HW1 Rubric: ECO 395M

James Scott

2/8/2021

There are 100 points on the HW. Your first 20 points come from submitting the homework in the correct format:

- A single .md or .pdf file with the entire writeup, including all figures and prose.

- A .Rmd file with the raw code used to produce the writeup.

You'll notice that my rubric below includes lots of R code. This is so you can see how I solved the problem. In general, you shouldn't include R code in a write up, which is intended as something that focused on the questions, evidence, and conclusions (and not the computer code used to run the analysis). That's what the .Rmd file is for: to give anyone access to the code who wants to dig further.

1) Data visualization: gas prices (20 points)

There are five plots to make here. Each is worth four points:

- 3 points for the figure correct figure itself
- 1 point for including an informative caption with the claim and the conclusion. What conclusion you reached isn't important as long as you had a plausible explanation based on the figure.

For part C, the bar plot, it is very important not to truncate the y axis, i.e. the y axis must begin at zero. (Otherwise the size of the bar no long corresponds to the size of the number it represents). This results in two bars that look really similar. If you left it like that, that's OK. But one way to "zoom in" on a bar plot is to mean-center the data, i.e. to compute a number $e_i = y_i - \bar{y}$ for each gas station i, and to then make a bar plot of e_i rather than y_i for each brand. This focuses on the brand differences while also ensuring that the y axis contains zero.

For part E (free choice), good choices here included a boxplot, a bar plot of means, or a faceted histogram.

2) Data visualization: a bike share network

For all three parts, you will need to load the `tidyverse` and `ggplot2` libraries and reading in the data set.

Below I show you the R code I used to create these figures. Ideally, you wouldn't include this R code in your write-up, and you wouldn't narrate your process of analyzing the data. Instead, you'd focus on the figures and the conclusions. I'm showing the R code here so you can know how to make the figures.

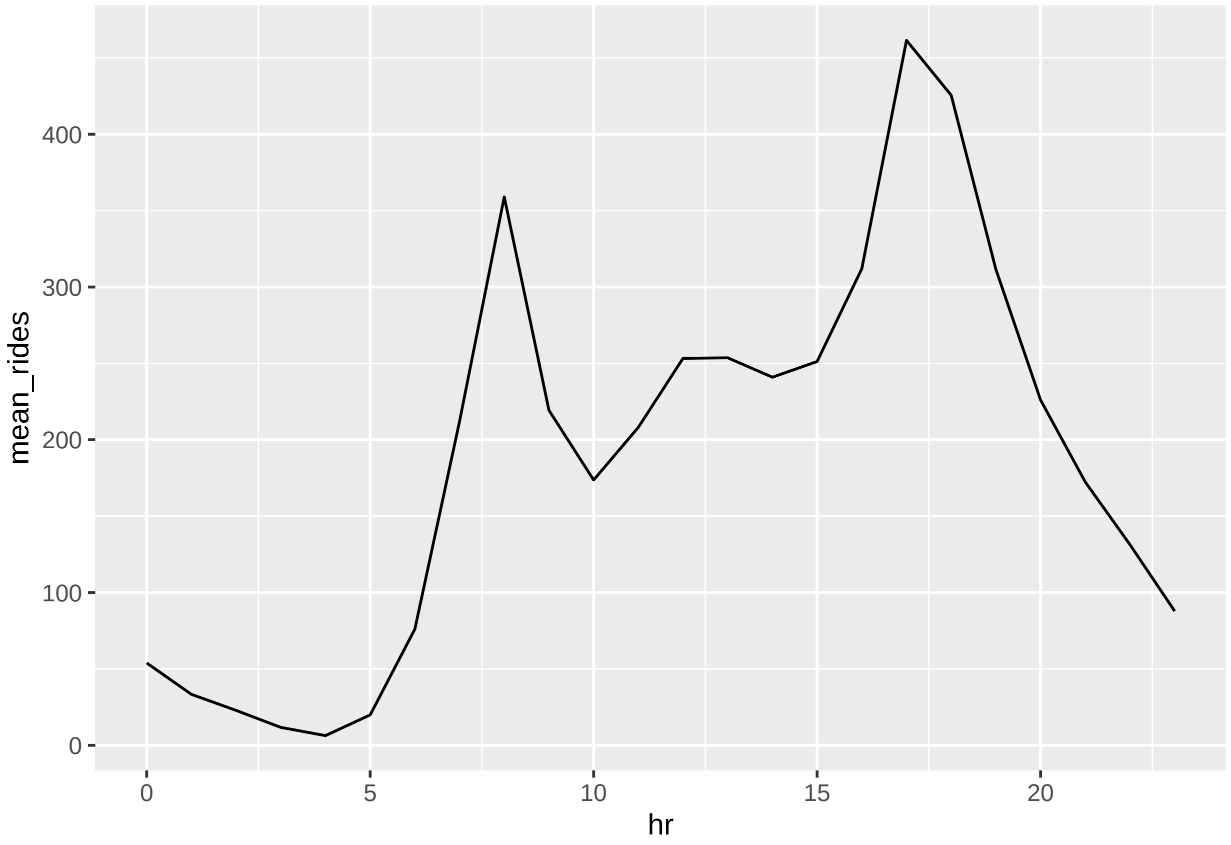
Part A: (6 points)

