   7.398 4.13e-13 ***
## V6           0.090392   0.062541   1.445  0.14883
## V8           0.320577   0.059047   5.429 7.91e-08 ***
## V9           0.007293   0.044486   0.164  0.86983
## V10         -0.075230   0.059331  -1.268  0.20524
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.274 on 674 degrees of freedom
## Multiple R-squared:  0.615,  Adjusted R-squared:  0.6104
## F-statistic: 134.6 on 8 and 674 DF,  p-value: < 2.2e-16

#Need to select only the best variables so we can use stepwise regression
step(reg_m)

## Start:  AIC=1131.43
## V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 + V10
##
##        Df Sum of Sq    RSS    AIC
## - V9    1     0.139 3486.8 1129.5
## - V3    1     4.120 3490.8 1130.2
## - V10   1     8.317 3495.0 1131.0
## <none>              3486.6 1131.4
## - V6    1    10.806 3497.5 1131.5
## - V4    1   111.227 3597.9 1150.9
## - V8    1   152.482 3639.1 1158.7
## - V2    1   157.657 3644.3 1159.6
## - V5    1   283.119 3769.8 1182.8
##
## Step:  AIC=1129.45
## V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V10
##
##        Df Sum of Sq    RSS    AIC
## - V3    1     4.028 3490.8 1128.2
## - V10   1     8.179 3495.0 1129.0
## <none>              3486.8 1129.5
## - V6    1    11.211 3498.0 1129.7
## - V4    1   114.768 3601.6 1149.6
```

```
## - V2    1   158.696 3645.5 1157.8
## - V8    1   160.776 3647.6 1158.2
## - V5    1   285.902 3772.7 1181.3
##
## Step:  AIC=1128.24
## V7 ~ V2 + V4 + V5 + V6 + V8 + V10
##
##        Df Sum of Sq    RSS    AIC
## - V6    1     8.606 3499.4 1127.9
## - V10   1     8.889 3499.7 1128.0
## <none>              3490.8 1128.2
## - V4    1   153.078 3643.9 1155.6
## - V2    1   155.308 3646.1 1156.0
## - V8    1   157.123 3647.9 1156.3
## - V5    1   282.133 3772.9 1179.3
##
## Step:  AIC=1127.92
## V7 ~ V2 + V4 + V5 + V8 + V10
##
##        Df Sum of Sq    RSS    AIC
## - V10   1     5.562 3505.0 1127.0
## <none>              3499.4 1127.9
## - V2    1   159.594 3659.0 1156.4
## - V8    1   169.954 3669.4 1158.3
## - V4    1   206.785 3706.2 1165.1
## - V5    1   295.807 3795.2 1181.3
##
## Step:  AIC=1127.01
## V7 ~ V2 + V4 + V5 + V8
##
##        Df Sum of Sq    RSS    AIC
## <none>              3505.0 1127.0
## - V2    1    155.70 3660.7 1154.7
## - V8    1    172.42 3677.4 1157.8
## - V4    1    201.22 3706.2 1163.1
## - V5    1    290.68 3795.7 1179.4

##
## Call:
## lm(formula = V7 ~ V2 + V4 + V5 + V8, data = reg_imp_data)
##
## Coefficients:
## (Intercept)           V2           V4           V5           V8
##     -0.5360       0.2262       0.3173       0.3323       0.3238

#Generate a better model now
reg_ms <- lm(V7~V2+V4+V5+V8, data = reg_imp_data)
summary(reg_ms)
```

```
##
## Call:
## lm(formula = V7 ~ V2 + V4 + V5 + V8, data = reg_imp_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8115 -0.9531 -0.3111  0.6678  8.6889
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.53601    0.17514  -3.060   0.0023 **
## V2           0.22617    0.04121   5.488 5.75e-08 ***
## V4           0.31729    0.05086   6.239 7.76e-10 ***
## V5           0.33227    0.04431   7.499 2.03e-13 ***
## V8           0.32378    0.05606   5.775 1.17e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.274 on 678 degrees of freedom
## Multiple R-squared:  0.6129, Adjusted R-squared:  0.6107
## F-statistic: 268.4 on 4 and 678 DF,  p-value: < 2.2e-16
```

```
#10-fold CV to check quality of model
reg_mcv <- cv.lm(reg_imp_data, reg_ms, m=10)
```

```
## Warning in cv.lm(reg_imp_data, reg_ms, m = 10):
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
##
## fold 1
## Observations in test set: 68
##                      4         12        22         36        51         5
6
## Predicted    4.6629181  1.2134523 6.5754051  1.2134523  5.017641  5.597553
4
## cvpred       4.5695959  1.2335534 6.5065661  1.2335534  4.902679  5.505644
1
## V7           4.0000000  1.0000000 7.0000000  1.0000000  3.000000  5.000000
0
## CV residual -0.5695959 -0.2335534 0.4934339 -0.2335534 -1.902679 -0.505644
1
##                     59         67        82       87         89         91
## Predicted    3.497886  1.9895750  1.998056 4.670420  1.9895750  1.3110677
## cvpred       3.480637  1.9994761  2.005937 4.626138  1.9994761  1.3400727
## V7          10.000000  1.0000000  1.000000 8.000000  1.0000000  1.0000000
## CV residual  6.519363 -0.9994761 -1.005937 3.373862 -0.9994761 -0.3400727
##                    109        118       123        125        129        130
## Predicted    0.9872832  7.544252  7.182002  7.4146942  4.512626 0.6634986
## cvpred       1.0137523  7.517230  7.042942  7.3693667  4.441823 0.6874319
## V7           1.0000000 10.000000 10.000000  7.0000000 10.000000 1.0000000
## CV residual -0.0137523  2.482770  2.957058 -0.3693667  5.558177 0.3125681
##                    131        169        194        200        215        221
## Predicted    2.526532  1.763406  1.3110677  1.4396214 11.459052  1.6433336
## cvpred       2.501676  1.779675  1.3400727  1.4533546 11.336796  1.6728542
## V7           1.000000  1.000000  1.0000000  1.0000000 10.000000  1.0000000
```

```
## CV residual -1.501676 -0.779675 -0.3400727 -0.4533546 -1.336796 -0.6728542
##                  231      255       259        302       327       354
## Predicted   6.8637366 5.975998  1.763406  1.3110677  3.994159  7.706992
## cvpred      6.7886636 5.837719  1.779675  1.3400727  3.970924  7.620465
## V7          7.0000000 8.000000  1.000000  1.0000000 10.000000 10.000000
## CV residual 0.2113364 2.162281 -0.779675 -0.3400727  6.029076  2.379535
##                  358        364        377        378        408        4
18
## Predicted    9.038563 2.94991561  0.9872832  0.9872832  0.9872832  0.98728
32
## cvpred       8.919888 2.91901614  1.0137523  1.0137523  1.0137523  1.01375
23
## V7          10.000000 3.00000000  1.0000000  1.0000000  1.0000000  1.00000
00
## CV residual  1.080112 0.08098386 -0.0137523 -0.0137523 -0.0137523 -0.01375
23
##                  429       439        455       459        473        47
7
## Predicted    0.9872832  2.641111 0.88966773  1.885461  1.7943441  1.659292
3
## cvpred       1.0137523  2.621118 0.90723301  1.870996  1.7864375  1.651194
9
## V7           1.0000000  1.000000 1.00000000  1.000000  1.0000000  1.000000
0
## CV residual -0.0137523 -1.621118 0.09276699 -0.870996 -0.7864375 -0.651194
9
##                  492        498        500        502       511        524
## Predicted    7.045971  1.3420059  1.6657904  1.6657904 0.6634986  7.892476
## cvpred       6.957781  1.3468353  1.6731557  1.6731557 0.6874319  7.828352
## V7          10.000000  1.0000000  1.0000000  1.0000000 1.0000000 10.000000
## CV residual  3.042219 -0.3468353 -0.6731557 -0.6731557 0.3125681  2.171648
##                  528       536       537        543        551        56
3
## Predicted    1.9895750  2.277906  2.215744  1.5681750  1.4396214  1.311067
7
## cvpred       1.9994761  2.281574  2.219277  1.5666364  1.4533546  1.340072
7
## V7           1.0000000  1.000000  1.000000  1.0000000  1.0000000  1.000000
0
## CV residual -0.9994761 -1.281574 -1.219277 -0.5666364 -0.4533546 -0.340072
7
##                  566       567       616        617      626       628
## Predicted   10.3282067  2.080692  2.300363  1.4396214 1.750410 0.8896677
## cvpred      10.2377906  2.084035  2.281875  1.4533546 1.735753 0.9072330
## V7          10.0000000  1.000000  1.000000  1.0000000 4.000000 5.0000000
## CV residual -0.2377906 -1.084035 -1.281875 -0.4533546 2.264247 4.0927670
##                  636       638       641        642        663        673
## Predicted    2.067696  3.282182  2.006538  1.4396214  1.6218560  1.5372368
## cvpred       2.040113  3.251798  2.012398  1.4533546  1.6224716  1.5598738
## V7           1.000000  2.000000  1.000000  1.0000000  1.0000000  1.0000000
```

