HW 3_Group 6_DSCI 5340

Loading the Packages

```
pacman::p_load(caret, data.table, ggplot2, FNN, MASS)
knitr::opts_chunk$set(echo = TRUE, fig.width=12, fig.height=6)
theme_set(theme_classic())
options(digits = 3)
```

1.Using createDataPartition() function from the Caret package to partition the data into two parts – 80% into training data and 20% into test data.

```
set.seed(42)
partitionindex <- createDataPartition(y = Boston$medv, p = 0.8, list = FALSE)
train_Boston <- Boston[partitionindex, ]
test_Boston <- Boston[-partitionindex, ]
```

- The above chunk created data partition into two parts. training data set with 407 observation and 14 variables and the test data set with 99 observation and 14 variables.

2.Using train() function from the Caret package, run a k-NN model with medv as the response or target variable with

a. Standardize the dataset using center and scale options in the preProcess() function in the Caret package

```
set.seed(42)
prep <- preProcess(train_Boston, method = c("center", "scale"))
train_norm_Boston <- predict(prep, train_Boston)
test_norm_Boston <- predict(prep, test_Boston)
```

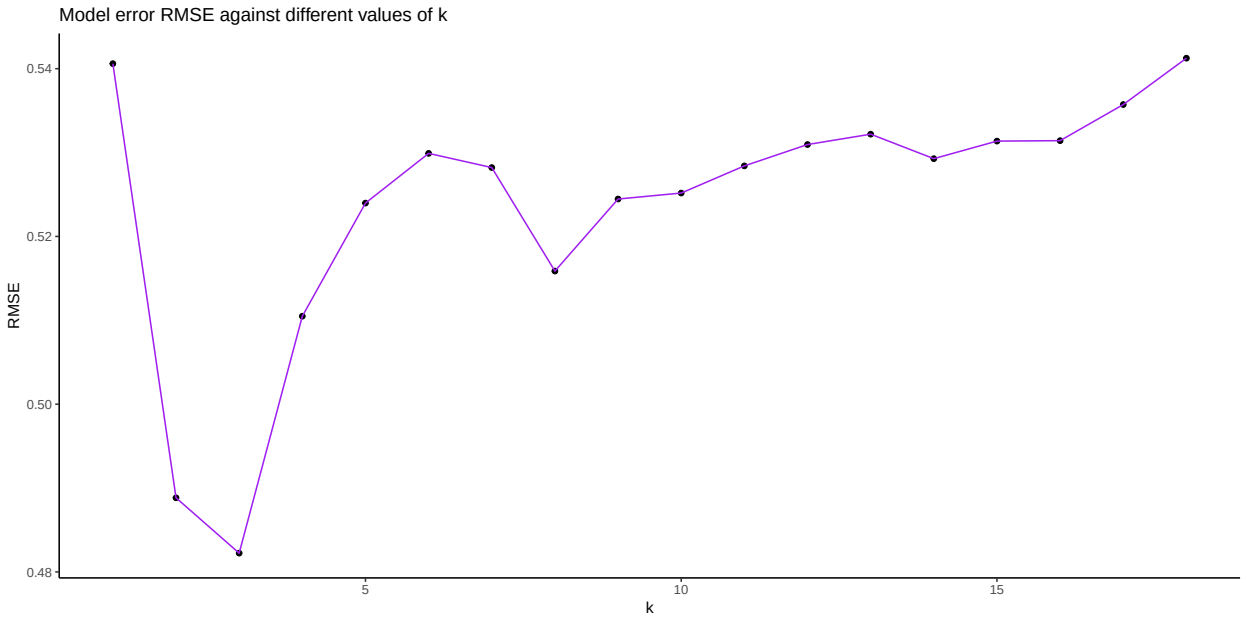
b. Use a 10-fold cross validation

```
set.seed(42)
kNN_model <- train(medv ~ .,
                   data = train_norm_Boston,
                   method = "knn",
                   tuneGrid = data.frame(k = 1:18),
                   preProcess = c("center", "scale"),
                   trControl = trainControl(method = "cv", number = 10))
kNN_model
```

```
## k-Nearest Neighbors
##
## 407 samples
##  13 predictor
##
## Pre-processing: centered (13), scaled (13)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 367, 365, 366, 366, 367, 366, ...
## Resampling results across tuning parameters:
##
##   k   RMSE   Rsquared  MAE
##    1  0.541  0.717     0.338
##    2  0.489  0.760     0.307
##    3  0.482  0.768     0.312
##    4  0.510  0.742     0.325
##    5  0.524  0.726     0.331
##    6  0.530  0.722     0.332
##    7  0.528  0.722     0.335
##    8  0.516  0.736     0.333
##    9  0.524  0.729     0.341
##   10  0.525  0.732     0.340
##   11  0.528  0.731     0.342
##   12  0.531  0.729     0.346
##   13  0.532  0.730     0.348
##   14  0.529  0.733     0.347
##   15  0.531  0.733     0.348
##   16  0.531  0.735     0.352
##   17  0.536  0.733     0.356
##   18  0.541  0.727     0.360
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 3.
```

3.Generate a plot showing model error RMSE against different values of k.

```
ggplot(kNN_model$results, mapping = aes(x = k, y = RMSE)) +
  geom_point(colour = "black") +
  geom_line(colour = "purple") +
  ggtitle("Model error RMSE against different values of k")
```



4.Choosing the optimal value of k and Explanation

```
kNN_model$results
```

```
##     k  RMSE Rsquared   MAE RMSESD RsquaredSD  MAESD
## 1   1 0.541    0.717 0.338 0.1222     0.1231 0.0588
## 2   2 0.489    0.760 0.307 0.1043     0.1045 0.0385
## 3   3 0.482    0.768 0.312 0.0679     0.0880 0.0298
## 4   4 0.510    0.742 0.325 0.0691     0.0989 0.0387
## 5   5 0.524    0.726 0.331 0.0887     0.1205 0.0506
## 6   6 0.530    0.722 0.332 0.0873     0.1229 0.0491
## 7   7 0.528    0.722 0.335 0.0866     0.1281 0.0456
## 8   8 0.516    0.736 0.333 0.0865     0.1288 0.0433
## 9   9 0.524    0.729 0.341 0.0875     0.1334 0.0436
## 10 10 0.525    0.732 0.340 0.0876     0.1341 0.0445
## 11 11 0.528    0.731 0.342 0.0842     0.1312 0.0424
## 12 12 0.531    0.729 0.346 0.0888     0.1366 0.0441
## 13 13 0.532    0.730 0.348 0.0894     0.1394 0.0436
## 14 14 0.529    0.733 0.347 0.0918     0.1413 0.0441
## 15 15 0.531    0.733 0.348 0.0899     0.1426 0.0433
## 16 16 0.531    0.735 0.352 0.0907     0.1431 0.0457
## 17 17 0.536    0.733 0.356 0.0923     0.1445 0.0462
## 18 18 0.541    0.727 0.360 0.0904     0.1470 0.0438
```

```
kNN_model$bestTune
```

```
##   k
## 3 3
```

- RMSE was used to select the optimal model using the smallest value. Based on the results from the model, the smallest value is 0.482 when k = 3. We also used best Tune from the model which also suggest the optimal value at k = 3.