N14724679
Kunal B. Relia (kbr263)

Question 1

a)

```
Load In & Clean Up

Loading the data
> data<-read.csv(file.choose(),header = T,skip =4,sep=",")
> View(data)
> dataDF<-data.frame(data)

Remove the extra non essential columns 1 & 7
> dataDF<-dataDF[,-7]
> dataDF<-dataDF[,-1]

Change the date into YYYY-mm-dd format for ease of graph plotting
> dataDF[,19]<-as.Date(dataDF[,19],"%d/%m/%Y")

Change the “” & “-“ values in APARTMENT.NUMBER to NA
> library("stringr")
> install.packages("stringr")
> dataDF$APARTMENT.NUMBER[str_trim(dataDF$APARTMENT.NUMBER)==""] <- NA
> dataDF$APARTMENT.NUMBER[str_trim(dataDF$APARTMENT.NUMBER)=="-"] <- NA

Set the 0 values in LAND.SQUARE.FEET & GROSS.SQUARE.FEET to NA
> dataDF$LAND.SQUARE.FEET[dataDF$LAND.SQUARE.FEET==0] <- NA
> dataDF$GROSS.SQUARE.FEET[dataDF$GROSS.SQUARE.FEET==0] <- NA

Find & Convert the non-numeric data into numeric data (as.numeric won’t work
directly so we need to remove the commas first)
> sapply(dataDF,class)
             NEIGHBORHOOD            BUILDING.CLASS.CATEGORY            TAX.CLASS.AT.PRESENT
                 "factor"                           "factor"                        "factor"
                    BLOCK                                LOT       BUILDING.CLASS.AT.PRESENT
                "integer"                          "integer"                        "factor"
                  ADDRESS                   APARTMENT.NUMBER                        ZIP.CODE
                 "factor"                           "factor"                       "integer"
        RESIDENTIAL.UNITS                   COMMERCIAL.UNITS                     TOTAL.UNITS
                "integer"                          "integer"                       "integer"
         LAND.SQUARE.FEET                  GROSS.SQUARE.FEET                      YEAR.BUILT
                 "factor"                           "factor"                       "integer"
TAX.CLASS.AT.TIME.OF.SALE BUILDING.CLASS.AT.TIME.OF.SALE                      SALE.PRICE
                "integer"                           "factor"                        "factor"
                SALE.DATE
                   "Date"
> dataDF[,'LAND.SQUARE.FEET']<-as.numeric(gsub(",","",dataDF[,'LAND.SQUARE.FE
ET']))
> dataDF[,'GROSS.SQUARE.FEET']<-as.numeric(gsub(",","",dataDF[,'GROSS.SQUARE.
FEET']))
> dataDF[,'SALE.PRICE']<-as.numeric(gsub(",","",dataDF[,'SALE.PRICE']))
> sapply(dataDF,class)
             NEIGHBORHOOD            BUILDING.CLASS.CATEGORY            TAX.CLASS.AT.PRESENT
                 "factor"                           "factor"                        "factor"
                    BLOCK                                LOT       BUILDING.CLASS.AT.PRESENT
                "integer"                          "integer"                        "factor"
                  ADDRESS                   APARTMENT.NUMBER                        ZIP.CODE
                 "factor"                           "factor"                       "integer"
        RESIDENTIAL.UNITS                   COMMERCIAL.UNITS                     TOTAL.UNITS
                "integer"                          "integer"                       "integer"
         LAND.SQUARE.FEET                  GROSS.SQUARE.FEET                      YEAR.BUILT
                "numeric"                          "numeric"                       "integer"
TAX.CLASS.AT.TIME.OF.SALE BUILDING.CLASS.AT.TIME.OF.SALE                      SALE.PRICE
```

```
                 "integer"                    "factor"                   "numeric"
                 SALE.DATE
                    "Date"
```

Conclusion:

