Homework 8

The purpose of this homework is to practice fitting multiple regression with categorical and non-linear predictors, and to examine overfitting. Please fill in the appropriate code and write answers to all questions in the answer sections, then submit a compiled pdf with your answers through Gradescope by 11pm on Sunday November 12th.

As always, if you need help with the homework, please attend the TA office hours which are listed on Canvas and/or ask questions on Ed Discussions. Also, if you have completed the homework, please help others out by answering questions on Ed Discussions, which will count toward your class participation grade.

Part 1: Fitting linear regression models with categorical predictors

As you will recall in the last two problem sets we explored the relationship between the price used Toyota Corollas sold for and other explanatory variables. This analysis was motivated by the example of when I had to sell my used Toyota Corolla after it broke down on the side of the highway. However, not only did I need to sell my Corolla, but I also needed to buy a new car. Let's continue to explore multiple regression models on the Edmunds.com data to see if we can get some insight into which car brands would be the best to buy.

Part 1.1 (5 points):

Toward the end of my search for a new car, the car models I was considering were the Mazda 3 and the Subaru Impreza. When buying the new car, I knew that at some point I was going to have to sell the car, so I was still interested in looking at how the prices of these car models decline as they are driven more miles. Let's build some regression models that could be helpful for seeing how different makes/models of cars decline in price as a function of the number of miles driven and other variables.

To start our analyses, please use the Edmunds `car_transactions` data frame to create a data frame called `three_car_data` that only has the car brands Mazda 3's, Subaru Imprezas, and BMW 5 series (one can dream, right?). To create this data frame, please complete the following steps:

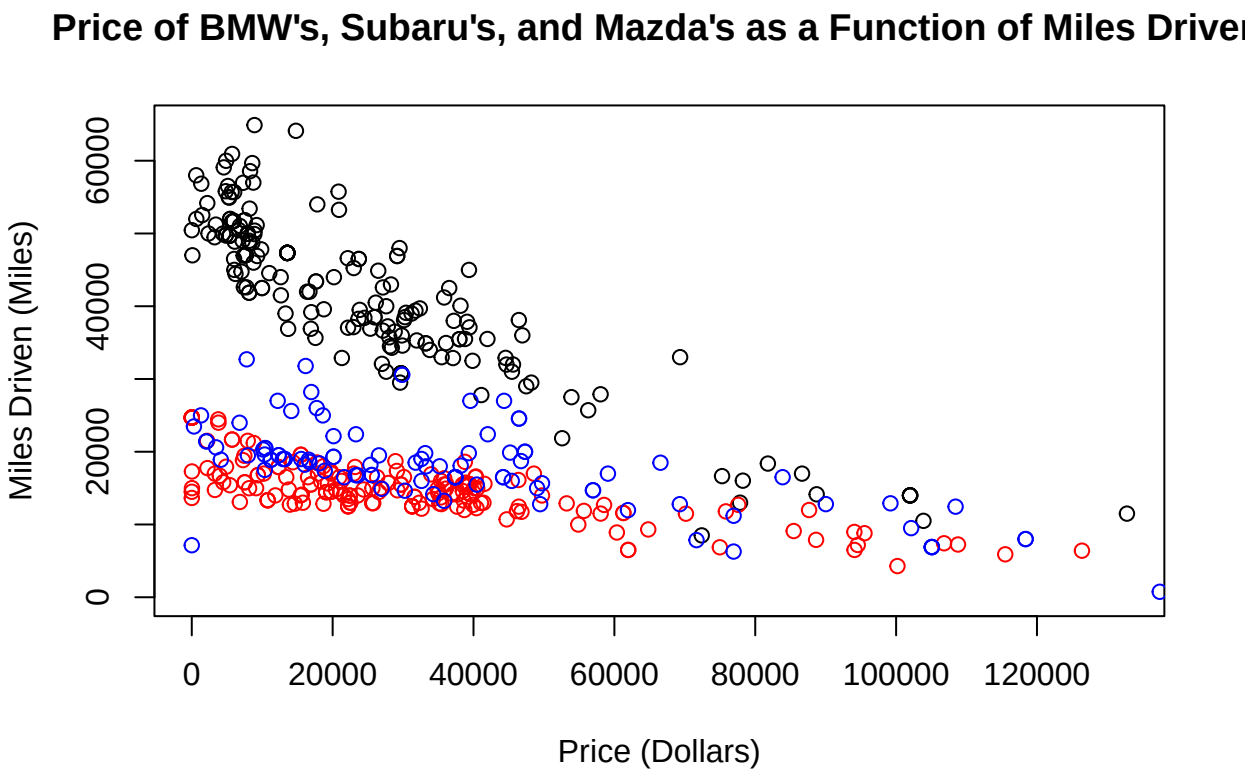
1. Get only the models of Mazda 3's, Subaru Imprezas, and BMW 5 series car makes (i.e., `MAZDA3`, `Impreza` and `5 Series`). Be sure to use the `%in%` operator instead of `==`, or connect multiple equality `==` statements with the "or" operator `|`, when you filter for the models of the cars - using a single `==` statement will produce an incorrect answer.
2. Get the data for only used cars.
3. Use the `droplevels()` function on this data frame to remove all levels of the categorical data we are not using (i.e., remove levels for other makes of cars).

