Question 1.0: Setup and Identification of Fields with Missing Data

Setup

Clear the environment, set the seed, load dplyr lib, and create a function to allow for easy reload/update of dataset. I created the load_data function since I'll want to start with the original dataset for each question part.

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

# Comment in set.seed(33) to repeat results
set.seed(33)

# Load dplyr lib
require(dplyr)

# Create function to re-load data, since we'll want to start with a fresh
dataset for each part
load_data <- function() {
  # Load cancer data into a data frame
  data_df <- read.table("breast-cancer-wisconsin.data.txt", header=FALSE,
sep=",", stringsAsFactors = TRUE)

  # Update V11 (response) field from 2/4 to 0/1
  data_df$V11[data_df$V11 == 2] <- 0
  data_df$V11[data_df$V11 == 4] <- 1

  # Replace ? with NA in data_df
  data_df[data_df=='?'] <- ''

  # Return data_df
  return(data_df)
}
```

Identification of Fields with Missing Data

To identify which fields had missing data, I created a function, Find_Columns_with_Missing(), that filters the data_tbl and counts the number of missing rows. I identified that column V7 was the only column missing data and had 16 obs. Missing.

```
# Load data into a data frame
data_df <- load_data()

# Change data_df into dplyr table
data_tbl <- tbl_df(data_df)
```

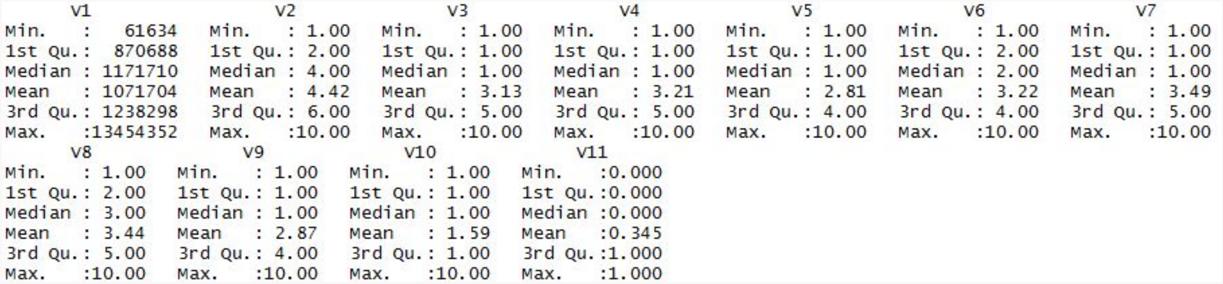
```r
# Function to identify columns with missing data
Find_Columns_with_Missing <- function(table, column) {
  filtered_tbl <- filter(table, is.na(table[column]))
  records <- nrow(filtered_tbl)

  return(records)
}

# Create placeholder for Find_Columns_with_Missing results
missing_tbl <- tbl_df(colnames(data_tbl))

# Loop through each column in data_tbl
for (i in 1:nrow(missing_tbl)) {
  missing_tbl[i,2] <- Find_Columns_with_Missing(data_tbl, i)
}

# Filter to only show columns with missing variables
cols_w_na_data <- filter(missing_tbl, missing_tbl[2]>0)
# V7 has 16 missing values
cols_w_na_data
# A tibble: 1 x 2
  value    V2
  <chr> <int>
1    V7    16
```

Question 1.1: Imputing Using Mode

For part 1, I decided to impute values using mode because the factors were ordinal. I used a mode function (courtesy of Ken Williams) and replaced the missing values for V7 with the mode.

```r
# Load data into a data frame
data_df <- load_data()
# Function to calculate the mode
# Source: https://stackoverflow.com/users/169947/ken-williams
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

# Impute nulls with mode (due to ordinal scale of bare_nuclei)
data_df$V7[is.na(data_df[,'V7'])] <- Mode(data_df[,'V7'])
data_df <- transform(data_df, V7 = as.numeric(as.character(V7)))
```