To calculate average ridership by hour, you first need to use group/pipe/summarize. In the code below, we create a new data frame of summary statistics called `d1`:

```
d1 = bikeshare %>%
  group_by(hr) %>%
  summarize(mean_rides = mean(total))
```

Every row of d1 now shows the average ridership (`mean_rides`, a variable we created). Now we can feed the `d1` data frame into `ggplot` to make our line graph:

```
ggplot(d1) +
  geom_line(aes(x=hr, y=mean_rides))
```



Example caption: This figure shows a plot of mean ridership by hour for bike rentals in the Capital Bikeshare system. Take-home lesson: we see two distinct peaks in ridership, around 8 AM and around 5 PM, corresponding to morning and evening rush hour. (You didn't need to come up with this specific message, just some relevant comment on the figure.)

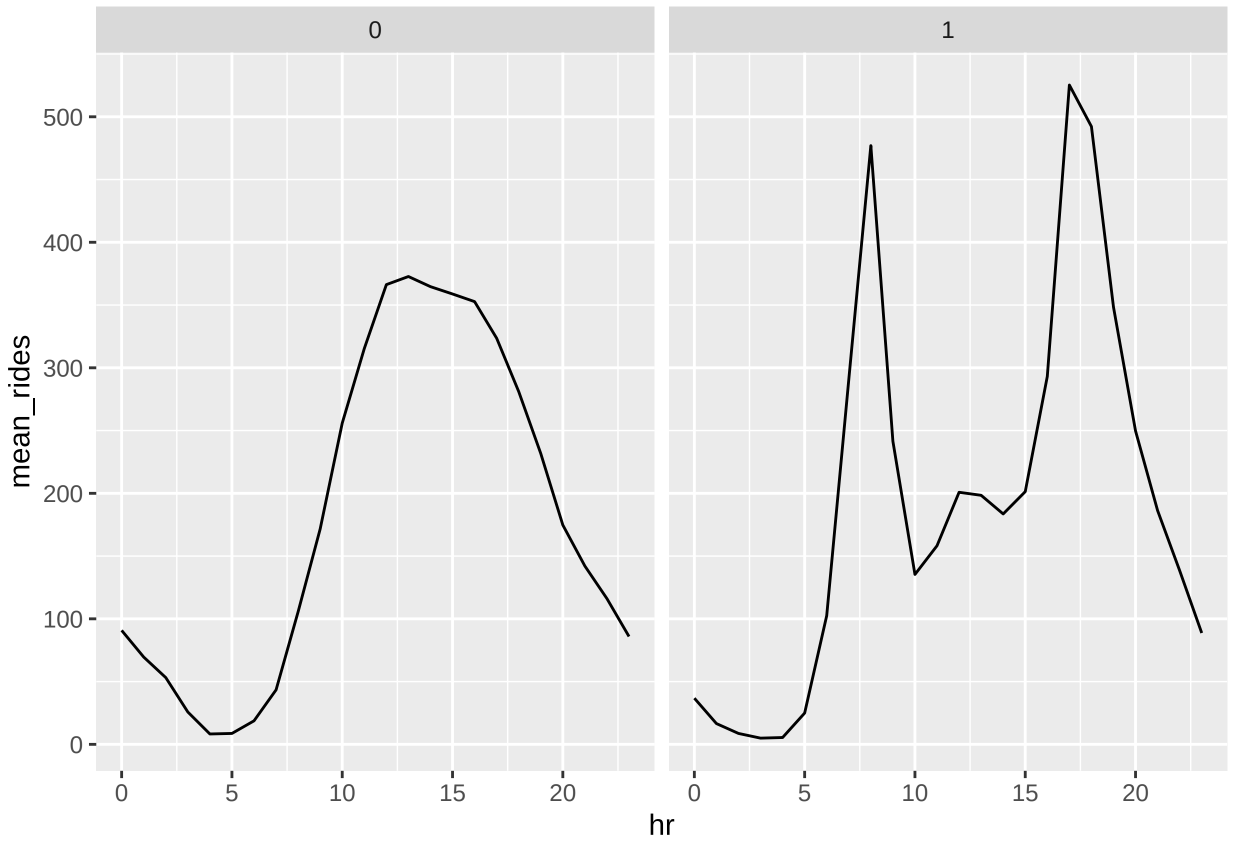
Part B: (6 points)