If you've filtered the data correctly the data frame should have 497 cases, so be sure to check that this is the case before moving on to other exercises.

Once you have created the `three_car_data` data frame, use the `plot()` and `points()` functions to create a scatter plot of the the price of a car as a function of miles driven, where the BMW's are plotted as *black* circles, the Subaru's are plotted as *blue* circles, and the Mazda's are plotted as *red* circles. Be sure to set the `ylim` argument in the plot function to be from 0 to $65,000 so that it captures all the points in the plot.

Note: For the rest of this question the term `brand` will refer to the make/model of the different types of cars.

```
load("car_transactions.rda")

three_car_data <- car_transactions %>%
  filter(model_bought %in% c("MAZDA3", "Impreza", "5 Series")) %>%
  filter(new_or_used_bought == "U") %>%
  droplevels()

# price of car as function of miles driven
# bmw - black, subaru - blue, mazda - red
# ylim from 0 to 65000
plot(price_bought ~ mileage_bought, data = filter(three_car_data, make_bought == "BMW"),
     ylim = c(0, 65000), xlab = "Price (Dollars)", ylab = "Miles Driven (Miles)",
     main = "Price of BMW's, Subaru's, and Mazda's as a Function of Miles Driven")

points(price_bought ~ mileage_bought,
       data = filter(three_car_data, make_bought == "Mazda"), col = "red")

points(price_bought ~ mileage_bought,
       data = filter(three_car_data, make_bought == "Subaru"), col = "blue")
```



Part 1.2 (12 points): Let's now fit a linear model for predicting price as a function of miles driven *with a separate intercept for each brand*. Use the `summary()` function to extract information about the linear model, and then answer the following questions:

1. How much does the price of a car decrease for each additional mile driven?
2. What is the reference car brand that the other car brands are being compared to?
3. What is the difference in car prices for each of the other brands relative to the reference car brand?
4. Does there appear to be statistically significant differences between the y-intercept of reference brand and each of the other car brands?
5. How much of the total sum of squares of car prices is mileage and car brand accounting for in this model based on the R^2 statistic?

```
# price as function of miles driven with separate intercept for each brand
lm_car <- lm(price_bought ~ mileage_bought + make_bought, data = three_car_data)

summary(lm_car)
```

```
##
## Call:
## lm(formula = price_bought ~ mileage_bought + make_bought, data = three_car_data)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -22653.3  -3635.8   -175.6   3207.9  20104.5
##
## Coefficients:
##                         Estimate    Std. Error t value            Pr(>|t|)
## (Intercept)        47307.312097    463.920519  101.97 <0.0000000000000002 ***
## mileage_bought        -0.223103      0.009718  -22.96 <0.0000000000000002 ***
## make_boughtMazda  -25472.300077    574.150783  -44.37 <0.0000000000000002 ***
## make_boughtSubaru -20849.590947    687.926849  -30.31 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5514 on 472 degrees of freedom
##   (21 observations deleted due to missingness)
## Multiple R-squared:  0.8667, Adjusted R-squared:  0.8659
## F-statistic:  1023 on 3 and 472 DF,  p-value: < 0.00000000000000022
```

Answers:

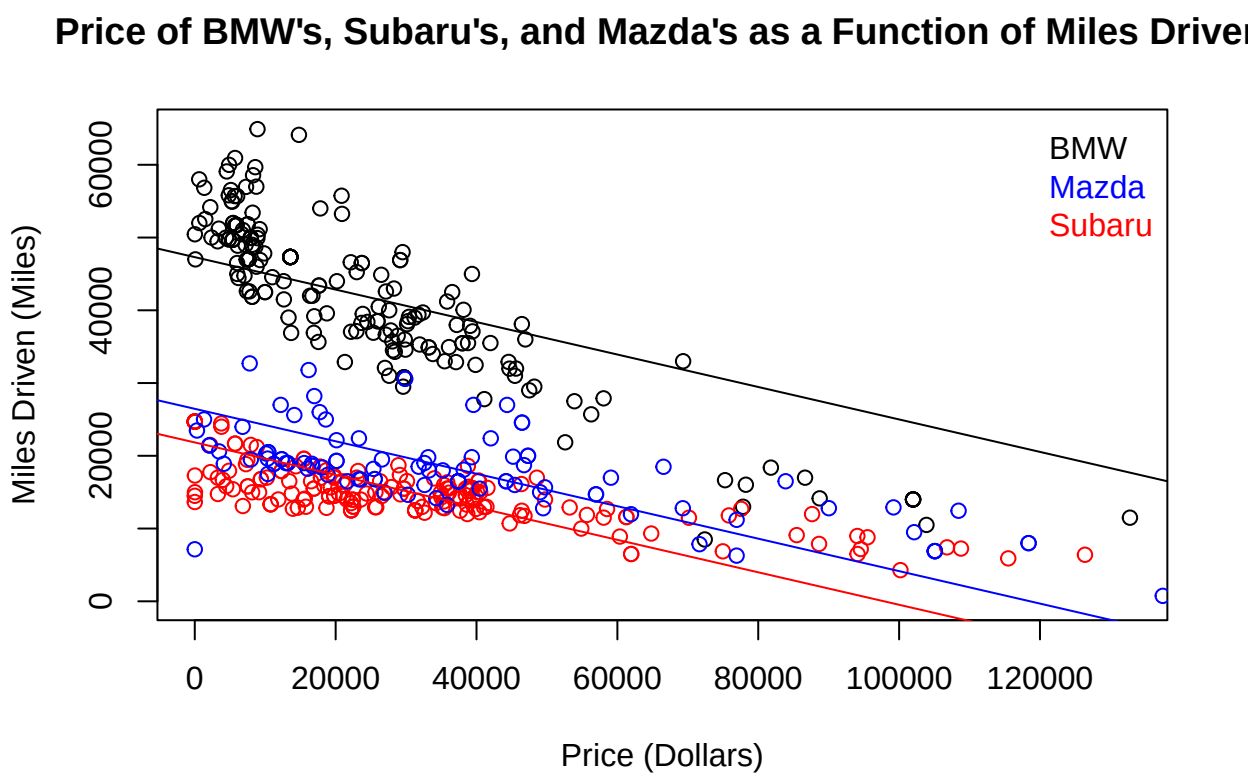
1. The price decreases by $0.22 for each mile driven
2. BMW
3. For Mazda's, the price difference is $25472.30.
   For Subaru's, the price difference is $20849.59.
4. Yes
5. 86.67%

