Group_Project

Kyle Ayiku

2023-03-25

Regression

This is the data set that I chose for this project.

```
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.2.3

# Reading in the data set.
setwd("C:/Users/amark/Downloads")
housing <- read.csv("housing.csv")
```

Dividing the data into train and test (Part A)

```
# Eliminating the NAs from the dataset.
housing <- housing[complete.cases(housing),]

set.seed(1234)
num <- sample(1:nrow(housing), 0.80 * nrow(housing), replace = FALSE)
train <- housing[num,]
test <- housing[-num,]

sapply(train, function(x) sum(is.na(x) == TRUE))

##          longitude           latitude housing_median_age
total_rooms
##                  0                  0                  0
0
##     total_bedrooms         population         households
median_income
##                  0                  0                  0
0
## median_house_value    ocean_proximity
##                  0                  0
```

I split the data into 80% train and 20% test and also checked to see if there any null values and unfortunately, I can see that total_bedrooms has 165 missing values.

```
library(dplyr)

## Warning: package 'dplyr' was built under R version 4.2.3
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(janitor)

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

##
## Attaching package: 'janitor'

## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test

# Cleaning the training data.
clean <- clean_names(train)
colnames(train)

##  [1] "longitude"          "latitude"           "housing_median_age"
##  [4] "total_rooms"        "total_bedrooms"     "population"
##  [7] "households"         "median_income"      "median_house_value"
## [10] "ocean_proximity"
```

I attempted to see if there is any of the training data that needs to be cleaned.

Exploratory Data Analysis (Part B)

```
# Brief look at the data.
head(train) # Displays the first few rows of the training data.

##      longitude latitude housing_median_age total_rooms total_bedrooms
## 7534   -118.22    33.91                 31         571            153
## 8103   -118.22    33.80                 36        1285            347
## 7242   -118.12    34.00                 31        3281            768
## 8173   -118.12    33.80                 36        1257            205
## 9288   -122.55    38.07                 38        3392            709
## 627    -122.18    37.70                 36        2639            533
##      population households median_income median_house_value
ocean_proximity
## 7534        841        158        2.6154              89200        <1H
OCEAN
## 8103       1291        337        3.7708             157100       NEAR
OCEAN
## 7242       2385        733        2.7308             173800        <1H
OCEAN
```

```
## 8173        530        211        5.3701             251400       <1H
OCEAN
## 9288       1894        713        3.0573             350800        NEAR
BAY
## 627        1209        519        4.0268             205500        NEAR
BAY
```

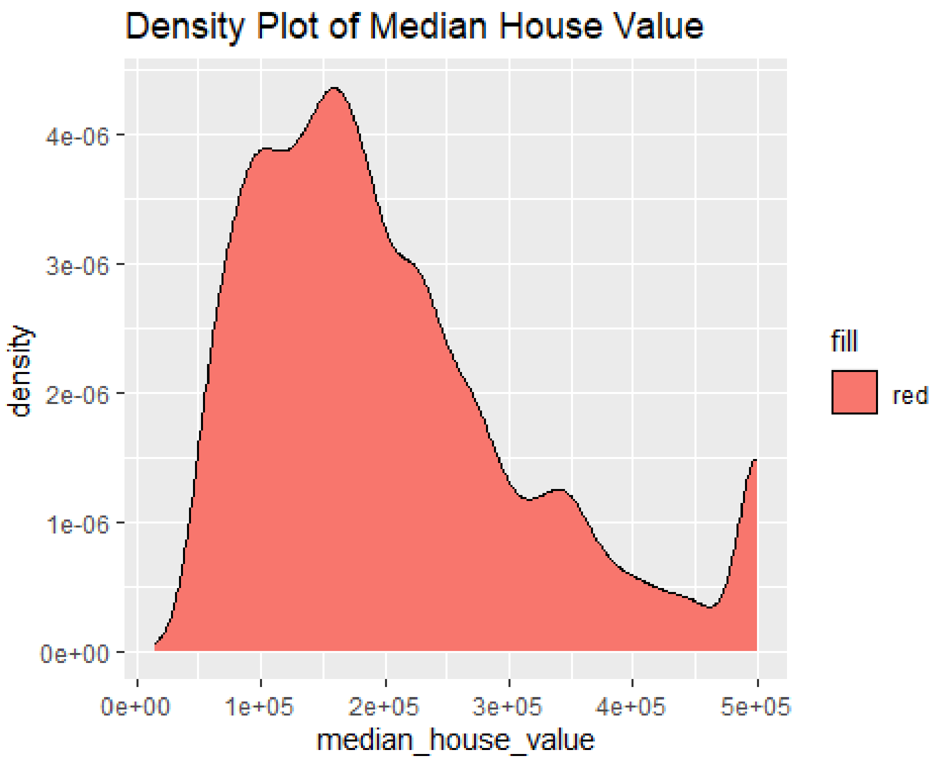
```r
dim(train)  # How many rows and columns are in the training data.
```

```
## [1] 16346    10
```