To make a faceted line graph, we have to calculate average ridership by hour *and* by whether it is a working day. This requires using group/pipe/summarize, but grouping on both the `hr` and `workingday` variable. In the code below, we create a new data frame of summary statistics called `d2`:

```r
d2 = bikeshare %>%
 group_by(hr, workingday) %>%
 summarize(mean_rides = mean(total))
```

The data frame `d2` now has average ridership (`mean_rides` for each combination of `hr` and `workingday`. We can now feed this into `ggplot` to make a faceted line graph:

```r
ggplot(d2) +
 geom_line(aes(x=hr, y=mean_rides)) +
 facet_wrap(~workingday)
```



Example caption: This figure shows a plot of mean ridership by hour for bike rentals in the Capital Bikeshare system, faceted according to whether it is a working day (1) or not (0). Take-home lesson: we see two distinct peaks in ridership around rush hours on working days, but only a single mid-day peak on non-working days.

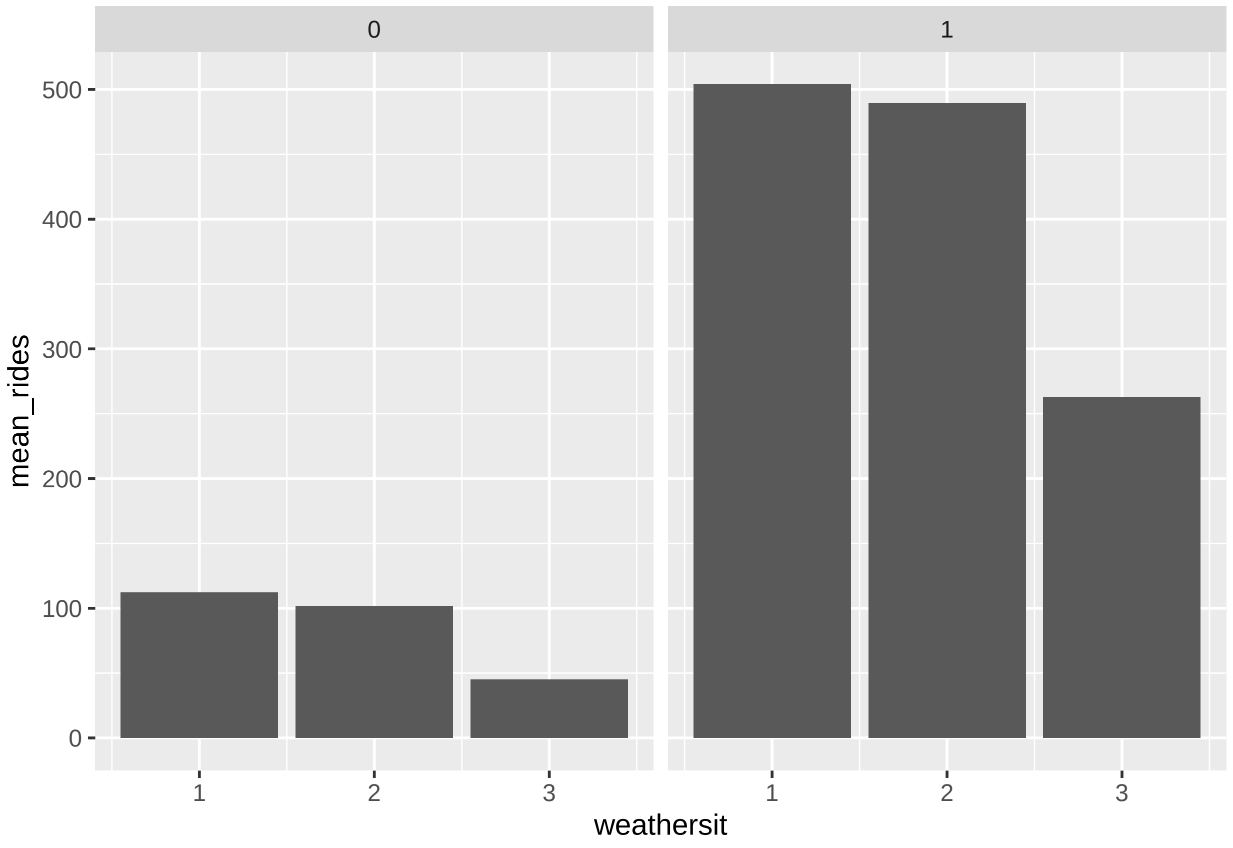
Part C: (8 points)

To make a faceted bar plot by weather situation for the 8 AM hour alone, we need to combine group/pipe/summarize with filter. In the code below, we create a new data frame of summary statistics called `d3`, where we first filter the rows to the 8 AM hour only, then we group by hour and weather situation, and then finally we calculate mean ridership:

```
d3 = bikeshare %>%
  filter(hr==8) %>%
  group_by(weathersit, workingday) %>%
  summarize(mean_rides = mean(total))
```

Now we can feed `d3` into `ggplot`:

```
ggplot(d3) +
  geom_col(aes(x=weathersit, y=mean_rides)) +
  facet_wrap(~workingday)
```



Example caption: This figure shows a plot of mean ridership for bike rentals in the Capital Bikeshare system for the 8 AM hour, according to the weather situation (1 = nice weather, 2 = mist or clouds, and 3 = snow or rain). The bar plot is faceted according to whether it is a working day (1) or not (0). Take-home lesson: we see a sharp dropoff in ridership during poor weather, both for working and non-working days.

