ECO 395M: Exercises 1

Areeya Aksornpan

2/7/2021

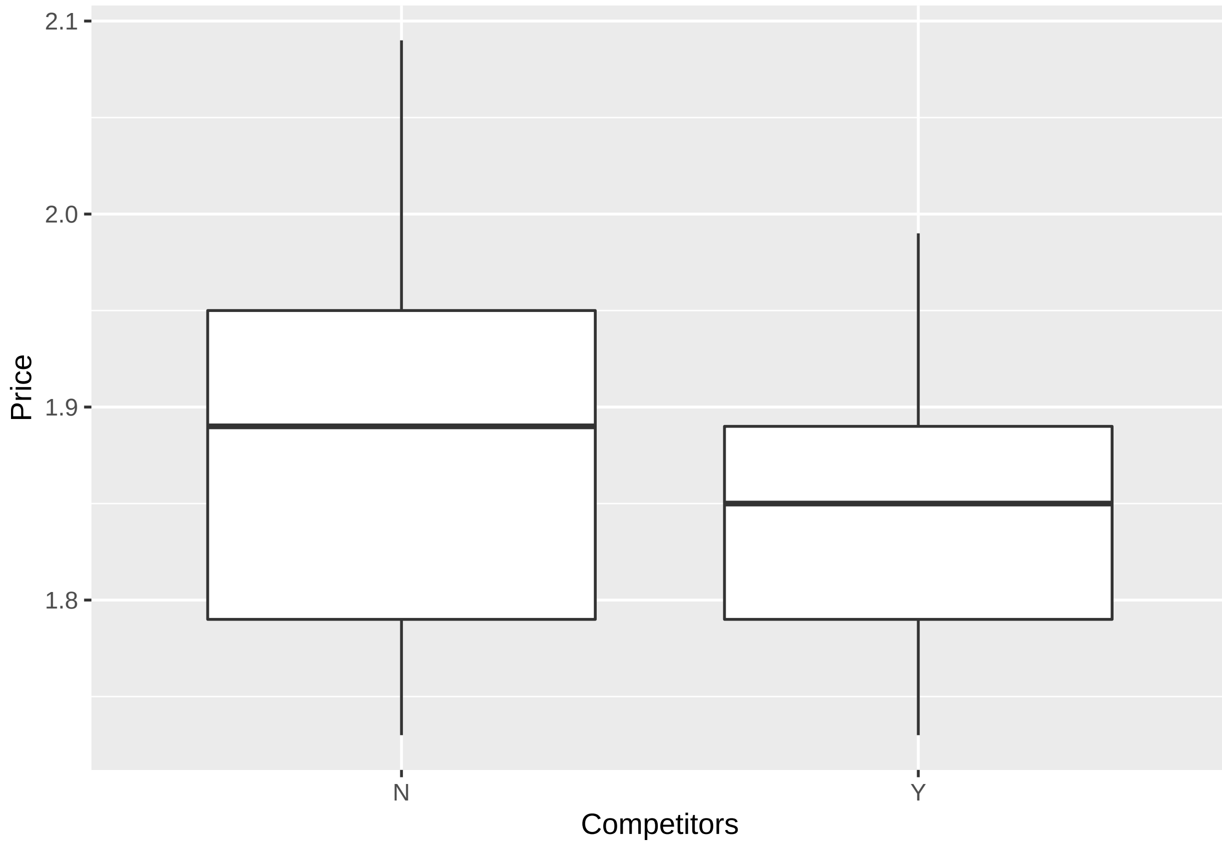
R Markdown

##1) Data visualization: gas prices

```r
library(ggplot2)
library(tidyverse)
```

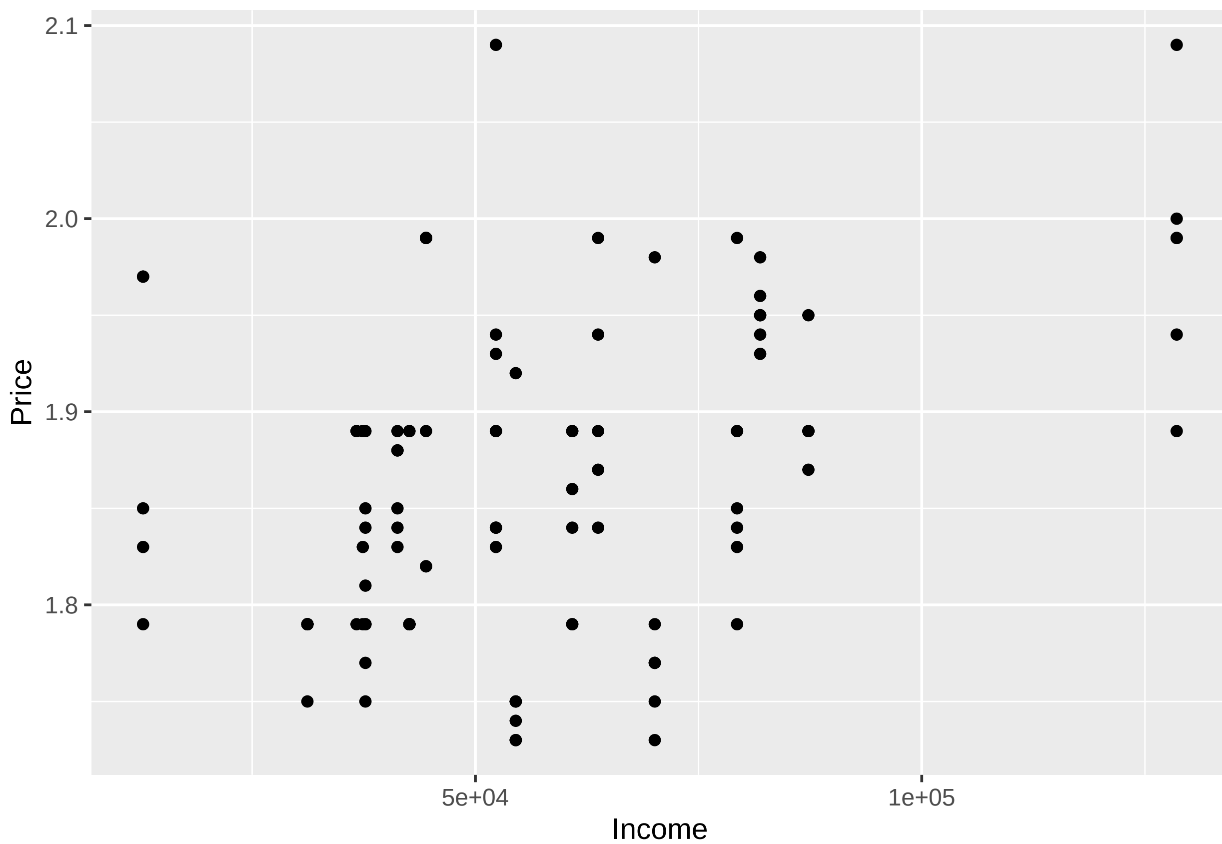
```
## -- Attaching packages ------------------------------------------- tidyverse 1.3.0 --

## v tibble  3.0.3     v dplyr   1.0.1
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## v purrr   0.3.4

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

```r
GasPrices = read.csv('~/Desktop/GasPrices.csv')