Part 1.3 (6 points): Now recreate the scatter plot you created in part 1.1 using the same colors but also add on the regression lines with different y-intercepts that you fit in part 1.2 (using the appropriate colors to match the colors of the points). You can plot the regression lines by passing the coefficients for intercept and slope to the `abline` function.

Based on this visualization, does it appear the model you fit in part 1.2 is doing a good job capturing the trends in this data?

```r
# plot from 1.1
plot(price_bought ~ mileage_bought, data = filter(three_car_data, make_bought == "BMW"),
     ylim = c(0, 65000), xlab = "Price (Dollars)", ylab = "Miles Driven (Miles)",
     main = "Price of BMW's, Subaru's, and Mazda's as a Function of Miles Driven")

points(price_bought ~ mileage_bought,
       data = filter(three_car_data, make_bought == "Mazda"), col = "red")

points(price_bought ~ mileage_bought,
       data = filter(three_car_data, make_bought == "Subaru"), col = "blue")

# extract coefs and plot regressions lines
```

```r
coefs <- coef(lm_car)

abline(coefs[1], coefs[2], col = "black")
abline(coefs[1] + coefs[3], coefs[2], col = "red")
abline(coefs[1] + coefs[4], coefs[2], col = "blue")



# legend
legend("topright", c("BMW", "Mazda", "Subaru"), text.col = c("black", "blue", "red"),
       bty = "n")
```



Answer

It appears that the model from part 1.2 is doing a good job of capturing the trends in this data, but it could be better.

Part 1.4 (6 points): Next fit a linear regression model for car price as a function of miles driven, but use separate y-intercepts **and slopes** for each of the 3 car brands. Once you have fit this model, please answer the following questions:

1. How much of the total sum of squares of car prices is this model capturing?
2. Based on this model, if a BMW 5 Series and Mazda 3 both had been driven 50,000 miles, what would be the difference in the car prices that the model predicts?

Hint: Think about what you would write differently in defining the model compared to the previous questions.

```
lm_car_2 <- lm(price_bought ~ mileage_bought + make_bought + mileage_bought * make_bought,
               data = three_car_data)

summary(lm_car_2)
```

```
##
## Call:
## lm(formula = price_bought ~ mileage_bought + make_bought + mileage_bought *
##     make_bought, data = three_car_data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -15670  -2416   -376   1798  18192
##
## Coefficients:
##                                     Estimate   Std. Error t value
## (Intercept)                        52067.23393    427.22099  121.87
## mileage_bought                        -0.41567      0.01240  -33.53
## make_boughtMazda                  -33605.59528    651.01035  -51.62
## make_boughtSubaru                 -29246.89438    782.73157  -37.37
## mileage_bought:make_boughtMazda       0.30011      0.01741   17.24
## mileage_bought:make_boughtSubaru      0.28728      0.01821   15.77
##                                          Pr(>|t|)
## (Intercept)                       <0.0000000000000002 ***
## mileage_bought                    <0.0000000000000002 ***
## make_boughtMazda                  <0.0000000000000002 ***
## make_boughtSubaru                 <0.0000000000000002 ***
## mileage_bought:make_boughtMazda   <0.0000000000000002 ***
## mileage_bought:make_boughtSubaru <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4135 on 470 degrees of freedom
##   (21 observations deleted due to missingness)
## Multiple R-squared:  0.9254, Adjusted R-squared:  0.9246
## F-statistic:  1166 on 5 and 470 DF,  p-value: < 0.00000000000000022
```

```
(-.41567 * 50000) - (-.30011 * 50000)
```

```
## [1] -5778
```