3) Data visualization: flights at ABIA (20 points)

This is an open-ended problem, and so there is no "key", per se. Instead, we followed these rough guidelines in grading.

- 15 points for the the figure(s), 5 points for a clear annotation.

- the better figures show more than one explanatory variable. If you had only one "basic" figure (histogram/boxplot/bar plot/scatter plot/line graph) showing one explanatory variable (e.g. average delay by day of week), your maximum score was 10 points for the figure and 5 for the caption. This data set was quite rich and we gave you plenty of tools for exploring richer relationships (e.g. delay vs day of week AND airline).

- all figures should have clear axis labels and titles.

4) K-nearest neighbors (20 points)

5 points each for the four plots:

- RMSE versus k for each subset of the data (S350 and 65 AMG)
- plot of optimal fit for each subset of the data (S350 and 65 AMG). My code below shows these plots for the full training data, and in the case of the S350, there's an interesting surprise. You didn't have to comment on this surprise or notice it, but there's a valuable lesson (see below).

General points:

- The precise grid of K values isn't important as long as you can clearly see the "down, then up" shape of RMSE as k increases.

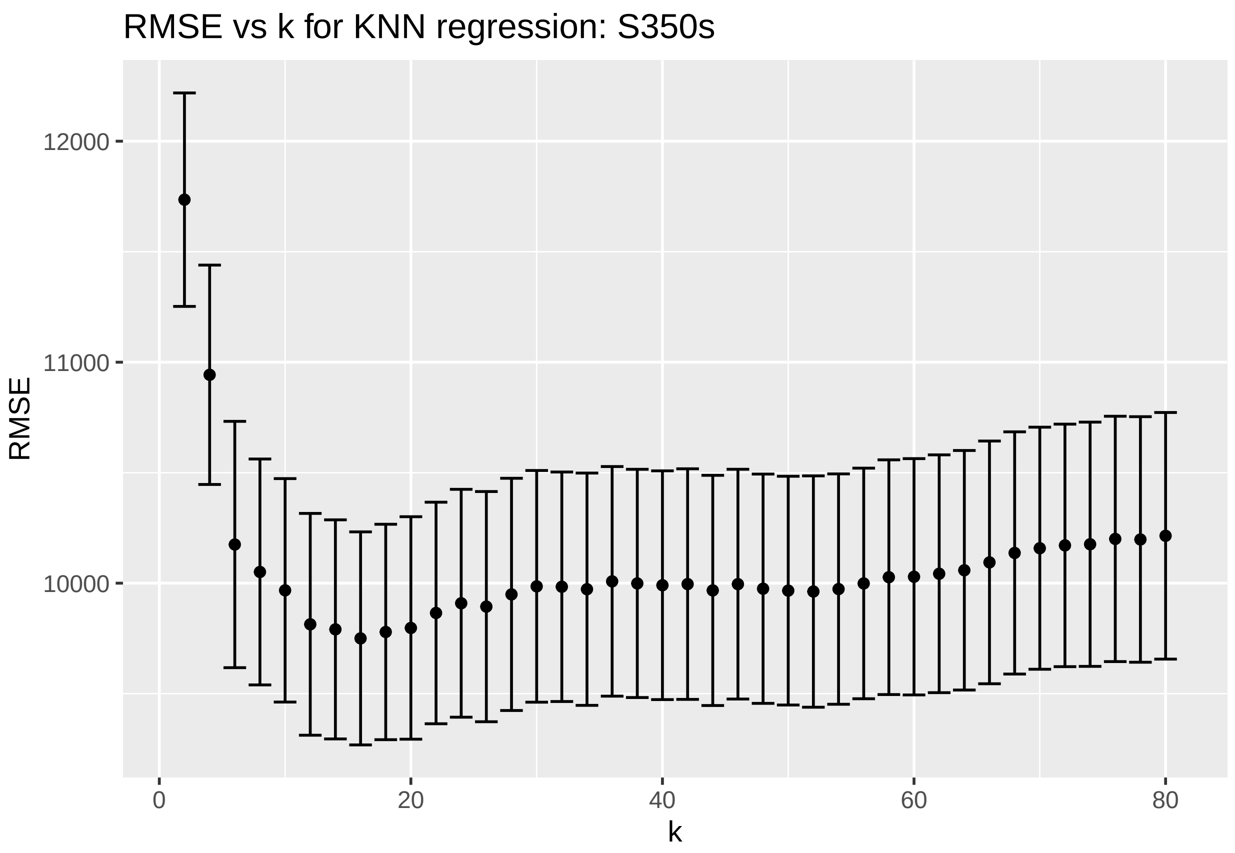
- The wording of the question did allow you to get away with just a single train/test split and still receive full credit. But best practice is definitely to average over multiple train/test splits, as my code does below.