##A
ggplot(GasPrices, aes(x=Competitors, y=Price)) + geom_boxplot()
```



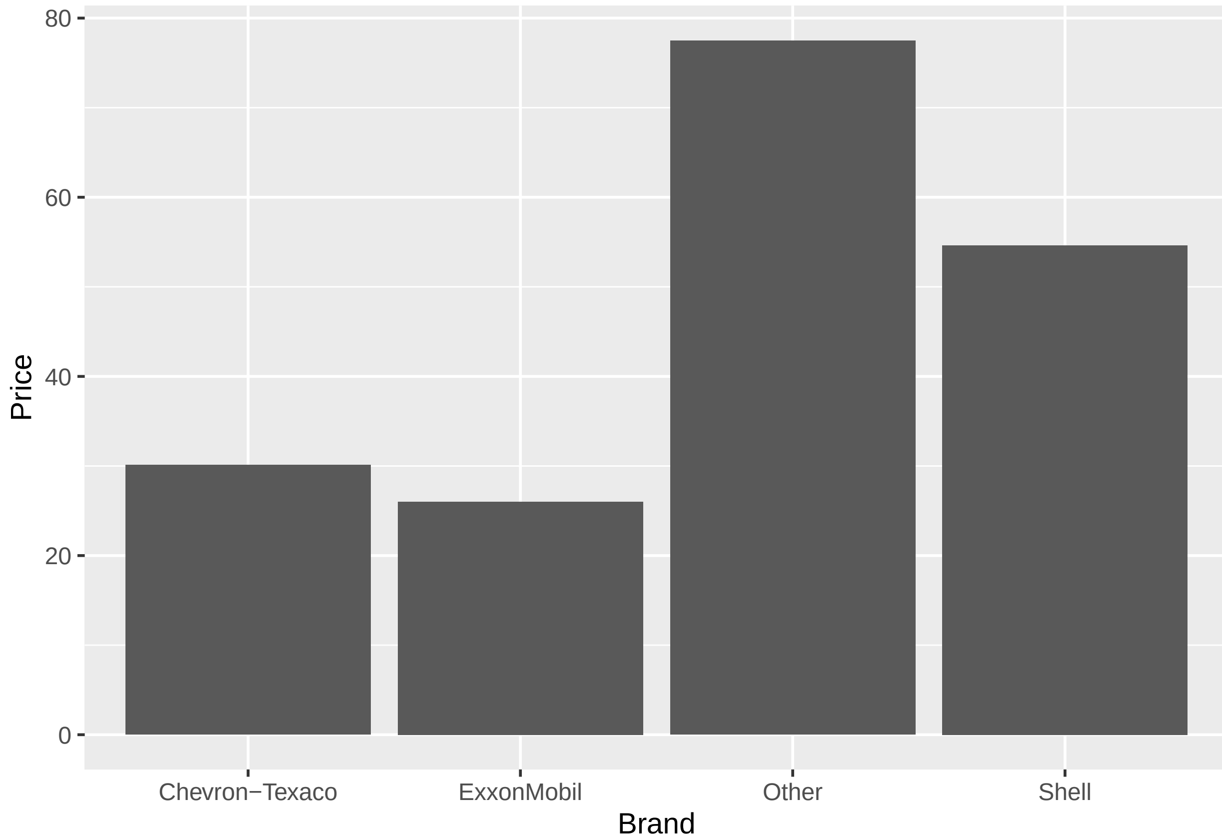
This boxplot shows that gas stations charge more if they lack direct competition in sight. When there are competitors, the maximum and average price decreases.

```r
##B
ggplot(GasPrices, aes(x=Income, y=Price)) + geom_point()
```



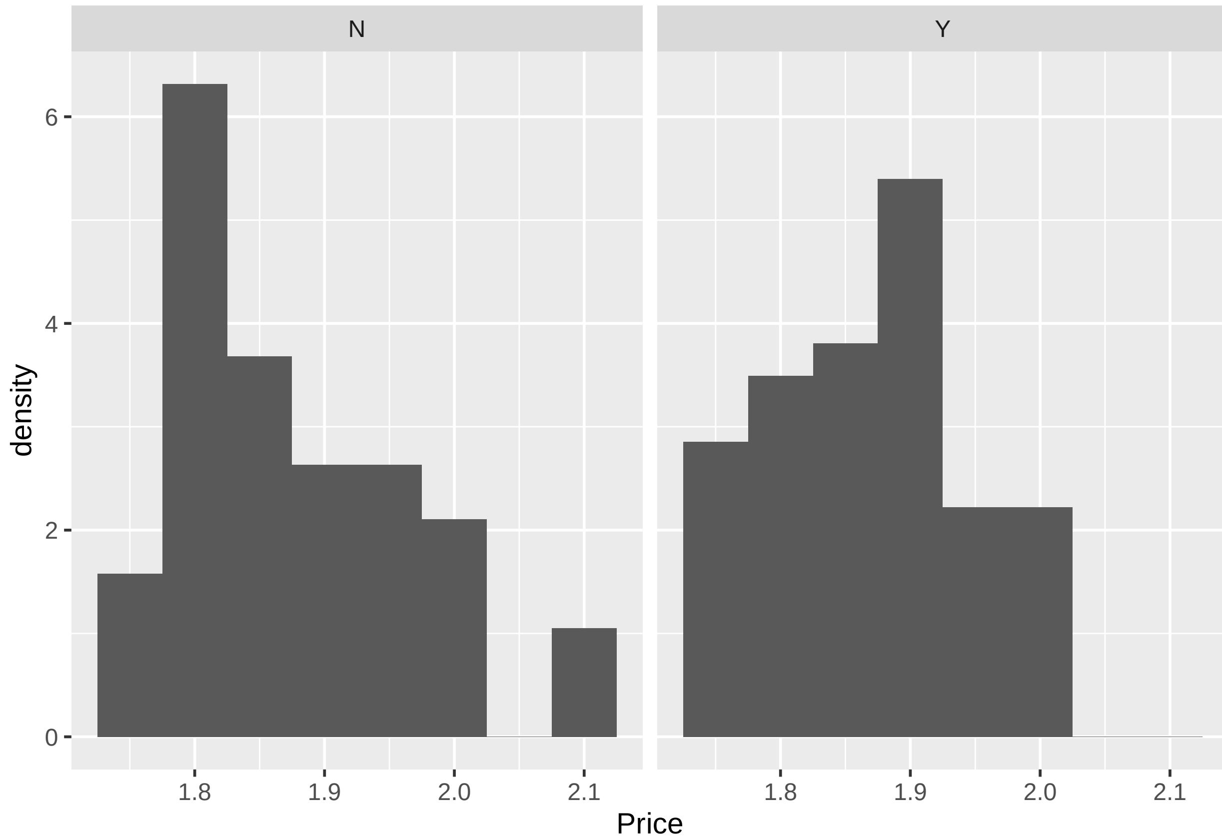
This scatter plot shows that the richer the area, the higher the gas price. Lower gas prices are not sold in the richer area. There is a positive correlation between price and income.

```r
##C
ggplot(GasPrices, aes(x=Brand, y=Price)) + geom_col()
```



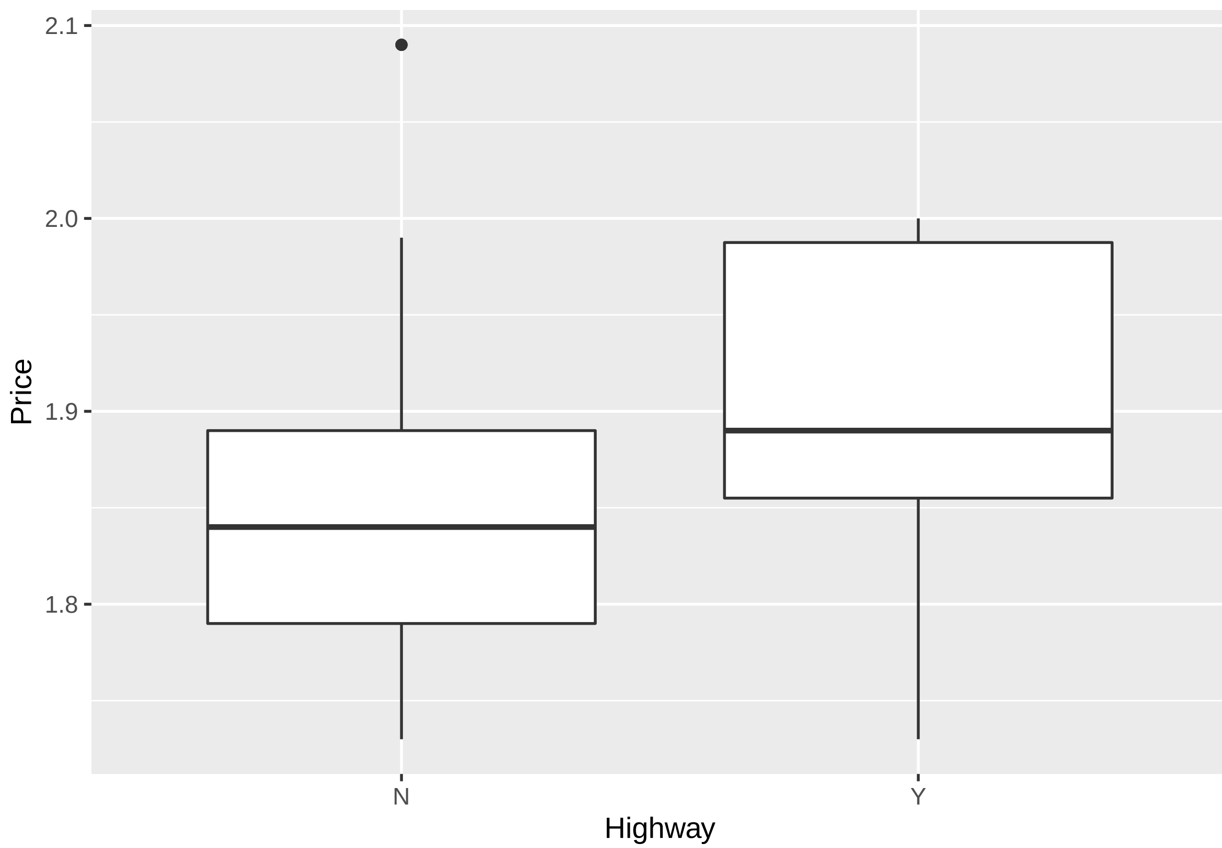
Although it is claimed that Shell charges more than other brands, this bar plot shows that the theory is unsupported by the data. Shell's price is higher than Chevron-Texaco and ExxonMobil, but there are other brands that sell gas in a higher price compare to Shell.