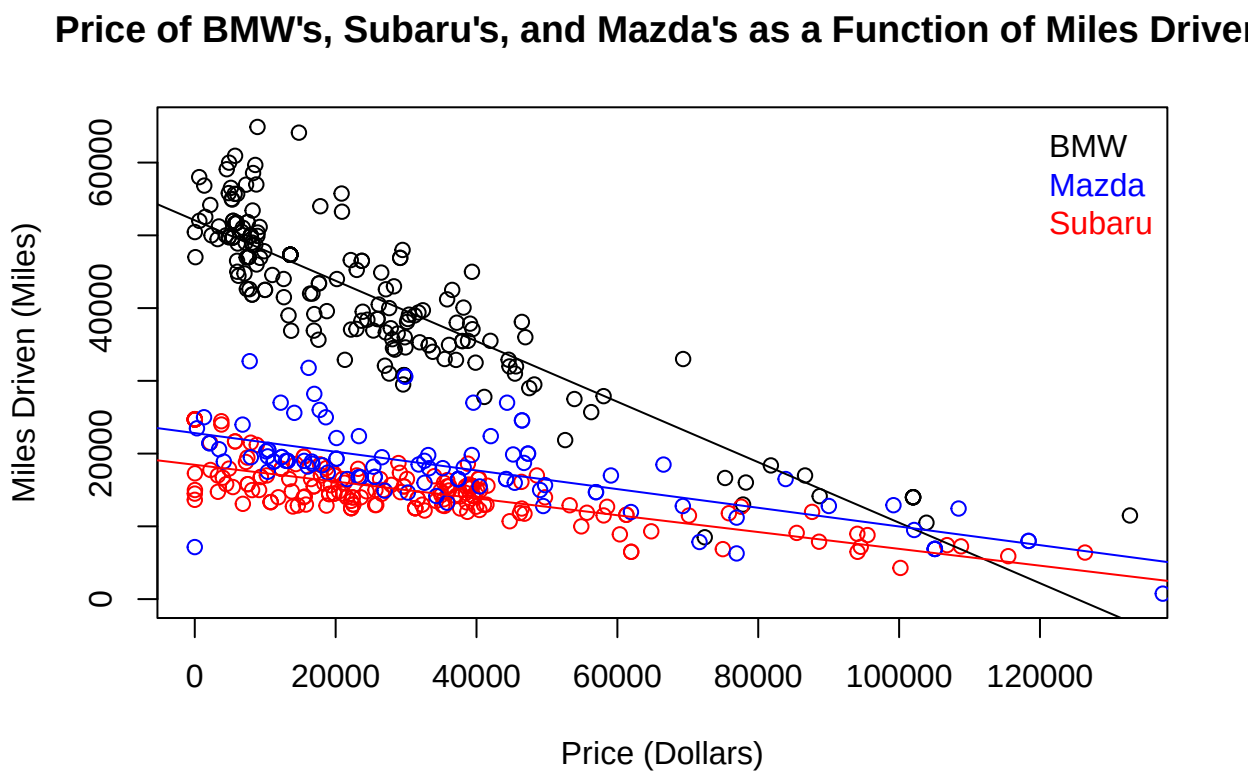
Answers

1. 92.54%
2. $5778

Part 1.5 (6 points): Now let's visualize these regression models. Start by recreating the scatter plot you created in part 1.1 using the same colors, but also add on the regression line with different y-intercepts and

different slopes based on the model you fit in part 1.4 (again use the appropriate colors). Based on this visualization, does it seem that the slopes are different for all 3 car brands?

```
# plot from 1.1
plot(price_bought ~ mileage_bought, data = filter(three_car_data, make_bought == "BMW"),
     ylim = c(0, 65000), xlab = "Price (Dollars)", ylab = "Miles Driven (Miles)",
     main = "Price of BMW's, Subaru's, and Mazda's as a Function of Miles Driven")

points(price_bought ~ mileage_bought,
       data = filter(three_car_data, make_bought == "Mazda"), col = "red")

points(price_bought ~ mileage_bought,
       data = filter(three_car_data, make_bought == "Subaru"), col = "blue")

# extract coefs and plot regressions lines
coefs <- coef(lm_car_2)

abline(coefs[1], coefs[2], col = "black")
abline(coefs[1] + coefs[3], coefs[2] + coefs[5], col = "red")
abline(coefs[1] + coefs[4], coefs[2] + coefs[6], col = "blue")

# legend
legend("topright", c("BMW", "Mazda", "Subaru"), text.col = c("black", "blue", "red"),
       bty = "n")
```



Answer

It seems that the slope is different BMW's but close to the same for Mazda's and Subaru's.

Part 2: Polynomial regression

As discussed in class, we can create models that better fit our data by adding polynomial expanded terms; i.e., we can create models of the form: $\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_1^3 + ...$ Let's explore fitting these models now. In particular, let's use polynomial regression to build models that predict the price that a car was purchased for (`price_bought`) as a function `mileage_bought` taken to different powers (i.e., *mileage*, $mileage^2$, etc.). To build these models, let's explore used Toyota Corolla data again so that we can compare some of the linear models we fit on the previous homework to models that contain non-linear functions of the predictors.

