Assignment3_2

Amirali Khatib

6/3/2022

Question 2

Import libraries

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.3
```

```
##
## 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(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.3
```

```
library(lattice) # caret requirement
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.1.3
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.1.3
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.7     v purrr   0.3.4
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## Warning: package 'tibble' was built under R version 4.1.3
```

```
## Warning: package 'tidyr' was built under R version 4.1.3
```

```
## Warning: package 'forcats' was built under R version 4.1.3
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.1.3
```

```
library(cowplot)
```

```
## Warning: package 'cowplot' was built under R version 4.1.3
```

Read Sacramento dataset

```
data(Sacramento)
sac_data = Sacramento
sac_data$limits = factor(ifelse(sac_data$city == "SACRAMENTO", "in", "out"))
sac_data = subset(sac_data, select = -c(city, zip, baths))
```

Data splitting

```
set.seed(42)
sac_trn_idx = sample(nrow(sac_data), size = 0.8 * nrow(sac_data))
sac_trn = sac_data[sac_trn_idx,]
sac_tst = sac_data[-sac_trn_idx,]

sac_est_idx = sample(nrow(sac_trn), size = 0.8 * nrow(sac_trn))
sac_est = sac_trn[sac_est_idx,]
sac_val = sac_trn[-sac_est_idx,]
```

RMSE function

```
rmse = function(predicted, actual){sqrt(mean((actual - predicted)^2))}
```

Part A: Linear model

```
sac_reg_model_list = list(
  model_1 = lm(formula = price ~ sqft + beds + limits, data = sac_est),
  model_2 = lm(formula = price ~ ., data = sac_est),
  model_3 = step(lm(formula = price ~ ., data = sac_est), trace = FALSE, direction = 'forward'),
  model_4 = lm(formula = price ~ I(sqft ^ 2) + beds + limits + (latitude + longitude) ^ 2, data = sac_est),
  model_5 = lm(formula = price ~ . -limits -latitude -longitude, data = sac_est))

sac_est_reg_predicted_list = lapply(sac_reg_model_list, predict, sac_est)
sac_val_reg_predicted_list = lapply(sac_reg_model_list, predict, sac_val)


sac_rmse_reg_est_list = sapply(sac_est_reg_predicted_list, rmse, sac_est$price)
sac_rmse_reg_val_list = sapply(sac_val_reg_predicted_list, rmse, sac_val$price)

report_reg_models = data.frame(data_type = c('estimation', 'validation'),
                               model_1_rmse = c(sac_rmse_reg_est_list[1], sac_rmse_reg_val_list[1]),
                               model_2_rmse = c(sac_rmse_reg_est_list[2], sac_rmse_reg_val_list[2]),
                               model_3_rmse = c(sac_rmse_reg_est_list[3], sac_rmse_reg_val_list[3]),
                               model_4_rmse = c(sac_rmse_reg_est_list[4], sac_rmse_reg_val_list[4]),
                               model_5_rmse = c(sac_rmse_reg_est_list[5], sac_rmse_reg_val_list[5]))

report_reg_models
```

```
##    data_type model_1_rmse model_2_rmse model_3_rmse model_4_rmse model_5_rmse
## 1 estimation     77639.83     75572.25     75572.25     80431.33     78144.34
## 2 validation     91497.72     89761.84     89761.84     95220.69     92986.50
```

By fitting model_5(`price ~ . -limits -latitude -longitude`), we didn't encounter with significant change in RMSE in comparison with the other models' RMSE value. . Hence, we can conclude that the absence of **limits**, **latitude**, **longitude** do not causes a considerable shift in RMSE value of the model.

Part B: KNN model

Feature scaling

Estimation set

```
beds.s = scale(sac_est$beds)
beds.center = attr(beds.s, "scaled:center")
beds.scale = attr(beds.s, "scaled:scale")

latitude.s = scale(sac_est$latitude)
latitude.center = attr(latitude.s, "scaled:center")
latitude.scale = attr(latitude.s, "scaled:scale")

longitude.s = scale(sac_est$longitude)
longitude.center = attr(longitude.s, "scaled:center")
longitude.scale = attr(longitude.s, "scaled:scale")

sqft.s = scale(sac_est$sqft)
sqft.center = attr(sqft.s, "scaled:center")
sqft.scale = attr(sqft.s, "scaled:scale")

sac_est_scaled = data.frame(beds_s = beds.s, sqft_s = sqft.s,
                            type = sac_est$type, price = sac_est$price,
                            latitude_s = latitude.s,
                            longitude_s = longitude.s,
                            limits = sac_est$limits)
head(sac_est_scaled, 5)
```