- there were no points explicitly assigned for answering the question at the end, about which subset seemed to lead to a larger optimal value of K. That was for you to reflect on, but wasn't graded. In general, you should have seen the 65 AMGs having larger optimal values of K, because of a smaller sample size (meaning it's better to reduce variance and accept some more bias).

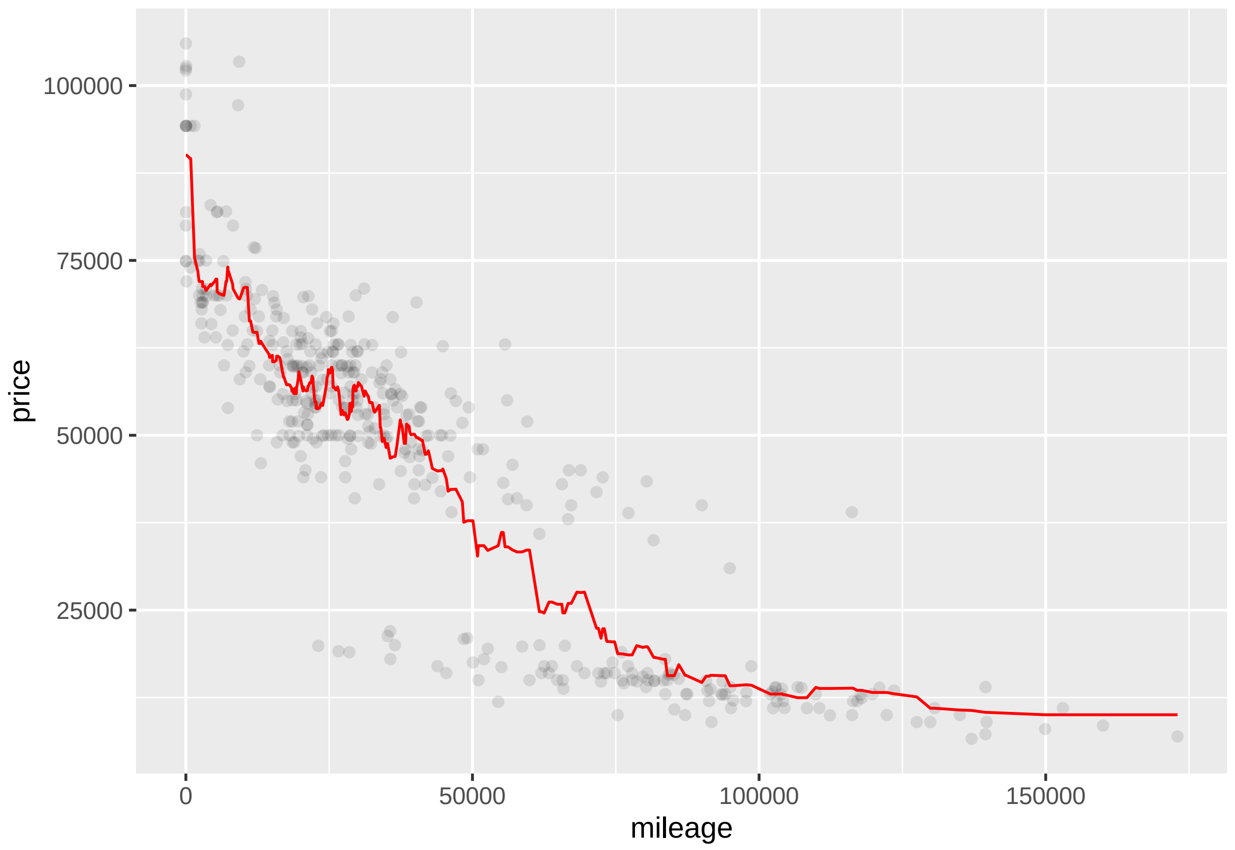
Here's some R code that can be used to solve this problem, along with the plots it produces. We first focus on the S 350 subset:

```r
library(tidyverse)
library(ggplot2)
library(caret)
library(modelr)
library(rsample)
library(foreach)
library(parallel)

sclass = read.csv('../data/sclass.csv')

# Focus on first trim level: 350
sclass350 = filter(sclass, trim == '350')

# K-fold cross validation
K_folds = 20
sclass350_folds = crossv_kfold(sclass350, k=K_folds)

# create a grid of K values -- the precise grid isn't important as long
# as you cover a wide range
k_grid = seq(2, 80, by=2)
```

```
# For each value of k, map the model-fitting function over the folds
# Using the same folds is important, otherwise we're not comparing
# models across the same train/test splits
cv_grid = foreach(k = k_grid, .combine='rbind') %do% {
  models = map(sclass350_folds$train, ~ knnreg(price ~ mileage, k=k, data = ., use.all=FALSE))
  errs = map2_dbl(models, sclass350_folds$test, modelr::rmse)
  c(k=k, err = mean(errs), std_err = sd(errs)/sqrt(K_folds))
} %>% as.data.frame

# plot means and std errors versus k
ggplot(cv_grid) +
  geom_point(aes(x=k, y=err)) +
  geom_errorbar(aes(x=k, ymin = err-std_err, ymax = err+std_err)) +
  labs(y="RMSE", title="RMSE vs k for KNN regression: S350s")
```