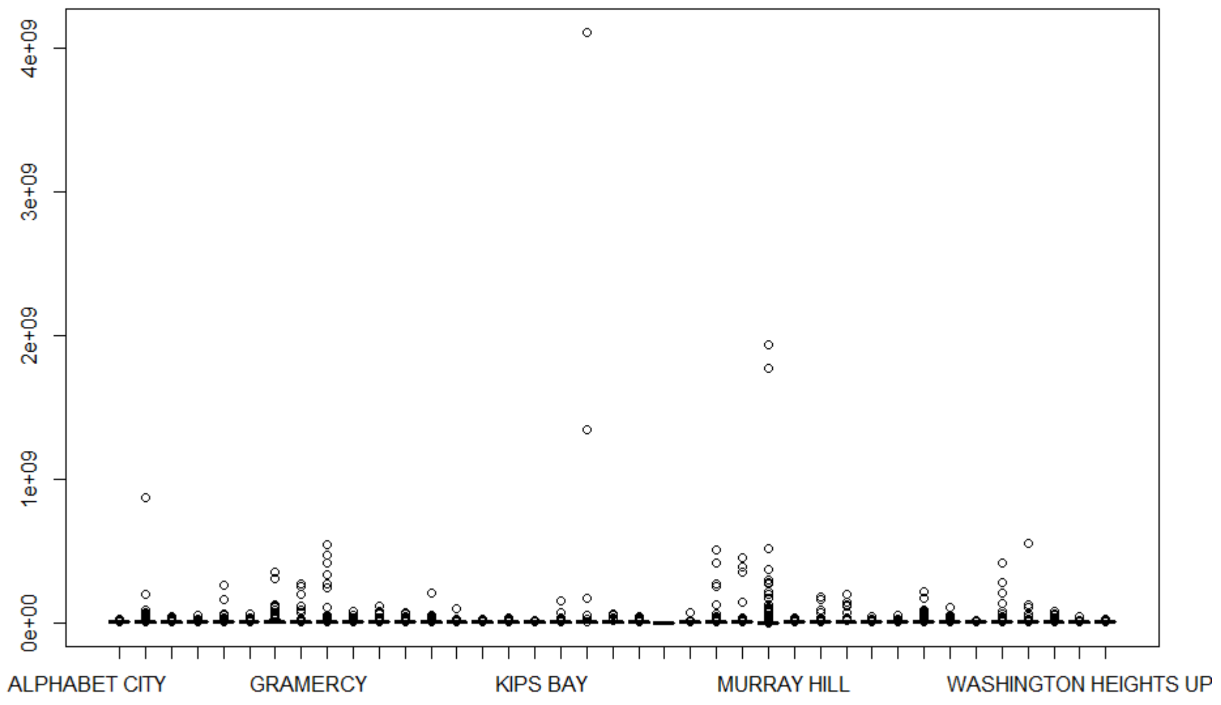
The following changes were analyzed and made during the first stage of data exploration:

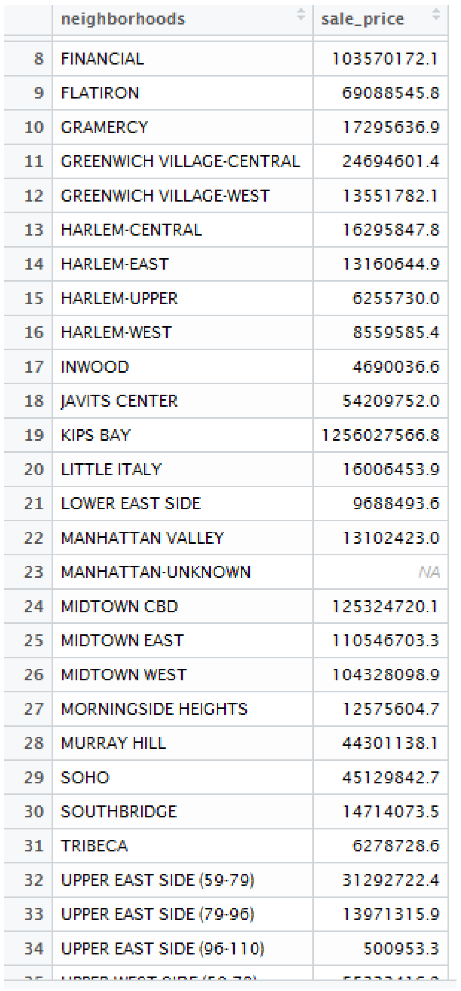
1) All the missing values (for eg, “” in APARTMENT.NUMBER) were set to NA. This not only took care of the missing values but during further analysis we can simple omit such observation from our analysis.
2) Also values that were 0 were set to NA for the same reason.
3) Columns 1 & 7 were removed from the dataset as they had same values and were of no use in our data analysis.
4) Date was formatted to *‘Y-m-d’* for ease of graph plotting.
5) Finally using sapply() function we discovered 3 columns were expected to be numeric but were non-numeric. This was changed too.
6) Outliers like a couple of properties in Kips Bay are identified and considered in the analysis.

b)

i) Analysis of Neighborhood vs Sale Rate

```
> plot(dataDF$NEIGHBORHOOD,dataDF$SALE.PRICE)
>
> numberOfNeighborhoods<-unique(dataDF$NEIGHBORHOOD)
> dataDF.sale<-dataDF[which(dataDF$GROSS.SQUARE.FEET>0 & dataDF$LAND.SQUARE.F
EET>0 & dataDF$SALE.PRICE>0),]
> neighborhoodsDF<-data.frame(neighborhoods=unique(dataDF$NEIGHBORHOOD))
>
> for(i in 1:length(numberOfNeighborhoods)){
+ neighborhoodsDF$sale_price[i]<-sd(dataDF.sale$SALE.PRICE[dataDF.sale$NEIGHB
ORHOOD == neighborhoodsDF$neighborhoods[i]],na.rm=TRUE)
+ }
> View(neighborhoodsDF)
```



