Thomas Huelsnitz
ISYE 6501
Week 6 Homework

HW06B

Question 14.1

The breast cancer data set 14.1breast-cancer-wisconsin.dataSummer2018.txt has missing values.

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

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

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

4. (Optional) Compare the results and quality of classification models (e.g., SVM, KNN) build using (1) the data sets from questions 1,2,3; (2) the data that remains after data points with missing values are removed; and (3) the data set when a binary variable is introduced to indicate missing values.

Begin by loading the data. Note that only the 7th column of the data set is missing values, so that will be the focus for imputation. Also, R surprisingly seems to not have a built in mode function, so we define one here.

```
bcd_original <- read.table("14.1breast-cancer-wisconsin.dataSummer2018.txt",
header=FALSE, sep=",")
bcd_original[,ncol(bcd_original)] <-
as.factor(bcd_original[,ncol(bcd_original)])

mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

#Column 7 is the only column with missing data. We will focus on it.
colnum <- 7
```

Begin with simply replacing the missing values with the average of the other values in column 7.

```
# Impute with mean value
bcd_mean_impute <- bcd_original
missingVector <- bcd_mean_impute[,colnum] == "?"
bcd_mean_impute[which(missingVector),colnum] <- NA
bcd_mean_impute[,colnum] <- as.numeric(bcd_mean_impute[,colnum])
avgValue <- mean(bcd_mean_impute[,colnum], na.rm=TRUE)
bcd_mean_impute[which(missingVector),colnum] <- avgValue
bcd_mean_impute[which(missingVector),colnum]
```

```
##  [1] 3.216691 3.216691 3.216691 3.216691 3.216691 3.216691 3.216691
##  [8] 3.216691 3.216691 3.216691 3.216691 3.216691 3.216691 3.216691
## [15] 3.216691 3.216691
```

Next with the mode.

```
# Impute with mode value
bcd_mode_impute <- bcd_original
missingVector <- bcd_mode_impute[,colnum] == "?"
bcd_mode_impute[which(missingVector),colnum] <- NA
bcd_mode_impute[,colnum] <- as.numeric(bcd_mode_impute[,colnum])
avgValue <- mode(bcd_mode_impute[,colnum])
bcd_mode_impute[which(missingVector),colnum] <- avgValue
bcd_mode_impute[which(missingVector),colnum]

##  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

The next approach is a bit more sophisticated. We will use a linear regression model where the dependant variables are all of the columns other than column 7 and the response variable (the last column).

```
#Impute with regression
bcd_reg_impute <- bcd_original
missingVector <- bcd_reg_impute[,colnum] == "?"
bcd_reg_impute[which(missingVector),colnum] <- NA
bcd_reg_impute[,colnum] <- as.numeric(bcd_reg_impute[,colnum])
linmodel <- lm(V7~.-V11, bcd_reg_impute)
preds <- predict(linmodel, bcd_reg_impute[which(missingVector),])
bcd_reg_impute[which(missingVector),colnum] <- preds
bcd_reg_impute[which(missingVector),colnum]

##  [1] 3.990889 4.285384 2.300996 2.564637 2.452592 2.656333 2.849674
##  [8] 2.595942 2.754044 5.504548 2.385635 3.420991 4.333062 2.555046
## [15] 2.296994 2.266393
```

Finally we exxtend the regression approach by adding a perturbation factor. The perturbation factor is taken from a normal distribution where the mean and the standard deviation of the normal distribution are the mean and standard deviation of the linear model fit residuals.

```
#Impute with regression + perturbation
bcd_reg_pert_impute <- bcd_original
missingVector <- bcd_reg_pert_impute[,colnum] == "?"
bcd_reg_pert_impute[which(missingVector),colnum] <- NA
bcd_reg_pert_impute[,colnum] <- as.numeric(bcd_reg_pert_impute[,colnum])
linmodel <- lm(V7~.-V11, bcd_reg_pert_impute)
preds <- predict(linmodel, bcd_reg_pert_impute[which(missingVector),])
residualsMean <- mean(summary(linmodel)$residuals)
residualsStd <- sd(summary(linmodel)$residuals)
perturbs <- rnorm(preds, residualsMean, residualsStd)
perturbed_preds <- preds + perturbs
```

```
bcd_reg_pert_impute[which(missingVector),colnum] <- perturbed_preds
bcd_reg_pert_impute[which(missingVector),colnum]

##  [1]  5.4701938  4.6581979  1.5739337  2.7920360 -0.7344594  1.5364108
##  [7]  4.8862987  0.3373842  2.2587536  6.3243256 -1.9986596  2.9279006
## [13]  4.6786683  3.3511059  1.6547919  3.7075744
```

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?

Every day I deal with traffic during my commute to and from work. The timing of traffic lights is an important factor in how long I and other drivers have to wait. The length of time that each light at an intersection is green should be optimized such that the average time required for all drivers passing through the intersection is minimzed. Optimization will depend directly upon the rates of traffic incoming to the intersection and should be subject to some constraints such as maximum allowed red light times. Rates of traffic may depend on the time of day, time of year, the weather conditions, and any major events happening near the intersection (could be sporting events or other road outages).