```
##       beds_s      sqft_s        type  price latitude_s longitude_s limits
## 1 -0.2841249 -0.71540995 Residential 181000 -0.3359412 -0.05852174     in
## 2  0.8226622 -0.43704621 Residential 293996  0.3695965 -0.52070228     in
## 3 -0.2841249 -0.83735978 Residential 122000 -2.5317650  0.34311540    out
## 4 -1.3909119 -0.28328337 Residential 195000  0.6980035  2.59984090    out
## 5 -1.3909119 -0.09903308 Residential 220702  0.2745050  0.16372701    out
```

Validation set

```
beds.s.val = scale(sac_val$beds, center = beds.center, scale = beds.scale)
latitude.s.val = scale(sac_val$latitude, center = latitude.center, scale = latitude.scale)
longitude.s.val = scale(sac_val$longitude, center = longitude.center, scale = longitude.scale)
sqft.s.val = scale(sac_val$sqft, center = sqft.center, scale = sqft.scale)


sac_val_scaled = data.frame(beds_s = beds.s.val, sqft_s = sqft.s.val,
                            type = sac_val$type, price = sac_val$price,
                            latitude_s = latitude.s.val,
                            longitude_s = longitude.s.val,
                            limits = sac_val$limits)

head(sac_val_scaled, 5)
```

```
##       beds_s     sqft_s        type  price latitude_s longitude_s limits
## 1 -0.2841249 -0.5351363 Residential 205000  1.0876503   4.0512084    out
## 2  0.8226622 -0.3111197 Residential 176850 -1.0656536  -0.1036670    out
## 3  0.8226622  0.2628398 Residential 300500  0.6039712  -0.9823655     in
## 4 -0.2841249 -0.3389561 Residential 225000  0.7884474   0.8827248    out
## 5 -0.2841249 -0.9274966 Residential  90000 -0.7879952  -0.9396848     in
```

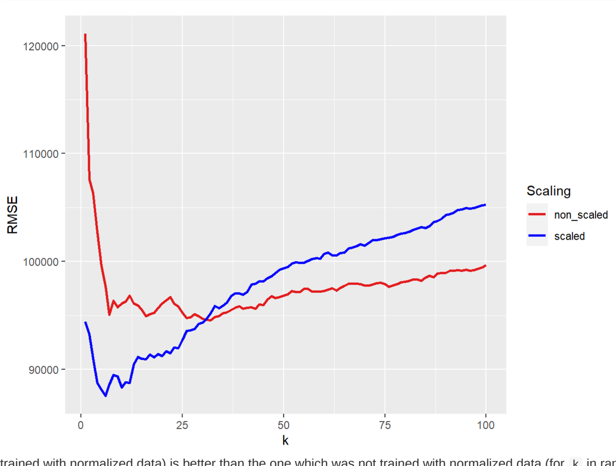
Train KNN models

```
sac_knn_model_list = list()
for (k in 1:100){
  sac_knn_model_list[[k]] = knnreg(formula = price ~ . , data = sac_est, k = k)
}
sac_knn_predicted_list = lapply(sac_knn_model_list, predict, sac_val)
sac_knn_rmse_list = sapply(sac_knn_predicted_list, rmse, sac_val$price)

sac_scaled_knn_model_list = list()
for (k in 1:100){
  sac_scaled_knn_model_list[[k]] = knnreg(formula = price ~., data = sac_est_scaled, k = k)
}
sac_scaled_knn_predicted_list = lapply(sac_scaled_knn_model_list, predict, sac_val_scaled)
sac_scaled_knn_rmse_list = sapply(sac_scaled_knn_predicted_list, rmse, sac_val$price)

report_knn_models = data.frame(k = as.numeric(1:100),
                               rmse = as.numeric(sac_knn_rmse_list),
                               rmse_scaled = as.numeric(sac_scaled_knn_rmse_list))
report_knn_models
```

