Sections

Contents

Using Mean to Inpute Values for the Missing Data

```
#Clearing the environment
rm(list = ls())

#Loading the Library
library(dplyr)
library(purrr)
library(kernlab)

set.seed(10)

#Pulling in the data
Wis_data <- read.table("breast-cancer-wisconsin.data.txt", header=FALSE, stringsAsFactors = TRUE,
sep=",")
head(Wis_data)
```

```
     V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11
1 1000025  5  1  1  1  2  1  3  1   1   2
2 1002945  5  4  4  5  7 10  3  2   1   2
3 1015425  3  1  1  1  2  2  3  1   1   2
4 1016277  6  8  8  1  3  4  3  7   1   2
5 1017023  4  1  1  3  2  1  3  1   1   2
6 1017122  8 10 10  8  7 10  9  7   1   4
```

Here I am checking the top few entries of the data table so that I get an idea of how to handle the data in the next few steps.

```
#Changing the V11 value to a binary value
Wis_data$V11[Wis_data$V11 == 2] <- 0
Wis_data$V11[Wis_data$V11 == 4] <- 1
Wis_data[Wis_data == '?'] <- ''

Wis_data
```

```
          V1 V2 V3 V4 V5 V6   V7 V8 V9 V10 V11
1    1000025  5  1  1  1  2    1  3  1   1   0
2    1002945  5  4  4  5  7   10  3  2   1   0
3    1015425  3  1  1  1  2    2  3  1   1   0
4    1016277  6  8  8  1  3    4  3  7   1   0
5    1017023  4  1  1  3  2    1  3  1   1   0
6    1017122  8 10 10  8  7   10  9  7   1   1
7    1018099  1  1  1  1  2   10  3  1   1   0
8    1018561  2  1  2  1  2    1  3  1   1   0
9    1033078  2  1  1  1  2    1  1  1   5   0
10   1033078  4  2  1  1  2    1  2  1   1   0
11   1035283  1  1  1  1  1    1  3  1   1   0
12   1036172  2  1  1  1  2    1  2  1   1   0
13   1041801  5  3  3  3  2    3  4  4   1   1
14   1043999  1  1  1  1  2    3  3  1   1   0
15   1044572  8  7  5 10  7    9  5  5   4   1
16   1047630  7  4  6  4  6    1  4  3   1   1
17   1048672  4  1  1  1  2    1  2  1   1   0
18   1049815  4  1  1  1  2    1  3  1   1   0
19   1050670 10  7  7  6  4   10  4  1   2   1
20   1050718  6  1  1  1  2    1  3  1   1   0
21   1054590  7  3  2 10  5   10  5  4   4   1
22   1054593 10  5  5  3  6    7  7 10   1   1
23   1056784  3  1  1  1  2    1  2  1   1   0
24   1057013  8  4  5  1  2 <NA>  7  3   1   1
```

This is the top snip of the output of the data table and I can see the values of V11 have been changed.

```
#Changing the data into a dplyr table.
Wis_table <- tbl_df(Wis_data)

#Creating a function to find the rows with blank values
Find_Blank <- function(dataframe, column){
  temp_table <- filter(dataframe, is.na(dataframe[column]))
  rows <- nrow(temp_table)
  return(rows)
}

#Creating a table with the blank rows
Blank_table <- tbl_df(colnames(Wis_table))

#Running through the whole table to find all the values
for (i in 1: nrow(Blank_table)) {
  Blank_table[i,2] <- Find_Blank(Wis_table,i)
}

#Finding the blank values
Blank_column <- filter(Blank_table, Blank_table[2]>0)

#The output of the filter
Blank_column
```

```
# A tibble: 1 x 2
  value    V2
  <chr> <int>
1 V7       16
```

Here we can see the blank values are from column V7 and there are 16 blank values.

```
#Calculating the mean
Wis_dataone <- filter(Wis_data, !is.na(Wis_data$V7))
V7_mean <- mean(as.numeric(Wis_dataone[["V7"]]), rm.na=TRUE)
V7_mean
```

```
[1] 3.216691
```