```
##D
ggplot(GasPrices) +
  geom_histogram(aes(x=Price, after_stat(density)), binwidth = 0.05) +
  facet_wrap(~Stoplight)
```



This faceted histogram shows that gas stations at stoplights charge more. Gas are sold the most (high frequency) at $1.8 when there's no stoplight and $1.9 at stoplights.

```
##E
ggplot(GasPrices, aes(x=Highway, y=Price)) + geom_boxplot()
```



This boxplot shows that gas stations with direct highway access charge more. The average price increases when there is direct highway access to the gas station. The minimum price increases from below \$1.8 to approximately \$1.85 and the maximum price increases from nearly \$1.9 to close to \$2.0.

##2) Data visualization: a bike share network

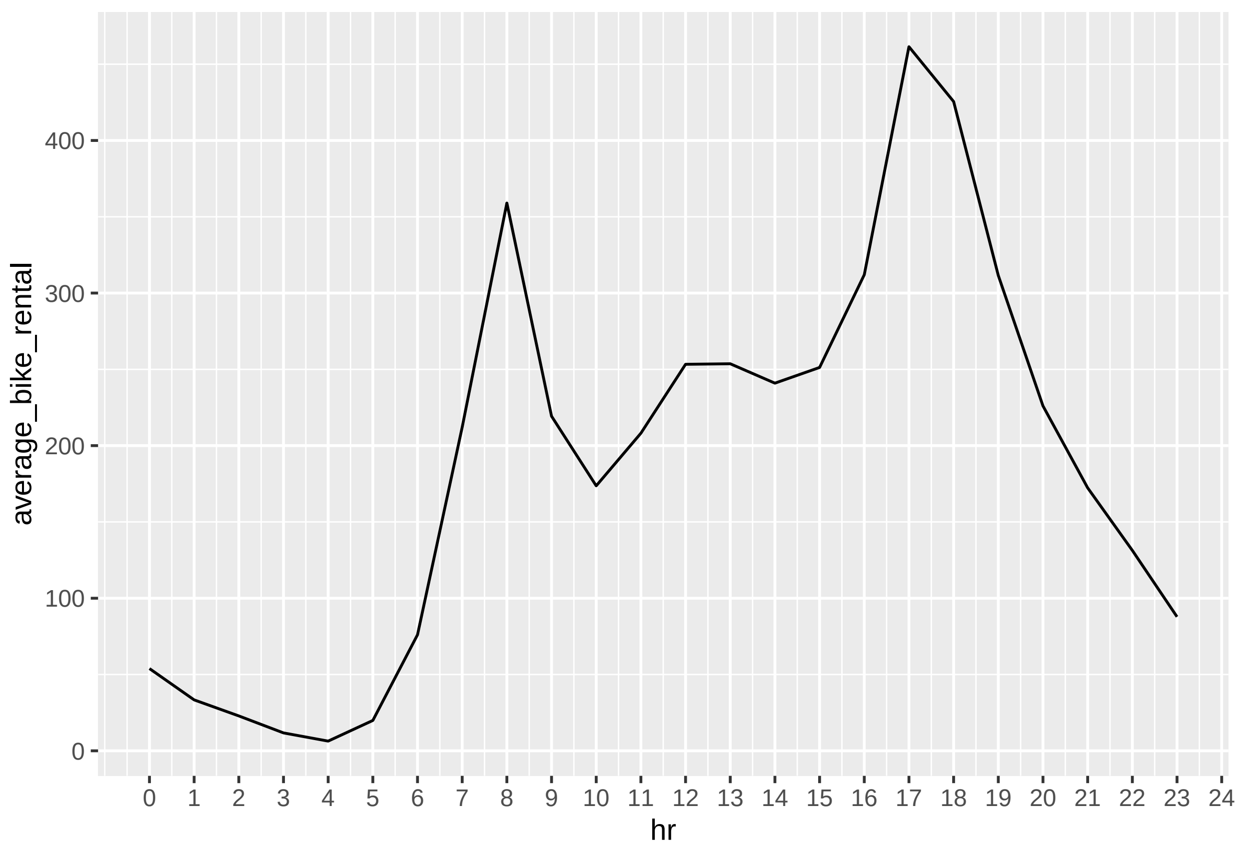
```r
library(ggplot2)
library(tidyverse)

bikeshare = read.csv('~/Desktop/bikeshare.csv')

head(bikeshare)
```

```
##   instant     dteday season yr mnth hr holiday weekday workingday weathersit
## 1       1 2011-01-01      1  0    1  0       0       6          0          1
## 2       2 2011-01-01      1  0    1  1       0       6          0          1
## 3       3 2011-01-01      1  0    1  2       0       6          0          1
## 4       4 2011-01-01      1  0    1  3       0       6          0          1
## 5       5 2011-01-01      1  0    1  4       0       6          0          1
## 6       6 2011-01-01      1  0    1  5       0       6          0          2
##   temp total
## 1 0.24    16
## 2 0.22    40
## 3 0.22    32
## 4 0.24    13
## 5 0.24     1
## 6 0.24     1
```

```r
##Plot A: a line graph showing average bike rentals (total) versus hour of the day (hr).

#Average bike rentals
bikerent_total1 = bikeshare %>%
  group_by(hr) %>%
  summarize(average_bike_rental = mean(total))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
#Plot the result over time in a line graph
ggplot(bikerent_total1) +
  geom_line(aes(x=hr, y=average_bike_rental)) + scale_x_continuous(breaks = 0:24)
```



The x-axis is the hour which bikers rent bicycles and the y-axis is the average number of total bike rentals in that hour, including both casual and registered users.

Bicycle renters prefer to rent bicycles mostly around 8am and 5pm, which is before and after their working hours. There is also a slight increase from 10am to 12pm, which is when workers could have their lunch breaks. It also shows that people started leaving the house around 5am and going back home around 6pm. We could expect that if we attempt to rent a bicycle at 8am or 5pm, there is a high possibility that there is no bicycle available.

```r
##Plot B: a faceted line graph showing average bike rentals versus hour of the day, faceted according t


bikerent_total2 = bikeshare %>%
```

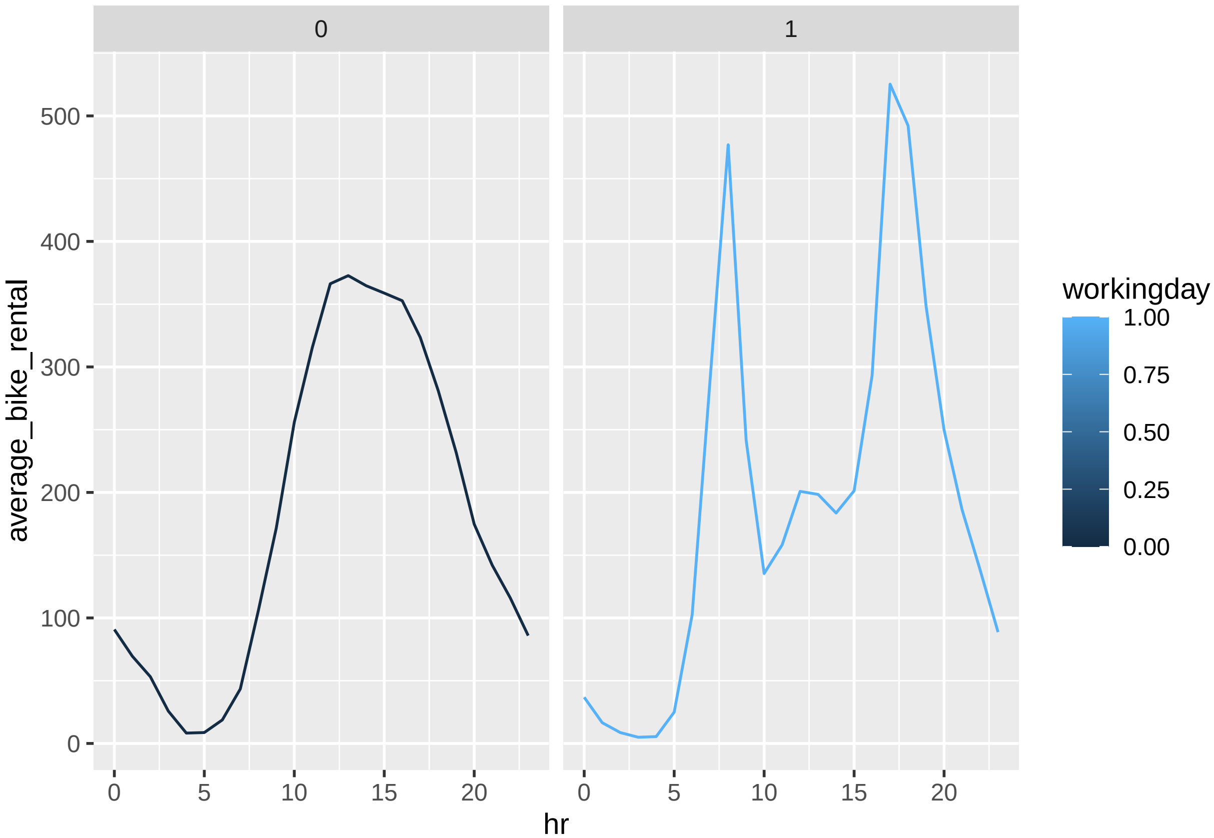
```r
  group_by(hr, workingday) %>%
  summarize(average_bike_rental = mean(total))
```

