KNN Practice

Bao Doquang, Dhwanit Agarwal, Akksay Singh and Shristi Singh

March 13, 2020

```
library(foreach)
```

```
## Warning: package 'foreach' was built under R version 3.6.3
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.6.2
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
library(class)
library(mosaic)
```

```
## Warning: package 'mosaic' was built under R version 3.6.2
```

```
## Warning: package 'ggstance' was built under R version 3.6.2
```

```
## Warning: package 'mosaicData' was built under R version 3.6.2
```

```
library(FNN)
```

```
## Warning: package 'FNN' was built under R version 3.6.2
```

```
sclass = read.csv('../Question1/sclass.csv')

sclass350 = subset(sclass, trim == '350')
sclass65AMG = subset(sclass, trim == '65 AMG')
```

(1) First I created subsets of the data which only included observations with 'trim' of 350 and 65 AMD and separated them thusly.

```
n350 = nrow(sclass350)
n_train350 = round(0.8*n350)
n_test350 = n350 - n_train350
train_cases350 = sample.int(n350, n_train350, replace=FALSE)
test_cases350 = setdiff(1:n350, train_cases350)
sclass350_train = sclass350[train_cases350,]
sclass350_test = sclass350[test_cases350,]

n65AMG = nrow(sclass65AMG)
n_train65AMG = round(0.8*n65AMG)
n_test65AMG = n65AMG - n_train65AMG
train_cases65AMG = sample.int(n65AMG, n_train65AMG, replace=FALSE)
test_cases65AMG = setdiff(1:n65AMG, train_cases65AMG)
sclass65AMG_train = sclass65AMG[train_cases65AMG,]
sclass65AMG_test = sclass65AMG[test_cases65AMG,]
```

(2) Here, I created training and testing sets for each of the individual subsets.

```
Xtrain350 = model.matrix(~ mileage, data=sclass350_train)
Xtest350 = model.matrix(~ mileage, data=sclass350_test)

ytrain350 = sclass350_train$price
ytest350 = sclass350_test$price

Xtrain65AMG = model.matrix(~ mileage, data=sclass65AMG_train)
Xtest65AMG = model.matrix(~ mileage, data=sclass65AMG_test)

ytrain65AMG = sclass65AMG_train$price
ytest65AMG = sclass65AMG_test$price
```

(3) This step splits the feature of the model from the outcome; however this model only has 1 model feature, mileage.

```
K = 2
rmse = function(y, yhat) {
  sqrt( mean( (y - yhat)^2 ) )
}
```

(4) Created the root mean square error helper function.

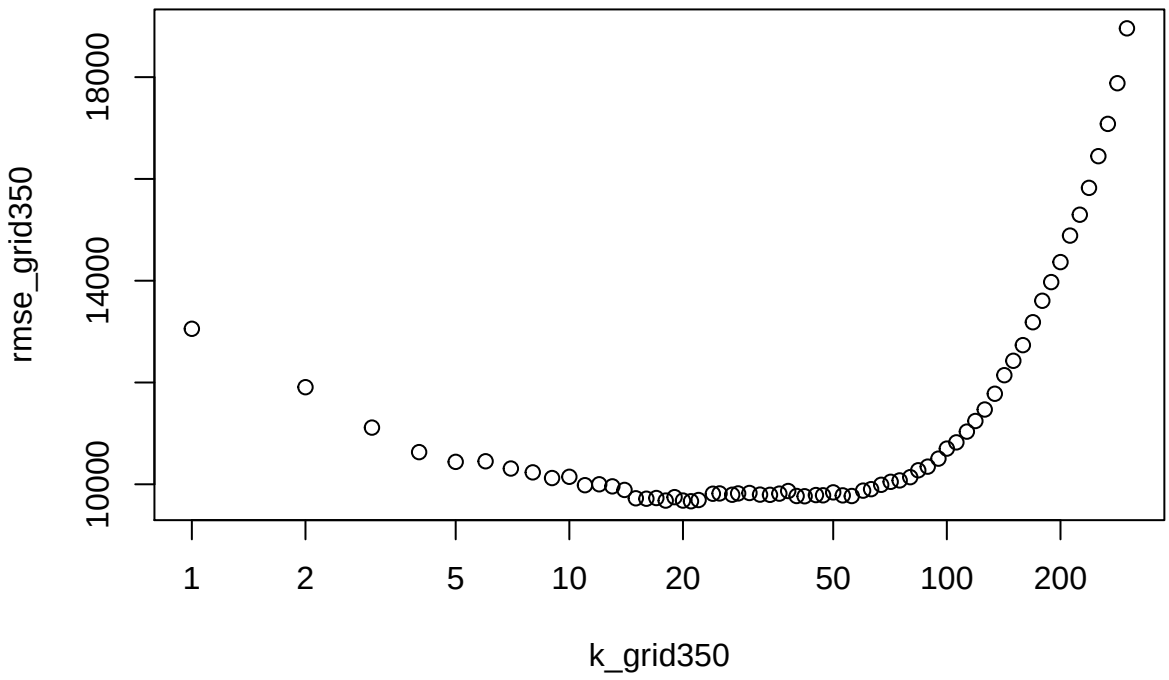
```
knn_model350 = knn.reg(Xtrain350, Xtest350, ytrain350, k=K)

rmse(ytest350, knn_model350$pred)
```

```
## [1] 11907.86
```

```
k_grid350 = exp(seq(log(1), log(300), length=100)) %>% round %>% unique
rmse_grid350 = foreach(K = k_grid350, .combine='c') %do% {
  knn_model350 = knn.reg(Xtrain350, Xtest350, ytrain350, k=K)
  rmse(ytest350, knn_model350$pred)
}

plot(k_grid350, rmse_grid350, log='x')
```