The code below recreates the `used_corollas_all` data frame you created on homework 7. Please use this data frame for the next two parts of this homework.

```
# load the data set
load("car_transactions.rda")


# use dplyr to reduce the data set to only used Corollas with under 150,000 miles
used_corollas_all <- car_transactions |>
  select(price_bought,
         mileage_bought,
         model_bought,
         new_or_used_bought) |>
  filter(model_bought == "Corolla",
         new_or_used_bought == "U") |>
  na.omit(used_corollas)


# check the size of the used_corollas_all data frame
dim(used_corollas_all)
```

```
## [1] 249   4
```

Part 2.1 (10 points):

Let's start exploring how polynomial regression can lead to better model fits, by predicting Corolla prices as a function of miles driven using polynomial models of degree 1, up to degree 5. For each model, we will then print the coefficient of determination, R^2, to see how much of the variability in the y values the model is capturing.

To do this, use a for loop where the counter index i of the for loop goes from 1 to 5. Then, inside the for loop do the following:

1. Use the `lm()` and `poly()` functions to fit a model of degree `i` and save it to an object called `curr_model`.
2. Use the `summary()` function to extract summary statistics from the `curr_model` and save these statistics in an object `curr_summary`.
3. Use the `print()` function to print the R^2 value that is stored in the `curr_summary` object.

Once you have the code working, in the answer section below fill in a table reporting the different R^2 values. Based on these values, discuss which model you think has the best fit to the used Corolla data. Should we trust the model with the highest R-squared value? Why or why not?

Hint: We only used 3 lines of code inside the for loop.

```
for(i in 1:5){
  curr_model <- lm(price_bought ~ poly(mileage_bought, i, raw = TRUE),
                   data = used_corollas_all)
  curr_summary <- summary(curr_model)
  print(curr_summary$r.squared)
}
```

```
## [1] 0.5961973
## [1] 0.6422709
## [1] 0.6424999
## [1] 0.643618
## [1] 0.6540874
```

Answer

Model degree	R^2
1	0.5961973
2	0.6422709
3	0.6424999
4	0.643618
5	0.6540874

I think the model with degree 3 has the best fit to the used Corolla data.

We should not trust the model with the highest r-squared value because it is likely overfit.

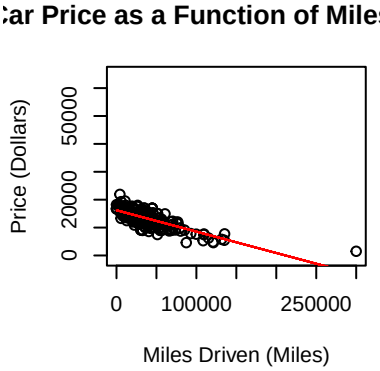
Part 2.2 (15 points): To gain better insight into which model to use, let's plot all of these polynomial models. To do this, again create a for loop that goes from 1 to 5, and inside the for loop do the following steps:

1. Use the `lm()` and `poly()` functions to fit a model of degree `i` and save it to an object called `curr_model`.
2. Use the data `x_vals_df` data frame below, in conjunction with the `predict()` function, to make predicted $\hat{y}$ values for x-values in the `x_vals_df` using the current model. Save this vector of predicted $\hat{y}$ values to an object called `y_vals_predicted`.
3. Use the `plot()` function to create a scatter plot of the data; i.e., plot `price_bought` as a function of the `mileage_bought`.
4. Use the `points()` function to add a the regression line values (from step 2) on this plot as a red line.

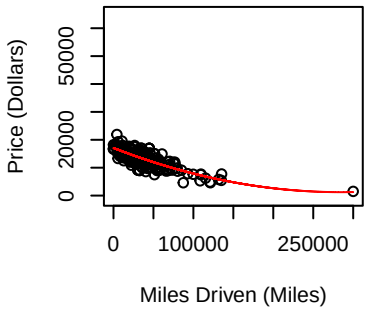
If the predicted regression line values are being cut off by the edge of the figure, go back to step 3 and adjust the y-limits of the scatter plot using the `ylim = c(lower_lim, upper_lim)` argument.

Based on these plots, describe in the answer section below, which model do you think is the best model and why it is better than each of the other models.

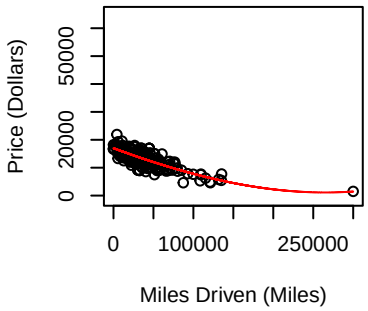
```r
for(i in 1:5){
  curr_model <- lm(price_bought ~ poly(mileage_bought, i, raw = TRUE),
                   data = used_corollas_all)
  par(mfrow = c(2, 3))
  x_vals_df <- data.frame(mileage_bought = 0:300000)
  y_vals_predicted <- predict(curr_model, newdata = x_vals_df)
  plot(used_corollas_all$mileage_bought, used_corollas_all$price_bought,
       ylim = c(-1000, 65000), xlab = "Miles Driven (Miles)",
       ylab = "Price (Dollars)", main = "Car Price as a Function of Miles Driven")
  points(x_vals_df$mileage_bought, y_vals_predicted, type = "l", col = "red")
}
```