```
## `summarise()` regrouping output by 'hr' (override with `.groups` argument)
```

```r
head(bikerent_total2, 30)
```

```
## # A tibble: 30 x 3
## # Groups:   hr [15]
##       hr workingday average_bike_rental
##    <int>      <int>               <dbl>
##  1     0          0               90.8
##  2     0          1               36.8
##  3     1          0               69.5
##  4     1          1               16.6
##  5     2          0               53.2
##  6     2          1                8.68
##  7     3          0               25.8
##  8     3          1                4.94
##  9     4          0                8.26
## 10     4          1                5.43
## # ... with 20 more rows
```

```r
ggplot(bikerent_total2) +
  geom_line(aes(x=hr, y=average_bike_rental, color=workingday)) +
  facet_wrap(~workingday)
```



The x-axis is the hour which bikers rent bicycles and the y-axis is the average number of total bike rentals in that hour, including both casual and registered users.

The left graph is the average bike rentals versus hour of weekend or holiday. Bicycle renters prefer to rent bicycles mostly around noon. We can assume that people started leaving the house around 6am and going back home around 1pm.

The right graph is the average bike rentals versus hour of workingday. Bicycle renters prefer to rent bicycles mostly around 8am and 5pm, which is before and after working hours. We can assume that people started leaving the house around 5am and going back home around 6pm.

```
##Plot C: a faceted bar plot showing average ridership during the 8 AM hour by weather situation code (

bikerent_total3 = bikeshare %>%
  filter(hr==8) %>%
group_by(weathersit, workingday) %>%
  summarise(average_bike_rental = mean(total))
```

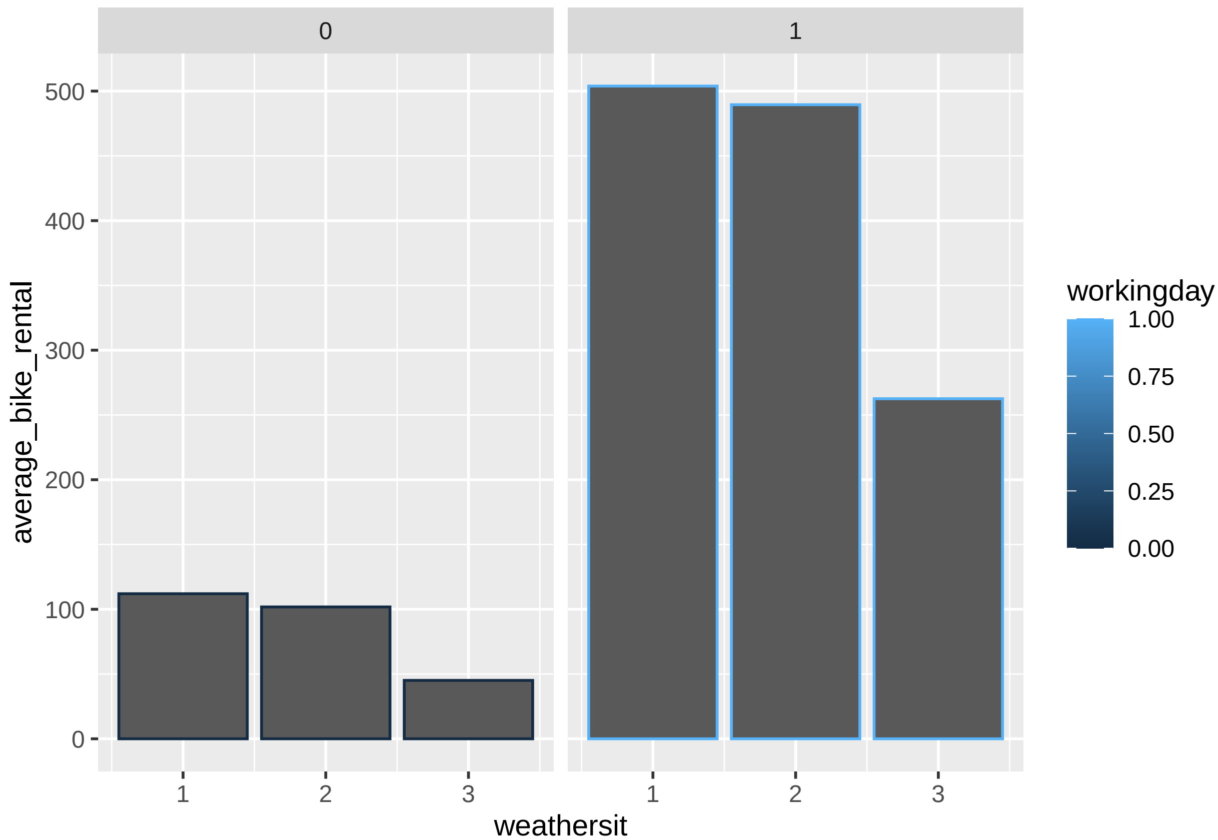
```
## `summarise()` regrouping output by 'weathersit' (override with `.groups` argument)
```

```
head(bikerent_total3, 30)
```

```
## # A tibble: 6 x 3
## # Groups:   weathersit [3]
##   weathersit workingday average_bike_rental
##        <int>      <int>               <dbl>
## 1          1          0                112.
```

```
## 2           1          1                 504.
## 3           2          0                 102.
## 4           2          1                 489.
## 5           3          0                  45.1
## 6           3          1                 263.
```

```
ggplot(bikerent_total3) +
  geom_col(aes(x=weathersit, y=average_bike_rental, color=workingday)) +
  facet_wrap(~workingday)
```



The y-axis is average ridership during the 8 AM hour and the x-axis is the weather situation which is sorted as follow

1: Clear, Few clouds, Partly cloudy, Partly cloudy

2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist

3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds

4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

The left graph is the average bike rentals versus weather situation on weekends or holidays, while the right graph is the average bike rentals versus weather situation on workdays.

Numbers of bike rentals on both graphs decreased as the weather situation got worsened. When there is light snow, light rain with scattered clouds or thunderstorm (3), the numbers of average bike rentals lessened by half. When it is mist(2), the number of average bike rental does not decrease much compare to when it is clear or cloudy (1). Since the weather condition lessens the number of bike rentals, we could expect a fewer number of bike rentals on a snowy or rainy day.

##3) Data visualization: flights at ABIA

```r
library(ggplot2)
library(tidyverse)

ABIA = read.csv('~/Desktop/ABIA.csv')

head(ABIA)
```

```
##   Year Month DayofMonth DayOfWeek DepTime CRSDepTime ArrTime CRSArrTime
## 1 2008     1          1         2     120       1935     309       2130
## 2 2008     1          1         2     555        600     826        835
## 3 2008     1          1         2     600        600     728        729
## 4 2008     1          1         2     601        605     727        750
## 5 2008     1          1         2     601        600     654        700
## 6 2008     1          1         2     636        645     934        932
##   UniqueCarrier FlightNum TailNum ActualElapsedTime CRSElapsedTime AirTime
## 1            9E      5746  84129E               109            115      88
## 2            AA      1614  N438AA               151            155     133
## 3            YV      2883  N922FJ               148            149     125
## 4            9E      5743  89189E                86            105      70
## 5            AA      1157  N4XAAA                53             60      38
## 6            NW      1674   N967N               178            167     145
##   ArrDelay DepDelay Origin Dest Distance TaxiIn TaxiOut Cancelled
## 1      339      345    MEM  AUS      559      3      18         0
## 2       -9       -5    AUS  ORD      978      7      11         0
## 3       -1        0    AUS  PHX      872      7      16         0
## 4      -23       -4    AUS  MEM      559      4      12         0
## 5       -6        1    AUS  DFW      190      5      10         0
## 6        2       -9    AUS  MSP     1042     11      22         0
##   CancellationCode Diverted CarrierDelay WeatherDelay NASDelay SecurityDelay
## 1                         0          339            0        0             0
## 2                         0           NA           NA       NA            NA
## 3                         0           NA           NA       NA            NA
## 4                         0           NA           NA       NA            NA
## 5                         0           NA           NA       NA            NA
## 6                         0           NA           NA       NA            NA
##   LateAircraftDelay
## 1                 0
## 2                NA
## 3                NA
## 4                NA
## 5                NA
## 6                NA
```

```r
##What is the best time of year to fly to minimize delays, and does this change by destination?