```
## CV residual -1.040113 -1.251798 -1.012398 -0.4533546 -0.6224716 -0.5598738
##                   676       695
## Predicted    2.293865 1.1158368
## cvpred       2.259914 1.1270341
## V7           1.000000 2.0000000
## CV residual -1.259914 0.8729659
##
## Sum of squares = 217.84    Mean square = 3.2    n = 68
##
## fold 2
## Observations in test set: 69
##                     3         17       26       40        62        66
## Predicted   1.7634059  1.6657904 3.847115 4.131480 0.9872832  3.987660
## cvpred      1.7149697  1.6065034 3.843866 4.116213 1.0424053  3.679787
## V7          2.0000000  1.0000000 7.000000 7.000000 2.0000000  2.000000
## CV residual 0.2850303 -0.6065034 3.156134 2.883787 0.9575947 -1.679787
##                    73        77       79        83         92      112
## Predicted    3.573045  1.939142 1.763406  2.215744  1.4481027 4.510642
## cvpred       3.578629  2.098673 1.714970  2.091035  1.4715226 4.410502
## V7           1.000000  1.000000 3.000000  1.000000  1.0000000 9.000000
## CV residual -2.578629 -1.098673 1.285030 -1.091035 -0.4715226 4.589498
##                   120        139       150       152       178       182
## Predicted   1.7634059  1.9830769  7.089906  4.267511  6.383423 0.6634986
## cvpred      1.7149697  1.9585926  7.002317  4.116366  6.551937 0.7459064
## V7          2.0000000  1.0000000 10.000000 10.000000  1.000000 1.0000000
## CV residual 0.2850303 -0.9585926  2.997683  5.883634 -5.551937 0.2540936
##                   204        208        211       222       233       235
## Predicted    2.215744  1.3110677 10.3591449  6.905688  5.46900  2.080692
## cvpred       2.091035  1.3389043 10.4523881  6.565561  5.41118  2.067059
## V7           1.000000  1.0000000 10.0000000 10.000000  1.00000  1.000000
## CV residual -1.091035 -0.3389043 -0.4523881  3.434439 -4.41118 -1.067059
##                   237       242      244       265      274        282
## Predicted    6.223645  2.427938 1.958637  7.257904 3.619941  1.8695027
## cvpred       6.141873  2.414071 1.931902  7.262324 3.523368  1.8764878
## V7          10.000000  1.000000 5.000000  3.000000 4.000000  1.0000000
## CV residual  3.858127 -1.414071 3.068098 -4.262324 0.476632 -0.8764878
##                   287      289       291       299       306       312
## Predicted    9.5163449 3.724055 0.6634986  2.246682  5.295246 0.6634986
## cvpred       9.6424577 3.576244 0.7459064  2.062135  5.136119 0.7459064
## V7          10.0000000 5.000000 1.0000000  1.000000 10.000000 1.0000000
## CV residual  0.3575423 1.423756 0.2540936 -1.062135  4.863881 0.2540936
##                   317         328       349       353       366        395
## Predicted    4.140940  0.98728319  5.705658  4.020868  1.213452  1.6218560
## cvpred       4.145442  1.04240531  5.917195  4.118927  1.230438  1.7465836
## V7          10.000000  1.00000000  1.000000  3.000000  1.000000  1.0000000
## CV residual  5.854558 -0.04240531 -4.917195 -1.118927 -0.230438 -0.7465836
##                   398      404        416       428       444        475
## Predicted    1.3420059 1.115837  3.7420218  6.095091 0.6634986  1.5681750
## cvpred       1.3100045 1.121972  3.8148131  6.062306 0.7459064  1.4980372
## V7           1.0000000 4.000000  3.0000000  2.000000 2.0000000  1.0000000
```

```
## CV residual -0.3100045 2.878028 -0.8148131 -4.062306 1.2540936 -0.4980372
##                    495        519        525       531       541       550
## Predicted    5.1996390  1.7653891  1.4396214  5.255827 1.8919595  6.591389
## cvpred       5.2925829  1.8236118  1.4184707  5.111967 1.7945361  6.454580
## V7           5.0000000  1.0000000  1.0000000 10.000000 2.0000000  5.000000
## CV residual -0.2925829 -0.8236118 -0.4184707  4.888033 0.2054639 -1.454580
##                    553       557         574        577       593       600
## Predicted    2.736743  2.541512  0.98728319  1.8919595  5.951297  2.520034
## cvpred       2.713109  2.496176  1.04240531  1.7945361  5.782168  2.554305
## V7           1.000000  1.000000  1.00000000  1.0000000 10.000000  1.000000
## CV residual -1.713109 -1.496176 -0.04240531 -0.7945361  4.217832 -1.554305
##                    607       611       624       633        662       666
## Predicted    1.6742718  8.266694 0.6634986 0.6634986  1.9895750 0.6634986
## cvpred       1.6595553  8.067330 0.7459064 0.7459064  1.9030024 0.7459064
## V7           1.0000000 10.000000 1.0000000 1.0000000  1.0000000 1.0000000
## CV residual -0.6595553  1.932670 0.2540936 0.2540936 -0.9030024 0.2540936
##                    683       691       697
## Predicted    2.215744  1.328030  7.354776
## cvpred       2.091035  1.445008  7.441434
## V7           1.000000  1.000000  3.000000
## CV residual -1.091035 -0.445008 -4.441434
##
## Sum of squares = 403.36    Mean square = 5.85    n = 69
##
## fold 3
## Observations in test set: 69
##                      7         10         11       15         23       50
## Predicted    1.311068  1.6657904  1.3110677 7.801359  1.4396214 4.904067
## cvpred       1.257795  1.6868818  1.2577948 7.736532  1.4286388 4.816286
## V7          10.000000  1.0000000  1.0000000 9.000000  1.0000000 8.000000
## CV residual  8.742205 -0.6868818 -0.2577948 1.263468 -0.4286388 3.183714
##                    63         65        71        84       101       108
## Predicted   5.975998 0.98728319  2.526532  3.058544 4.6157352  7.170035
## cvpred      5.862017 0.91215283  2.509187  3.156849 4.7101056  6.774924
## V7          8.000000 1.00000000  1.000000  2.000000 5.0000000 10.000000
## CV residual 2.137983 0.08784717 -1.509187 -1.156849 0.2898944  3.225076
##                    124       147       149        151       162       164
## Predicted    4.132459 3.688602  3.075507  1.3110677  1.989575 0.9957645
## cvpred       4.110175 3.564894  3.099260  1.2577948  2.032524 0.8833584
## V7          10.000000 8.000000  1.000000  1.0000000  1.000000 3.0000000
## CV residual  5.889825 4.435106 -2.099260 -0.2577948 -1.032524 2.1166416
##                    167       175       184       201       219       241
## Predicted    6.445324  6.131548  8.057748  7.648341  7.958150  3.191064
## cvpred       6.200885  6.181088  7.865484  7.665883  7.870491  3.142882
## V7          10.000000 10.000000 10.000000 10.000000  4.000000  2.000000
## CV residual  3.799115  3.818912  2.134516  2.334117 -3.870491 -1.142882
##                    260        270       296       330       334       337
## Predicted   4.119463  1.3110677  7.740462  7.224957  5.787290  6.036920
## cvpred      3.982953  1.2577948  7.583453  7.325247  5.630801  5.895067
## V7          8.000000  1.0000000 10.000000 10.000000 10.000000 10.000000
```