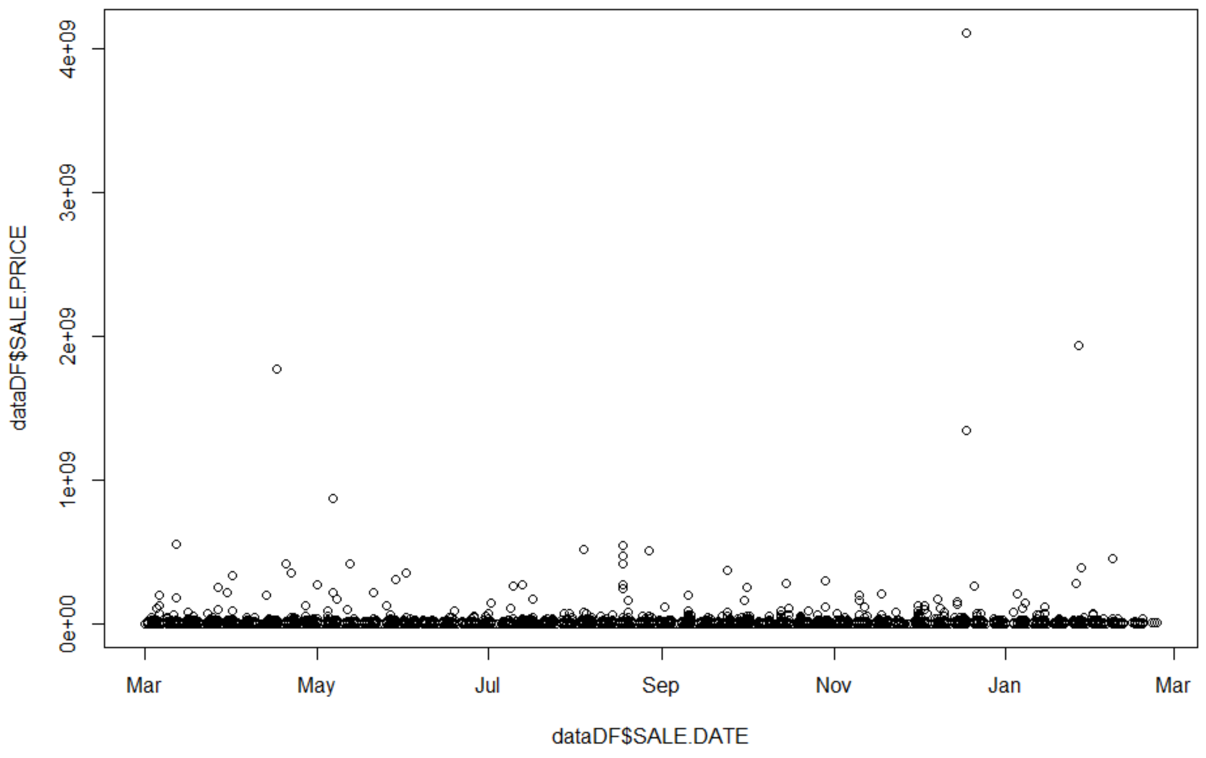


	neighborhoods	sale_price
8	FINANCIAL	103570172.1
9	FLATIRON	69088545.8
10	GRAMERCY	17295636.9
11	GREENWICH VILLAGE-CENTRAL	24694601.4
12	GREENWICH VILLAGE-WEST	13551782.1
13	HARLEM-CENTRAL	16295847.8
14	HARLEM-EAST	13160644.9
15	HARLEM-UPPER	6255730.0
16	HARLEM-WEST	8559585.4
17	INWOOD	4690036.6
18	JAVITS CENTER	54209752.0
19	KIPS BAY	1256027566.8
20	LITTLE ITALY	16006453.9
21	LOWER EAST SIDE	9688493.6
22	MANHATTAN VALLEY	13102423.0
23	MANHATTAN-UNKNOWN	NA
24	MIDTOWN CBD	125324720.1
25	MIDTOWN EAST	110546703.3
26	MIDTOWN WEST	104328098.9
27	MORNINGSIDE HEIGHTS	12575604.7
28	MURRAY HILL	44301138.1
29	SOHO	45129842.7
30	SOUTHBRIDGE	14714073.5
31	TRIBECA	6278728.6
32	UPPER EAST SIDE (59-79)	31292722.4
33	UPPER EAST SIDE (79-96)	13971315.9
34	UPPER EAST SIDE (96-110)	500953.3

Conclusion:

Based on the graph and the mean values, we can say the Midtown CBD neighborhood is the most expensive one. Though Kips Bay has the highest mean, it has a couple of outliers which drag its mean value upwards. Also as the number of properties sold here is less, the mean is affected more by the outlier. Hence, based on the properties sold, we can say the three Midtown areas and the Financial neighborhood is the costliest one. Further based on graph, we can say that Financial neighborhood has no outlier and most of the properties are sold around its mean price.

```
ii) Analysis of Time vs Sale Rate
> plot(dataDF$SALE.DATE,dataDF$SALE.PRICE)
```



Conclusion:

Based on the graph we can see that barring a few outliers, the cost of the property remains nearly the same throughout the year. We don't see any seasonal effect. From March to February, the sale price of properties nearly remain the same.