```
summary(data_df)
```



```
       V1                 V2               V3               V4               V5               V6               V7
 Min.   :   61634   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00
 1st Qu.:  870688   1st Qu.: 2.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 2.00   1st Qu.: 1.00
 Median : 1171710   Median : 4.00   Median : 1.00   Median : 1.00   Median : 1.00   Median : 2.00   Median : 1.00
 Mean   : 1071704   Mean   : 4.42   Mean   : 3.13   Mean   : 3.21   Mean   : 2.81   Mean   : 3.22   Mean   : 3.49
 3rd Qu.: 1238298   3rd Qu.: 6.00   3rd Qu.: 5.00   3rd Qu.: 5.00   3rd Qu.: 4.00   3rd Qu.: 4.00   3rd Qu.: 5.00
 Max.   :13454352   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00
       V8              V9              V10              V11
 Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   :0.000
 1st Qu.: 2.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.:0.000
 Median : 3.00   Median : 1.00   Median : 1.00   Median :0.000
 Mean   : 3.44   Mean   : 2.87   Mean   : 1.59   Mean   :0.345
 3rd Qu.: 5.00   3rd Qu.: 4.00   3rd Qu.: 1.00   3rd Qu.:1.000
 Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :1.000
```

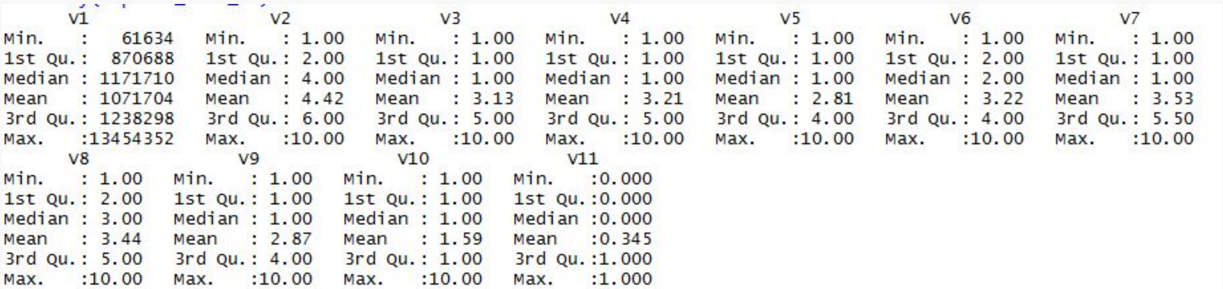
Question 1.2: Impute using Linear Regression

For Part 2, I first split the dataset into one containing all records that had complete data (data_df_wo_na) and one containing all records missing data (data_df_wo_na). Using the data_df_wo_na dataset, I created an imputation_model that used all factors (except V1 (i.e. ID) and V11 (i.e Response)). Using imputation_model, I used step() to perform backward step factor selection. Using the step recommended factors, I retrained the model and used this to predict values for the missing V7s:

```
# Load data into a data frame
data_df <- load_data()

# Splice table into records with/without missing data
data_df_w_na <- filter(data_df, is.na(data_df$V7))
data_df_wo_na <- filter(data_df, !is.na(data_df$V7))

# Create a linear regression model
imputation_model <- lm(as.numeric(V7) ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 +
V10, data_df_wo_na)
summary(imputation_model)
Call:
lm(formula = as.numeric(V7) ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 +
    V10, data = data_df_wo_na)

Residuals:
   Min     1Q Median     3Q    Max
-4.114 -0.718 -0.473 -0.299  7.385

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.86282    0.16250   11.46   <2e-16 ***
V2           0.06812    0.03475    1.96   0.0504 .
V3           0.08794    0.06348    1.39   0.1664
V4           0.11005    0.06119    1.80   0.0726 .
```

```
V5          -0.07695    0.03827   -2.01   0.0448 *
V6           0.04322    0.05212    0.83   0.4073
V8           0.04454    0.04921    0.90   0.3658
V9           0.11942    0.03708    3.22   0.0013 **
V10          0.00141    0.04945    0.03   0.9773
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.9 on 674 degrees of freedom
Multiple R-squared:  0.233,   Adjusted R-squared:  0.224
F-statistic: 25.5 on 8 and 674 DF,  p-value: <2e-16

# Use stepwise for factor selection
step(imputation_model, direction = "backward")
Step:  AIC=878
as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9

       Df Sum of Sq  RSS AIC
<none>              2428 878
- V5    1      11.5 2439 879
- V3    1      12.8 2440 880
- V4    1      13.8 2441 880
- V2    1      15.5 2443 880
- V9    1      47.9 2475 889

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

Coefficients:
(Intercept)           V2           V3           V4           V5
V9
     1.9696       0.0717       0.1132       0.1193      -0.0657
0.1305

# Re-train the linear regression using stepwise recommended factors
step_model <- lm(as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9, data_df_wo_na)
summary(step_model)
Call:
lm(formula = as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9, data = data_df_wo_na)

Residuals:
   Min     1Q Median     3Q    Max
```

```
-4.053 -0.741 -0.482 -0.339  7.367

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.9696     0.1371   14.37  < 2e-16 ***
V2            0.0717     0.0345    2.08  0.03823 *
V3            0.1132     0.0600    1.89  0.05963 .
V4            0.1193     0.0607    1.97  0.04981 *
V5           -0.0657     0.0367   -1.79  0.07374 .
V9            0.1305     0.0357    3.66  0.00028 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.89 on 677 degrees of freedom
Multiple R-squared:  0.231,   Adjusted R-squared:  0.225
F-statistic: 40.6 on 5 and 677 DF,  p-value: <2e-16