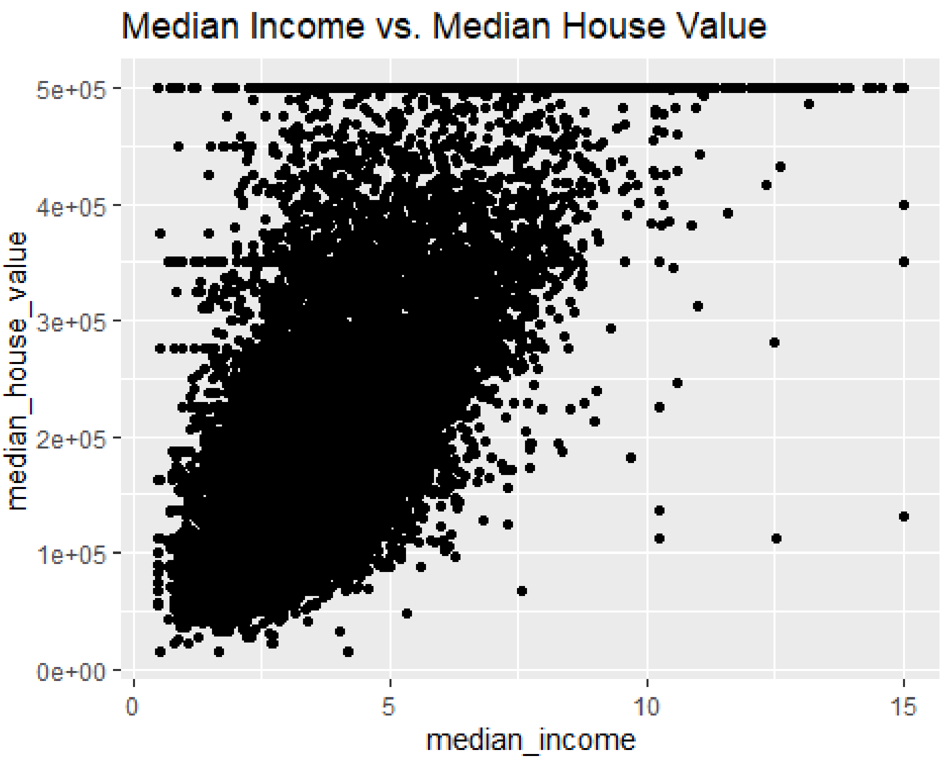
```r
summary(train) # General statistics of the training data.
```

```
##    longitude         latitude     housing_median_age  total_rooms
##  Min.   :-124.3   Min.   :32.54   Min.   : 1.00      Min.   :    2
##  1st Qu.:-121.8   1st Qu.:33.93   1st Qu.:18.00      1st Qu.: 1451
##  Median :-118.5   Median :34.25   Median :29.00      Median : 2126
##  Mean   :-119.6   Mean   :35.62   Mean   :28.59      Mean   : 2637
##  3rd Qu.:-118.0   3rd Qu.:37.71   3rd Qu.:37.00      3rd Qu.: 3138
##  Max.   :-114.3   Max.   :41.95   Max.   :52.00      Max.   :39320
##  total_bedrooms     population      households     median_income
##  Min.   :   1.0   Min.   :    5   Min.   :   1.0   Min.   : 0.4999
##  1st Qu.: 297.0   1st Qu.:  788   1st Qu.: 281.0   1st Qu.: 2.5550
##  Median : 435.0   Median : 1166   Median : 409.0   Median : 3.5313
##  Mean   : 538.4   Mean   : 1425   Mean   : 499.5   Mean   : 3.8674
##  3rd Qu.: 647.0   3rd Qu.: 1718   3rd Qu.: 604.0   3rd Qu.: 4.7437
##  Max.   :6445.0   Max.   :35682   Max.   :6082.0   Max.   :15.0001
##  median_house_value ocean_proximity
##  Min.   : 14999     Length:16346
##  1st Qu.:119400     Class :character
##  Median :179650     Mode  :character
##  Mean   :206728
##  3rd Qu.:264575
##  Max.   :500001
```