ABIA_DepDelay1 = ABIA %>%
  group_by(Month) %>%
  summarize(ABIA_total1 = mean(na.omit(DepDelay)))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

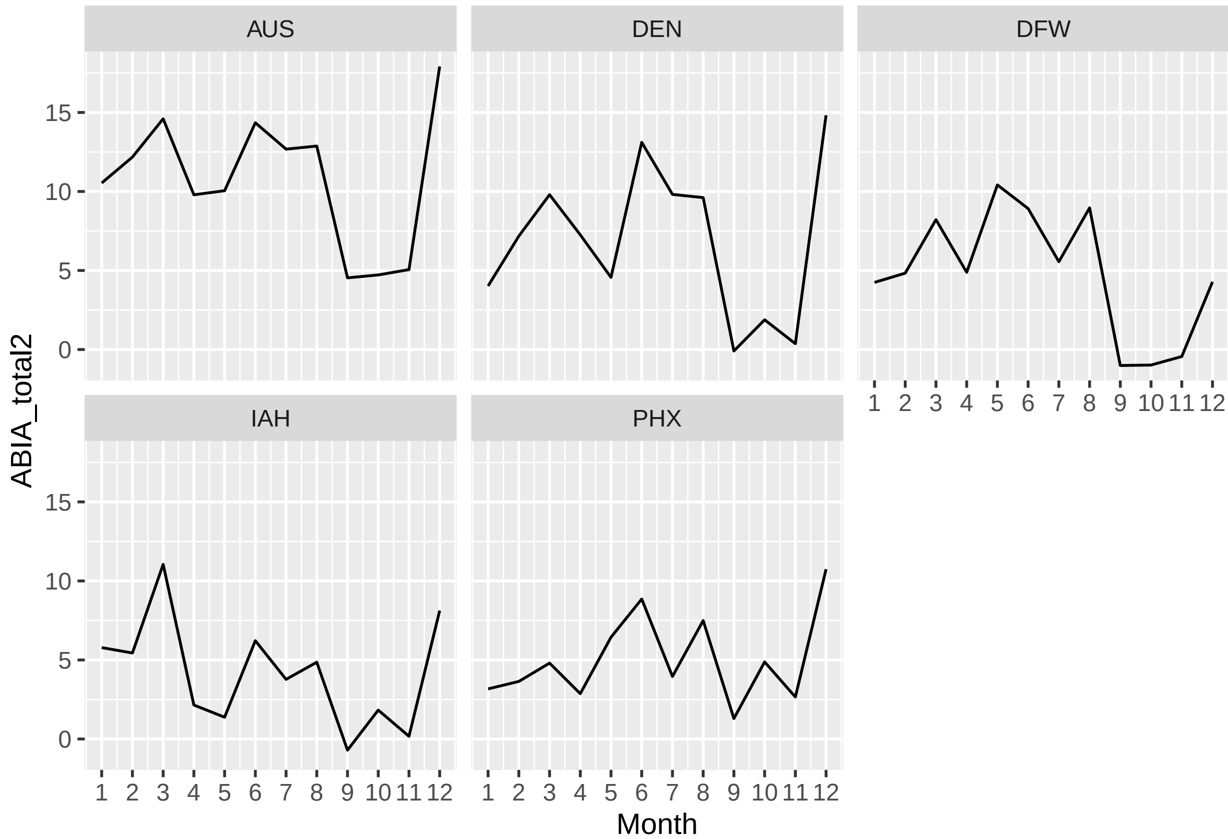
```r
Desination = c('AUS', 'DFW', 'IAH', 'PHX', 'DEN')

ABIA_DepDelay2 = ABIA %>%
  filter(Dest %in% Desination) %>%
  group_by(Month, Dest) %>%
  summarize(ABIA_total2 = mean(na.omit(DepDelay)))
```

```
## `summarise()` regrouping output by 'Month' (override with `.groups` argument)
```

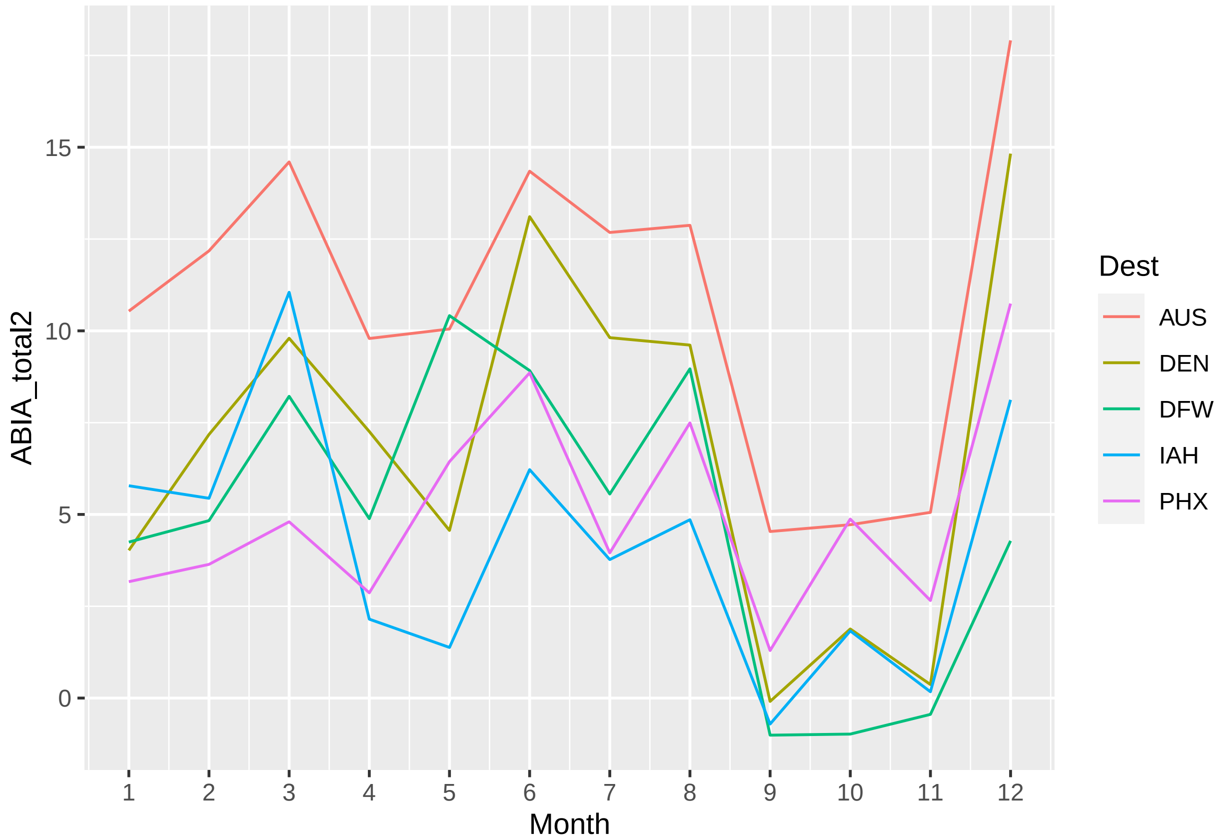
```r
head(ABIA_DepDelay1, 100)
```

```
## # A tibble: 12 x 2
##    Month ABIA_total1
##    <int>       <dbl>
##  1     1        8.37
##  2     2       10.3
##  3     3       13.3
##  4     4        8.17
##  5     5        8.85
##  6     6       12.2
##  7     7       10.2
##  8     8       10.3
##  9     9        3.33
## 10    10        3.88
## 11    11        4.36
## 12    12       15.3
```

```r
ggplot(ABIA_DepDelay2) +
  geom_line(aes(x=Month, y=ABIA_total2)) +
  facet_wrap(~Dest) +
  scale_x_continuous(breaks = 1:12)
```




```
ggplot(ABIA_DepDelay2) +
  geom_line(aes(x=Month, y=ABIA_total2, color=Dest)) +
  scale_x_continuous(breaks = 1:12)
```



All these five airports commonly have the least amount of delays in September and the most amount of delays in December, which refers that the destination does not affect the departure time and its' delay. It's possible that the weather is a major factor in delay. It could alternatively be the air traffic in December since it is the peak time of the high season.