```
##       k      rmse rmse_scaled
## 1     1 121087.40    94422.83
## 2     2 107618.61    93238.05
## 3     3 106328.80    91025.64
## 4     4 102745.41    88759.24
## 5     5  99603.08    88103.08
## 6     6  97682.81    87550.50
## 7     7  95060.63    88597.70
## 8     8  96348.80    89500.65
## 9     9  95771.40    89386.17
## 10   10  96094.96    88318.71
## 11   11  96260.37    88796.01
## 12   12  96819.48    88756.78
## 13   13  96109.25    90471.27
## 14   14  95956.32    91161.36
## 15   15  95549.12    90985.38
## 16   16  94945.73    90965.14
## 17   17  95142.65    91396.23
## 18   18  95233.32    91115.35
## 19   19  95686.44    91437.44
## 20   20  96098.69    91263.43
## 21   21  96404.20    91705.15
## 22   22  96725.45    91503.40
## 23   23  96093.45    92043.80
## 24   24  95843.59    91993.39
## 25   25  95241.17    92700.86
## 26   26  94763.49    93589.96
## 27   27  94828.34    93628.03
## 28   28  95110.12    93739.42
## 29   29  94957.06    94200.70
## 30   30  94677.88    94339.98
## 31   31  94611.46    94679.12
## 32   32  94543.58    95204.71
## 33   33  94883.90    95893.53
## 34   34  94955.62    95694.59
## 35   35  95216.97    95908.81
## 36   36  95314.22    96182.94
## 37   37  95491.90    96777.81
## 38   38  95725.83    97063.12
## 39   39  95862.74    97053.49
## 40   40  95618.86    96936.54
## 41   41  95707.98    97167.49
## 42   42  95749.34    97864.97
## 43   43  95657.92    97976.73
## 44   44  96006.59    98191.22
## 45   45  95980.29    98177.84
## 46   46  96501.99    98488.47
## 47   47  96813.94    98642.29
## 48   48  96644.68    98947.43
## 49   49  96713.34    99248.73
## 50   50  96834.36    99382.09
## 51   51  96959.80    99496.63
## 52   52  97273.94    99799.87
## 53   53  97203.80    99927.26
## 54   54  97179.21    99872.66
## 55   55  97491.07    99868.00
## 56   56  97483.85   100051.25
## 57   57  97231.33   100222.10
## 58   58  97245.05   100299.36
## 59   59  97207.12   100289.68
## 60   60  97260.73   100702.04
## 61   61  97394.53   100837.91
## 62   62  97510.16   100563.04
## 63   63  97330.55   100577.74
## 64   64  97582.73   100794.06
## 65   65  97752.33   100851.44
## 66   66  97967.40   101198.88
## 67   67  97972.57   101290.82
## 68   68  97934.79   101446.76
## 69   69  97898.51   101597.45
## 70   70  97781.21   101495.16
## 71   71  97789.26   101718.04
## 72   72  97851.24   102008.69
## 73   73  98012.30   101971.29
## 74   74  98039.26   102074.41
## 75   75  97933.77   102163.41
## 76   76  97672.40   102218.50
## 77   77  97802.93   102278.85
## 78   78  97930.01   102446.39
## 79   79  98092.16   102609.07
## 80   80  98108.18   102637.33
## 81   81  98209.55   102755.85
## 82   82  98329.26   102921.06
## 83   83  98325.79   103051.95
## 84   84  98223.07   103198.35
## 85   85  98509.68   103102.66
## 86   86  98672.63   103272.97
## 87   87  98558.26   103654.83
## 88   88  98881.52   103767.57
## 89   89  98943.12   103973.55
## 90   90  98952.03   104317.18
## 91   91  99138.99   104393.06
## 92   92  99137.01   104514.77
## 93   93  99181.43   104785.27
## 94   94  99152.15   104811.70
## 95   95  99229.42   104939.21
## 96   96  99167.51   104890.09
## 97   97  99209.74   104970.81
## 98   98  99326.34   105096.04
## 99   99  99459.39   105214.12
## 100 100  99671.73   105250.27
```

Plot line chart for KNN models' RMSE value

```
ggplot(data = report_knn_models, aes(x = k))+
  geom_line(aes(y = rmse, color = 'non_scaled'), size = 1)+
  geom_line(aes(y = rmse_scaled, color = 'scaled'), size = 1)+
  labs(x = 'k', y = 'RMSE', color = 'Scaling')+
  scale_color_manual(values = c("non_scaled" = "#E31A1C", "scaled" = "blue"))
```



The model performance (which was trained with normalized data) is better than the one which was not trained with normalized data (for `k` in range 1 to 31) The model performance (which was not trained with normalized data) is better that the one which was trained with normalized data (for `k` in range 32 to 100)

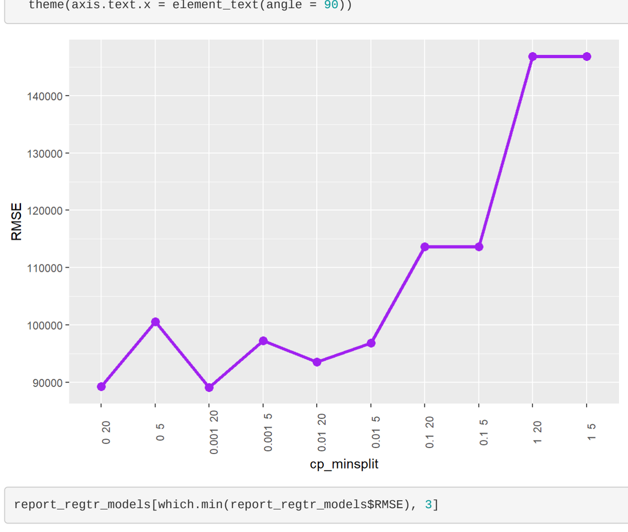
Part C: Regression tree model