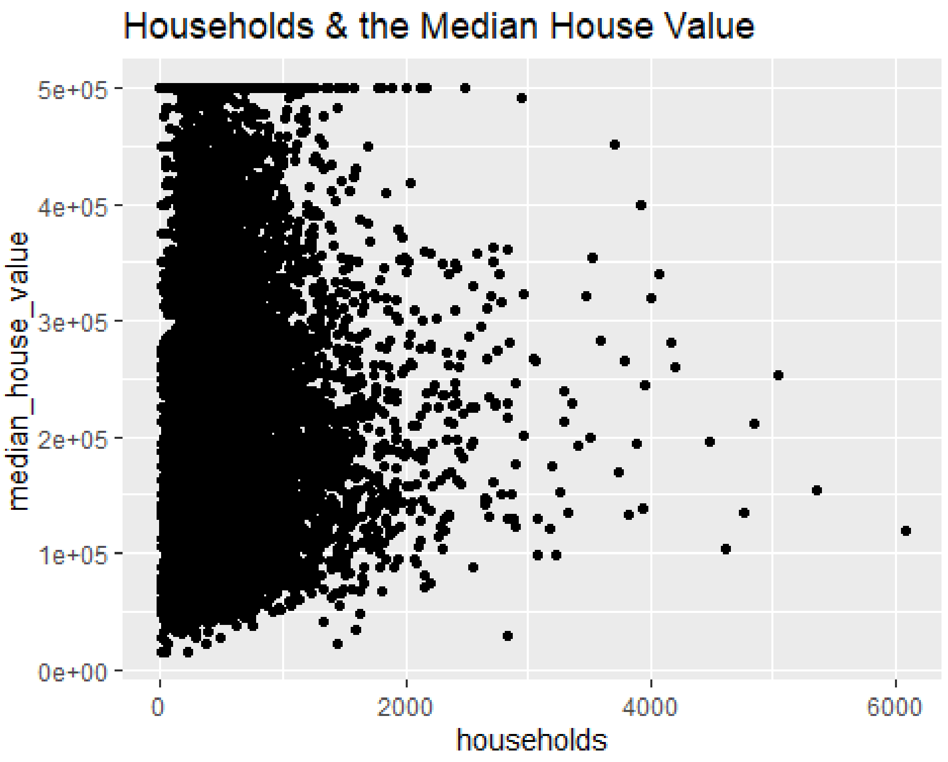
```r
# Density plot for the response variable, "median_house_value."
ggplot(data = train) +
  geom_density(mapping = aes(x = median_house_value,
                             fill = "red")) +
  labs(title = "Density Plot of Median House Value")
```