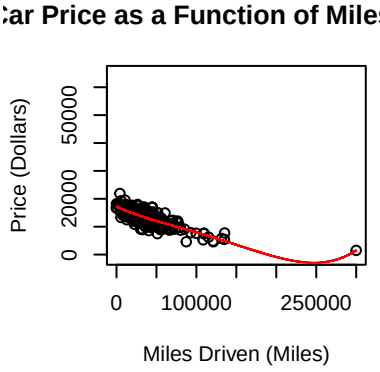


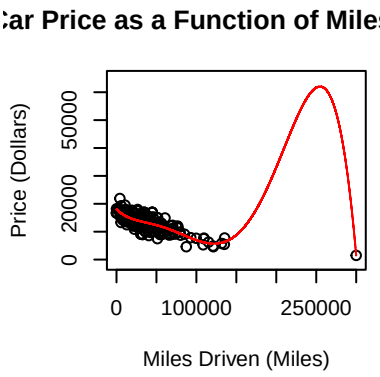
:ar Price as a Function of Miles D



:ar Price as a Function of Miles D




ar Price as a Function of Miles D
Price (Dollars)
0
20000
50000
0
100000
250000
Miles Driven (Miles)



Answer

I think the third model is best. The first models does not include the outlier, the fourth model predicts a negative price around 250000 miles, and the fifth model seems overfit.

Part 2.3 (15 points): As we discussed in class, the R^2 value always increases (or stays the same) as more variables (or a higher degrees polynomials) are added to the model. Thus if one was to judge how "good" a model was based on the R^2 statistic value, one would always choose the most complex model that has the most explanatory variables in it.

As we also discussed in class, there are other statistics and methods that can potentially give better measures of how well a model will be able to make predictions on new data. Let's empirically evaluate how well these methods work in terms of there ability to assess how well a model fits the data.

The statistics for model fit that we will compare are: R^2, R^2_{adj}, AIC and BIC. To assess how good these methods are for evaluating models, please create objects called `all_r_squared`, `all_adj_r_squared`, `all_aic`, and `all_bic` that will store the results from evaluating our different model evaluation statistics. Then create a for loop that does the following:

1. Use the `lm()` and `poly()` functions to fit a model of degree `i` and save it to an object called `curr_model`.
2. Use the `summary()` function to get a summary of the model. Then extract the R^2 and adjusted R^2 statistics and save them in the `i`th slot of the `all_r_squared` and `all_adj_r_squared` vectors.
3. Use the `AIC()` and `BIC()` functions on the model object to extract the AIC and BIC statistics and save these values to the `i`th slot of the `all_aic` and `all_bic` vectors.

Then, outside of the for loop, use the `which.max()` function and `which.min()` functions to determine which model each of these statistics would select. Fill in the table below by placing an x in the appropriate column for the degree model that you would select based on each statistic (you will fill in the last row of the table in part 2.4). Also comment on which statistics you think lead to the best model choice (looking at the plots you created in part 2.2 could be helpful to answer this question).

```
all_r_squared <- NULL
all_adj_r_squared <- NULL
all_aic <- NULL
all_bic <- NULL

for(i in 1:5){
  curr_model <- lm(price_bought ~ poly(mileage_bought, i, raw = TRUE),
                   data = used_corollas_all)
  curr_summary <- summary(curr_model)
  all_r_squared[i] <- curr_summary$r.squared
  all_adj_r_squared[i] <- curr_summary$adj.r.squared

  all_aic[i] <- AIC(curr_model)
  all_bic[i] <- BIC(curr_model)
}

which.max(all_r_squared)
```

```
## [1] 5
```

```
which.max(all_adj_r_squared)
```

```
## [1] 5
```

```
which.min(all_aic)
```

```
## [1] 5
```

```
which.min(all_bic)
```

```
## [1] 2
```

Answer

	1	2	3	4	5
R^2					*
R^2_{adj}					*
AIC					*
BIC		*			
cross-val		*			

Answer

I think the BIC statistic led to the best model choice, since (as noted in 2.2) model five is overfit.

Part 2.4 (15 points):

As have discussed in class, we can also use cross-validation to assess model fit. To do this we build a model on a *training set* of data and then evaluating the accuracy of the predictions on a separate *test set* of data. When evaluating the model, we can use the *mean squared prediction error* (MSPE) as a measure of how accurately the model is making predictions on new data. This measure is defined as:

$$MSPE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

Where the y_i come from the m points in the *test set*.

For more information on this measure, see Wikipedia.