Question 2

a)

```
> dataDF.sale<-dataDF[which(dataDF$GROSS.SQUARE.FEET>0 & dataDF$LAND.SQUARE.F
EET>0 & dataDF$SALE.PRICE>0),]
> model<-lm(SALE.PRICE~BUILDING.CLASS.CATEGORY+GROSS.SQUARE.FEET+LAND.SQUARE.
FEET+COMMERCIAL.UNITS+RESIDENTIAL.UNITS+NEIGHBORHOOD+YEAR.BUILT,data=dataDF.s
ale)
> summary(model)

Call:
lm(formula = SALE.PRICE ~ BUILDING.CLASS.CATEGORY + GROSS.SQUARE.FEET +
    LAND.SQUARE.FEET + COMMERCIAL.UNITS + RESIDENTIAL.UNITS +
    NEIGHBORHOOD + YEAR.BUILT, data = dataDF.sale)

Residuals:
       Min         1Q     Median         3Q        Max
-505472926   -3706790   -1960861    1654785 1467169159

Coefficients:
                                                                    Estimat
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 54060000 on 1839 degrees of freedom
Multiple R-squared:  0.8068,   Adjusted R-squared:  0.7995
F-statistic: 111.3 on 69 and 1839 DF,  p-value: < 2.2e-16
```

Conclusion:

Regression model is a good choice in this case because of high R-squared values we get. Had we got low R-squared values, this definitely would have been a bad model. Further, high R-Squared value suggest the significance of predictors in determining the value of Sale.Rate.

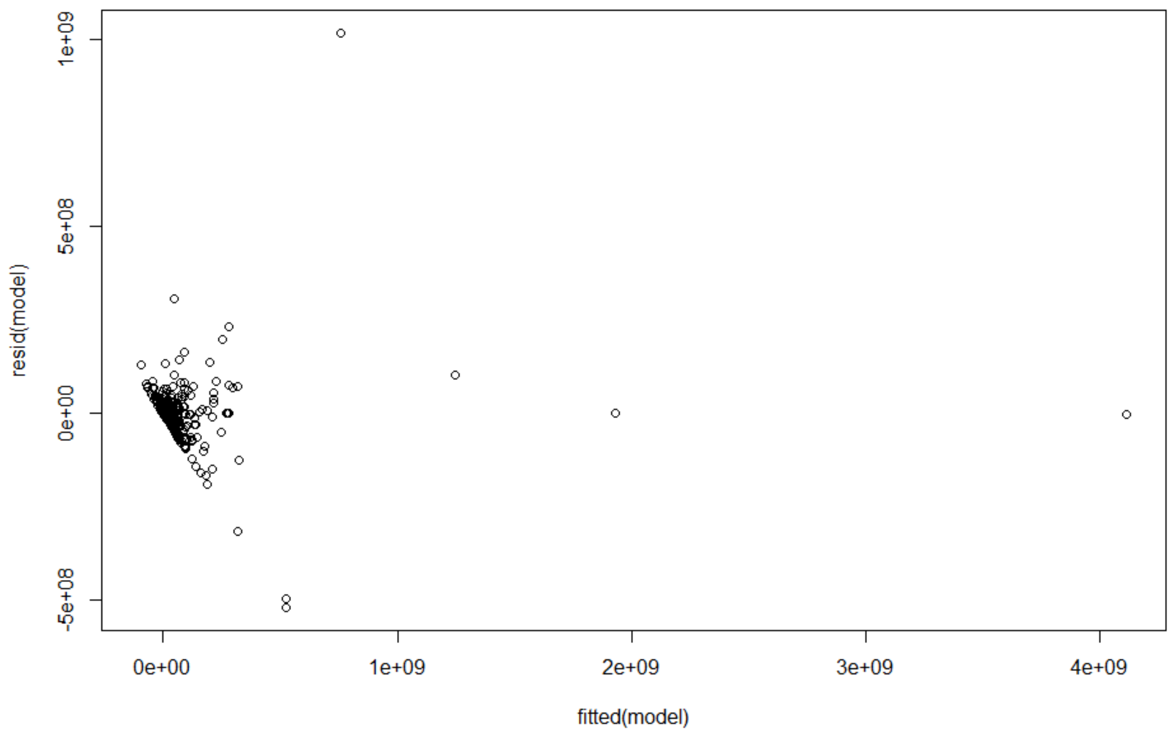
BUILDING.CLASS.CATEGORY+GROSS.SQUARE.FEET+LAND.SQUARE.FEET+COMMERCIAL.UNITS+RESIDENTIAL.UNITS+NEIGHBORHOOD+YEAR.BUILT are all good predictors.

Further this claim logically is also justifiable as a property’s value is definitely determined by factors like year built, its zip code, total units it houses, etc.