```
## CV residual 4.017047 -0.2577948  2.416547  2.674753  4.369199  4.104933
##                   346       357       359       360       367       373
## Predicted   0.6634986 2.8503169  5.151713 5.612533  9.5830222  1.9830769
## cvpred      0.5665108 2.8548287  5.172957 5.660648  9.3214845  1.9689128
## V7          1.0000000 3.0000000  4.000000 7.000000 10.0000000  1.0000000
## CV residual 0.4334892 0.1451713 -1.172957 1.339352  0.6785155 -0.9689128
##                   382       384       389       391       437       446
## Predicted    6.937630 0.8896677  1.2134523  1.3195491  5.878383 0.8896677
## cvpred       7.112849 0.8247538  1.1703958  1.2290004  5.774618 0.8247538
## V7          10.000000 1.0000000  1.0000000  1.0000000  1.000000 1.0000000
## CV residual  2.887151 0.1752462 -0.1703958 -0.2290004 -4.774618 0.1752462
##                   449       461       464       472       479       491
## Predicted   0.6634986  2.232707  1.3420059  2.458876  1.5681750 0.6634986
## cvpred      0.5665108  2.233178  1.3412398  2.491421  1.5994828 0.5665108
## V7          1.0000000  1.000000  1.0000000  1.000000  1.0000000 1.0000000
## CV residual 0.4334892 -1.233178 -0.3412398 -1.491421 -0.5994828 0.4334892
##                   512       521       529       534       546       548
## Predicted    1.8919595 0.6634986  2.761183  1.4396214  1.8919595 0.8896677
## cvpred       1.9451248 0.5665108  2.738635  1.4286388  1.9451248 0.8247538
## V7           1.0000000 1.0000000  1.000000  1.0000000  1.0000000 1.0000000
## CV residual -0.9451248 0.4334892 -1.738635 -0.4286388 -0.9451248 0.1752462
##                   555       561       598       605       606       614
## Predicted    1.1158368  2.215744  2.850317  6.477814  8.1618372  1.2134523
## cvpred       1.0829968  2.290767  2.854829  6.518941  8.1365235  1.1703958
## V7           1.0000000  1.000000  1.000000 10.000000  8.0000000  1.0000000
## CV residual -0.0829968 -1.290767 -1.854829  3.481059 -0.1365235 -0.1703958
##                   622       634       637       639       659       675
## Predicted    4.712371  5.490477  9.842661  1.3420059  8.177821 0.98728319
## cvpred       4.783693  5.553415  9.911594  1.3412398  8.065123 0.91215283
## V7           2.000000  3.000000  1.000000  1.0000000 10.000000 1.00000000
## CV residual -2.783693 -2.553415 -8.911594 -0.3412398  1.934877 0.08784717
##                   681       690       693
## Predicted   11.459052 0.6634986  1.1158368
## cvpred      11.391382 0.5665108  1.0829968
## V7          10.000000 1.0000000  1.0000000
## CV residual -1.391382 0.4334892 -0.0829968
##
## Sum of squares = 455.21    Mean square = 6.6    n = 69
##
## fold 4
## Observations in test set: 69
##                     5          8        20         34       39         48
## Predicted    2.654107  1.8545232  2.441913  1.8695027  6.47330 0.98728319
## cvpred       2.611545  1.8627682  2.406653  1.8488227  6.44151 0.97359493
## V7           1.000000  1.0000000  1.000000  1.0000000 10.00000 1.00000000
## CV residual -1.611545 -0.8627682 -1.406653 -0.8488227  3.55849 0.02640507
##                   52        58        76       103      110       116
## Predicted   3.8471146  4.493680 1.9521387  2.306861 5.685708 0.6634986
## cvpred      3.8271412  4.521483 1.9752741  2.302952 5.714108 0.6409970
## V7          4.0000000  1.000000 2.0000000  1.000000 9.000000 5.0000000
```

```
## CV residual 0.1728588 -3.521483 0.0247259 -1.302952 3.285892 4.3590030
##                   119        122       133        163        181        18
8
## Predicted   0.6634986 1.98957497  7.427142  1.7634059  1.3110677  9.218552
8
## cvpred      0.6409970 1.96646899  7.444785  1.7463769  1.3061928  9.202670
2
## V7          3.0000000 2.00000000 10.000000  1.0000000  1.0000000 10.000000
0
## CV residual 2.3590030 0.03353101  2.555215 -0.7463769 -0.3061928  0.797329
8
##                   212        213      223       227       264       283
## Predicted    8.7362557  1.3110677 2.330322  7.866028  7.936411  5.583603
## cvpred       8.7903771  1.3061928 2.278947  7.936524  7.864508  5.547196
## V7           8.0000000  1.0000000 5.000000 10.000000 10.000000 10.000000
## CV residual -0.7903771 -0.3061928 2.721053  2.063476  2.135492  4.452804
##                   300       309      310       311       321       332
## Predicted    6.394174  7.852053 2.410975  1.213452  4.930059  2.215744
## cvpred       6.346493  7.921324 2.411573  1.193687  4.952641  2.186561
## V7          10.000000  3.000000 5.000000  1.000000 10.000000  1.000000
## CV residual  3.653507 -4.921324 2.588427 -0.193687  5.047359 -1.186561
##                   339       343        355        365       390       392
## Predicted   0.98728319 0.8896677 0.98728319  1.5372368 1.8919595  7.542244
## cvpred      0.97359493 0.8610891 0.97359493  1.5262849 1.8539631  7.603926
## V7          1.00000000 1.0000000 1.00000000  1.0000000 2.0000000 10.000000
## CV residual 0.02640507 0.1389109 0.02640507 -0.5262849 0.1460369  2.396074
##                   402        405       434       436        453      456
## Predicted    1.1158368  1.3280304  2.097655  6.849736  1.7803686 3.016307
## cvpred       1.0811811  1.2860727  2.062740  6.980567  1.7262568 2.958309
## V7           1.0000000  1.0000000  1.000000 10.000000  1.0000000 6.000000
## CV residual -0.0811811 -0.2860727 -1.062740  3.019433 -0.7262568 3.041691
##                   458        465        474       483        485        48
8
## Predicted    9.360364  1.3420059  1.3420059 11.232883  1.8854614 10.811483
0
## cvpred       9.479578  1.3012732  1.3012732 11.326305  1.8578485 10.881201
4
## V7           3.000000  1.0000000  1.0000000  5.000000  1.0000000 10.000000
0
## CV residual -6.479578 -0.3012732 -0.3012732 -6.326305 -0.8578485 -0.881201
4
##                   496        503       504        506        509       516
## Predicted    1.439621  1.9980563  1.989575  1.1158368  1.5681750  6.877737
## cvpred       1.413779  1.9564089  1.966469  1.0811811  1.5213652  6.906734
## V7           1.000000  1.0000000  1.000000  1.0000000  1.0000000 10.000000
## CV residual -0.413779 -0.9564089 -0.966469 -0.0811811 -0.5213652  3.093266
##                   527       538       540      554       559       562
## Predicted    1.3420059  2.533030  2.118129 1.983077  1.213452  2.215744
## cvpred       1.3012732  2.523044  2.074055 1.970354  1.193687  2.186561
## V7           1.0000000  1.000000  1.000000 5.000000  1.000000  1.000000
```

```
## CV residual -0.3012732 -1.523044 -1.074055 3.029646 -0.193687 -1.186561
##                   573       589       595       599       604        668
## Predicted    1.439621  8.323337  5.535677  1.439621  7.746960  1.7634059
## cvpred       1.413779  8.335213  5.624353  1.413779  7.789732  1.7463769
## V7           1.000000  3.000000 10.000000  1.000000  1.000000  1.0000000
## CV residual -0.413779 -5.335213  4.375647 -0.413779 -6.789732 -0.7463769
##                   674        678       698
## Predicted    1.8854614  1.5681750  6.839297
## cvpred       1.8578485  1.5213652  6.944685
## V7           1.0000000  1.0000000  4.000000
## CV residual -0.8578485 -0.5213652 -2.944685
##
## Sum of squares = 424.19    Mean square = 6.15    n = 69
##
## fold 5
## Observations in test set: 68
##                     9         13       14        43        44        49
## Predicted   0.8896677  3.8386332 1.311068  6.924895  5.146219  2.654107
## cvpred      0.8757238  3.8665604 1.299128  6.875909  5.121808  2.690652
## V7          1.0000000  3.0000000 3.000000 10.000000  1.000000  1.000000
## CV residual 0.1242762 -0.8665604 1.700872  3.124091 -4.121808 -1.690652
##                    60        68        126        128        135        13
7
## Predicted    5.369401  3.491388 0.98728319  1.7634059  1.4396214  1.665790
4
## cvpred       5.443698  3.505635 0.96553808  1.7866794  1.4530894  1.696865
1
## V7           2.000000 10.000000 1.00000000  1.0000000  1.0000000  1.000000
0
## CV residual -3.443698  6.494365 0.03446192 -0.7866794 -0.4530894 -0.696865
1
##                   142       144        157        158       161       174
## Predicted   0.8896677 0.6634986  1.3045696  1.5372368  6.894675 10.554376
## cvpred      0.8757238 0.6319481  1.2648097  1.5429037  6.946213 10.517465
## V7          1.0000000 5.0000000  1.0000000  1.0000000 10.000000 10.000000
## CV residual 0.1242762 4.3680519 -0.2648097 -0.5429037  3.053787 -0.517465
##                   189        190       192       205       224       230
## Predicted    8.1007036  1.9456406  8.837838  1.311068 6.214184  8.809406
## cvpred       8.1444698  1.8976712  8.759701  1.299128 6.252696  8.848290
## V7           8.0000000  1.0000000 10.000000  1.000000 8.000000 10.000000
## CV residual -0.1444698 -0.8976712  1.240299 -0.299128 1.747304  1.151710
##                   251        257       262       266        277       285
## Predicted   0.98078507  1.1158368  8.999143  3.167603  1.4396214  6.927621
## cvpred      0.93121972  1.1194994  8.953296  3.172045  1.4530894  6.929753
## V7          1.00000000  1.0000000 10.000000  1.000000  1.0000000 10.000000
## CV residual 0.06878028 -0.1194994  1.046704 -2.172045 -0.4530894  3.070247
##                   286        320       326        370       383        399
## Predicted   11.006714  5.2333701  1.756908  1.6218560  2.412958  1.4396214
## cvpred      11.005016  5.2529245  1.752361  1.5640813  2.416049  1.4530894
## V7          10.000000  5.0000000  1.000000  1.0000000  1.000000  1.0000000
```