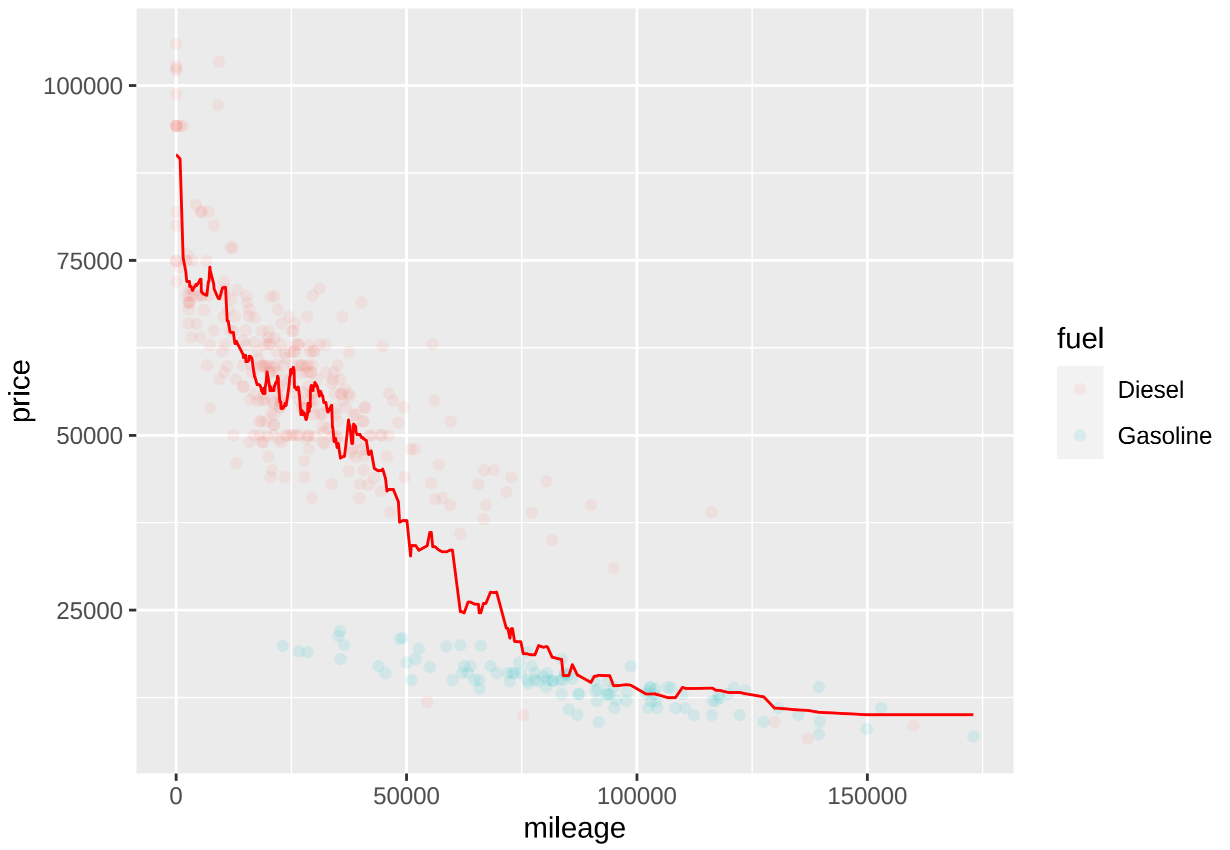


```
# fit at optimal k to show predicts on full data set
k_best = k_grid[which.min(cv_grid$err)]
knn_best = knnreg(price ~ mileage, k=k_best, data = sclass350)

# add predictions to data frame
sclass350  = sclass350 %>%
  mutate(price_pred = predict(knn_best, sclass350))

ggplot(sclass350) +
  geom_point(aes(x=mileage, y=price), alpha=0.1) +
  geom_line(aes(x=mileage, y = price_pred), col='red')
```