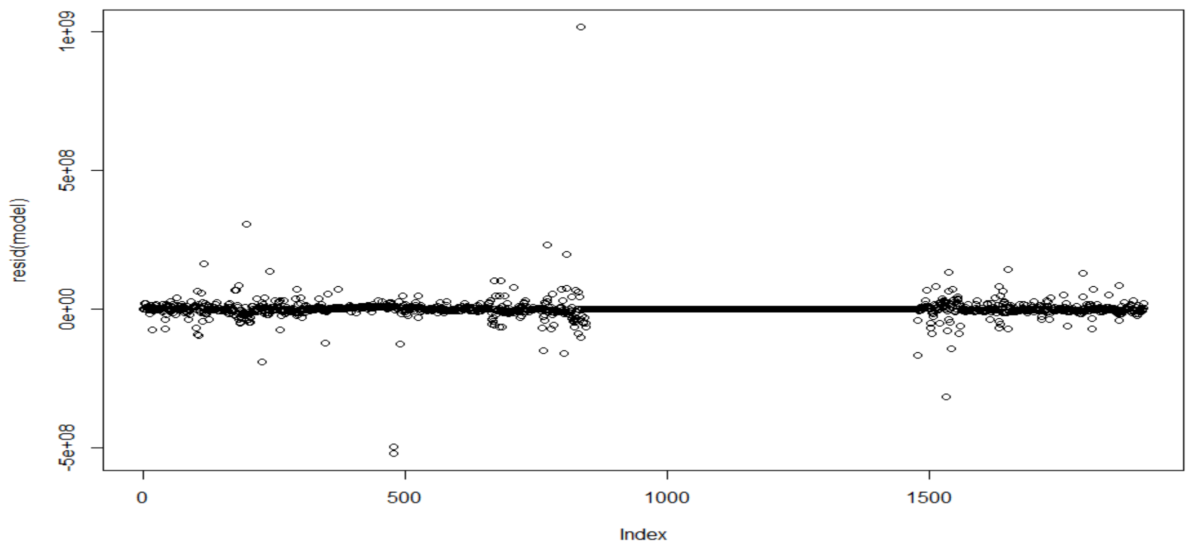
b)

```
> model<-lm(SALE.PRICE~factor(BUILDING.CLASS.CATEGORY)+GROSS.SQUARE.FEET+LAND
.SQUARE.FEET+COMMERCIAL.UNITS+RESIDENTIAL.UNITS+NEIGHBORHOOD+factor(YEAR.BUIL
T),data=dataDF.sale)
> summary(model)
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 38880000 on 1731 degrees of freedom
Multiple R-squared:  0.9059,	Adjusted R-squared:  0.8963
F-statistic: 94.17 on 177 and 1731 DF,  p-value: < 2.2e-16
```

```
> plot(resid(model)~fitted(model))
```



```
> plot(resid(model))
```



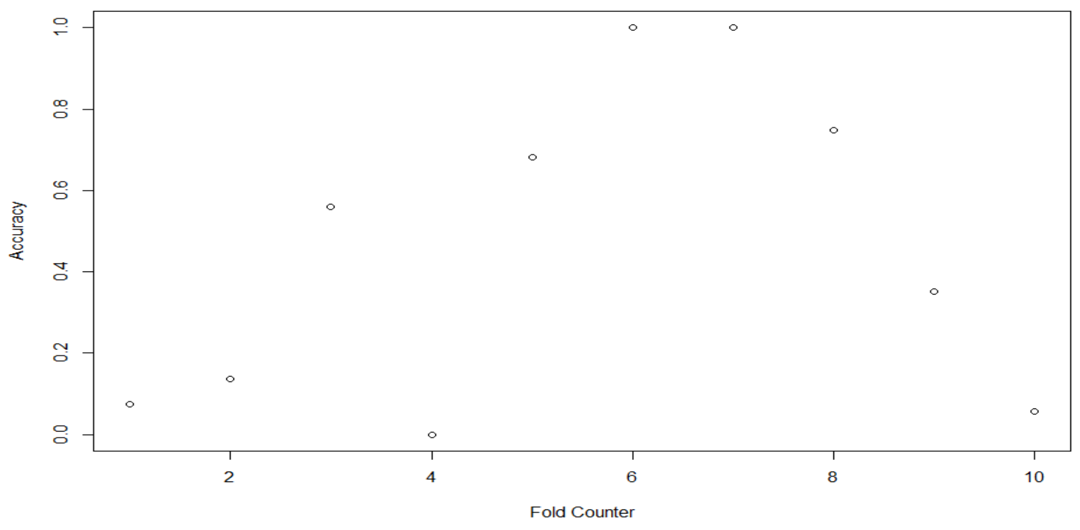
Conclusion:

The plots for Fitted Regression model look good. We know that in the residual graph, the estimated regression line for the model we built is denoted by residual = 0. Hence, we can clearly see that most of the points are around the residual line which is a good sign.