```
## CV residual -1.005016 -0.2529245 -0.752361 -0.5640813 -1.416049 -0.4530894
##                  401        419        422        430       466       467
## Predicted   7.920713  2.8652964  9.8146854  1.2134523  8.856328  7.207995
## cvpred      7.863710  2.9036006  9.8350929  1.2093137  8.902960  7.321105
## V7          9.000000  2.0000000 10.0000000  1.0000000  4.000000 10.000000
## CV residual 1.136290 -0.9036006  0.1649071 -0.2093137 -4.902960  2.678895
##                  470        476      486        493       508       510
## Predicted    1.3045696  1.1158368 1.328030  1.6657904 0.6634986 0.8896677
## cvpred       1.2648097  1.1194994 1.292145  1.6968651 0.6319481 0.8757238
## V7           1.0000000  1.0000000 3.000000  1.0000000 4.0000000 1.0000000
## CV residual -0.2648097 -0.1194994 1.707855 -0.6968651 3.3680519 0.1242762
##                  515       545       565       575       576        602
## Predicted    8.954947  2.180291  1.989575  7.209978  2.533030 0.98728319
## cvpred       8.952470  2.137955  2.030455  7.283295  2.573502 0.96553808
## V7          10.000000  1.000000  1.000000  2.000000  1.000000 1.00000000
## CV residual  1.047530 -1.137955 -1.030455 -5.283295 -1.573502 0.03446192
##                  610       613        619        621       625       629
## Predicted    1.5681750 11.006714  1.6657904  1.4396214  2.224225 0.8896677
## cvpred       1.6070508 11.005016  1.6968651  1.4530894  2.270739 0.8757238
## V7           1.0000000 10.000000  1.0000000  1.0000000  1.000000 1.0000000
## CV residual -0.6070508 -1.005016 -0.6968651 -0.4530894 -1.270739 0.1242762
##                  645        647        648      651        654        657
## Predicted   0.8896677 0.98078507  1.3280304 1.448103  1.6657904  1.8919595
## cvpred      0.8757238 0.93121972  1.2921447 1.449598  1.6968651  1.9406407
## V7          1.0000000 1.00000000  1.0000000 4.000000  1.0000000  1.0000000
## CV residual 0.1242762 0.06878028 -0.2921447 2.550402 -0.6968651 -0.9406407
##                  669       671
## Predicted    4.462741 7.2881239
## cvpred       4.506405 7.2273568
## V7           1.000000 8.0000000
## CV residual -3.506405 0.7726432
##
## Sum of squares = 250.79    Mean square = 3.69    n = 68
##
## fold 6
## Observations in test set: 68
##                     6        21         29        37        55        57
## Predicted   10.018398  6.623331  1.2134523 10.146951 7.5722028 5.7498291
## cvpred      10.173255  6.536414  1.2299342 10.215597 7.5324166 5.7851995
## V7          10.000000 10.000000  1.0000000  1.000000 8.0000000 6.0000000
## CV residual -0.173255  3.463586 -0.2299342 -9.215597 0.4675834 0.2148005
##                     64        74         80        81       104       127
## Predicted    3.393772  7.703744  1.2134523  3.149661  4.823962  6.591389
## cvpred       3.375680  7.586849  1.2299342  3.320914  4.836939  6.596374
## V7           2.000000 10.000000  1.0000000  1.000000  3.000000 10.000000
## CV residual -1.375680  2.413151 -0.2299342 -2.320914 -1.836939  3.403626
##                   138        145       156       160        166
173
## Predicted    1.11583682  1.2134523  5.146219  7.935668 1.98957497  0.98728
319
```

```
## cvpred       1.07561017  1.2299342  5.145130  8.037857 1.97425503  1.03326
857
## V7           1.00000000  1.0000000 10.000000 10.000000 2.00000000  1.00000
000
## CV residual -0.07561017 -0.2299342  4.854870  1.962143 0.02574497 -0.03326
857
##                   179     187        195       197       202       210
## Predicted    1.989575 6.48854  1.7634059 6.4703123  8.183315  2.215744
## cvpred       1.974255 6.77052  1.7775894 6.4770373  8.239467  2.170921
## V7           1.000000 8.00000  1.0000000 7.0000000 10.000000  1.000000
## CV residual -0.974255 1.22948 -0.7775894 0.5229627  1.760533 -1.170921
##                    226       238      248       272        275        281
## Predicted    0.98728319  6.994273 3.846111  2.215744  1.7634059  1.7634059
## cvpred       1.03326857  6.964859 3.769011  2.170921  1.7775894  1.7775894
## V7           1.00000000  2.000000 9.000000  1.000000  1.0000000  1.0000000
## CV residual -0.03326857 -4.964859 5.230989 -1.170921 -0.7775894 -0.7775894
##                    303        304        318       324       331       341
## Predicted    9.4909008  1.3110677  8.2904156  7.116354 6.553928  5.27630
## cvpred       9.5385901  1.3842582  8.4208401  7.136218 6.549422  5.23027
## V7          10.0000000  1.0000000  8.0000000 10.000000 8.000000 10.00000
## CV residual  0.4614099 -0.3842582 -0.4208401  2.863782 1.450578  4.76973
##                   350       376       381       388        397       403
## Predicted   5.248350 0.6634986 0.6634986  3.182583  1.7634059  2.526532
## cvpred      5.354343 0.6822790 0.6822790  3.169000  1.7775894  2.491994
## V7          8.000000 1.0000000 1.0000000  1.000000  1.0000000  1.000000
## CV residual 2.645657 0.3177210 0.3177210 -2.169000 -0.7775894 -1.491994
##                   414       417       421       426       447       482
## Predicted    2.533030  7.739458 2.7452241 11.232883 0.6634986  2.882259
## cvpred       2.506952  7.821163 2.7656544 11.372944 0.6822790  2.783023
## V7           1.000000 10.000000 3.0000000 10.000000 1.0000000  1.000000
## CV residual -1.506952  2.178837 0.2343456 -1.372944 0.3177210 -1.783023
##                   501        533        539        544        552       55
6
## Predicted    2.441913  1.3110677  1.6657904  1.6657904  1.3110677  2.31336
0
## cvpred       2.367586  1.3842582  1.6232654  1.6232654  1.3842582  2.32524
5
## V7           1.000000  1.0000000  1.0000000  1.0000000  1.0000000  1.00000
0
## CV residual -1.367586 -0.3842582 -0.6232654 -0.6232654 -0.3842582 -1.32524
5
##                   569       570       571        580        590        597
## Predicted    3.522326 10.824479  6.461831  1.3110677  1.5681750  1.9830769
## cvpred       3.418022 10.897547  6.502010  1.3842582  1.4689414  1.9592967
## V7          10.000000  5.000000 10.000000  1.0000000  1.0000000  1.0000000
## CV residual  6.581978 -5.897547  3.497990 -0.3842582 -0.4689414 -0.9592967
##                   649       665        677       679       682        688
## Predicted   10.328207  2.104153  1.3045696 0.6634986  8.709284  1.4396214
## cvpred      10.586281  2.078633  1.3692999 0.6822790  8.831333  1.4265998
## V7           2.000000  1.000000  1.0000000 1.0000000 10.000000  1.0000000
```