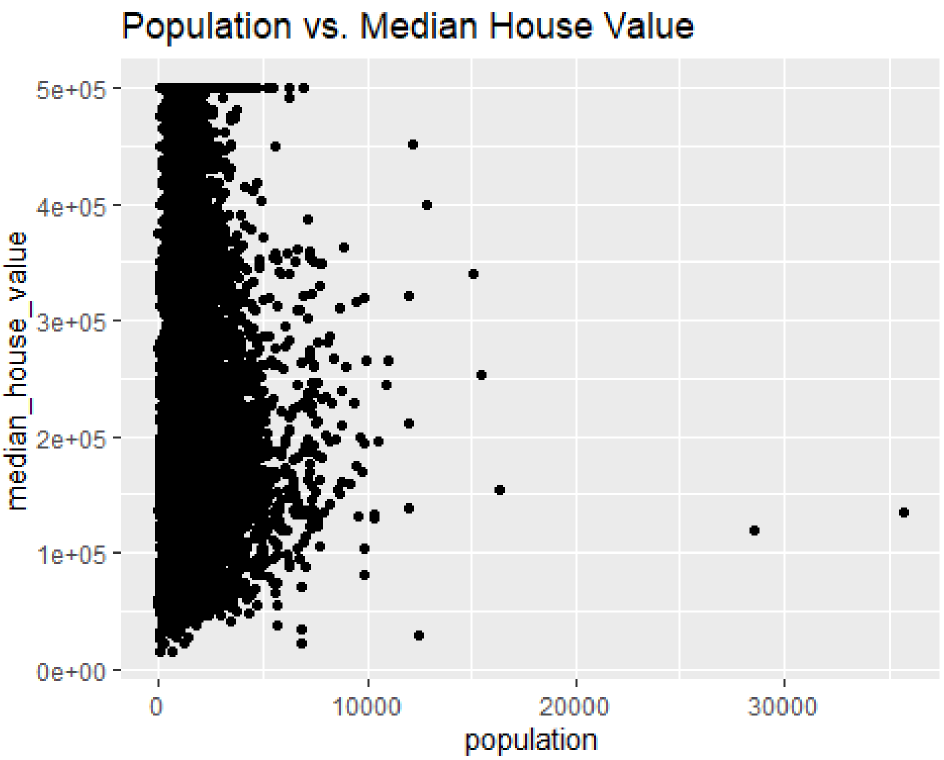
```
# Scatter plots for some predictors and median_house_value.
ggplot(data = train) +
  geom_point(mapping = aes(x = median_income,
                           y = median_house_value)) +
  labs(title = "Median Income vs. Median House Value")
```



```
ggplot(data = train) +
  geom_point(mapping = aes(x = households,
                           y = median_house_value)) +
  labs(title = "Households & the Median House Value")
```



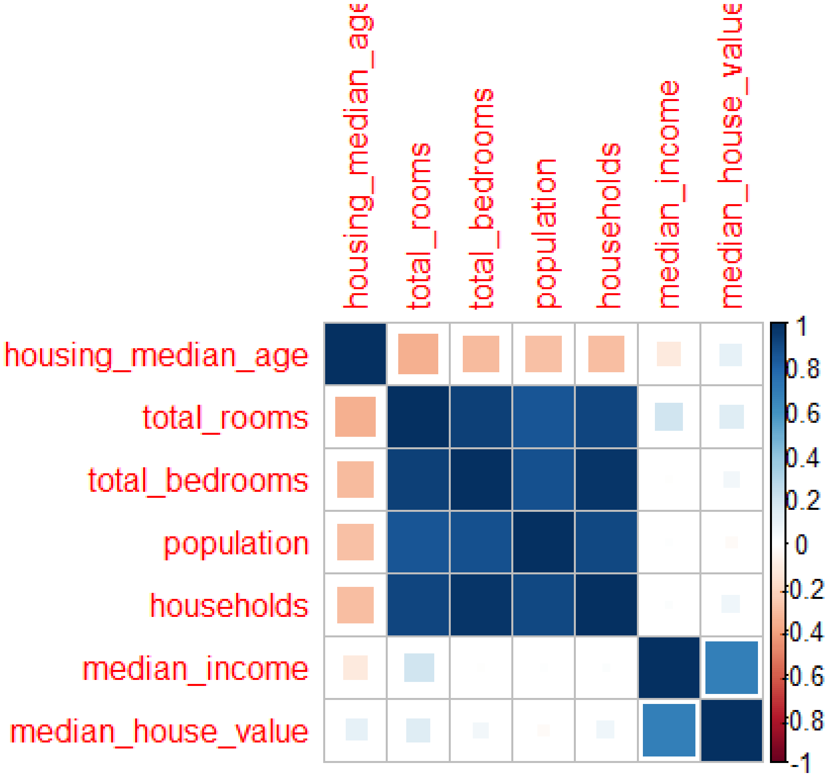
```
ggplot(data = train) +
  geom_point(mapping = aes(x = population,
                           y = median_house_value)) +
  labs(title = "Population vs. Median House Value")
```



```
# Displaying the correlation matrix.
library(Hmisc)

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

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':
##
##     src, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, units

library(corrplot)

## Warning: package 'corrplot' was built under R version 4.2.2

## corrplot 0.92 loaded

cor(train[,c(3:9)])

##                      housing_median_age total_rooms total_bedrooms
population
## housing_median_age          1.0000000  -0.3595297    -0.318486183 -
0.291600322
## total_rooms                -0.3595297   1.0000000     0.930276347
```

```
0.855524116
## total_bedrooms            -0.3184862   0.9302763    1.000000000
0.874904940
## population                -0.2916003   0.8555241    0.874904940
1.000000000
## households                -0.3000743   0.9178393    0.978089318
0.906235866
## median_income             -0.1176870   0.1982394   -0.008214284
0.005999249
## median_house_value         0.1071152   0.1354307    0.051267879 -
0.023011587
##                      households median_income median_house_value
## housing_median_age -0.30007428  -0.117687025         0.10711521
## total_rooms         0.91783925   0.198239385         0.13543067
## total_bedrooms      0.97808932  -0.008214284         0.05126788
## population          0.90623587   0.005999249        -0.02301159
## households          1.00000000   0.013615731         0.06699831
## median_income       0.01361573   1.000000000         0.68588807
## median_house_value  0.06699831   0.685888071         1.00000000
```