Further when we analyze the residual vs fitted plot, we see that most of the points are in very close proximity. There are only a few outliers (about 4) that can be seen which are far from the regression line.

c) + d) + e)

```
> KNNDF<-data.frame(dataDF.sale[,9])
> KNNDFColl<-data.frame(dataDF.sale$NEIGHBORHOOD)

> folds <- cut(seq(1,nrow(KNNDF)),breaks=10,labels=FALSE)

> accuracy<-integer()
> icol<-integer()

> for(i in 1:10) {
+ icol <- c(icol,i)
+ testIndex<-which(folds==i,arr.ind=TRUE)
+ testData<-KNNDF[testIndex,]
+ trainData<-KNNDF[-testIndex,]
+ trainres<-(KNNDFColl[-testIndex,])
+ testres<-(KNNDFColl[testIndex,])
+ library(e1071)
+ library(class)
+ library(caret)
+ knn_req<-knn(train=data.frame(trainData),test=data.frame(testData),cl=train
res,k=3)
+ getAccuracy<-confusionMatrix(testres,knn_req)
+ accuracy<-c(accuracy,getAccuracy$overall[1])
+ }

> plot(icol,accuracy,xlab="Fold Counter",ylab="Accuracy")

> sum=0

> for(acc in accuracy) {
+ sum=sum+acc
+ }

> print(sum/length(accuracy))
[1] 0.4612565
```



Conclusion:

c)
```
Overall Statistics

               Accuracy : 0.0576
                 95% CI : (0.0291, 0.1007)
    No Information Rate : 0.288
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0536
 Mcnemar's Test P-Value : NA
```

From the above derived graph and confusion matrix, the accuracy levels match and they tend to be about 6%. The low accuracy is not good for our analysis and this is the error. Ideally, the accuracy is expected to be pretty high. But here, as per the confusion matrix, we are getting a very low accuracy.

d) The graph is kind of parabolic in nature. In other words, except for fold 4, it tends to increase from 1 to 5 and then after a peak at folds 6 & 7 it decreases. Hence, this means that values for predictors are likely to be similar for folds 6 & 7 and they will be very different for folds 4 & 10. Thus predictors tend to be repetitive for the purpose of learning the model and such predictor values are not good.

e) Finally we can conclude that Fold 4 & Fold 10 are very bad selections as accuracy tends to be 0. We can conclude that Fold 6 & 7 are the most accurate and hence best suited for this dataset. Hence, action can be taken using Folds 6 & 7 which will provide excellent accuracy. As a complete analysis, we need to improve the overall accuracy as it is not so high.

Question 3

a)

```
> trans<-preProcess(dataDF[,-(13:14)],method=c("BoxCox","center","scale","pca
"))
> attach(trans)
> str(dataDF)
'data.frame':   23757 obs. of  19 variables:
 $ NEIGHBORHOOD                  : Factor w/ 39 levels "ALPHABET CITY
",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ BUILDING.CLASS.CATEGORY       : Factor w/ 43 levels "01  ONE FAMILY DWELLI
NGS                       ",..: 1 3 3 4 5 5 5 5 5 5 ...
 $ TAX.CLASS.AT.PRESENT          : Factor w/ 9 levels "  ","1","1A",..: 2 2 2
4 7 7 5 5 5 5 ...
 $ BLOCK                         : int  400 376 377 399 373 373 377 377 379 3
87 ...
 $ LOT                           : int  19 24 66 1101 16 17 2 2 37 119 ...
 $ BUILDING.CLASS.AT.PRESENT     : Factor w/ 130 levels "  ","A1","A4",..: 3
11 11 97 12 12 18 18 18 18 ...
 $ ADDRESS                       : Factor w/ 12234 levels "1-3 MINETTA STREET
, 2CD                     ",..: 10253 5849 5272 5113 7337 7356 687 687 1433 1098
0 ...
 $ APARTMENT.NUMBER              : Factor w/ 2738 levels "-           ",..: N
A NA NA NA NA NA NA NA NA NA ...
 $ ZIP.CODE                      : int  10009 10009 10009 10009 10009 10009 1
0009 10009 10009 10009 ...
 $ RESIDENTIAL.UNITS             : int  1 3 3 1 10 10 22 22 20 22 ...
 $ COMMERCIAL.UNITS              : int  0 0 0 0 0 0 3 3 2 2 ...
 $ TOTAL.UNITS                   : int  1 3 3 1 10 10 25 25 22 24 ...
 $ LAND.SQUARE.FEET              : num  1883 2059 2381 NA 2204 ...
 $ GROSS.SQUARE.FEET             : num  5200 3696 3084 NA 8625 ...
 $ YEAR.BUILT                    : int  1900 1900 1899 1955 1899 1900 1900 19
00 1930 1920 ...
 $ TAX.CLASS.AT.TIME.OF.SALE     : int  1 1 1 1 2 2 2 2 2 2 ...
 $ BUILDING.CLASS.AT.TIME.OF.SALE: Factor w/ 128 levels "A1","A4","A5",..: 2
10 10 95 11 11 17 17 17 17 ...
 $ SALE.PRICE                    : num  6500000 3775000 2900000 6995000 20000
000 ...
 $ SALE.DATE                     : Date, format: "2015-03-18" "2015-10-22" "2
015-06-24" ...
> PCA_comp<-prcomp(~ RESIDENTIAL.UNITS+COMMERCIAL.UNITS+TOTAL.UNITS+LAND.SQUA
RE.FEET+GROSS.SQUARE.FEET,data=dataDF,
+ na.action=na.omit,scale=T,center=T)
> str(PCA_comp)
List of 7
 $ sdev     : num [1:5] 1.93293 0.99701 0.46419 0.23296 0.00464
 $ rotation : num [1:5, 1:5] 0.5024 0.0705 0.5046 0.5024 0.4854 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:5] "RESIDENTIAL.UNITS" "COMMERCIAL.UNITS" "TOTAL.UNITS" "LA
ND.SQUARE.FEET" ...
  .. ..$ : chr [1:5] "PC1" "PC2" "PC3" "PC4" ...
 $ center   : Named num [1:5] 13.31 2.73 16 7692.92 65721.79
  ..- attr(*, "names")= chr [1:5] "RESIDENTIAL.UNITS" "COMMERCIAL.UNITS" "TOT
AL.UNITS" "LAND.SQUARE.FEET" ...
```