The code below splits the data in half and creates a *training set* that we will use to learn the estimated coefficients $\hat{\beta}$'s and also creates a *test set* with the other half of the data that we can evaluate how accurately the model can make predictions. Use a for loop to create models of degree 1 to 5 using the training data, have the models make predictions on the test set, and then calculate the MSPE based on their predictions. Store all the results accumulated in the for loop in a vector called `all_MSPE`. Then, add to the table above (in part 2.3) which model has the minimum MPSE for cross-validation by putting an x in the appropriate column. Also be sure to print out the MPSE values you get to show your work.

```
# create the training set and the test set
total_num_points <- dim(used_corollas_all)[1]
num_training_points <- floor(total_num_points/2)

training_data <- used_corollas_all[1:num_training_points, ]
test_data <- used_corollas_all[(num_training_points + 1):total_num_points, ]



# run a for loop to calculate the MSPE for models of degree 1 to 5
# then find the model with the minimal MSPE

all_MSPE <- NULL
for(i in 1:5){
  curr_fit <- lm(price_bought ~ poly(mileage_bought, i, raw = TRUE),
                 data = training_data)
  curr_test_predictions <- predict(curr_fit, newdata = test_data)
  all_MSPE[i] <- mean((test_data$price_bought - curr_test_predictions)^2)
}
which.min(all_MSPE)
```

```
## [1] 2
```

```
print(all_MSPE)
```

```
## [1]    4780843    4681972  263498684 3490880923 7138348751
```

Part 2.5 (5 points):

This is a "challenge problem" that you should try to figure out without getting help from the TAs.

Note: this challenge question is difficult, so try your best and if you can't get it, it's only worth 5 points so it will not have much impact on your grade in the class.

Please run a three-fold cross-validation where you:

1. Learn the model parameter estimates on two-thirds of the data.
2. Make predictions on the remaining data (1/3 of the data).
3. Repeat steps 1 and 2 putting aside a different 1/3 of the data each time for the test set and training on the remaining data.
4. Average the MSPE results over the 3 cross-validation splits to get the mean cross-validation results for a model of a given degree.
5. Repeat this procedure for models of degree 1-5 (i.e., by having an outer for loop). Hint 1: a good strategy is to get the code working for the degree 1 polynomial fit and once this is working then add the outer loop to get the code to work for the other degree polynomial models.

Print the mean cross-validation results from the models of each degree (i.e., print the 5 MSPE values). Does the three-fold cross-validation results lead to the same conclusions compared to using only 1 training and test split as was done in part 2.4?

Hint 2: In each cross-validation loop, it could be useful to first find the indices (i.e., row numbers) of data points that should be in the test set. You can then use the `setdiff()` function from all indices (row numbers) to get the remaining points that should be in the training set.

```
all_MSPE_1 <- NULL
all_MSPE_2 <- NULL
all_MSPE_3 <- NULL
avg_MSPE <- NULL

for(i in 1:5){
  # first set
  total_num_points <- dim(used_corollas_all)[1]
  num_training_points <- floor(total_num_points/3)

  # first two-thirds
  training_data_1 <- used_corollas_all[1:2*num_training_points, ]
  # last third
  test_data_1 <- used_corollas_all[(2*num_training_points + 1):total_num_points, ]
  curr_fit_1 <- lm(price_bought ~ poly(mileage_bought, i, raw = TRUE),
                   data = training_data_1)
  curr_test_predictions_1 <- predict(curr_fit_1, newdata = test_data_1)

  all_MSPE_1[i] <- mean((test_data_1$price_bought - curr_test_predictions_1)^2)

  # second set
  # last two-thirds
  training_data_2 <- used_corollas_all[num_training_points+1:total_num_points, ]
  # first third
  test_data_2 <- used_corollas_all[1:num_training_points, ]
```

```
  curr_fit_2 <- lm(price_bought ~ poly(mileage_bought, i, raw = TRUE),
                   data = training_data_2)
  curr_test_predictions_2 <- predict(curr_fit_2, newdata = test_data_2)

  all_MSPE_2[i] <- mean((test_data_2$price_bought - curr_test_predictions_2)^2)

  # third set
  # first and last third
  training_data_3 <- c(used_corollas_all[1:num_training_points, ],
                       used_corollas_all[(num_training_points*2 + 1):total_num_points, ])
  # middle third
  test_data_3 <- used_corollas_all[num_training_points:(2*num_training_points), ]

  curr_fit_3 <- lm(price_bought ~ poly(mileage_bought, i, raw = TRUE),
                   data = training_data_3)
  curr_test_predictions_3 <- predict(curr_fit_3, newdata = test_data_3)

  all_MSPE_3[i] <- mean((test_data_3$price_bought - curr_test_predictions_3)^2)
  avg_MSPE[i] <- mean(c(which.min(all_MSPE_1), which.min(all_MSPE_2),
                        which.min(all_MSPE_3)))
}
```

```
## Warning in predict.lm(curr_fit_1, newdata = test_data_1): prediction from a
## rank-deficient fit may be misleading

## Warning in predict.lm(curr_fit_1, newdata = test_data_1): prediction from a
## rank-deficient fit may be misleading

## Warning in predict.lm(curr_fit_1, newdata = test_data_1): prediction from a
## rank-deficient fit may be misleading

## Warning in predict.lm(curr_fit_1, newdata = test_data_1): prediction from a
## rank-deficient fit may be misleading
```

```
print(avg_MSPE)
```