```
cp_list = c(1,0.1,0.01,0.001,0)
minsplit_list = c(5,20)
sac_regtr_list = list()
cp_column = list()
minsplit_column = list()
cp_minsplit = list()
sac_rmse_regtr_list = list()
for (m in minsplit_list){
  for (cp in cp_list){
    minsplit_column = append(minsplit_column, m)
    cp_column = append(cp_column , cp)
    cp_minsplit = append(cp_minsplit, paste(cp,'',m))
    regtr_model = rpart(price ~ ., data = sac_est, cp = cp, minsplit = m)
    predict_regtr = predict(regtr_model, sac_val)
    rmse_regtr = rmse(predict_regtr, sac_val$price)
    sac_rmse_regtr_list = append(sac_rmse_regtr_list, rmse_regtr)
  }
}

report_regtr_models = data.frame(cp = as.numeric(cp_column),
                                 minsplit = as.numeric(minsplit_column),
                                 cp_minsplit = as.character(cp_minsplit),
                                 RMSE = as.numeric(sac_rmse_regtr_list))
head(report_regtr_models,5)
```

```
##      cp minsplit cp_minsplit      RMSE
## 1 1.000        5        1  5 146867.07
## 2 0.100        5      0.1  5 113666.72
## 3 0.010        5     0.01  5  96828.55
## 4 0.001        5    0.001  5  97246.15
## 5 0.000        5        0  5 100560.04
```

Plot line chart for regression tree line chart

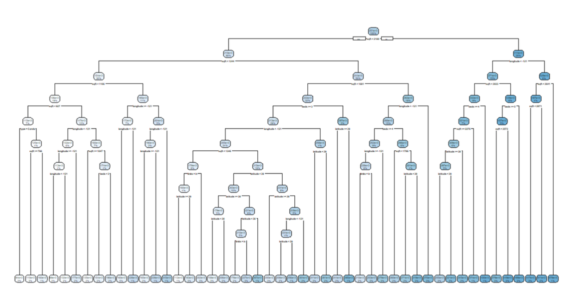
```
ggplot(data = report_regtr_models, aes(x = cp_minsplit, y = RMSE, group = 1))+
  geom_line(color = 'purple',size = 1.25) + geom_point(color = 'purple', size = 3)+
  theme(axis.text.x = element_text(angle = 90))
```




```
report_regtr_models[which.min(report_regtr_models$RMSE), 3]
```

```
## [1] "0.001  20"
```

```
rpart.plot(rpart(price ~ ., data = sac_est, cp = 0.001, minsplit = 20))
```



Part D: Plot RMSE value of best model for each 3 types of method

```
# choose from linear models:
which.min(report_reg_models[2,2:5])
```

```
## model_2_rmse
##            2
```

```
final_reg_model = lm(formula = price ~ ., data = sac_est)
final_reg_predict = predict(final_reg_model, sac_val)
# choose from knn models:
which.min(report_knn_models[,2])
```

```
## [1] 32
```

```
which.min(report_knn_models[,3])
```

```
## [1] 6
```

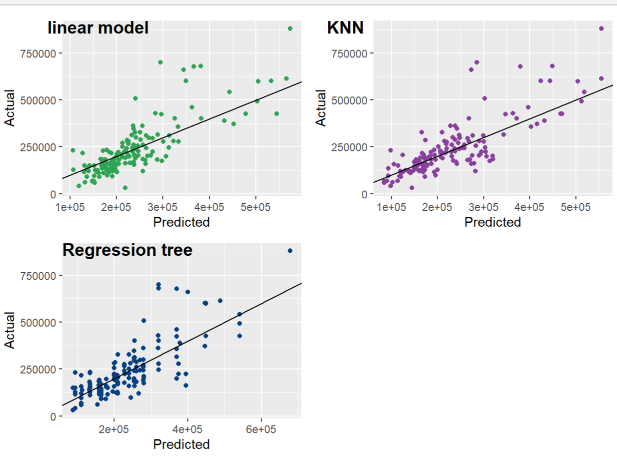
```
paste0('non_scaled:',min(report_knn_models[,2]), ' scaled:', min(report_knn_models[,3]))
```