##4) K-nearest neighbors

```
library(tidyverse)
library(ggplot2)
library(rsample)
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(modelr)
library(parallel)
library(foreach)
```

```
##
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```r
sclass = read.csv('~/Desktop/sclass.csv')


##350
model350 = sclass %>%
  filter(trim %in% '350')

#1.Split the data into a training and a testing set.

sclass350_split =  initial_split(model350, prop=0.9)
sclass350_train = training(sclass350_split)
sclass350_test  = testing(sclass350_split)


#2.Run K-nearest-neighbors, for many different values of K, starting at K=2 and going as high as you ne

K_folds = 5


model350 = model350 %>%
  mutate(fold_id = rep(1:K_folds, length=nrow(model350)) %>% sample)

head(model350)
```

```
##    id trim subTrim condition isOneOwner mileage year  color displacement   fuel
## 1 282  350    unsp       CPO          f   21929 2012  Black        3.0 L Diesel
## 2 284  350    unsp       CPO          f   17770 2012 Silver        3.0 L Diesel
## 3 285  350    unsp      Used          f   29108 2012  Black        3.0 L Diesel
## 4 288  350    unsp       CPO          f   35004 2013  White        3.0 L Diesel
## 5 289  350    unsp      Used          t   66689 2012  Black        3.0 L Diesel
## 6 290  350    unsp       CPO          f   19567 2012  Black        3.0 L Diesel
##   state region   soundSystem wheelType wheelSize featureCount price fold_id
## 1    MA    New          unsp      unsp      unsp           82 55994       4
## 2    IL    ENC       Premium     Alloy      unsp           72 60900       4
## 3    VA    SoA          unsp      unsp      unsp            5 54995       5
## 4    NH    New Harman Kardon      unsp      unsp           83 59988       1
## 5    NJ    Mid Harman Kardon     Alloy      unsp           79 37995       4
## 6    LA    WSC       Premium     Alloy      unsp           76 59977       3
```

```r
#3.Calculate the out-of-sample root mean-squared error (RMSE) for each value of K.

rmse_cv = foreach(fold = 1:K_folds, .combine='c') %do% {
  knn100 = knnreg(price ~ mileage,
                  data=filter(model350, fold_id != fold), k=100)
  modelr::rmse(knn100, data=filter(model350, fold_id == fold))
}
```

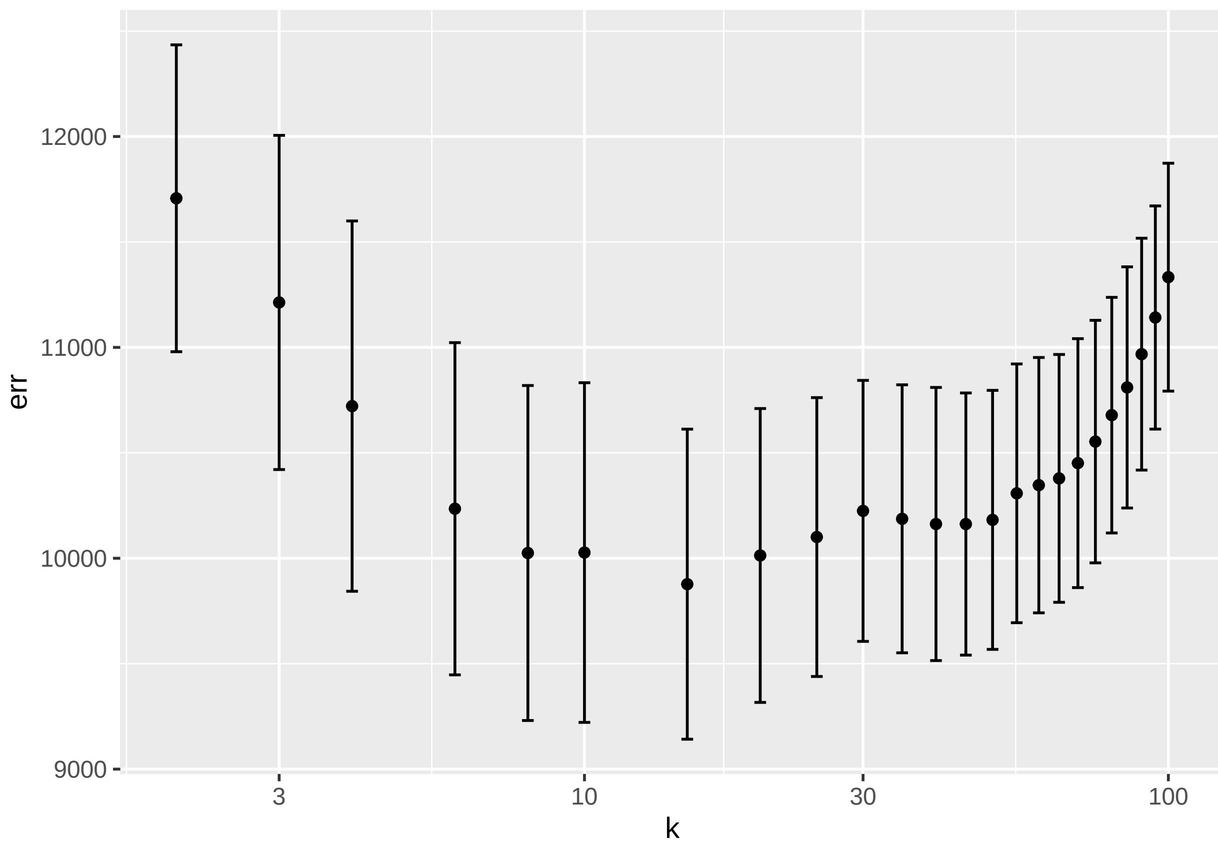
```r
k_grid = c(2, 3, 4, 6, 8, 10, 15, 20, 25, 30, 35, 40, 45,
           50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100)

model350_folds = crossv_kfold(model350, k=K_folds)

cv_grid = foreach(k = k_grid, .combine='rbind') %dopar% {
  models = map(model350_folds$train, ~ knnreg(price ~ mileage, k=k, data = ., use.all=FALSE))
  errs = map2_dbl(models, model350_folds$test, modelr::rmse)
  c(k=k, err = mean(errs), std_err = sd(errs)/sqrt(K_folds))
} %>% as.data.frame
```

```
## Warning: executing %dopar% sequentially: no parallel backend registered
```

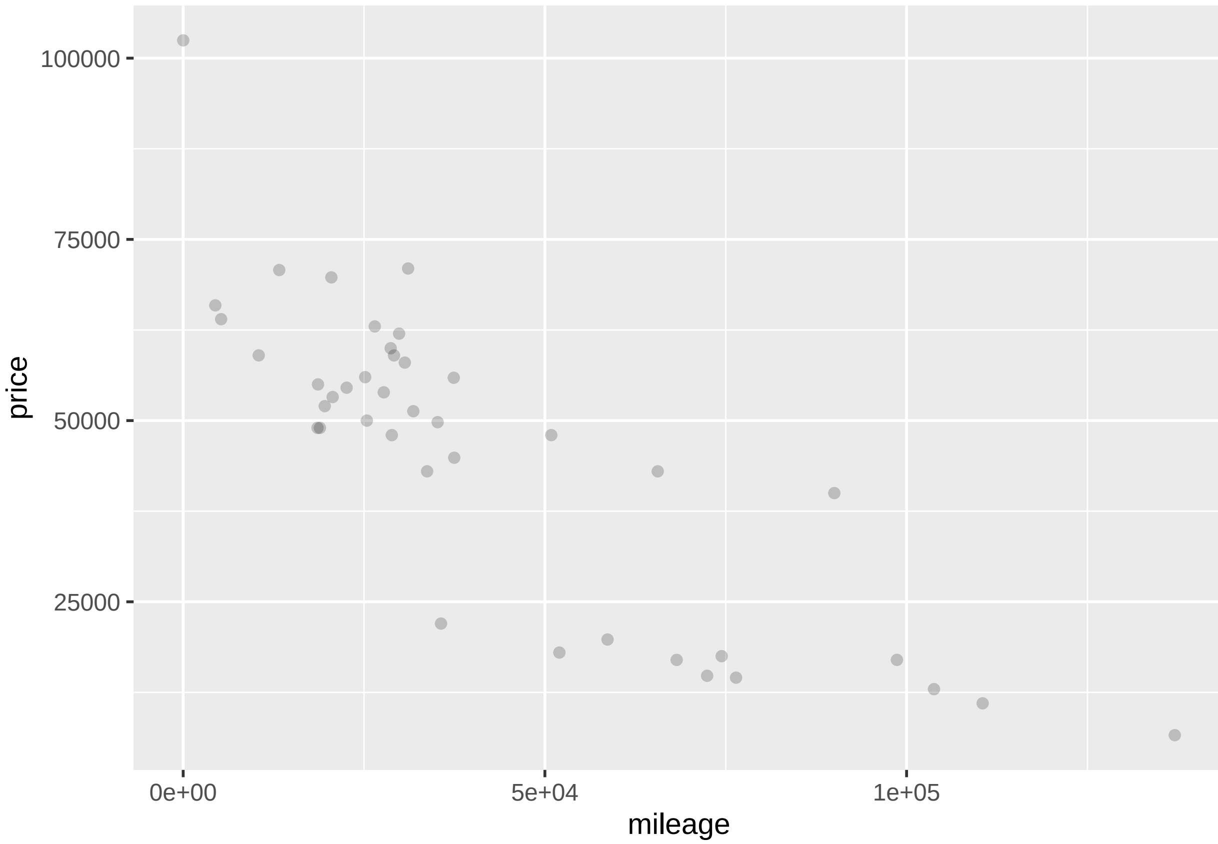
```r
head(cv_grid)
```

```
##           k      err  std_err
## result.1  2 11707.18 727.6667
## result.2  3 11213.11 792.3210
## result.3  4 10721.43 878.0608
## result.4  6 10234.84 787.7256
## result.5  8 10024.90 794.3343
## result.6 10 10027.05 805.4022
```