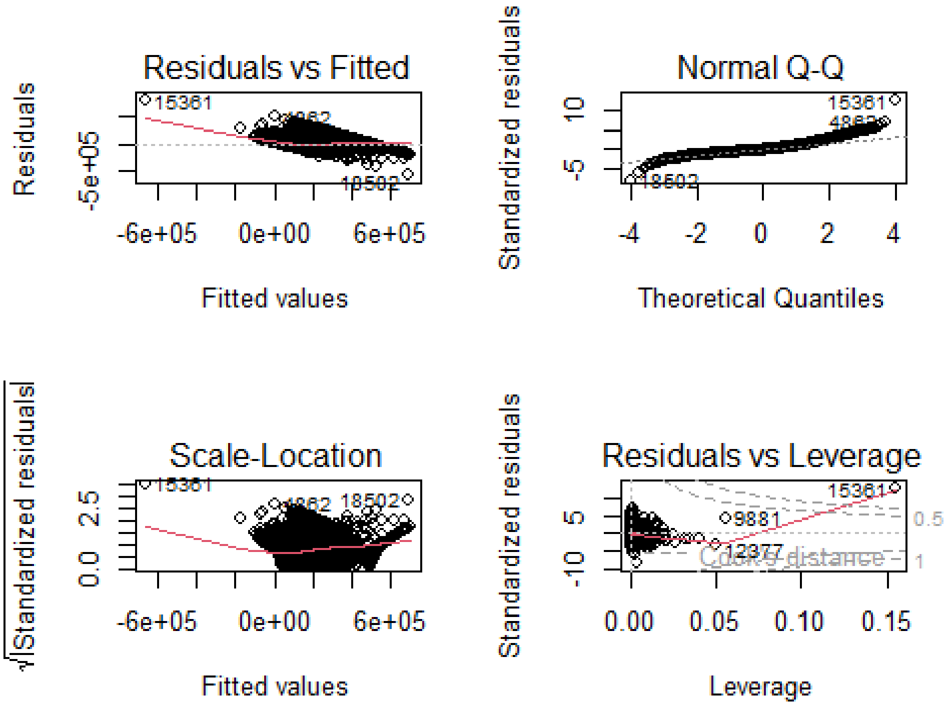
```
corrplot(cor(train[,c(3:9)]), method = "square")
```



Regression Techniques (Part C)