```
## CV residual -8.586281 -1.078633 -0.3692999 0.3177210  1.168667 -0.4265998
##                  696       699
## Predicted   0.8896677  7.806135
## cvpred      0.8789446  8.087469
## V7          1.0000000  5.000000
## CV residual 0.1210554 -3.087469
##
## Sum of squares = 458.38    Mean square = 6.74    n = 68
##
## fold 7
## Observations in test set: 68
##                   16         27        53       54       85        93
## Predicted    5.575097  1.4396214  5.589072 7.105889 7.288124  1.989575
## cvpred       5.544082  1.4614348  5.627713 6.975890 7.120335  2.018533
## V7           1.000000  1.0000000  5.000000 8.000000 9.000000  1.000000
## CV residual -4.544082 -0.4614348 -0.627713 1.024110 1.879665 -1.018533
##                   99       102        111       115        141        143
## Predicted   5.6671932 3.162109  2.2909027 2.0806923  1.1158368  5.7101483
## cvpred      5.6807195 3.097274  2.2591954 2.0907553  1.1419705  5.7094925
## V7          6.0000000 5.000000  2.0000000 3.0000000  1.0000000  5.0000000
## CV residual 0.3192805 1.902726 -0.2591954 0.9092447 -0.1419705 -0.7094925
##                  154        172       176       183        198       199
## Predicted   1.342006  1.3110677  6.760627  2.441913  3.212542 0.6634986
## cvpred      1.379604  1.3056322  6.642039  2.493799  3.200070 0.6667036
## V7          3.000000  1.0000000 10.000000  1.000000  1.000000 1.0000000
## CV residual 1.620396 -0.3056322  3.357961 -1.493799 -2.200070 0.3332964
##                  218       220       225       232        243       254
## Predicted    1.3110677  3.076486  7.572203 7.2936180  1.5372368  6.924895
## cvpred       1.3056322  3.113512  7.534788 7.2087963  1.5432657  6.874997
## V7           1.0000000  1.000000 10.000000 8.0000000  1.0000000 10.000000
## CV residual -0.3056322 -2.113512  2.465212 0.7912037 -0.5432657  3.125003
##                  261       267       268       271        279        288
## Predicted    9.2055566  5.921338  4.659956  4.797970  1.3110677  1.4396214
## cvpred       9.0839662  5.942348  4.612816  4.828203  1.3056322  1.4614348
## V7          10.0000000 10.000000 10.000000 10.000000  1.0000000  1.0000000
## CV residual  0.9160338  4.057652  5.387184  5.171797 -0.3056322 -0.4614348
##                  290        292       294      297        319        323
## Predicted    6.451822  1.3110677  5.635994 4.155920  1.3110677  1.7634059
## cvpred       6.327353  1.3056322  5.627610 4.134468  1.3056322  1.7808991
## V7          10.000000  1.0000000 10.000000 5.000000  1.0000000  1.0000000
## CV residual  3.672647 -0.3056322  4.372390 0.865532 -0.3056322 -0.7808991
##                  333       344       351       362       368       371
## Predicted    2.541512 0.6634986  2.232707  6.035941  9.046065  1.983077
## cvpred       2.561192 0.6667036  2.246507  5.871251  8.854192  2.008924
## V7           1.000000 1.0000000  1.000000 10.000000 10.000000  1.000000
## CV residual -1.561192 0.3332964 -1.246507  4.128749  1.145808 -1.008924
##                  374       413       432        433        441      442
## Predicted    2.224225  9.482419  2.880276  1.8919595  9.2380471 2.882259
## cvpred       2.251336  9.403533  2.885435  1.9367017  9.1320068 2.870997
## V7           1.000000  4.000000  1.000000  1.0000000 10.0000000 4.000000
```

```
## CV residual -1.251336 -5.403533 -1.885435 -0.9367017  0.8679932 1.129003
##                   450        452       454        471        481       505
## Predicted    8.698819  1.5681750  7.851074  1.4396214  1.5681750 0.6634986
## cvpred       8.534779  1.6172374  7.696649  1.4614348  1.6172374 0.6667036
## V7          10.000000  1.0000000 10.000000  1.0000000  1.0000000 1.0000000
## CV residual  1.465221 -0.6172374  2.303351 -0.4614348 -0.6172374 0.3332964
##                   518        549       558        560        564         5
79
## Predicted   0.98728319  1.1158368  2.232707  1.8919595  1.4396214 0.987283
19
## cvpred      0.98616792  1.1419705  2.246507  1.9367017  1.4614348 0.986167
92
## V7          1.00000000  1.0000000  1.000000  1.0000000  1.0000000 1.000000
00
## CV residual 0.01383208 -0.1419705 -1.246507 -0.9367017 -0.4614348 0.013832
08
##                   601        615       640       644       658        664
## Predicted    1.4396214  1.2134523  2.232707 0.6634986  3.484890  1.6218560
## cvpred       1.4614348  1.2238014  2.246507 0.6667036  3.495591  1.6058803
## V7           1.0000000  1.0000000  1.000000 1.0000000  1.000000  1.0000000
## CV residual -0.4614348 -0.2238014 -1.246507 0.3332964 -2.495591 -0.6058803
##                   670       685
## Predicted    8.692321 0.6634986
## cvpred       8.525171 0.6667036
## V7           5.000000 1.0000000
## CV residual -3.525171 0.3332964
##
## Sum of squares = 275.02    Mean square = 4.04    n = 68
##
## fold 8
## Observations in test set: 68
##                    30        32        42        45         46        61
## Predicted    1.2980715  1.537237  4.952516  8.817888 0.98728319  5.137738
## cvpred       1.3531581  1.506356  4.915405  8.887094 0.97979325  5.228491
## V7           1.0000000  1.000000  3.000000  1.000000 1.00000000  3.000000
## CV residual -0.3531581 -0.506356 -1.915405 -7.887094 0.02020675 -2.228491
##                   69        78         86        88         94         97
## Predicted   7.398711  2.224225  4.1259607  5.982521 0.98728319  1.2219336
## cvpred      7.405879  2.258156  4.1665584  6.034897 0.97979325  1.2668765
## V7          9.000000  1.000000  4.0000000 10.000000 1.00000000  1.0000000
## CV residual 1.594121 -1.258156 -0.1665584  3.965103 0.02020675 -0.2668765
##                   105       106        136       185       186        203
## Predicted   10.811483  4.616739  2.2157441  6.125050  1.537237  1.3110677
## cvpred      10.980669  4.716289  2.2034311  6.204626  1.506356  1.2739976
## V7           1.000000  3.000000  2.0000000 10.000000  1.000000  1.0000000
## CV residual -9.980669 -1.716289 -0.2034311  3.795374 -0.506356 -0.2739976
##                   207       240        245        249       269        307
## Predicted    6.549413  4.960997  1.3110677  1.9895750  7.518783  1.3110677
## cvpred       6.733358  4.970130  1.2739976  1.9710727  7.521666  1.2739976
## V7           5.000000 10.000000  1.0000000  1.0000000  4.000000  1.0000000
```

```
## CV residual -1.733358  5.029870 -0.2739976 -0.9710727 -3.521666 -0.2739976
##                    315        325       335       340      363        369
## Predicted   0.98728319  1.3110677  5.462502  5.816245 1.756908  1.2980715
## cvpred      0.97979325  1.2739976  5.601072  5.925566 1.778295  1.3531581
## V7          1.00000000  1.0000000 10.000000 10.000000 3.000000  1.0000000
## CV residual 0.02020675 -0.2739976  4.398928  4.074434 1.221705 -0.3531581
##                    375       380        385       387       415       427
## Predicted    1.7569078  3.167603 0.88966773  6.214184  5.784303  3.154607
## cvpred       1.7782946  3.204785 0.91794726  6.211747  5.800971  3.283945
## V7           1.0000000  1.000000 1.00000000 10.000000 10.000000  1.000000
## CV residual -0.7782946 -2.204785 0.08205274  3.788253  4.199029 -2.283945
##                    448       451       463       480       484       487
## Predicted    1.5681750  2.330322  2.458876  8.178825  8.387027  1.439621
## cvpred       1.6150224  2.374727  2.545239  8.290961  8.397941  1.444510
## V7           1.0000000  1.000000  1.000000 10.000000 10.000000  1.000000
## CV residual -0.6150224 -1.374727 -1.545239  1.709039  1.602059 -0.444510
##                    489        494       497       507        513       514
## Predicted    5.705658  9.0330686 0.6634986  9.064007  1.5681750  1.439621
## cvpred       5.880431  9.2304688 0.6855889  9.339135  1.6150224  1.444510
## V7           3.000000 10.0000000 1.0000000  5.000000  1.0000000  1.000000
## CV residual -2.880431  0.7695312 0.3144111 -4.339135 -0.6150224 -0.444510
##                    522       523        530        542       568       585
## Predicted    1.342006  6.907671  1.6657904  1.1158368 1.665790  3.229504
## cvpred       1.382664  6.923966  1.6768683  1.1503056 1.676868  3.359669
## V7           1.000000  5.000000  1.0000000  1.0000000 3.000000  1.000000
## CV residual -0.382664 -1.923966 -0.6768683 -0.1503056 1.323132 -2.359669
##                    588        603        627       630       650        65
5
## Predicted    1.8919595  1.6657904  6.2002088  1.342006  1.439621  1.763405
9
## cvpred       1.9092267  1.6768683  6.2125301  1.382664  1.444510  1.738714
3
## V7           1.0000000  1.0000000  6.0000000  1.000000  1.000000  1.000000
0
## CV residual -0.9092267 -0.6768683 -0.2125301 -0.382664 -0.444510 -0.738714
3
##                    656       660        661       672       684       686
## Predicted    1.439621 0.6634986 0.98728319  2.095672 0.6634986 0.6634986
## cvpred       1.444510 0.6855889 0.97979325  2.087644 0.6855889 0.6855889
## V7           1.000000 1.0000000 1.00000000  1.000000 1.0000000 1.0000000
## CV residual -0.444510 0.3144111 0.02020675 -1.087644 0.3144111 0.3144111
##                    692       694
## Predicted    6.724170  1.439621
## cvpred       6.897414  1.444510
## V7           5.000000  1.000000
## CV residual -1.897414 -0.444510
##
## Sum of squares = 388.7    Mean square = 5.72    n = 68
##
## fold 9
```