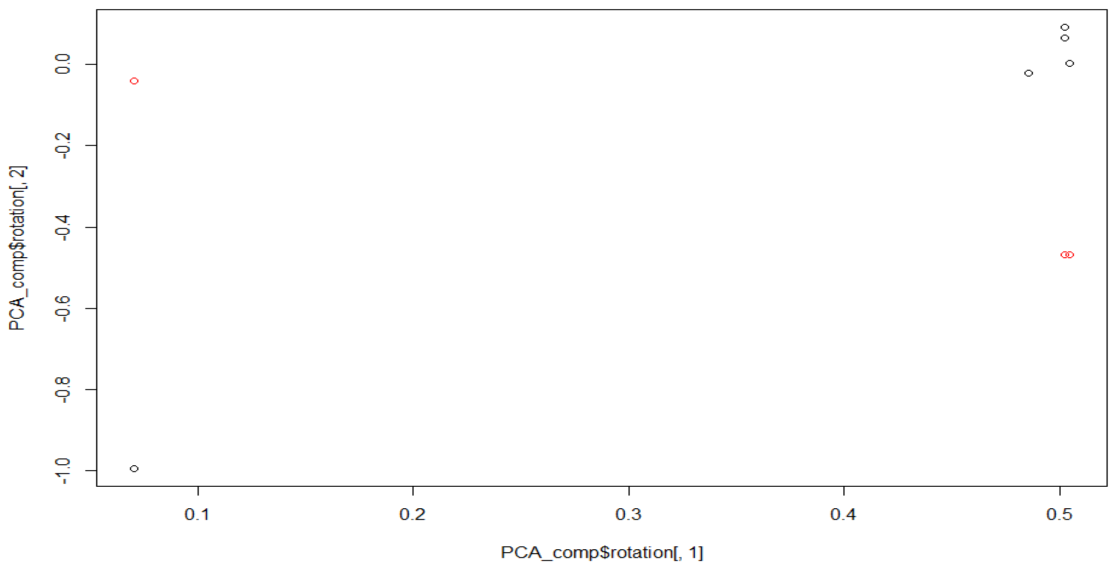
```
 $ scale    : Named num [1:5] 160.5 14.3 161.8 52505.8 191710.2
  ..- attr(*, "names")= chr [1:5] "RESIDENTIAL.UNITS" "COMMERCIAL.UNITS" "TOT
AL.UNITS" "LAND.SQUARE.FEET" ...
 $ x        : num [1:3346, 1:5] -0.308 -0.297 -0.296 -0.24 -0.24 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3346] "1" "2" "3" "5" ...
  .. ..$ : chr [1:5] "PC1" "PC2" "PC3" "PC4" ...
 $ call     : language prcomp(formula = ~RESIDENTIAL.UNITS + COMMERCIAL.UNITS
+ TOTAL.UNITS + LAND.SQUARE.FEET +      GROSS.SQUARE.FEET, data = dataDF, na.
action = na.omit, scale = T, center = T)
 $ na.action:Class 'omit'  Named int [1:20411] 4 26 27 28 29 30 31 32 33 34 .
..
  .. ..- attr(*, "names")= chr [1:20411] "4" "26" "27" "28" ...
 - attr(*, "class")= chr "prcomp"
> summary(PCA_comp)
Importance of components:
                          PC1    PC2     PC3     PC4      PC5
Standard deviation     1.9329 0.9970 0.46419 0.23296 0.004638
Proportion of Variance 0.7472 0.1988 0.04309 0.01085 0.000000
Cumulative Proportion  0.7472 0.9460 0.98914 1.00000 1.000000
```