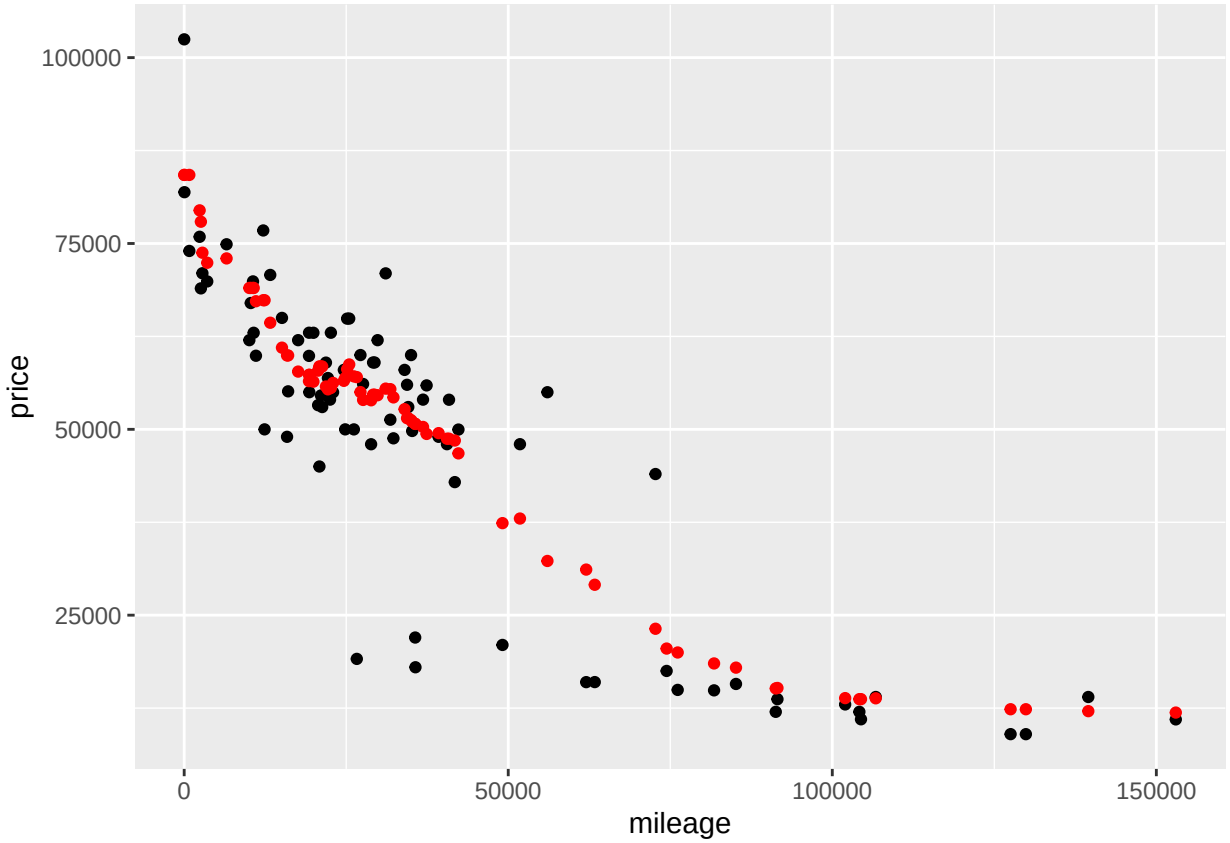



```
k_grid350[which.min(rmse_grid350)]
```

```
## [1] 21
```

```
finalknn350 = knn.reg(train = Xtrain350, test = Xtest350, y = ytrain350, k=k_grid350[which.min(rmse_gri
)
ypred_finalknn350 = finalknn350$pred

sclass350_test$ypred_finalknn350 = ypred_finalknn350
p_test350 = ggplot(data = sclass350_test) +
  geom_point(mapping = aes(x = mileage, y = price))
p_test350 + geom_point(aes(x = mileage, y = ypred_finalknn350), color='red')
```