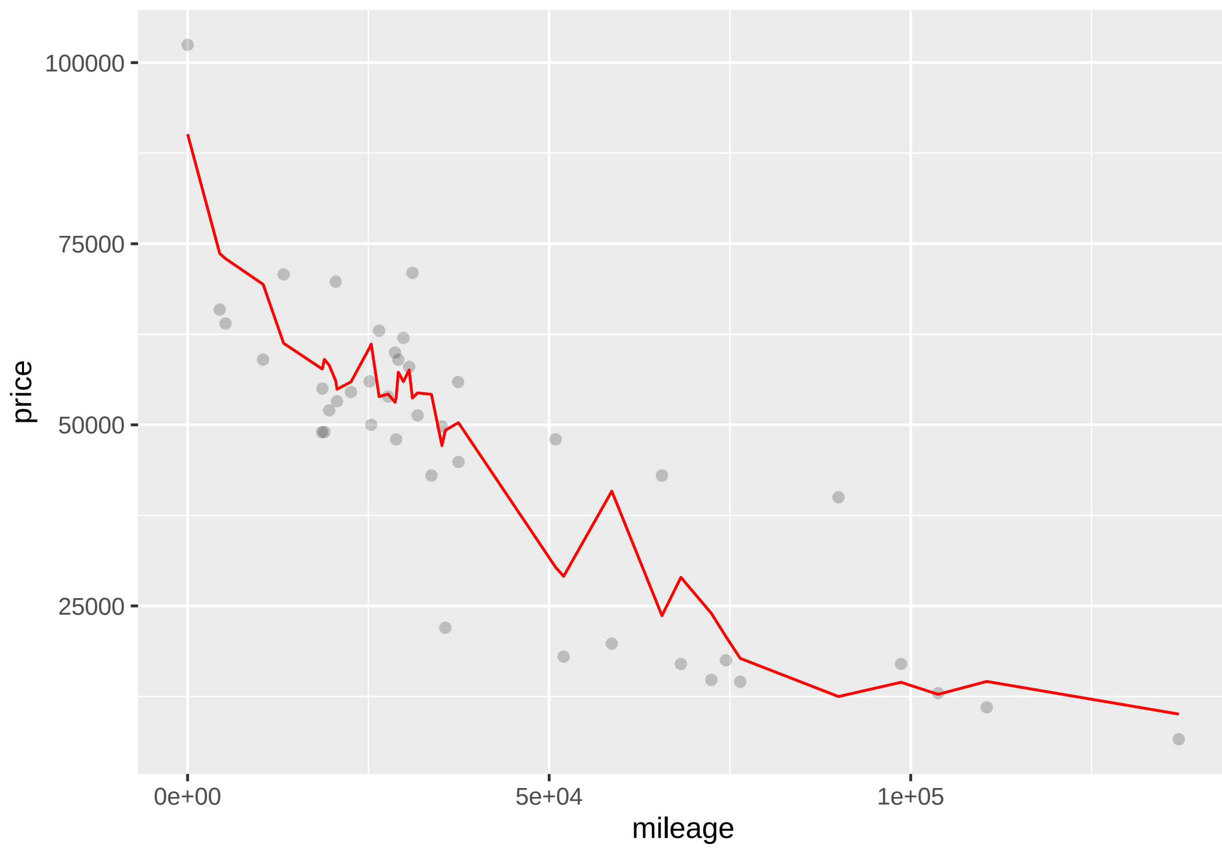
```r
#RMSE versus K plot

ggplot(cv_grid) +
  geom_point(aes(x=k, y=err)) +
  geom_errorbar(aes(x=k, ymin = err-std_err, ymax = err+std_err)) +
  scale_x_log10()
```




```
#For the optimal value of K (k=10), plot of the fitted model i.e. price prediction vs. mileage

knn10 = knnreg(price ~ mileage, data=sclass350_train, k=10)

sclass350_test = sclass350_test %>%
  mutate(price_pred = predict(knn10, sclass350_test))

p_test = ggplot(data = sclass350_test) +
  geom_point(mapping = aes(x = mileage, y = price), alpha=0.2)
p_test
```




```
p_test + geom_line(aes(x = mileage, y = price_pred), color='red', size=0.5)
```

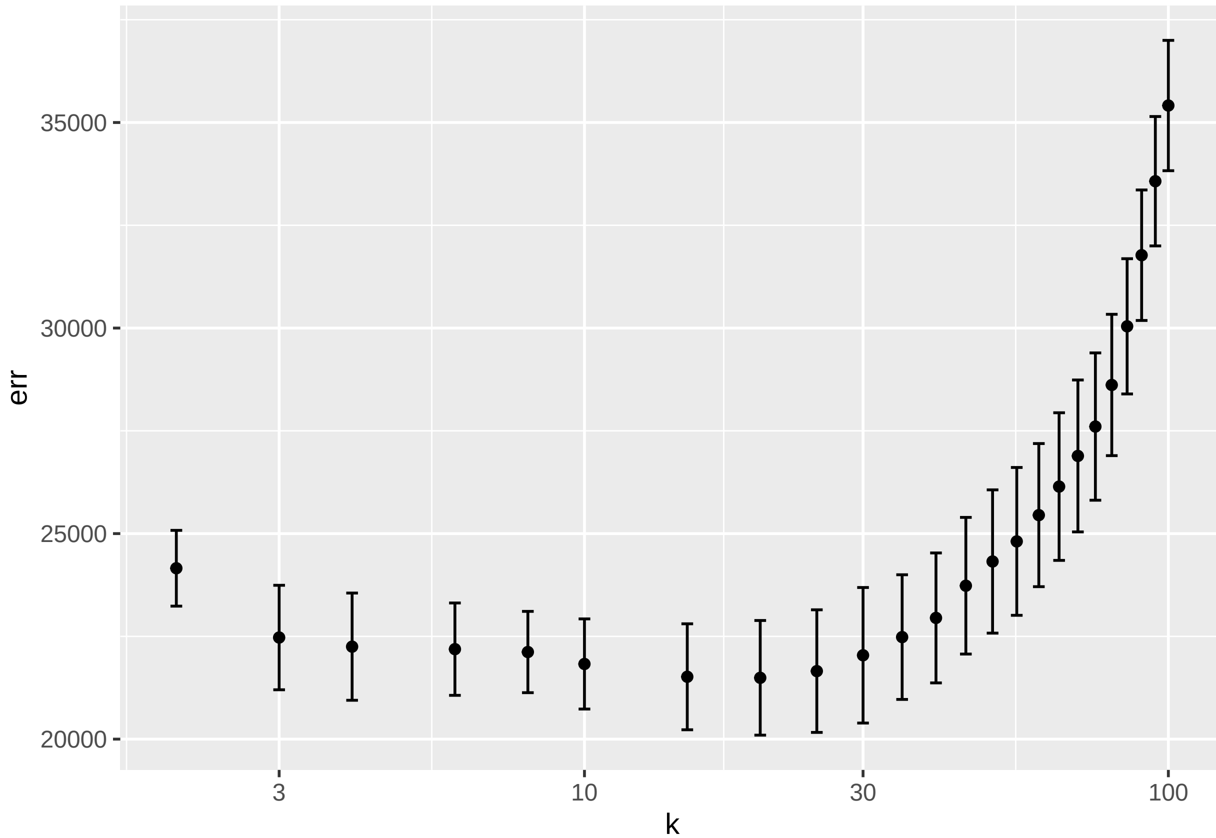



```
##65 AMG
model65AMG = sclass %>%
  filter(trim %in% '65 AMG')

#1.Split the data into a training and a testing set.

sclass65AMG_split =  initial_split(model65AMG, prop=0.9)
sclass65AMG_train = training(sclass65AMG_split)
sclass65AMG_test  = testing(sclass65AMG_split)

#2.Run K-nearest-neighbors, for many different values of K, starting at K=2 and going as high as you ne

K_folds = 5


model65AMG = model65AMG %>%
  mutate(fold_id = rep(1:K_folds, length=nrow(model65AMG)) %>% sample)


#3.Calculate the out-of-sample root mean-squared error (RMSE) for each value of K.

rmse_cv = foreach(fold = 1:K_folds, .combine='c') %do% {
  knn100 = knnreg(price ~ mileage,
                 data=filter(model65AMG, fold_id != fold), k=100)
  modelr::rmse(knn100, data=filter(model350, fold_id == fold))
}
```