Linear Regression

```r
# Creating a multiple linear regression model with
# the target, median_house_value.

lm1 <- lm(median_house_value ~ housing_median_age + total_rooms +
            total_bedrooms + population + households +
            median_income + longitude + latitude, data = train)
summary(lm1)

##
## Call:
## lm(formula = median_house_value ~ housing_median_age + total_rooms +
##     total_bedrooms + population + households + median_income +
##     longitude + latitude, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -559038  -43890  -11392   30223  796426
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -3.575e+06  7.088e+04 -50.438  < 2e-16 ***
## housing_median_age  1.172e+03  4.880e+01  24.019  < 2e-16 ***
## total_rooms        -7.925e+00  8.942e-01  -8.863  < 2e-16 ***
## total_bedrooms      1.082e+02  7.571e+00  14.294  < 2e-16 ***
## population         -3.851e+01  1.210e+00 -31.836  < 2e-16 ***
## households          5.362e+01  8.220e+00   6.523 7.08e-11 ***
## median_income       3.993e+04  3.772e+02 105.874  < 2e-16 ***
## longitude          -4.267e+04  8.081e+02 -52.802  < 2e-16 ***
## latitude           -4.258e+04  7.634e+02 -55.780  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 70010 on 16337 degrees of freedom
## Multiple R-squared:  0.6331, Adjusted R-squared:  0.6329
## F-statistic:  3523 on 8 and 16337 DF,  p-value: < 2.2e-16

# Test data evaluation.
pred1 <- predict(lm1, newdata = test)
print(cor1 <- cor(pred1, test$median_house_value)) # Correlation.

## [1] 0.8077368

print(mse1 <- mean((pred1 - test$median_house_value)^2)) # The MSE.

## [1] 4597020605

print(rmse1 <- sqrt(mse1)) # The RMSE.

## [1] 67801.33

par(mfrow = c(2, 2))
plot(lm1)
```



kNN Regression

```
library(caret)

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

## Loading required package: lattice

# Model fitting.
fit <- knnreg(train[,3:9], train[,2], k = 5)

# Evaluation.
pred2 <- predict(fit, test[,3:9])
print(cor_knn1 <- cor(pred2, test$median_house_value)) # Correlation.

## [1] -0.2835157

print(mse_knn1 <- mean((pred2 - test$median_house_value)^2)) # The MSE.

## [1] 56220849031

print(rmse2 <- sqrt(mse_knn1)) # The RMSE.

## [1] 237109.4

# Data scaling.
train_scaled <- train[,3:9]
means <- sapply(train_scaled, mean)
stdvs <- sapply(train_scaled, sd)
```

```r
train_scaled <- scale(train_scaled, center = means, scale = stdvs)
test_scaled <- scale(test[,3:9], center = means, scale = stdvs)

# kNN for the scaled data.
fit <- knnreg(train_scaled, train$median_house_value, k = 7)
pred3 <- predict(fit, test_scaled)
print(cor_knn2 <- cor(pred3, test$median_house_value))
```

```
## [1] 0.9947504
```

```r
print(mse_knn2 <- mean((pred3 - test$median_house_value)^2))
```

```
## [1] 143304622
```

```r
print(rmse3 <- sqrt(mse_knn2)) # The RMSE.
```