I had to create a temporary table of values to find the mean value for V7 without the NA's. I decided to use the mean because I felt the mode may be skewed.

```
#Filling in the blank values with the mean
Wis_data$V7[is.na(Wis_data['V7'])] <- V7_mean
Week_10 <- transform(Wis_data, V7 = as.numeric(as.character(V7)))
summary(Week_10)
```

```
V1              V2              V3
 Min.   :   61634   Min.   : 1.000   Min.   : 1.000
 1st Qu.:  870688   1st Qu.: 2.000   1st Qu.: 1.000
 Median : 1171710   Median : 4.000   Median : 1.000
 Mean   : 1071704   Mean   : 4.418   Mean   : 3.134
 3rd Qu.: 1238298   3rd Qu.: 6.000   3rd Qu.: 5.000
 Max.   :13454352   Max.   :10.000   Max.   :10.000

     V4              V5              V6
 Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
 1st Qu.: 1.000   1st Qu.: 1.000   1st Qu.: 2.000
 Median : 1.000   Median : 1.000   Median : 2.000
 Mean   : 3.207   Mean   : 2.807   Mean   : 3.216
 3rd Qu.: 5.000   3rd Qu.: 4.000   3rd Qu.: 4.000
 Max.   :10.000   Max.   :10.000   Max.   :10.000

     V7              V8              V9
 Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
 1st Qu.: 1.000   1st Qu.: 2.000   1st Qu.: 1.000
 Median : 1.000   Median : 3.000   Median : 1.000
 Mean   : 3.545   Mean   : 3.438   Mean   : 2.867
 3rd Qu.: 6.000   3rd Qu.: 5.000   3rd Qu.: 4.000
 Max.   :10.000   Max.   :10.000   Max.   :10.000
 NA's   :16
    V10             V11
 Min.   : 1.000   Min.   :0.0000
 1st Qu.: 1.000   1st Qu.:0.0000
 Median : 1.000   Median :0.0000
 Mean   : 1.589   Mean   :0.3448
 3rd Qu.: 1.000   3rd Qu.:1.0000
 Max.   :10.000   Max.   :1.0000
```

Here are the results of 14.1.1 where I have used the mean and imputed data.

```
#Pulling in the data
Wis_data2 <- read.table("breast-cancer-wisconsin.data.txt", header=FALSE, stringsAsFactors = TRUE, sep=",")

#Changing the V11 value to a binary value
Wis_data2$V11[Wis_data2$V11 == 2] <- 0
Wis_data2$V11[Wis_data2$V11 == 4] <- 1
Wis_data2[Wis_data2 == '?'] <- ''

Wis_data2
```

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11
1	1000025	5	1	1	1	2	1	3	1	1	0
2	1002945	5	4	4	5	7	10	3	2	1	0
3	1015425	3	1	1	1	2	2	3	1	1	0
4	1016277	6	8	8	1	3	4	3	7	1	0
5	1017023	4	1	1	3	2	1	3	1	1	0
6	1017122	8	10	10	8	7	10	9	7	1	1
7	1018099	1	1	1	1	2	10	3	1	1	0
8	1018561	2	1	2	1	2	1	3	1	1	0
9	1033078	2	1	1	1	2	1	1	1	5	0
10	1033078	4	2	1	1	2	1	2	1	1	0
11	1035283	1	1	1	1	1	1	3	1	1	0
12	1036172	2	1	1	1	2	1	2	1	1	0
13	1041801	5	3	3	3	2	3	4	4	1	1
14	1043999	1	1	1	1	2	3	3	1	1	0
15	1044572	8	7	5	10	7	9	5	5	4	1
16	1047630	7	4	6	4	6	1	4	3	1	1
17	1048672	4	1	1	1	2	1	2	1	1	0
18	1049815	4	1	1	1	2	1	3	1	1	0
19	1050670	10	7	7	6	4	10	4	1	2	1
20	1050718	6	1	1	1	2	1	3	1	1	0
21	1054590	7	3	2	10	5	10	5	4	4	1
22	1054593	10	5	5	3	6	7	7	10	1	1
23	1056784	3	1	1	1	2	1	2	1	1	0
24	1057013	8	4	5	1	2	<NA>	7	3	1	1