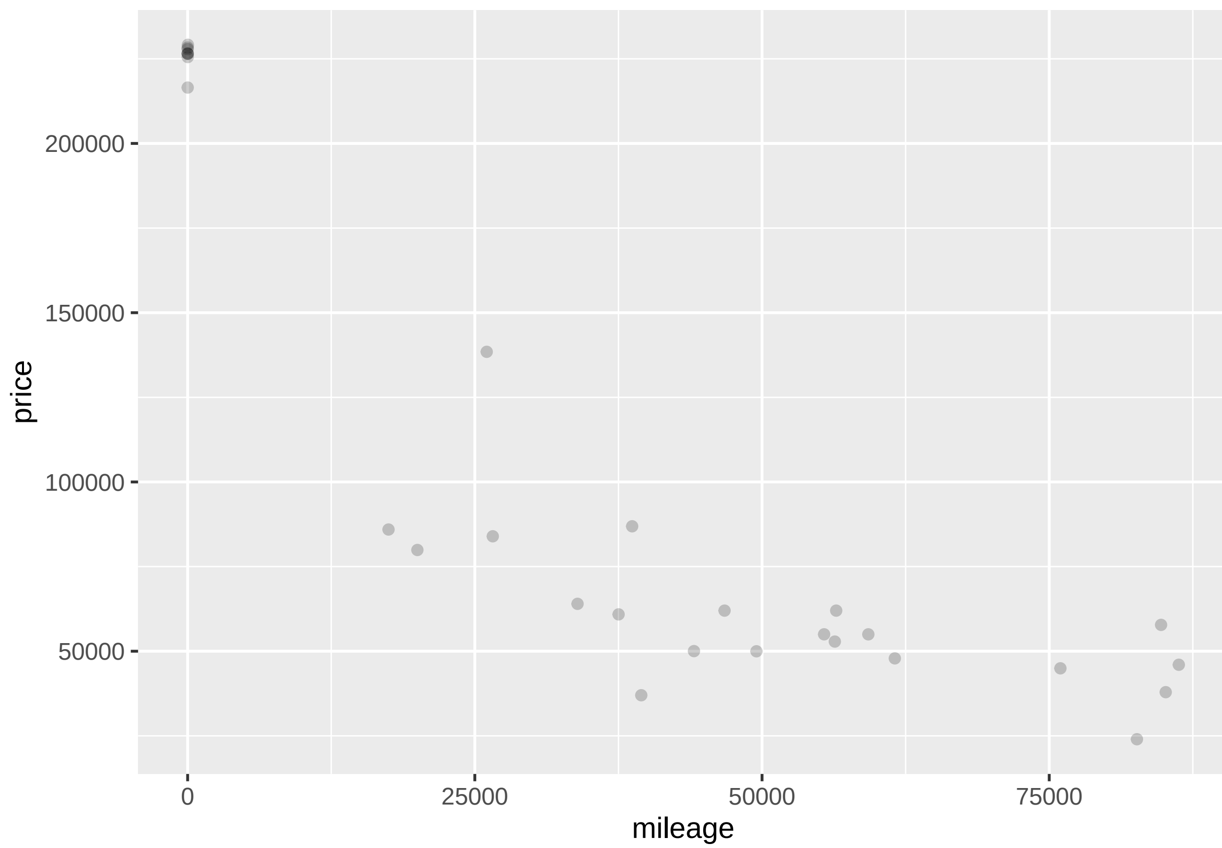
```r
k_grid = c(2, 3, 4, 6, 8, 10, 15, 20, 25, 30, 35, 40, 45,
           50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100)

model65AMG_folds = crossv_kfold(model65AMG, k=K_folds)

cv_grid = foreach(k = k_grid, .combine='rbind') %dopar% {
  models = map(model65AMG_folds$train, ~ knnreg(price ~ mileage, k=k, data = ., use.all=FALSE))
  errs = map2_dbl(models, model65AMG_folds$test, modelr::rmse)
  c(k=k, err = mean(errs), std_err = sd(errs)/sqrt(K_folds))
} %>% as.data.frame

head(cv_grid)
```

```
##           k      err   std_err
## result.1  2 24157.39  921.7563
## result.2  3 22471.92 1270.8941
## result.3  4 22249.78 1304.0722
## result.4  6 22189.05 1123.4124
## result.5  8 22119.43  989.0297
## result.6 10 21828.60 1096.7082
```

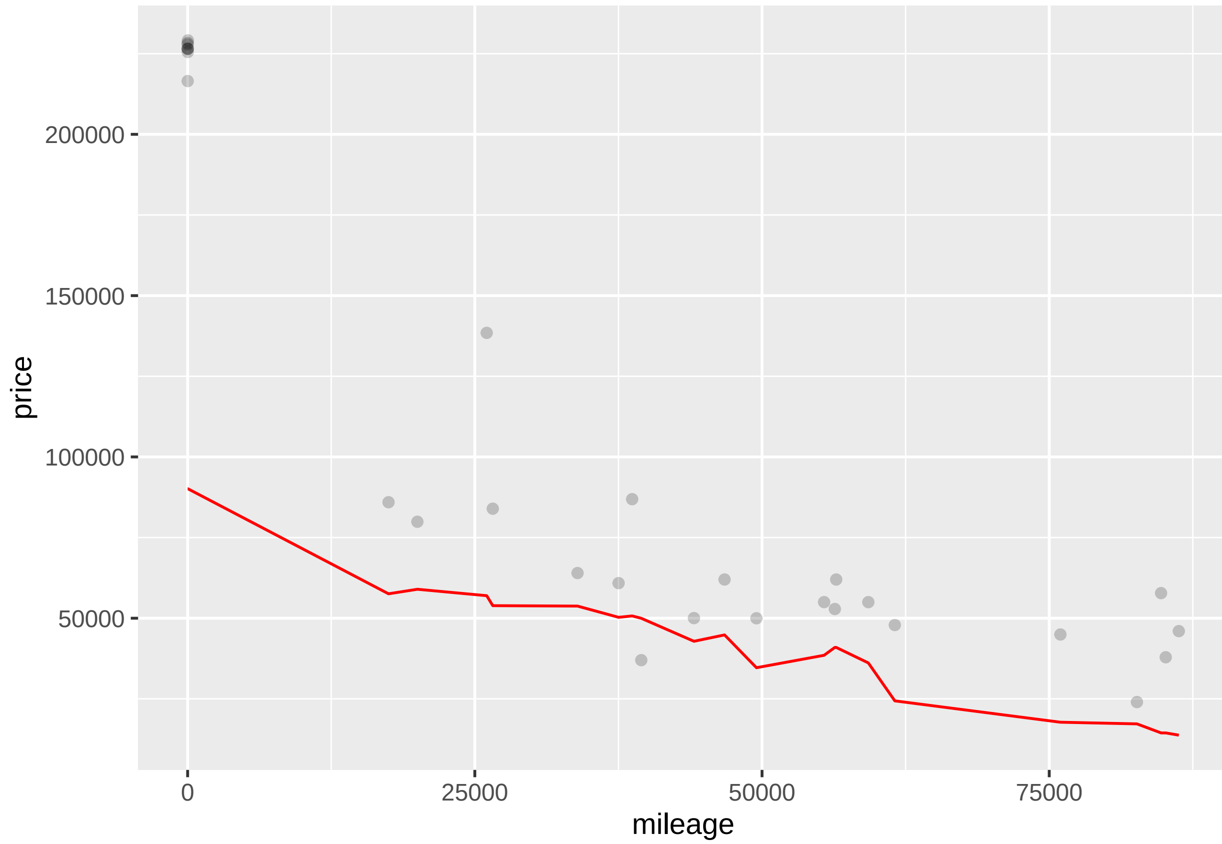
```r
#RMSE versus K plot

ggplot(cv_grid) +
  geom_point(aes(x=k, y=err)) +
  geom_errorbar(aes(x=k, ymin = err-std_err, ymax = err+std_err)) +
  scale_x_log10()
```




```
#For the optimal value of K (k=15), plot of the fitted model i.e. price prediction vs. mileage

knn15 = knnreg(price ~ mileage, data=sclass65AMG_train, k=15)

sclass65AMG_test = sclass65AMG_test %>%
  mutate(price_pred = predict(knn10, sclass65AMG_test))

p_test = ggplot(data = sclass65AMG_test) +
  geom_point(mapping = aes(x = mileage, y = price), alpha=0.2)
p_test
```




```
p_test + geom_line(aes(x = mileage, y = price_pred), color='red', size=0.5)
```



Which trim yields a larger optimal value of K? Why do you think this is?

The car's trim level 65 AMG yields a larger optimal value of K. The lowest out-of-sample root mean-squared error of 65 AMG is lower than 350.