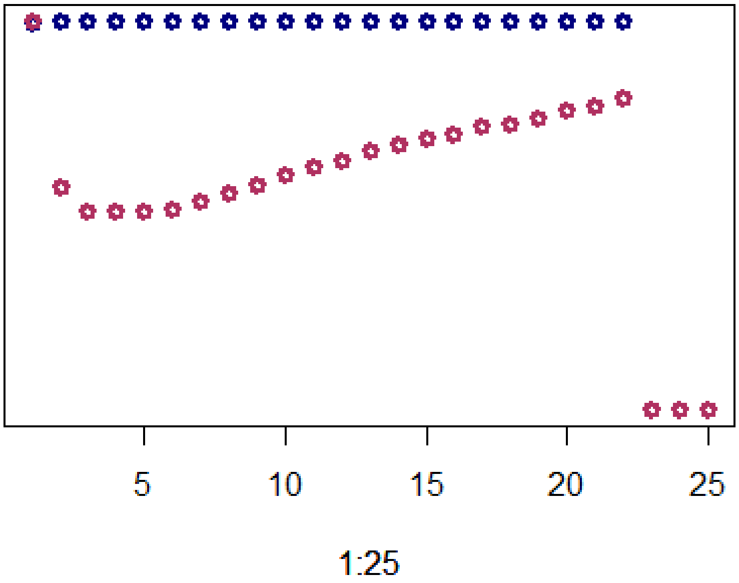
```
## [1] 11970.99
```

```r
# Finding the best k algorithm.

cor_k <- rep(0, 25)
mse_k <- rep(0, 25)

num <- 1
for (k in seq(1, 43, 2))
{
  fit_k <- knnreg(train_scaled, train$median_house_value, k = k)
  pred_k <- predict(fit_k, test_scaled)
  cor_k[num] <- cor(pred_k, test$median_house_value) # Correlation value.
  mse_k[num] <- mean((pred_k - test$median_house_value)^2) # The MSE.
  print(paste("k = ", k, cor_k[num], mse_k[num]))
  num <- num + 1
}
```

```
## [1] "k =  1 0.989279387434334 282376990.03034"
## [1] "k =  3 0.993897558621111 162107770.556398"
## [1] "k =  5 0.994639521093658 144492354.970546"
## [1] "k =  7 0.994750383640009 143304621.529022"
## [1] "k =  9 0.994778005188362 143773832.343323"
## [1] "k =  11 0.994747082359241 145897116.989246"
## [1] "k =  13 0.99459802755827 2151317348.69291"
## [1] "k =  15 0.994436361673998 156506027.781358"
## [1] "k =  17 0.994272519842331 162819376.379659"
## [1] "k =  19 0.994052564769095 170015279.548431"
## [1] "k =  21 0.993860263921137 176852526.423217"
## [1] "k =  23 0.993758472978828 181382385.607802"
## [1] "k =  25 0.99355383593146 188172512.248087"
## [1] "k =  27 0.993460491962776 192305629.276772"
## [1] "k =  29 0.993347069622814 196723299.419251"
## [1] "k =  31 0.993287721370641 199973395.641524"
## [1] "k =  33 0.993151928890091 205115339.198631"
## [1] "k =  35 0.993116345580429 207652602.101231"
## [1] "k =  37 0.992994140712848 211966216.934172"
## [1] "k =  39 0.992861386146755 217211630.00099"
## [1] "k =  41 0.992814959484625 220293272.57596"
## [1] "k =  43 0.992668002402656 225787838.949563"
```

```
# Plotting the kNN regression model.
plot(1:25, cor_k, lwd = 3, col = 'navy', ylab = "", yaxt = 'n')
par(new = TRUE)
plot(1:25, mse_k, lwd = 3, col = 'maroon', labels = FALSE,
     ylab = "", yaxt = 'n')
```

```
## Warning in plot.window(...): "labels" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "labels" is not a graphical parameter

## Warning in box(...): "labels" is not a graphical parameter

## Warning in title(...): "labels" is not a graphical parameter
```



```
# Lowest and highest values.
which.min(mse_k) # The lowest.

## [1] 23

which.max(cor_k) # The highest.

## [1] 5
```

As seen from the algorithm and diagram above, the lowest MSE came from “k = 23” and the highest correlation coefficient came from “k = 5.”

Comparison & Analysis (Parts C & D)

Given the regression techniques which I implemented, the correlation coefficients and mean squared errors were varied. Logically speaking, this is expected in the sense that which kNN generally had higher correlation coefficients of 99% whereas the linear regression technique only had one of roughly 80%, meaning that for the data set, kNN is superior in this case. Also, kNN had a lower mean squared error than linear regression.

Analyzing the techniques, there are clearly varying methods of retrieving the results as seen from the techniques above. Additionally, kNN has much more flexibility in terms of the methodologies the technique can process as the possibilities are theoretically endless. This is due to the fact that it does not that there is a response variable, per se. Conversely, linear regression is much simpler to utilize as there is a general approach to how the model will fit the data, given that there is not only a target, but also the predictors which contribute towards the target variable.