Pulling the data for #2 of the prompt.

```
#Filtering the data between na's and non-na's
Data_na <- filter(Wis_data2, is.na(Wis_data2$V7))
Data_no_na <- filter(Wis_data2, !is.na(Wis_data2$V7))

#Imputing the data
Data_imputation <- lm(as.numeric(V7) ~ V2+V3+V4+V5+V6+V8+V9+V10,
                      Data_no_na)
summary(Data_imputation)
```

```
Call:
lm(formula = as.numeric(V7) ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 +
   V10, data = Data_no_na)

Residuals:
   Min     1Q  Median     3Q    Max
-4.1137 -0.7185 -0.4731 -0.2994  7.3848

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.862817   0.162497  11.464  < 2e-16 ***
V2           0.068118   0.034746   1.960  0.05035 .
V3           0.087939   0.063482   1.385  0.16643
V4           0.110046   0.061190   1.798  0.07255 .
V5          -0.076950   0.038270  -2.011  0.04475 *
V6           0.043216   0.052123   0.829  0.40733
V8           0.044536   0.049211   0.905  0.36579
V9           0.119422   0.037076   3.221  0.00134 **
V10          0.001405   0.049448   0.028  0.97733
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.896 on 674 degrees of freedom
Multiple R-squared:  0.2326,      Adjusted R-squared:  0.2235
F-statistic: 25.54 on 8 and 674 DF,  p-value: < 2.2e-16
```

This is a summary of the imputated data. I am using the data set with no NAs.

```
#Using a backward factor selection
step(Data_imputation, direction="backward")
```

```
- V3    1     9.964 2434.4 880.06
- V4    1    12.613 2437.0 880.80
- V5    1    13.821 2438.2 881.14
- V2    1    14.269 2438.7 881.27
- V9    1    41.469 2465.9 888.84

Step:  AIC=878.14
as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9

       Df Sum of Sq    RSS    AIC
<none>              2427.6 878.14
- V5    1    11.502 2439.1 879.37
- V3    1    12.764 2440.3 879.72
- V4    1    13.847 2441.4 880.03
- V2    1    15.460 2443.0 880.48
- V9    1    47.920 2475.5 889.49

Call:
lm(formula = as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9, data = D
ata_no_na)

Coefficients:
(Intercept)           V2           V3           V4
    1.96962      0.07169      0.11320      0.11926
         V5           V9
   -0.06574      0.13053
```

This is a snippet of the important part of the factor selection which is the remaining factors left of the selection.

```
#Creating a lm based on the backwards factor selection
Data_step <- lm(as.numeric(V7)~ V2+V3+V4+V5+V9, data=Data_no_na)
summary(Data_step)
```

```
Call:
lm(formula = as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9, data = D
ata_no_na)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0534 -0.7407 -0.4819 -0.3385  7.3673

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.96962    0.13706  14.370  < 2e-16 ***
V2           0.07169    0.03453   2.076 0.038230 *
V3           0.11320    0.06000   1.887 0.059628 .
V4           0.11926    0.06069   1.965 0.049810 *
V5          -0.06574    0.03671  -1.791 0.073735 .
V9           0.13053    0.03570   3.656 0.000276 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.894 on 677 degrees of freedom
Multiple R-squared:  0.2308,    Adjusted R-squared:  0.2251
F-statistic: 40.63 on 5 and 677 DF,  p-value: < 2.2e-16
```