```
## [1] "non_scaled:94543.5809894794 scaled:87550.4969798713"
```

```
final_knn_model = knnreg(formula = price ~ ., data = sac_est_scaled, k = 6)
final_knn_predict = predict(final_knn_model, sac_val_scaled)
# choose from regression tree models:
report_regtr_models[which.min(report_regtr_models$RMSE),3]
```

```
## [1] "0.001  20"
```

```
final_regtr_model  =rpart(formula = price ~ ., data = sac_est, cp = 0.001, minsplit = 20)
final_regtr_predict = predict(final_regtr_model, sac_val)

#par(mfrow = c(1,3))
# linear regression
plot1 = ggplot()+
  geom_point(aes(x = final_reg_predict, y = sac_val$price),  color = '#31A354')+
  geom_abline(intercept = 0, slope = 1)+
  labs(x = 'Predicted', y = 'Actual')
# knn
plot2 = ggplot()+
  geom_point(aes(x = final_knn_predict, y = sac_val$price),color = '#88419D')+
  geom_abline(intercept = 0, slope = 1)+
  labs(x = 'Predicted', y = 'Actual')
# regression tree
plot3 = ggplot()+
  geom_point(aes(x = final_regtr_predict, y = sac_val$price),color = '#084081')+
  geom_abline(intercept = 0, slope = 1)+
  labs(x = 'Predicted', y = 'Actual')

plot_grid(plot1,plot2,plot3, labels = c('linear model', 'KNN', 'Regression tree'))
```



We can say:

The lower RMSE value a model have, the closer distance(to the `y = x` line) its points (predicted, actual) have.

We can deduce that the RMSE of above plotted model are approximately the same as each other.

```
final_rmse = paste('Linear model: ', min(report_reg_models[2,2:5]),
                   'KNN: ', min(report_knn_models[,2]),
                   'KNN_S: ', min(report_knn_models[,3]),
                   'Regtr: ', min(report_regtr_models[,4]))
final_rmse
```

```
## [1] "Linear model:  89761.841927725 KNN:  94543.5809894794 KNN_S:  87550.4969798713 Regtr:  89127.3514891204"
```

Part E: RMSE of final model on test set

```
beds.s = scale(sac_trn$beds)
beds.center = attr(beds.s, "scaled:center")
beds.scale = attr(beds.s, "scaled:scale")

latitude.s = scale(sac_trn$latitude)
latitude.center = attr(latitude.s, "scaled:center")
latitude.scale = attr(latitude.s, "scaled:scale")

longitude.s = scale(sac_trn$longitude)
longitude.center = attr(longitude.s, "scaled:center")
longitude.scale = attr(longitude.s, "scaled:scale")

sqft.s = scale(sac_trn$sqft)
sqft.center = attr(sqft.s, "scaled:center")
sqft.scale = attr(sqft.s, "scaled:scale")

sac_trn_scaled = data.frame(beds_s = beds.s, sqft_s = sqft.s,
                            type = sac_trn$type, price = sac_trn$price,
                            latitude_s = latitude.s,
                            longitude_s = longitude.s,
                            limits = sac_trn$limits)
final_model = knnreg(formula = price ~ ., data = sac_trn_scaled, k = 6)

beds.s = scale(sac_tst$beds, center = beds.center, scale = beds.scale)
latitude.s = scale(sac_tst$latitude, center = latitude.center, scale = latitude.scale)
longitude.s = scale(sac_tst$longitude, center = longitude.center, scale = longitude.scale)
sqft.s = scale(sac_tst$sqft, center = sqft.center, scale = sqft.scale)

sac_tst_scaled = data.frame(beds_s = beds.s, sqft_s = sqft.s,
                            type = sac_tst$type, price = sac_tst$price,
                            latitude_s = latitude.s,
                            longitude_s = longitude.s,
                            limits = sac_tst$limits)

final_predict = predict(final_model, sac_tst_scaled)
final_rmse = rmse(final_predict, sac_tst$price)
final_rmse
```

```
## [1] 81090.39
```