```
## Observations in test set: 68
##                      2        35       47        70        72         75
## Predicted    4.496667  1.756908 4.987707  1.311068  6.380199  4.2984487
## cvpred       4.508080  1.754327 4.991610  1.291313  6.352392  4.2994208
## V7          10.000000  1.000000 9.000000  1.000000  2.000000  4.0000000
## CV residual  5.491920 -0.754327 4.008390 -0.291313 -4.352392 -0.2994208
##                     90         95        132       155       168
170
## Predicted    1.5457182  1.5372368  1.5372368 0.6634986  9.192560  0.995764
544
## cvpred       1.5426069  1.5186415  1.5186415 0.6665928  9.250653  1.002918
279
## V7           1.0000000  1.0000000  1.0000000 1.0000000  1.000000  1.000000
000
## CV residual -0.5426069 -0.5186415 -0.5186415 0.3334072 -8.250653 -0.002918
279
##                    171        177        193       206       216        21
7
## Predicted    1.1158368  1.5372368  1.8919595  9.024587  7.756421 0.9872831
9
## cvpred       1.1212497  1.5186415  1.8882668  9.027126  7.770306 0.9789529
1
## V7           1.0000000  1.0000000  1.0000000 10.000000  5.000000 1.0000000
0
## CV residual -0.1212497 -0.5186415 -0.8882668  0.972874 -2.770306 0.0210470
9
##                    228       234       239        247       252       253
## Predicted    8.056744  6.268583 8.0756904  9.3708285  7.936411  4.405550
## cvpred       8.043092  6.277800 8.1233454  9.3727859  7.951071  4.414692
## V7           5.000000 10.000000 9.0000000 10.0000000 10.000000 10.000000
## CV residual -3.043092  3.722200 0.8766546  0.6272141  2.048929  5.585308
##                    263       273       305       336       338       342
## Predicted    7.724479  4.659956  6.504238 0.6634986  1.311068  1.311068
## cvpred       7.706767  4.614605  6.535368 0.6665928  1.291313  1.291313
## V7          10.000000 10.000000 10.000000 1.0000000  1.000000  1.000000
## CV residual  2.293233  5.385395  3.464632 0.3334072 -0.291313 -0.291313
##                    345       347       348        352        356         36
1
## Predicted    7.888510  2.217727 0.6634986  1.5372368  1.6657904  9.9068067
1
## cvpred       7.845875  2.232949 0.6665928  1.5186415  1.6609383  9.9031401
3
## V7          10.000000  1.000000 1.0000000  1.0000000  1.0000000 10.0000000
0
## CV residual  2.154125 -1.232949 0.3334072 -0.5186415 -0.6609383  0.0968598
7
##                    372       379        386        393       394        40
6
## Predicted    1.2980715  2.436419  1.7653891  1.4396214 0.6634986 0.9872831
9
```

```
## cvpred       1.3080271  2.442586  1.7782924  1.4336099 0.6665928 0.9789529
1
## V7           1.0000000  1.000000  1.0000000  1.0000000 1.0000000 1.0000000
0
## CV residual -0.3080271 -1.442586 -0.7782924 -0.4336099 0.3334072 0.0210470
9
##                    407       410        411       420       423        431
## Predicted    1.9830769  1.756908 0.98728319 0.8896677  2.624148 0.98728319
## cvpred       1.9816555  1.754327 0.97895291 0.8939213  2.614733 0.97895291
## V7           1.0000000  1.000000 1.00000000 1.0000000  1.000000 1.00000000
## CV residual -0.9816555 -0.754327 0.02104709 0.1060787 -1.614733 0.02104709
##                    438       443        445       457       462        490
## Predicted    1.3420059 1.328030  3.553289  8.560519 1.115837 3.0829841
## cvpred       1.3485782 1.339244  3.569894  8.536217 1.121250 3.0610327
## V7           1.0000000 3.000000  1.000000 10.000000 5.000000 4.0000000
## CV residual -0.3485782 1.660756 -2.569894  1.463783 3.878750 0.9389673
##                    520        526        535       572        578       581
## Predicted    6.817819  1.4481027  1.2134523  8.795431 0.98728319  2.209246
## cvpred       6.765130  1.4575752  1.2062814  8.761201 0.97895291  2.208984
## V7          10.000000  1.0000000  1.0000000 10.000000 1.00000000  1.000000
## CV residual  3.234870 -0.4575752 -0.2062814  1.238799 0.02104709 -1.208984
##                    584       586        591        609        632        64
6
## Predicted    1.1158368 0.6634986  7.806135 10.3282067  1.8919595  1.439621
4
## cvpred       1.1212497 0.6665928  7.750142 10.3005319  1.8882668  1.433609
9
## V7           1.0000000 1.0000000  1.000000 10.0000000  1.0000000  1.000000
0
## CV residual -0.1212497 0.3334072 -6.750142 -0.3005319 -0.8882668 -0.433609
9
##                    667        689
## Predicted    2.217727  1.3420059
## cvpred       2.232949  1.3485782
## V7           1.000000  1.0000000
## CV residual -1.232949 -0.3485782
##
## Sum of squares = 354.73    Mean square = 5.22    n = 68
##
## fold 10
## Observations in test set: 68
##                      1        18        19        25         28         31
## Predicted    2.215744  1.989575  7.235422  1.311068  1.8919595  1.4396214
## cvpred       2.261519  2.024431  7.290197  1.313169  1.9346683  1.4604934
## V7           1.000000  1.000000 10.000000  1.000000  1.0000000  1.0000000
## CV residual -1.261519 -1.024431  2.709803 -0.313169 -0.9346683 -0.4604934
##                     33        38        96        98       100        107
## Predicted    7.209978  3.737051  1.311068  2.215744  8.540046  8.851813
## cvpred       7.237259  3.806008  1.313169  2.261519  8.638487  8.876143
## V7           5.000000  1.000000  1.000000  1.000000 10.000000 10.000000
```

```
## CV residual -2.237259 -2.806008 -0.313169 -1.261519  1.361513  1.123857
##                   113        114       117        121        134       148
## Predicted    8.266694  8.4856218  3.528824  1.9606200  1.4396214 0.9872832
## cvpred       8.452796  8.4610631  3.570750  1.9566501  1.4604934 0.9863185
## V7          10.000000  8.0000000  2.000000  1.0000000  1.0000000 2.0000000
## CV residual  1.547204 -0.4610631 -1.570750 -0.9566501 -0.4604934 1.0136815
##                   153        180       191        196        209       214
## Predicted    8.847847  3.514849  9.823167  1.989575  1.311068 10.4876985
## cvpred       8.896583  3.548481  9.856695  2.024431  1.313169 10.5456879
## V7           5.000000 10.000000  8.000000  1.000000  1.000000 10.0000000
## CV residual -3.896583  6.451519 -1.856695 -1.024431 -0.313169 -0.5456879
##                   229        246       256        258        278       280
## Predicted    1.311068  2.5480100  4.134442  1.4396214 0.98728319 5.914840
## cvpred       1.313169  2.6060153  4.100938  1.4604934 0.98631851 5.929857
## V7           1.000000  2.0000000 10.000000  1.0000000 1.00000000 7.000000
## CV residual -0.313169 -0.6060153  5.899062 -0.4604934 0.01368149 1.070143
##                   284        301       308        313        314       329
## Predicted    5.580591  8.374031  1.311068  6.836504 0.6634986 3.8610901
## cvpred       5.595580  8.349031  1.313169  6.970203 0.6594680 3.9152469
## V7          10.000000  4.000000  1.000000  1.000000 1.0000000 4.0000000
## CV residual  4.404420 -4.349031 -0.313169 -5.970203 0.3405320 0.0847531
##                   396        400        409        424        425       435
## Predicted    1.4396214  1.2980715  2.1867891  2.526532  1.1158368 5.99848
## cvpred       1.4604934  1.2574372  2.1937376  2.532638  1.1336429 5.99735
## V7           1.0000000  1.0000000  2.0000000  1.000000  1.0000000 8.00000
## CV residual -0.4604934 -0.2574372 -0.1937376 -1.532638 -0.1336429 2.00265
##                   440        460        468        469        478        49
9
## Predicted    1.5681750  2.202748  6.652547  1.3420059  1.3420059  1.665790
4
## cvpred       1.6078178  2.205787  6.678917  1.3707304  1.3707304  1.697580
9
## V7           1.0000000  1.000000 10.000000  1.0000000  1.0000000  1.000000
0
## CV residual -0.6078178 -1.205787  3.321083 -0.3707304 -0.3707304 -0.697580
9
##                   517        532        547        582        583       587
## Predicted   0.6634986  1.9830769  9.5830222  8.027790  6.011476 11.006714
## cvpred      0.6594680  1.9965655  9.5973381  7.982265  6.053082 11.052064
## V7          1.0000000  1.0000000 10.0000000 10.000000 10.000000 10.000000
## CV residual 0.3405320 -0.9965655  0.4026619  2.017735  3.946918 -1.052064
##                   592        594        596        608        612        620
## Predicted    6.397423  1.8854614  1.8919595 0.6634986  9.680638  1.8919595
## cvpred       6.374049  1.9068024  1.9346683 0.6594680  9.687101  1.9346683
## V7          10.000000  1.0000000  1.0000000 1.0000000  2.000000  1.0000000
## CV residual  3.625951 -0.9068024 -0.9346683 0.3405320 -7.687101 -0.9346683
##                   623        631        635        643        652        65
3
## Predicted    3.326116  2.428917  1.1158368  1.4396214  1.6518150  1.891959
5
```