This is the snapshot of the summary of the lm based on the backward selection results.

```
#Creating the prediction on the V7 value and convert to integer
V7 <- data.frame(round(predict(Data_step, Data_na)))
colnames(V7) <- c("V7")

#Impute the data together
Data_na <- cbind(Data_na[,1:6], V7, Data_na[,8:11])

#Combine all the information together
Data_imputed2 <- rbind(Wis_data2[,1:11], Data_no_na[,1:11])
Data_imputed2 <- transform(Data_imputed2, V7 = as.numeric(V7))
summary(Data_imputed2)
```

```
       V1                V2              V3
 Min.   :   61634   Min.   : 1.00   Min.   : 1.000
 1st Qu.:  873549   1st Qu.: 2.00   1st Qu.: 1.000
 Median : 1171710   Median : 4.00   Median : 1.000
 Mean   : 1074183   Mean   : 4.43   Mean   : 3.143
 3rd Qu.: 1238450   3rd Qu.: 6.00   3rd Qu.: 5.000
 Max.   :13454352   Max.   :10.00   Max.   :10.000

       V4              V5              V6
 Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
 1st Qu.: 1.000   1st Qu.: 1.000   1st Qu.: 2.000
 Median : 1.000   Median : 1.000   Median : 2.000
 Mean   : 3.211   Mean   : 2.818   Mean   : 3.225
 3rd Qu.: 5.000   3rd Qu.: 4.000   3rd Qu.: 4.000
 Max.   :10.000   Max.   :10.000   Max.   :10.000

       V7              V8              V9
 Min.   : 2.000   Min.   : 1.000   Min.   : 1.000
 1st Qu.: 2.000   1st Qu.: 2.000   1st Qu.: 1.000
 Median : 2.000   Median : 3.000   Median : 1.000
 Mean   : 3.217   Mean   : 3.441   Mean   : 2.868
 3rd Qu.: 3.000   3rd Qu.: 5.000   3rd Qu.: 4.000
 Max.   :11.000   Max.   :10.000   Max.   :10.000
 NA's   :16
      V10             V11
 Min.   : 1.000   Min.   :0.0000
 1st Qu.: 1.000   1st Qu.:0.0000
 Median : 1.000   Median :0.0000
 Mean   : 1.596   Mean   :0.3473
 3rd Qu.: 1.000   3rd Qu.:1.0000
 Max.   :10.000   Max.   :1.0000
```

This is the results of 14.1.2 solutions where regression have been used to impute the data.

```
#Pulling in the data
Wis_data3 <- read.table("breast-cancer-wisconsin.data.txt", header=FALSE, stringsAsFactors = TRUE, sep=",")

#Changing the V11 value to a binary value
Wis_data3$V11[Wis_data3$V11 == 2] <- 0
Wis_data3$V11[Wis_data3$V11 == 4] <- 1
Wis_data3[Wis_data3 == '?'] <- ''

Wis_data3
```

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11
1	1000025	5	1	1	1	2	1	3	1	1	0
2	1002945	5	4	4	5	7	10	3	2	1	0
3	1015425	3	1	1	1	2	2	3	1	1	0
4	1016277	6	8	8	1	3	4	3	7	1	0
5	1017023	4	1	1	3	2	1	3	1	1	0
6	1017122	8	10	10	8	7	10	9	7	1	1
7	1018099	1	1	1	1	2	10	3	1	1	0
8	1018561	2	1	2	1	2	1	3	1	1	0
9	1033078	2	1	1	1	2	1	1	1	5	0
10	1033078	4	2	1	1	2	1	2	1	1	0
11	1035283	1	1	1	1	1	1	3	1	1	0
12	1036172	2	1	1	1	2	1	2	1	1	0
13	1041801	5	3	3	3	2	3	4	4	1	1
14	1043999	1	1	1	1	2	3	3	1	1	0
15	1044572	8	7	5	10	7	9	5	5	4	1
16	1047630	7	4	6	4	6	1	4	3	1	1
17	1048672	4	1	1	1	2	1	2	1	1	0
18	1049815	4	1	1	1	2	1	3	1	1	0
19	1050670	10	7	7	6	4	10	4	1	2	1
20	1050718	6	1	1	1	2	1	3	1	1	0
21	1054590	7	3	2	10	5	10	5	4	4	1
22	1054593	10	5	5	3	6	7	7	10	1	1
23	1056784	3	1	1	1	2	1	2	1	1	0
24	1057013	8	4	5	1	2	<NA>	7	3	1	1