# Predict values for V7 and round to convert to integers
V7 <- data.frame(round(predict(step_model, data_df_w_na)))
colnames(V7) <- c("V7")

# Impute the predictions to data_df_w_na
data_df_w_na <- cbind(data_df_w_na[,1:6], V7, data_df_w_na[,8:11])

# Combine data_df_w_na and data_df_wo_na into imputed_data_df
imputed_data_df <- rbind(data_df_w_na[,1:11], data_df_wo_na[,1:11])
imputed_data_df <- transform(imputed_data_df, V7 = as.numeric(V7))
summary(imputed_data_df)
```

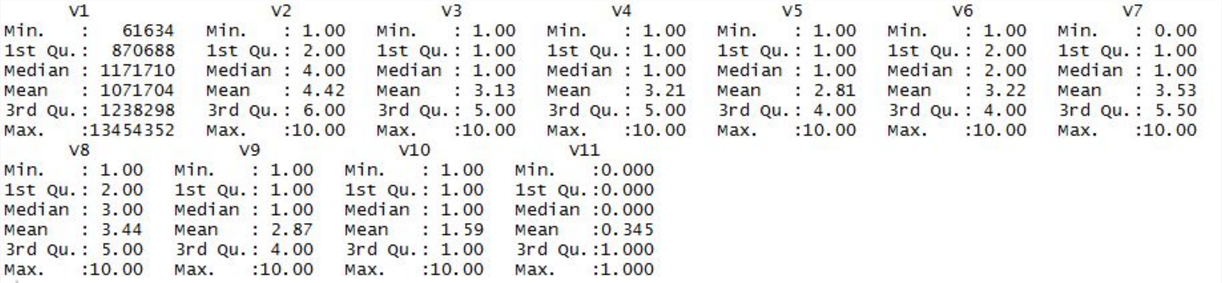


```
       V1                V2               V3               V4               V5               V6               V7
Min.   :   61634   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00
1st Qu.:  870688   1st Qu.: 2.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 2.00   1st Qu.: 1.00
Median : 1171710   Median : 4.00   Median : 1.00   Median : 1.00   Median : 1.00   Median : 2.00   Median : 1.00
Mean   : 1071704   Mean   : 4.42   Mean   : 3.13   Mean   : 3.21   Mean   : 2.81   Mean   : 3.22   Mean   : 3.53
3rd Qu.: 1238298   3rd Qu.: 6.00   3rd Qu.: 5.00   3rd Qu.: 5.00   3rd Qu.: 4.00   3rd Qu.: 4.00   3rd Qu.: 5.50
Max.   :13454352   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00
       V8               V9               V10              V11
Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   :0.000
1st Qu.: 2.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.:0.000
Median : 3.00   Median : 1.00   Median : 1.00   Median :0.000
Mean   : 3.44   Mean   : 2.87   Mean   : 1.59   Mean   :0.345
3rd Qu.: 5.00   3rd Qu.: 4.00   3rd Qu.: 1.00   3rd Qu.:1.000
Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :1.000
```

Question 1.3: Impute with Regression & Perturbation
The Part 3 process was similar to Part 2 but with the additional of creating a normal distribution of values and adding them to the predicted V7 values to create the perturbed V7 values. Initially the perturbed results ranged 0:10 which was outside the initial range of 1:10; therefore, I updated the 0 values to 1. Part 3 specific code has been bolded:

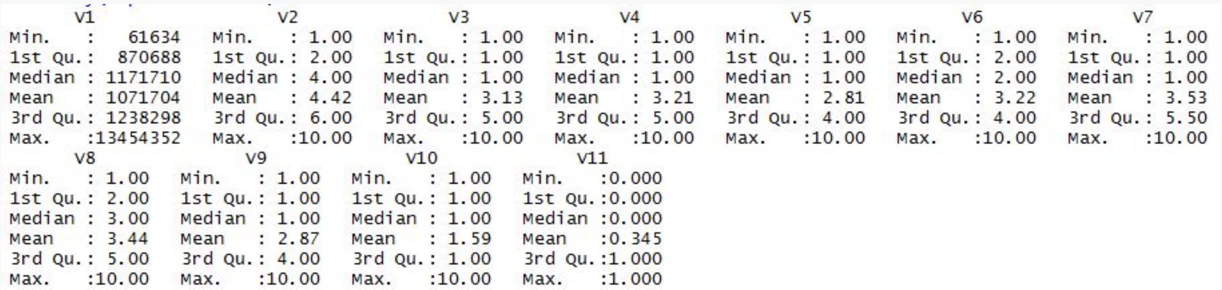
```
# Load data into a data frame
data_df <- load_data()