```
## cvpred       3.396821  2.442875  1.1336429  1.4604934  1.6753115  1.934668
3
## V7           1.000000  1.000000  1.0000000  1.0000000  1.0000000  1.000000
0
## CV residual -2.396821 -1.442875 -0.1336429 -0.4604934 -0.6753115 -0.934668
3
##                   680       687
## Predicted   0.8896677 0.6634986
## cvpred      0.8965555 0.6594680
## V7          1.0000000 1.0000000
## CV residual 0.1034445 0.3405320
##
## Sum of squares = 335.35    Mean square = 4.93    n = 68
##
## Overall (Sum over all 68 folds)
##       ms
## 5.217522

summary(reg_mcv)

##        V2              V3              V4              V5
##  Min.   : 1.000   Min.   : 1.000   Min.   : 1.000   Min.   : 1.00
##  1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 1.000   1st Qu.: 1.00
##  Median : 4.000   Median : 1.000   Median : 1.000   Median : 1.00
##  Mean   : 4.442   Mean   : 3.151   Mean   : 3.215   Mean   : 2.83
##  3rd Qu.: 6.000   3rd Qu.: 5.000   3rd Qu.: 5.000   3rd Qu.: 4.00
##  Max.   :10.000   Max.   :10.000   Max.   :10.000   Max.   :10.00
##        V6              V7              V8              V9
##  Min.   : 1.000   Min.   : 1.000   Min.   : 1.000   Min.   : 1.00
##  1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 2.000   1st Qu.: 1.00
##  Median : 2.000   Median : 1.000   Median : 3.000   Median : 1.00
##  Mean   : 3.234   Mean   : 3.545   Mean   : 3.445   Mean   : 2.87
##  3rd Qu.: 4.000   3rd Qu.: 6.000   3rd Qu.: 5.000   3rd Qu.: 4.00
##  Max.   :10.000   Max.   :10.000   Max.   :10.000   Max.   :10.00
##       V10           Predicted          cvpred             fold
##  Min.   : 1.000   Min.   : 0.6635   Min.   : 0.5665   Min.   : 1.000
##  1st Qu.: 1.000   1st Qu.: 1.3420   1st Qu.: 1.3612   1st Qu.: 3.000
##  Median : 1.000   Median : 2.0807   Median : 2.0741   Median : 5.000
##  Mean   : 1.603   Mean   : 3.5447   Mean   : 3.5433   Mean   : 5.489
##  3rd Qu.: 1.000   3rd Qu.: 5.7858   3rd Qu.: 5.7931   3rd Qu.: 8.000
##  Max.   :10.000   Max.   :11.4591   Max.   :11.3914   Max.   :10.000

#Calculate R-squared to see quality of the model
TSS <- sum((as.numeric(data[-missing,]$V7) - mean(as.numeric(data[-missing,]$
V7)))^2)
R2 <- 1 - attr(reg_mcv,"ms")*nrow(data[-missing,])/TSS
R2

## [1] 0.6064699
```

```
#Now it's time to predict the values
preds <- predict(reg_ms, newdata = data[missing,])
preds
```

```
##        24        41       140       146       159       165       236
250
## 5.4585352 7.9816106 0.9872832 1.6218560 0.9807851 2.2157441 2.7152652 1.76
34059
##       276       293       295       298       316       322       412
618
## 2.0741942 6.0866099 0.9872832 2.5265324 5.2438347 1.7634059 0.9872832 0.66
34986
```

```
impd <- data
impd[missing, ]$V7 <- preds

#Convert to numeric since they're character values right now
impd$V7 <- as.numeric(impd$V7)
impd[missing, ]$V7 <- round(preds)

#Convert back to integer finally
impd$V7 <- as.integer(impd$V7)

#Check to make sure none of the values are out of bounds
check <- impd[missing, "V7"]
check
```

```
##  [1] 5 8 1 2 1 2 3 2 2 6 1 3 5 2 1 1
```

Question 14.1.3

```
#This is regression imputation with perturbation
#Load Data
data <- read.table("C:\\Users\\User\\OneDrive\\Desktop\\Data 14.1\\breast-can
cer-wisconsin.data.txt", stringsAsFactors = F, header = F, sep=",")

#Get the missing indices
missing <- which(data$V7 == "?", arr.ind = T)
missing
```

```
##  [1]  24  41 140 146 159 165 236 250 276 293 295 298 316 322 412 618
```

```
#Prepare the needed data
reg_imp_data <- data[-missing, 2:10]
reg_imp_data$V7 <- as.integer(reg_imp_data$V7)

#Create a linear model
reg_m <- lm(V7~V2+V3+V4+V5+V6+V8+V9+V10, data=reg_imp_data)
summary(reg_m)
```

```
##
## Call:
## lm(formula = V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 + V10, data = reg_imp_d
ata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7316 -0.9426 -0.3002  0.6725  8.6998
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.616652   0.194975  -3.163  0.00163 **
## V2           0.230156   0.041691   5.521 4.83e-08 ***
## V3          -0.067980   0.076170  -0.892  0.37246
## V4           0.340442   0.073420   4.637 4.25e-06 ***
## V5           0.339705   0.045919   7.398 4.13e-13 ***
## V6           0.090392   0.062541   1.445  0.14883
## V8           0.320577   0.059047   5.429 7.91e-08 ***
## V9           0.007293   0.044486   0.164  0.86983
## V10         -0.075230   0.059331  -1.268  0.20524
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.274 on 674 degrees of freedom
## Multiple R-squared:  0.615,  Adjusted R-squared:  0.6104
## F-statistic: 134.6 on 8 and 674 DF,  p-value: < 2.2e-16

#Need to select only the best variables so we can use stepwise regression
step(reg_m)

## Start:  AIC=1131.43
## V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 + V10
##
##        Df Sum of Sq    RSS    AIC
## - V9    1     0.139 3486.8 1129.5
## - V3    1     4.120 3490.8 1130.2
## - V10   1     8.317 3495.0 1131.0
## <none>              3486.6 1131.4
## - V6    1    10.806 3497.5 1131.5
## - V4    1   111.227 3597.9 1150.9
## - V8    1   152.482 3639.1 1158.7
## - V2    1   157.657 3644.3 1159.6
## - V5    1   283.119 3769.8 1182.8
##
## Step:  AIC=1129.45
## V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V10
##
##        Df Sum of Sq    RSS    AIC
## - V3    1     4.028 3490.8 1128.2
## - V10   1     8.179 3495.0 1129.0
```

```
## <none>              3486.8 1129.5
## - V6    1    11.211 3498.0 1129.7
## - V4    1   114.768 3601.6 1149.6
## - V2    1   158.696 3645.5 1157.8
## - V8    1   160.776 3647.6 1158.2
## - V5    1   285.902 3772.7 1181.3
##
## Step:  AIC=1128.24
## V7 ~ V2 + V4 + V5 + V6 + V8 + V10
##
##        Df Sum of Sq    RSS    AIC
## - V6    1     8.606 3499.4 1127.9
## - V10   1     8.889 3499.7 1128.0
## <none>              3490.8 1128.2
## - V4    1   153.078 3643.9 1155.6
## - V2    1   155.308 3646.1 1156.0
## - V8    1   157.123 3647.9 1156.3
## - V5    1   282.133 3772.9 1179.3
##
## Step:  AIC=1127.92
## V7 ~ V2 + V4 + V5 + V8 + V10
##
##        Df Sum of Sq    RSS    AIC
## - V10   1     5.562 3505.0 1127.0
## <none>              3499.4 1127.9
## - V2    1   159.594 3659.0 1156.4
## - V8    1   169.954 3669.4 1158.3
## - V4    1   206.785 3706.2 1165.1
## - V5    1   295.807 3795.2 1181.3
##
## Step:  AIC=1127.01
## V7 ~ V2 + V4 + V5 + V8
##
##        Df Sum of Sq    RSS    AIC
## <none>              3505.0 1127.0
## - V2    1    155.70 3660.7 1154.7
## - V8    1    172.42 3677.4 1157.8
## - V4    1    201.22 3706.2 1163.1
## - V5    1    290.68 3795.7 1179.4

##
## Call:
## lm(formula = V7 ~ V2 + V4 + V5 + V8, data = reg_imp_data)
##
## Coefficients:
## (Intercept)           V2           V4           V5           V8
##     -0.5360       0.2262       0.3173       0.3323       0.3238
```

```r
#Generate a better model now
reg_ms <- lm(V7~V2+V4+V5+V8, data = reg_imp_data)
summary(reg_ms)
```