This is a snip of the table after the binary values have been inputted into V11.

```
#Filtering the data between na's and non-na's
Data_na2 <- filter(Wis_data3, is.na(Wis_data3$V7))
Data_no_na2 <- filter(Wis_data3, !is.na(Wis_data3$V7))

#Imputing the data
Data_imputation2 <- lm(as.numeric(V7) ~ V2+V3+V4+V5+V6+V8+V9+V10,
                       Data_no_na2)
summary(Data_imputation2)
```

```
Call:
lm(formula = as.numeric(V7) ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 +
    V10, data = Data_no_na2)

Residuals:
    Min      1Q  Median      3Q     Max
-4.1137 -0.7185 -0.4731 -0.2994  7.3848

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.862817   0.162497  11.464  < 2e-16 ***
V2           0.068118   0.034746   1.960  0.05035 .
V3           0.087939   0.063482   1.385  0.16643
V4           0.110046   0.061190   1.798  0.07255 .
V5          -0.076950   0.038270  -2.011  0.04475 *
V6           0.043216   0.052123   0.829  0.40733
V8           0.044536   0.049211   0.905  0.36579
V9           0.119422   0.037076   3.221  0.00134 **
V10          0.001405   0.049448   0.028  0.97733
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.896 on 674 degrees of freedom
Multiple R-squared:  0.2326,      Adjusted R-squared:  0.2235
F-statistic: 25.54 on 8 and 674 DF,  p-value: < 2.2e-16
```

Summary of the imputation and setting up for regression with perturbation.

```
#Using a backward factor selection
step(Data_imputation2, direction="backward")
```

```
- V3    1     9.964 2434.4 880.06
- V4    1    12.613 2437.0 880.80
- V5    1    13.821 2438.2 881.14
- V2    1    14.269 2438.7 881.27
- V9    1    41.469 2465.9 888.84

Step:  AIC=878.14
as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9

       Df Sum of Sq    RSS    AIC
<none>              2427.6 878.14
- V5    1    11.502 2439.1 879.37
- V3    1    12.764 2440.3 879.72
- V4    1    13.847 2441.4 880.03
- V2    1    15.460 2443.0 880.48
- V9    1    47.920 2475.5 889.49

Call:
lm(formula = as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9, data = D
ata_no_na2)

Coefficients:
(Intercept)           V2           V3           V4
    1.96962      0.07169      0.11320      0.11926
         V5           V9
   -0.06574      0.13053
```

This is a snip of the results of the backwards factor selection.

```
#Creating a lm based on the backwards factor selection
Data_step2 <- lm(as.numeric(V7)~ V2+V3+V4+V5+V9, data=Data_no_na2)
summary(Data_step2)

Call:
lm(formula = as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9, data = Data_no_na2)

Residuals:
   Min     1Q  Median     3Q    Max
-4.0534 -0.7407 -0.4819 -0.3385  7.3673

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.96962    0.13706  14.370  < 2e-16 ***
V2           0.07169    0.03453   2.076 0.038230 *
V3           0.11320    0.06000   1.887 0.059628 .
V4           0.11926    0.06069   1.965 0.049810 *
V5          -0.06574    0.03671  -1.791 0.073735 .
V9           0.13053    0.03570   3.656 0.000276 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.894 on 677 degrees of freedom
Multiple R-squared:  0.2308,      Adjusted R-squared:  0.2251
F-statistic: 40.63 on 5 and 677 DF,  p-value: < 2.2e-16
```