# Splice table into records with/without missing data
data_df_w_na <- filter(data_df, is.na(data_df$V7))
data_df_wo_na <- filter(data_df, !is.na(data_df$V7))

# Create a linear regression model
imputation_model <- lm(as.numeric(V7) ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 +
V10, data_df_wo_na)
summary(imputation_model)

# Use stepwise for factor selection
step(imputation_model, direction = "backward")

# Re-train the linear regression using stepwise recommended factors
step_model <- lm(as.numeric(V7) ~ V2 + V3 + V4 + V5 + V9, data_df_wo_na)

# CV the step_model
cv_step_model <- cv.lm(data_df_wo_na, step_model)

# Predict values for V7
V7 <- data.frame(predict(step_model, data_df_w_na))

# Create a normal distribution for perturbation
normal_dist <- data.frame(rnorm(nrow(V7), mean = 0, sd = 1))

# Add perturbation to predicted V7 values and round
perturbed_V7 <- data.frame(round(V7[,1] + normal_dist[,1]))
colnames(perturbed_V7) <- c("V7")

# Impute the predictions to data_df_w_na
data_df_w_na <- cbind(data_df_w_na[,1:6], perturbed_V7,
data_df_w_na[,8:11])


# Combine data_df_w_na and data_df_wo_na into imputed_data_df
```

```r
imputed_data_df <- rbind(data_df_w_na[,1:11], data_df_wo_na[,1:11])
imputed_data_df <- transform(imputed_data_df, V7 =
as.numeric(as.character(V7)))
summary(imputed_data_df)
```



```
       V1                V2               V3               V4               V5               V6               V7
 Min.   :   61634   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 0.00
 1st Qu.:  870688   1st Qu.: 2.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 2.00   1st Qu.: 1.00
 Median : 1171710   Median : 4.00   Median : 1.00   Median : 1.00   Median : 1.00   Median : 2.00   Median : 1.00
 Mean   : 1071704   Mean   : 4.42   Mean   : 3.13   Mean   : 3.21   Mean   : 2.81   Mean   : 3.22   Mean   : 3.53
 3rd Qu.: 1238298   3rd Qu.: 6.00   3rd Qu.: 5.00   3rd Qu.: 5.00   3rd Qu.: 4.00   3rd Qu.: 4.00   3rd Qu.: 5.50
 Max.   :13454352   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00
       V8               V9              V10              V11
 Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   :0.000
 1st Qu.: 2.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.:0.000
 Median : 3.00   Median : 1.00   Median : 1.00   Median :0.000
 Mean   : 3.44   Mean   : 2.87   Mean   : 1.59   Mean   :0.345
 3rd Qu.: 5.00   3rd Qu.: 4.00   3rd Qu.: 1.00   3rd Qu.:1.000
 Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :1.000
```

```r
# Update min value of V7 to fit 1:10 scale
imputed_data_df$V7[imputed_data_df$V7 == 0] <- 1
summary(imputed_data_df)
```



```
       V1                V2               V3               V4               V5               V6               V7
 Min.   :   61634   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00
 1st Qu.:  870688   1st Qu.: 2.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 2.00   1st Qu.: 1.00
 Median : 1171710   Median : 4.00   Median : 1.00   Median : 1.00   Median : 1.00   Median : 2.00   Median : 1.00
 Mean   : 1071704   Mean   : 4.42   Mean   : 3.13   Mean   : 3.21   Mean   : 2.81   Mean   : 3.22   Mean   : 3.53
 3rd Qu.: 1238298   3rd Qu.: 6.00   3rd Qu.: 5.00   3rd Qu.: 5.00   3rd Qu.: 4.00   3rd Qu.: 4.00   3rd Qu.: 5.50
 Max.   :13454352   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :10.00
       V8               V9              V10              V11
 Min.   : 1.00   Min.   : 1.00   Min.   : 1.00   Min.   :0.000
 1st Qu.: 2.00   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.:0.000
 Median : 3.00   Median : 1.00   Median : 1.00   Median :0.000
 Mean   : 3.44   Mean   : 2.87   Mean   : 1.59   Mean   :0.345
 3rd Qu.: 5.00   3rd Qu.: 4.00   3rd Qu.: 1.00   3rd Qu.:1.000
 Max.   :10.00   Max.   :10.00   Max.   :10.00   Max.   :1.000
```