```
##
## Call:
## lm(formula = V7 ~ V2 + V4 + V5 + V8, data = reg_imp_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8115 -0.9531 -0.3111  0.6678  8.6889
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.53601    0.17514  -3.060   0.0023 **
## V2           0.22617    0.04121   5.488 5.75e-08 ***
## V4           0.31729    0.05086   6.239 7.76e-10 ***
## V5           0.33227    0.04431   7.499 2.03e-13 ***
## V8           0.32378    0.05606   5.775 1.17e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.274 on 678 degrees of freedom
## Multiple R-squared:  0.6129, Adjusted R-squared:  0.6107
## F-statistic: 268.4 on 4 and 678 DF,  p-value: < 2.2e-16
```

```r
#Predict the values
preds <- predict(reg_ms, newdata = data[missing,])
preds
```

```
##        24        41       140       146       159       165       236
250
## 5.4585352 7.9816106 0.9872832 1.6218560 0.9807851 2.2157441 2.7152652 1.76
34059
##       276       293       295       298       316       322       412
618
## 2.0741942 6.0866099 0.9872832 2.5265324 5.2438347 1.7634059 0.9872832 0.66
34986
```

```r
#Perturbation values using a normal distribution
perts <- rnorm(nrow(data[missing,]), preds, sd(preds))
perts
```

```
##  [1]  6.93560160  3.14378473  0.08579384 -0.78969862  0.95708275  3.317976
05
##  [7]  0.79160608 -0.57705244  6.79889284  7.76637879  2.78578227  2.985562
38
## [13]  6.48492561  3.80582469 -0.73027027 -1.57838969
```

```r
#Update the df
pert_data <- data
```

```
pert_data[missing, ]$V7 <- perts

#Convert to numeric, round, then convert to integer
pert_data$V7 <- as.numeric(pert_data$V7)
pert_data[missing, ]$V7 <- round(perts)
pert_data$V7 <- as.integer(pert_data$V7)

#Check to make sure none of the values are out of bounds
check <- pert_data[missing, "V7"]
check

##  [1]  7  3  0 -1  1  3  1 -1  7  8  3  3  6  4 -1 -2

#Out of bound values (e.g., negatives) need to be corrected
pert_data$V7[pert_data$V7 > 10] <- 10
pert_data$V7[pert_data$V7 < 1] <- 1

#Check again
check <- pert_data[missing, "V7"]
check

##  [1] 7 3 1 1 1 3 1 1 7 8 3 3 6 4 1 1
```

Question 14.1.4

```
#This is comparing the different imputation methods to see which works best
#Load Libraries
library(missForest); library(mice)

## Warning: package 'missForest' was built under R version 4.2.3

## Warning: package 'mice' was built under R version 4.2.3

##
## Attaching package: 'mice'

## The following object is masked from 'package:stats':
##
##     filter

## The following objects are masked from 'package:base':
##
##     cbind, rbind

#Load Data
data <- read.table("C:\\Users\\User\\OneDrive\\Desktop\\Data 14.1\\breast-can
cer-wisconsin.data.txt", stringsAsFactors = F, header = F, sep=",")

#Split the dataset into training and test sets
set.seed(1)
train_index <- sample(nrow(data), round(0.7*nrow(data)), replace = FALSE)
train <- data[train_index, ]
```

```r
test <- data[-train_index, ]

#Set the missing values to NA
train_missing <- train
train_missing["V7"][train_missing["V7"] == "?"] <- NA
train_missing$V7 <- as.numeric(train_missing$V7)
test_missing <- test
test_missing["V7"][test_missing["V7"] == "?"] <- NA
test_missing$V7 <- as.numeric(test_missing$V7)

#Mean/mode imputation
train_imputed_mean <- train_missing
train_imputed_mean[] <- lapply(train_imputed_mean, function(x) ifelse(is.na(x
), mean(x, na.rm = TRUE), x))
train_imputed_mean$V7 <- pmax(pmin(train_imputed_mean$V7, 10), 1)
test_imputed_mean <- test_missing
test_imputed_mean[] <- lapply(test_imputed_mean, function(x) ifelse(is.na(x),
mean(x, na.rm = TRUE), x))
test_imputed_mean$V7 <- pmax(pmin(test_imputed_mean$V7, 10), 1)

#Linear regression imputation
train_imputed_lm <- mice(train_missing, method = "norm.predict", m = 10)

##
##  iter imp variable
##   1   1  V7
##   1   2  V7
##   1   3  V7
##   1   4  V7
##   1   5  V7
##   1   6  V7
##   1   7  V7
##   1   8  V7
##   1   9  V7
##   1   10  V7
##   2   1  V7
##   2   2  V7
##   2   3  V7
##   2   4  V7
##   2   5  V7
##   2   6  V7
##   2   7  V7
##   2   8  V7
##   2   9  V7
##   2   10  V7
##   3   1  V7
##   3   2  V7
##   3   3  V7
##   3   4  V7
##   3   5  V7
```

```
##   3   6  V7
##   3   7  V7
##   3   8  V7
##   3   9  V7
##   3   10  V7
##   4   1  V7
##   4   2  V7
##   4   3  V7
##   4   4  V7
##   4   5  V7
##   4   6  V7
##   4   7  V7
##   4   8  V7
##   4   9  V7
##   4   10  V7
##   5   1  V7
##   5   2  V7
##   5   3  V7
##   5   4  V7
##   5   5  V7
##   5   6  V7
##   5   7  V7
##   5   8  V7
##   5   9  V7
##   5   10  V7
```

```
test_imputed_lm <- mice(test_missing, method = "norm.predict", m = 10)
```

```
##
##  iter imp variable
##   1   1  V7
##   1   2  V7
##   1   3  V7
##   1   4  V7
##   1   5  V7
##   1   6  V7
##   1   7  V7
##   1   8  V7
##   1   9  V7
##   1   10  V7
##   2   1  V7
##   2   2  V7
##   2   3  V7
##   2   4  V7
##   2   5  V7
##   2   6  V7
##   2   7  V7
##   2   8  V7
##   2   9  V7
##   2   10  V7
```

```
##   3   1  V7
##   3   2  V7
##   3   3  V7
##   3   4  V7
##   3   5  V7
##   3   6  V7
##   3   7  V7
##   3   8  V7
##   3   9  V7
##   3   10  V7
##   4   1  V7
##   4   2  V7
##   4   3  V7
##   4   4  V7
##   4   5  V7
##   4   6  V7
##   4   7  V7
##   4   8  V7
##   4   9  V7
##   4   10  V7
##   5   1  V7
##   5   2  V7
##   5   3  V7
##   5   4  V7
##   5   5  V7
##   5   6  V7
##   5   7  V7
##   5   8  V7
##   5   9  V7
##   5   10  V7
```

```r
train_imputed_lm <- complete(train_imputed_lm, action = "long")
train_imputed_lm$V7 <- pmax(pmin(train_imputed_lm$V7, 10), 1)
test_imputed_lm <- complete(test_imputed_lm, action = "long")
test_imputed_lm$V7 <- pmax(pmin(test_imputed_lm$V7, 10), 1)

#Linear regression with perturbation imputation
train_imputed_lmp <- missForest(train_missing)
train_imputed_lmp$ximp$V7 <- pmax(pmin(train_imputed_lmp$ximp$V7, 10), 1)
test_imputed_lmp <- missForest(test_missing)
test_imputed_lmp$ximp$V7 <- pmax(pmin(test_imputed_lmp$ximp$V7, 10), 1)

#Fit a linear regression model and compute the prediction error
fit_mean <- lm(V11 ~ ., data = train_imputed_mean)
pred_mean <- predict(fit_mean, newdata = test_imputed_mean)
error_mean <- mean((pred_mean - test$V11)^2)
rsq_mean <- summary(fit_mean)$r.squared

fit_lm <- lm(V11 ~ ., data = train_imputed_lm)
pred_lm <- predict(fit_lm, newdata = test_imputed_lm)
```

```r
error_lm <- mean((pred_lm - test$V11)^2)
rsq_lm <- summary(fit_lm)$r.squared

fit_lmp <- lm(V11 ~ ., data = train_imputed_lmp$ximp)
pred_lmp <- predict(fit_lmp, newdata = test_imputed_lmp$ximp)
error_lmp <- mean((pred_lmp - test$V11)^2)
rsq_lmp <- summary(fit_lmp)$r.squared

#Print the prediction errors
cat("Mean imputation error:", error_mean, "\n")

## Mean imputation error: 0.1448082

cat("Mean imputation R-squared:", rsq_mean, "\n")

## Mean imputation R-squared: 0.833089

cat("Linear regression imputation error:", error_lm, "\n")

## Linear regression imputation error: 0.1398115

cat("Linear regression imputation R-squared:", rsq_lm, "\n")

## Linear regression imputation R-squared: 0.8343196

cat("Linear regression with perturbation imputation error:", error_lmp, "\n")

## Linear regression with perturbation imputation error: 0.141096

cat("Linear regression with perturbation imputation R-squared:", rsq_lmp, "\n
")

## Linear regression with perturbation imputation R-squared: 0.8296648
```