Creating the lm values based on the backwards factor selection above.

```
#Cross validate for linear regression
Data_step_cv <-cv.lm(Data_no_na2, Data_step2)

#creating the na values
V7_value <- data.frame(predict(Data_step2, Data_na2))

#Creating the normal distribution
Data_norm <- data.frame(rnorm(nrow(V7_value), mean=0, sd=1))

#Add perturbation to the new V7 values
Data_preturb <- data.frame(round(V7_value[,1] + Data_norm[,1]))
colnames(Data_preturb) <- c("V7")

#Combine the columns together
Data_na2 <- cbind(Data_na2[,1:6], Data_preturb, Data_na2[,8:11])

#Combine the data to create the imputed data
Data_imputed3 <- rbind(Data_na2[,1:11], Data_no_na2[,1:11])
Data_imputed3 <- transform(Data_imputed3, V7=as.numeric(as.character(V7)))
summary(Data_imputed3)
```

```
     V1                V2              V3
 Min.   :   61634   Min.   : 1.000   Min.   : 1.000
 1st Qu.:  870688   1st Qu.: 2.000   1st Qu.: 1.000
 Median : 1171710   Median : 4.000   Median : 1.000
 Mean   : 1071704   Mean   : 4.418   Mean   : 3.134
 3rd Qu.: 1238298   3rd Qu.: 6.000   3rd Qu.: 5.000
 Max.   :13454352   Max.   :10.000   Max.   :10.000
     V4              V5              V6
 Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
 1st Qu.: 1.000   1st Qu.: 1.000   1st Qu.: 2.000
 Median : 1.000   Median : 1.000   Median : 2.000
 Mean   : 3.207   Mean   : 2.807   Mean   : 3.216
 3rd Qu.: 5.000   3rd Qu.: 4.000   3rd Qu.: 4.000
 Max.   :10.000   Max.   :10.000   Max.   :10.000
     V7              V8              V9
 Min.   : 0.000   Min.   : 1.000   Min.   : 1.000
 1st Qu.: 1.000   1st Qu.: 2.000   1st Qu.: 1.000
 Median : 1.000   Median : 3.000   Median : 1.000
 Mean   : 3.515   Mean   : 3.438   Mean   : 2.867
 3rd Qu.: 5.000   3rd Qu.: 5.000   3rd Qu.: 4.000
 Max.   :10.000   Max.   :10.000   Max.   :10.000
    V10             V11
 Min.   : 1.000   Min.   :0.0000
 1st Qu.: 1.000   1st Qu.:0.0000
 Median : 1.000   Median :0.0000
 Mean   : 1.589   Mean   :0.3448
 3rd Qu.: 1.000   3rd Qu.:1.0000
 Max.   :10.000   Max.   :1.0000
```

This is the summary and results for the regression with perturbation.

```
#Fit the 1 - 10 scale
Data_imputed3$V7[Data_imputed3$V7 == 0] <- 1
summary(Data_imputed3)
```

```
      V1              V2            V3
 Min.   :   61634   Min.   : 1.000   Min.   : 1.000
 1st Qu.:  870688   1st Qu.: 2.000   1st Qu.: 1.000
 Median : 1171710   Median : 4.000   Median : 1.000
 Mean   : 1071704   Mean   : 4.418   Mean   : 3.134
 3rd Qu.: 1238298   3rd Qu.: 6.000   3rd Qu.: 5.000
 Max.   :13454352   Max.   :10.000   Max.   :10.000
      V4            V5            V6
 Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
 1st Qu.: 1.000   1st Qu.: 1.000   1st Qu.: 2.000
 Median : 1.000   Median : 1.000   Median : 2.000
 Mean   : 3.207   Mean   : 2.807   Mean   : 3.216
 3rd Qu.: 5.000   3rd Qu.: 4.000   3rd Qu.: 4.000
 Max.   :10.000   Max.   :10.000   Max.   :10.000
      V7            V8            V9
 Min.   : 1.000   Min.   : 1.000   Min.   : 1.000
 1st Qu.: 1.000   1st Qu.: 2.000   1st Qu.: 1.000
 Median : 1.000   Median : 3.000   Median : 1.000
 Mean   : 3.519   Mean   : 3.438   Mean   : 2.867
 3rd Qu.: 5.000   3rd Qu.: 5.000   3rd Qu.: 4.000
 Max.   :10.000   Max.   :10.000   Max.   :10.000
    V10           V11
 Min.   : 1.000   Min.   :0.0000
 1st Qu.: 1.000   1st Qu.:0.0000
 Median : 1.000   Median :0.0000
 Mean   : 1.589   Mean   :0.3448
 3rd Qu.: 1.000   3rd Qu.:1.0000
 Max.   :10.000   Max.   :1.0000
```

Fit the data to scale so that it won't fall outside the normal range.

Compare Inputed Datasets with the SVM model for Error Rate

```
#Using Model 14.1.1
Week_10_matrix <- data.matrix(Week_10)

#Create the ksvm model
model1_ksvm <- ksvm(
  Week_10_matrix[,1:10], Week_10_matrix[,11], kernel="vanilladot"
  , type="C-svc", C=100, scaled = TRUE
)

#Output the Error ratio
model1_ksvm@error
```

```
[1] 0.02781845
```