One thing that's interesting about this plot is that it shows how there's really two groups of points. If you dig into it, you'll find that this is due to some cars having gasoline (petrol) engines, and some having diesel engines, with the diesels selling at a significant premium.

```
ggplot(sclass350) +
  geom_point(aes(x=mileage, y=price, color=fuel), alpha=0.1) +
  geom_line(aes(x=mileage, y = price_pred), col='red')
```



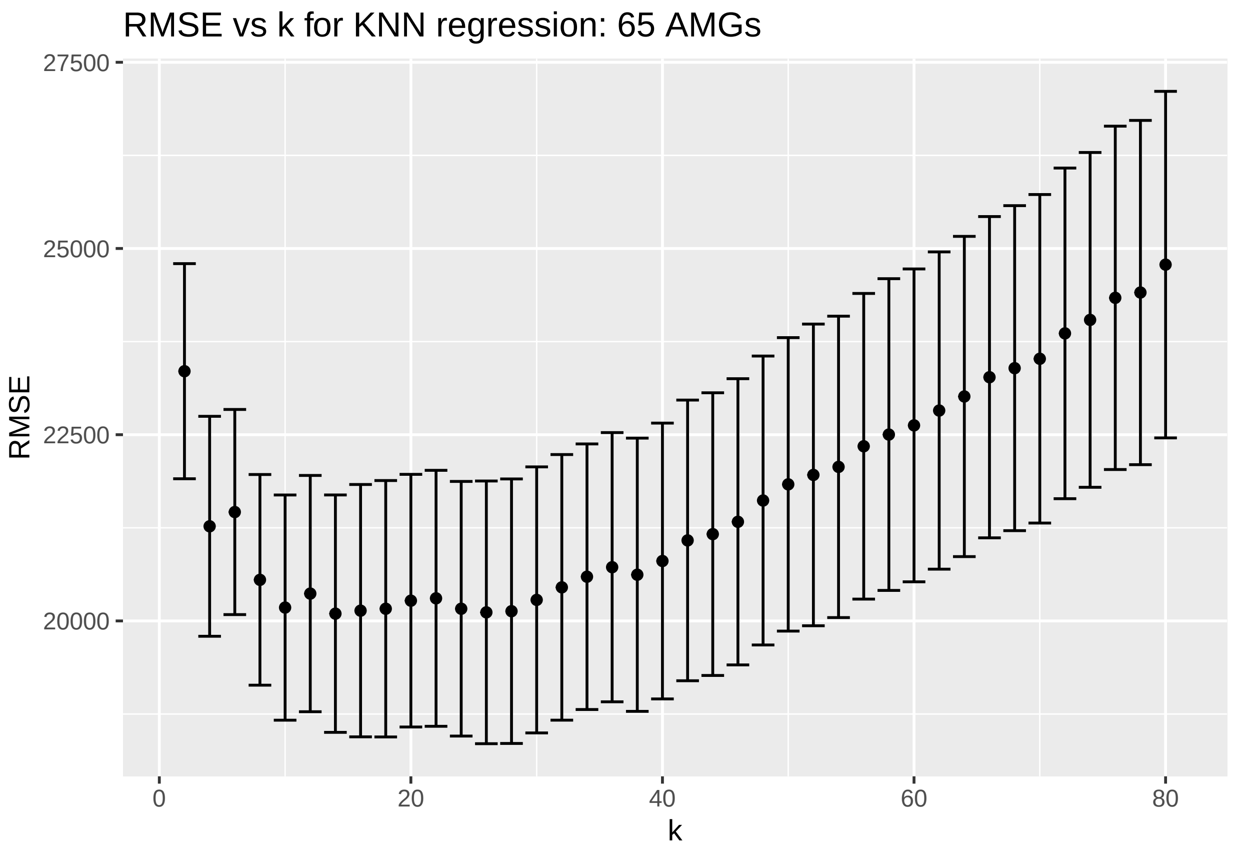
You didn't need to observe this or dig into it in order to get full credit, but it's a useful illustration of why **it's good to plot your data and your fitted model** if possible. Clearly the KNN regression model would be improved here by splitting it into two separate models, one for each fuel type. (ALthough another thing to point out that is fuel type is also confounded by model year, so don't read too much into the fuel type):

```
xtabs(~fuel + year, data=sclass350)
```