(5) Here, we define the predictive model as a K nearest neighbors, train on our training set, then plot RMSE from our helper function against various levels of K. From this we take the level of K with the lowest available error to use in our final predictive model where we plot the predictions of our model that was developed with the training set, against the testing set.

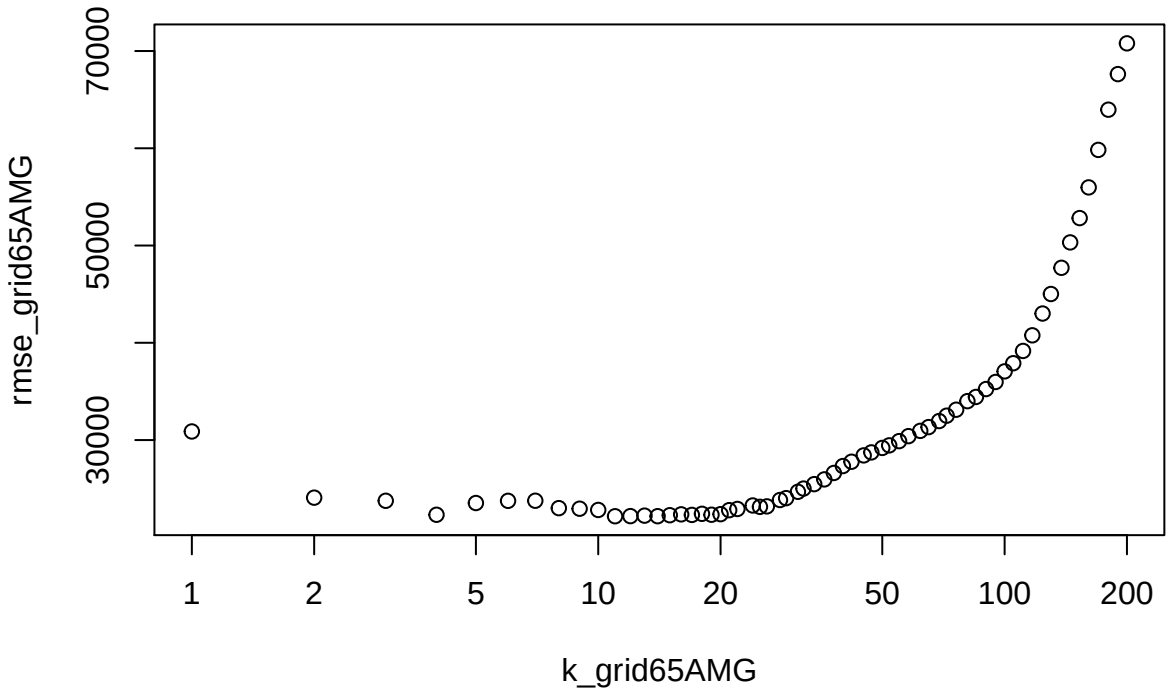
```
K =2
knn_model65AMG = knn.reg(Xtrain65AMG, Xtest65AMG, ytrain65AMG, k=K)

rmse(ytest65AMG, knn_model65AMG$pred)
```

```
## [1] 24077.63
```

```
k_grid65AMG = exp(seq(log(1), log(200), length=100)) %>% round %>% unique
rmse_grid65AMG = foreach(K = k_grid65AMG, .combine='c') %do% {
  knn_model65AMG = knn.reg(Xtrain65AMG, Xtest65AMG, ytrain65AMG, k=K)
  rmse(ytest65AMG, knn_model65AMG$pred)
}

plot(k_grid65AMG, rmse_grid65AMG, log='x')
```