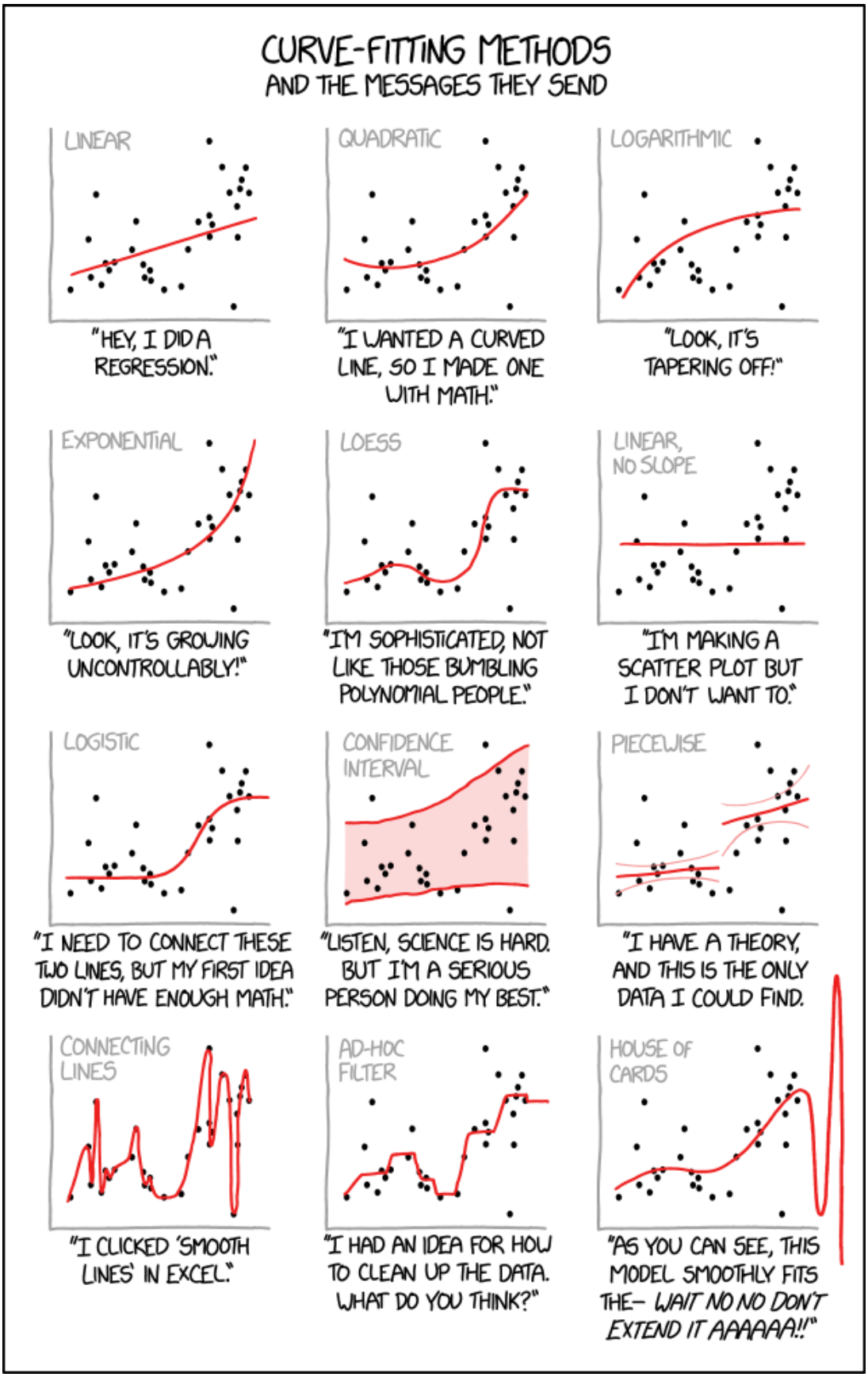
```
## [1] 1 1 1 1 1
```

Answer

The three-fold cross-validation result does not lead to the same conclusion compared to 2.4.

Reflection (3 points)

Please reflect on how the homework went by going to Canvas, going to the Quizzes link, and clicking on Reflection on homework 8.


CURVE-FITTING METHODS
AND THE MESSAGES THEY SEND
LINEAR
"HEY, I DID A REGRESSION."
QUADRATIC
"I WANTED A CURVED LINE, SO I MADE ONE WITH MATH."
LOGARITHMIC
"LOOK, IT'S TAPERING OFF!"
EXPONENTIAL
"LOOK, IT'S GROWING UNCONTROLLABLY!"
LOESS
"I'M SOPHISTICATED, NOT LIKE THOSE BUMBLING POLYNOMIAL PEOPLE."
LINEAR, NO SLOPE
"I'M MAKING A SCATTER PLOT BUT I DON'T WANT TO."
LOGISTIC
"I NEED TO CONNECT THESE TWO LINES, BUT MY FIRST IDEA DIDN'T HAVE ENOUGH MATH."
CONFIDENCE INTERVAL
"LISTEN, SCIENCE IS HARD. BUT I'M A SERIOUS PERSON DOING MY BEST."
PIECEWISE
"I HAVE A THEORY, AND THIS IS THE ONLY DATA I COULD FIND.
CONNECTING LINES
"I CLICKED 'SMOOTH LINES' IN EXCEL."
AD-HOC FILTER
"I HAD AN IDEA FOR HOW TO CLEAN UP THE DATA. WHAT DO YOU THINK?"
HOUSE OF CARDS
"AS YOU CAN SEE, THIS MODEL SMOOTHLY FITS THE– WAIT NO NO DON'T EXTEND IT AAAAAA!!"