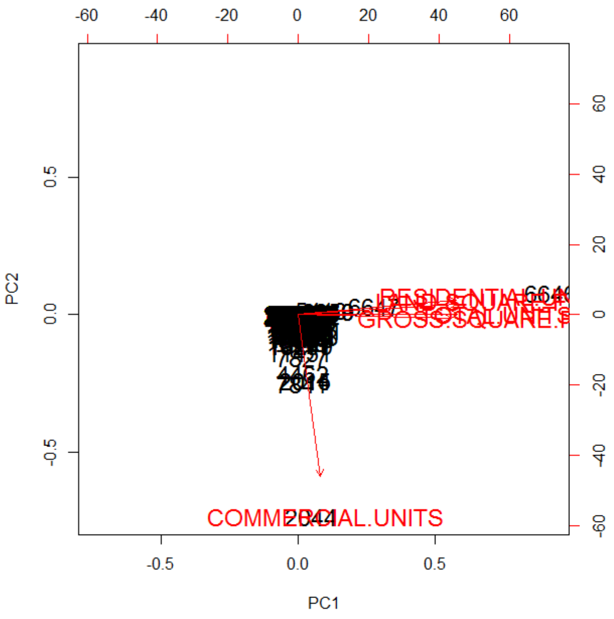
Conclusion:

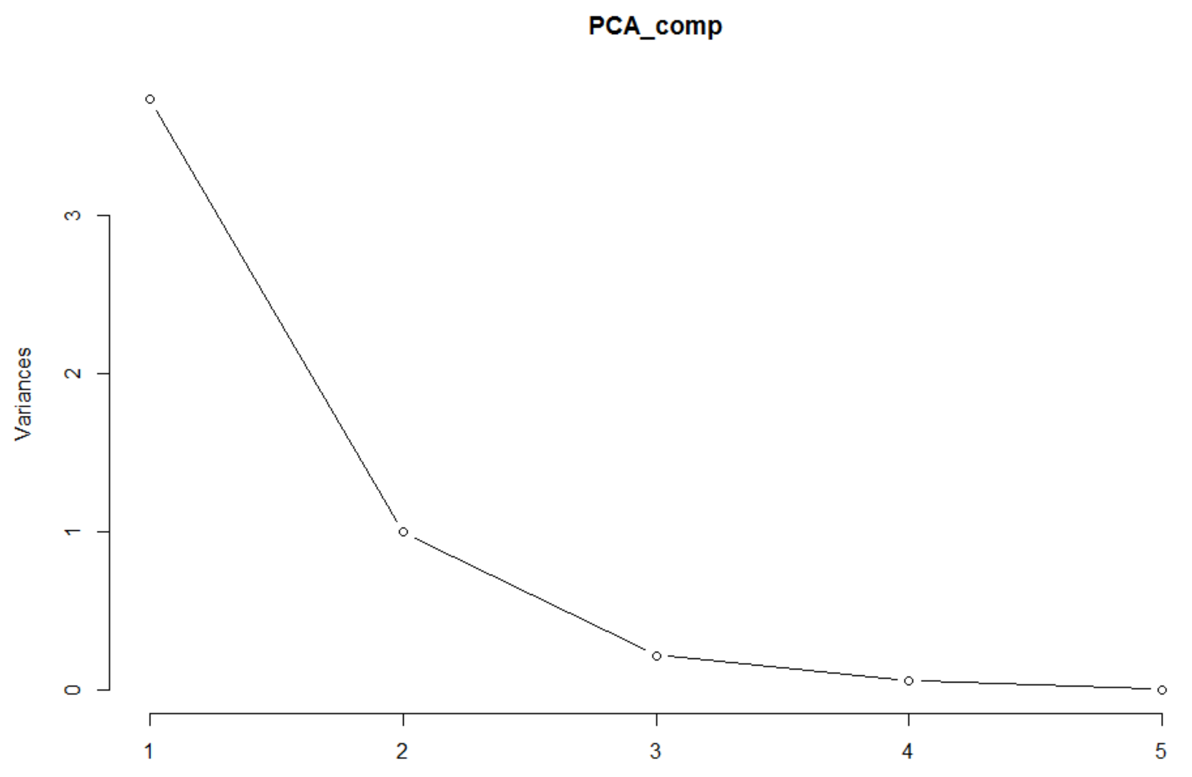
From the above summary of PCA_comp, we can conclude that we have transform standard deviation to one. To achieve this, prcomp() function is used. Hence it is clearly visible that we have scaled the variables to have a standard deviation of one.

b)

```
> plot(PCA_comp$rotation[,1],PCA_comp$rotation[,2])
> points(PCA_comp$rotation[,1],PCA_comp$rotation[,3],col="red")
> plot(PCA_comp$rotation[,1])
> plot(PCA_comp$rotation[,2])
> plot(PCA_comp$rotation[,3])
> biplot(PCA_comp,cex=1.5)
> plot(PCA_comp,type='l')
```







Conclusion:

The above three plots are a great help in visualization of data. The first graph is generated using the required method. Each point has been colored based on the vectors paired. Analyzing graph 2 we can say that the vector is orthogonal as it should be. And the final graph shows how the variance is decreasing.

Hence, the components are perpendicular to each other and the variance tends to become zero.

Assignment 3 N14724679
Foundations of Data Science Kunal B. Relia (kbr263)

Group Work:

Some of the questions of this assignment were discussed with Fenil Tailor. Only the idea of solving a few questions were discussed by us followed by individual application of the ideas.