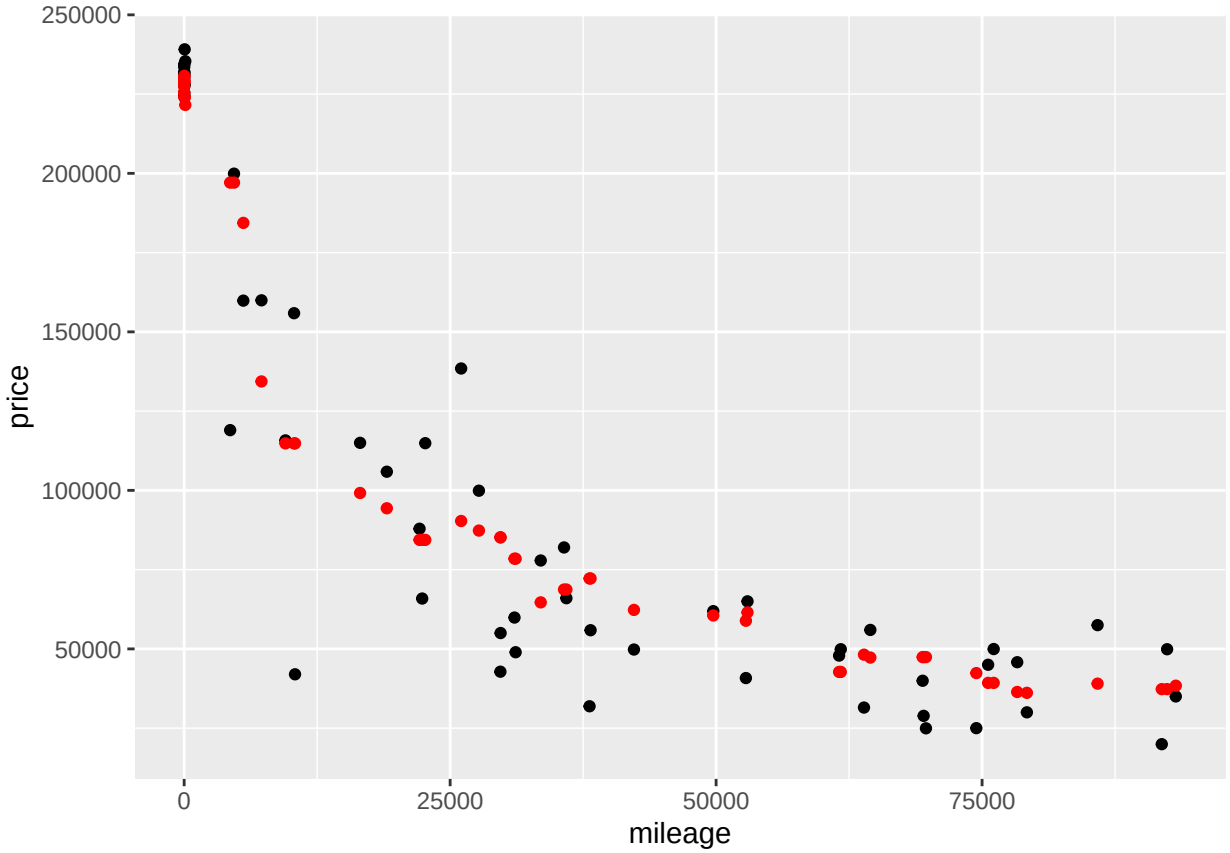



```
k_grid65AMG[which.min(rmse_grid65AMG)]
```

```
## [1] 11
```

```
finalknn65AMG = knn.reg(train = Xtrain65AMG, test = Xtest65AMG, y = ytrain65AMG, k=k_grid65AMG[which.mi
)
ypred_finalknn65AMG = finalknn65AMG$pred

sclass65AMG_test$ypred_finalknn65AMG = ypred_finalknn65AMG
p_test65AMG = ggplot(data = sclass65AMG_test) +
  geom_point(mapping = aes(x = mileage, y = price))
p_test65AMG + geom_point(aes(x = mileage, y = ypred_finalknn65AMG), color='red')
```



(6) Step '5' is repeated for the second subset. (7) Trim 350 appears to consistently have a higher optimal value of K versus the subset of trim 65 AMG. This may be due to the fact that the subset of 65 AMG is significantly smaller than the subset of trim 350; specifically the subset 350 having 416 observations while the subset 65 AMG has 292 observations. This increase allows for a higher value of K to be chosen while maintaining a similar proportion of datum chosen in the KNN model.