For this optional part of the assignment, I am using support vector machine to determine the accuracy and reliability of the model. Model1 is using the mean to impute the missing values.

```
#Using Model 14.1.2
Data_imputed2_matrix <- data.matrix(Data_imputed2)

#Create the ksvm model
model2_ksvm <- ksvm(
  Data_imputed2_matrix[,1:10], Data_imputed2_matrix[,11], kernel="vanilladot"
  , type="C-svc", C=100, scaled = TRUE
)

#Output the Error ratio
model2_ksvm@error
```

```
[1] 0.03074671
```

This is the error from using Regression to impute the values.

```
#Using Model 14.1.3
Data_imputed3_matrix <- data.matrix(Data_imputed3)
#Create the ksvm model
model3_ksvm <- ksvm(
  Data_imputed3_matrix[,1:10], Data_imputed3_matrix[,11], kernel="vanilladot"
  , type="C-svc", C=100, scaled = TRUE
)
#Output the Error ratio
model3_ksvm@error
```

```
[1] 0.03004292
```

This is the error from using regression with perturbation to impute the missing values.

```
#Using the derived data from 14.1.2 solutions
Data_no_na_matrix <- data.matrix(Data_no_na2)

#Creating the ksvm for data with no NA's
model4_ksvm <- ksvm(
  Data_no_na_matrix[,1:10], Data_no_na_matrix[,11], kernel="vanilladot"
  , type="C-svc", C=100, scaled = TRUE
)

#Output the Error ratio
model4_ksvm@error
```

```
[1] 0.03074671
```

This is the error ratio for just using rows with values in V7.

```
#Calling in the table
Wis_binaryd <- read.table("breast-cancer-wisconsin.data.txt", header=FALSE, stringsAsFactors = TRUE, sep=",")

#Creating the binary values
Wis_binaryd$V11[Wis_binaryd$V11 == 2] <- 0
Wis_binaryd$V11[Wis_binaryd$V11 == 4] <- 1
Wis_binaryd[Wis_binaryd == '?'] <- ''

#Creating the filtered views
Data_na <- filter(Wis_binaryd, is.na(Wis_binaryd$V7))
Data_no_na <- filter(Wis_binaryd, !is.na(Wis_binaryd$V7))

#Adding in the binary values 0 for NA and 1 for V7 with values
Data_na_bin <- cbind(0,Data_na[,1:11])
Data_no_na_bin <- cbind(1,Data_no_na[,1:11])

#Renaming the columns
names(Data_na_bin) <- c("Bin","V1","V2","V3","V4","V5","V6","V7","V8","V9","V10","V11")
names(Data_no_na_bin) <- c("Bin","V1","V2","V3","V4","V5","V6","V7","V8","V9","V10","V11")

#combining the tables by rows
Data_comple_bin <- rbind(Data_na_bin,Data_no_na_bin)
head(Data_comple_bin)
```

```
  Bin      V1 V2 V3 V4 V5 V6   V7 V8 V9 V10 V11
1   0 1057013  8  4  5  1  2 <NA>  7  3   1   1
2   0 1096800  6  6  6  9  6 <NA>  7  8   1   0
3   0 1183246  1  1  1  1  1 <NA>  2  1   1   0
4   0 1184840  1  1  3  1  2 <NA>  2  1   1   0
5   0 1193683  1  1  2  1  3 <NA>  1  1   1   0
6   0 1197510  5  1  1  1  2 <NA>  3  1   1   0
```

A snippet of the data to show that the binary value have been added.

```
#Converting the table into a matrix
Data_bin_matrix <- data.matrix(Data_comple_bin)

#Building the ksvm model
model5_ksvm <- ksvm(
  Data_bin_matrix[,1:11], Data_bin_matrix[,12], kernel="vanilladot"
  , type="C-svc", C=100, scaled = TRUE
)

#Output the error
model5_ksvm@error
```

```
[1] 0.3499268
```

This is the value for when a binary value is introduced into the table.

SVM Comparison Conclusion

Introducing a binary value (regression with perturbation) produced the highest error rate (34.99%) out of all the alternatives with is surprising when my initial thinking is that the binary value would help predict the value in V11. The model that produced the lowest error rate (2.78%) was when I use the mean to impute the missing values in V7. There is a high discrepancy between these two models which leads me to believe that I should continue to use a different model and comparison tool. My fear about Model 1 (using the mean) is that it is overfitting the data. The other models being close in results to Model 1 leads the results to be inconclusive.