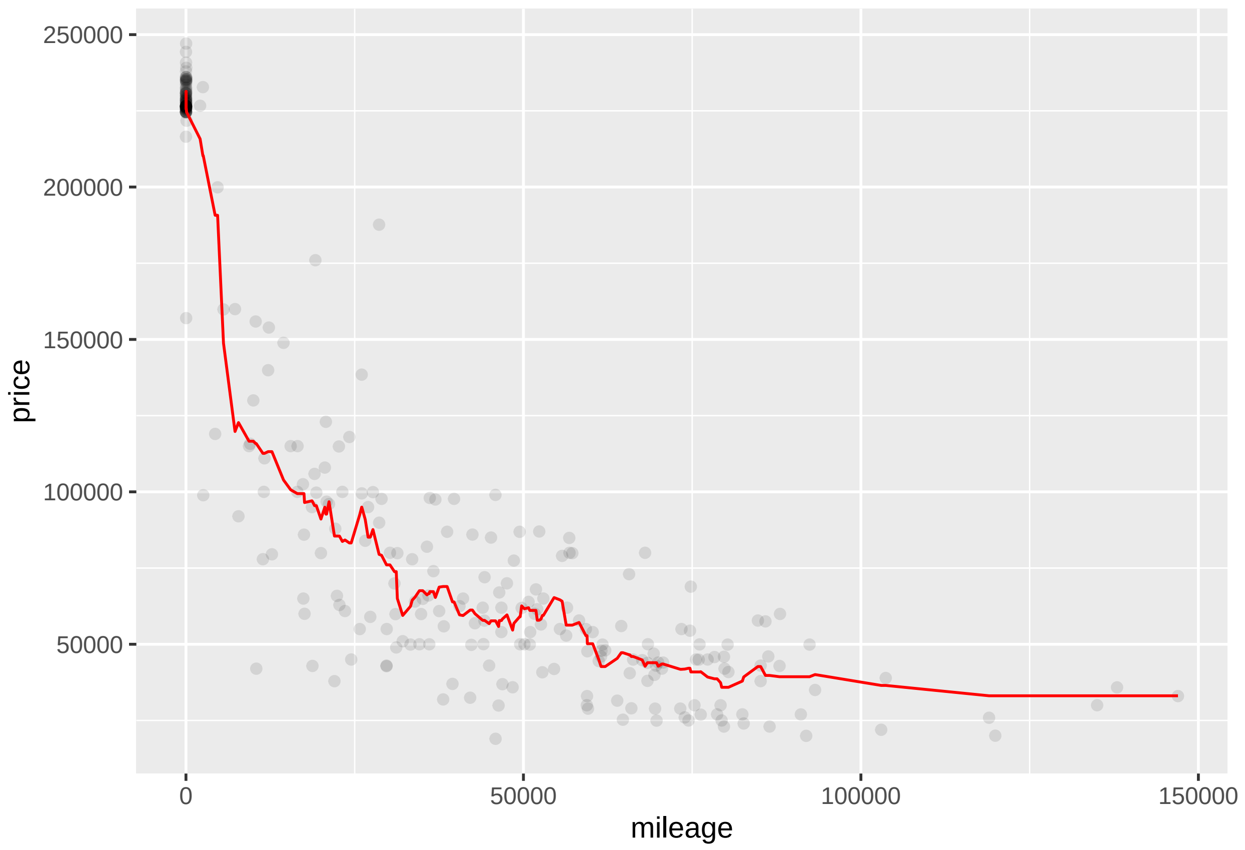
```
##           year
## fuel        1994 1995 2006 2012 2013
##   Diesel       4    1    0  217   85
##   Gasoline     0    0  109    0    0
```

Now we repeat the same steps for the 65AMGs.

```
# Now repeat for the 65 AMGs
sclass65AMG = subset(sclass, trim == '65 AMG')

# create the folds
sclass65AMG_folds = crossv_kfold(sclass65AMG, k=K_folds)

# For each value of k, map the model-fitting function over the folds
cv_grid2 = foreach(k = k_grid, .combine='rbind') %do% {
  models = map(sclass65AMG_folds$train, ~ knnreg(price ~ mileage, k=k, data = ., use.all=FALSE))
  errs = map2_dbl(models, sclass65AMG_folds$test, modelr::rmse)
  c(k=k, err = mean(errs), std_err = sd(errs)/sqrt(K_folds))
} %>% as.data.frame

# plot means and std errors versus k
ggplot(cv_grid2) +
  geom_point(aes(x=k, y=err)) +
  geom_errorbar(aes(x=k, ymin = err-std_err, ymax = err+std_err)) +
  labs(y="RMSE", title="RMSE vs k for KNN regression: 65 AMGs")
```



```
# fit at optimal k to show predicts on full data set
k_best = k_grid[which.min(cv_grid2$err)]
knn_best = knnreg(price ~ mileage, k=k_best, data = sclass65AMG)

# add predictions to data frame
sclass65AMG  = sclass65AMG %>%
  mutate(price_pred = predict(knn_best, sclass65AMG))

ggplot(sclass65AMG) +
  geom_point(aes(x=mileage, y=price), alpha=0.1) +
  geom_line(aes(x=mileage, y = price_pred), col='red')
```


250000
200000
150000
100000
50000
price
0
50000
100000
150000
mileage