Homework 9

The purpose of this homework is to examine overfitting and logistic regression. 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 11:59pm on Sunday November 15th.

As always, if you need help with any of the homework assignments, please attend the TA office hours which are listed on Canvas and/or ask questions on Piazza. Also, if you have completed the homework, please help others out by answering questions on Piazza.

Note: this homework was a little shorter an more straightforward than previous homework in order to give you time to start on your final project. Please take advantage of this and start thinking about your final project (e.g., it would be good to find a data set you can use for your final project).

Part 1: Model selection

On homework 8, you fit linear regression models of of degree, 1, 3, and 5 for predicting the price of a used Toyota Corolla based on the number of miles driven and calculated the proportion of the price explained by these different models using the R^2 statistic. 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.

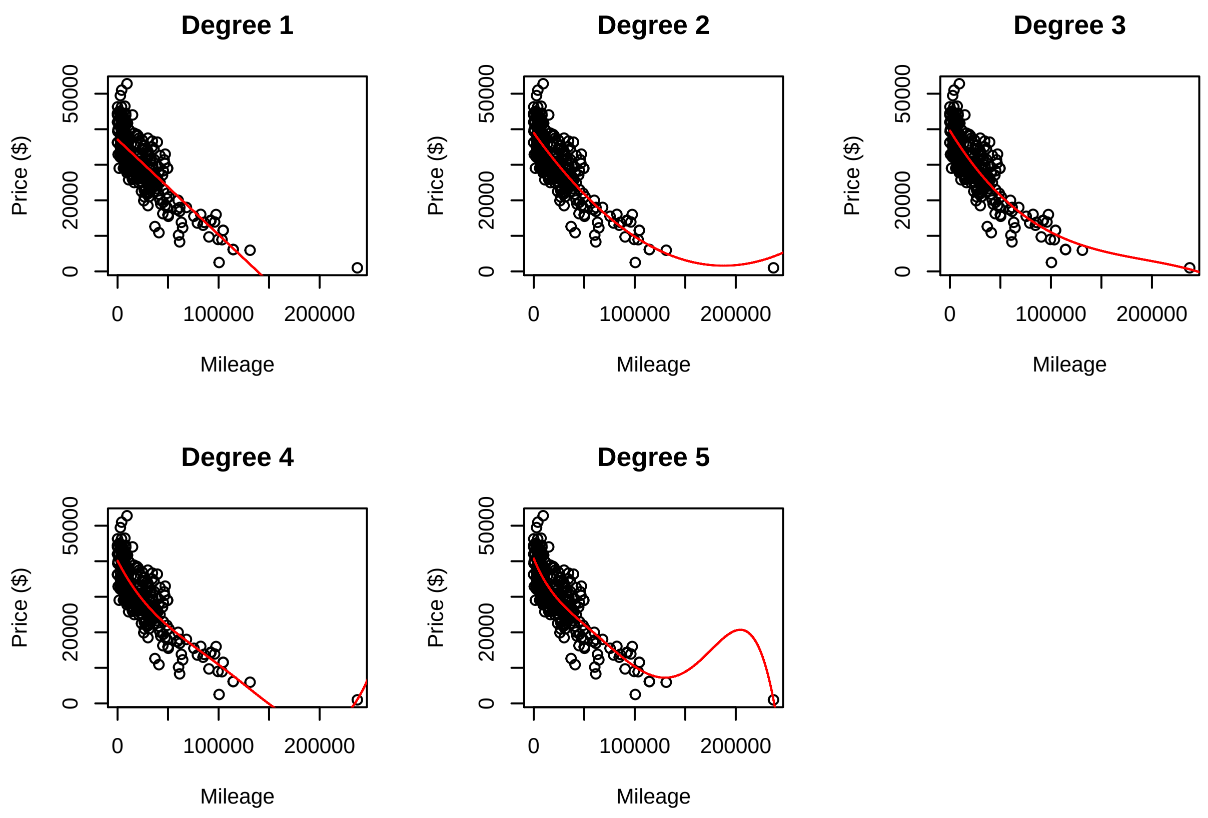
As we also discussed in class, there are other statistics and methods that can potentially give better measures of how well a model will be able to make predictions on new data. In the set of exercises below, you will empirically evaluate these different methods for assessing model fits to see how well they work.

Part 1.1 (5 points): The code below loads the Edmunds car transaction data and creates a data frame that has data from the used BWM model '3 Series'. It also plots the model fits for predicting price as a function of miles driven for models up to degree 5. Note: we are using the `poly()` function to create polynomial expansions of the `mileage_bought` variable which is easier than writing I(mileage_bought^2) + I(mileage_bought^3) etc. (The `poly()` function also creates 'orthogonal polynomials' which eliminates multicollinearity).

Based on looking at the model fits, which degree polynomial do you think is the best fit to the data?

```r
load('car_transactions.rda')


car_model_name <-  "3 Series"   # "MAZDA3"

```

```r
used_cars <- car_transactions%>%
  select(price_bought,mileage_bought, model_bought, make_bought, new_or_used_bought) %>%
  filter(model_bought == car_model_name, new_or_used_bought == "U") %>%
  na.omit()


par(mfrow = c(2, 3))
x_vals_df <- data.frame(mileage_bought = 0:300000)


for (i in 1:5) {

  curr_model <- lm(price_bought ~ poly(mileage_bought, degree = i), data = used_cars)

  model_summary <- summary(curr_model)

  y_vals_predicted <- predict(curr_model, newdata = x_vals_df)

  plot(price_bought ~ mileage_bought, data = used_cars, xlab = "Mileage",
       ylab = "Price ($)", main = paste("Degree", i))

  points(x_vals_df$mileage_bought, y_vals_predicted, type = "l", col = "red")


}
```



Answers

Based on this, I think that Degree 3 is the best fit to the data because it seems to fit the data without major overfitting. Degree 1 does not seem adequate since the data is not linear. Degree 2 is not as good of a model as Degree 3 since the data seems to be overfitting from 150000 onwards, but it still seems to be an adequate model for the range of most of the data points. Degree 4 also is an instance of overfitting since it does not make sense that the price dip and then increase again as the mileage increases from 80,000 to 120,000. Degree 5 also seems so overfit since it is concave down for the latter quarter of the graph qnd it would not make sense for the price to increase and then subsequently decrease faster and faster as the mileage increases.

Part 1.2 (15 points): Now let's try calculating different measures of model fit for polynomials from degree 1 to degree 5. Use a for loop that creates models of degree 1 to degree 5 and saves the following statistics:

1. R^2: save to a vector called `all_r_squared`
2. R^2_{adj}: save to a vector called `all_adj_r_squared`
3. AIC: save to a vector called `all_aic`
4. BIC: save to a vector called `all_bic`

Then 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 for each statistic (you will fill in the last row of the table in part 1.3). Also comment on which statistics you think are leading to the best model choice.

```r
# objects to store the results
all_r_squared <- NULL
all_adj_r_squared <- NULL
all_aic <- NULL
all_bic <- NULL


# create a for loop to extract the relevant statistics
for(i in 1:5){
  curr_model<-lm(price_bought~poly(mileage_bought, degree = i), data = used_cars)
  model_summary<-summary(curr_model)

  all_r_squared[i]<-model_summary$r.squared
  all_adj_r_squared[i]<-model_summary$adj.r.squared
  all_aic[i]<-AIC(curr_model)
  all_bic[i]<-BIC(curr_model)


}

all_r_squared
```

```
## [1] 0.6638238 0.7147759 0.7177723 0.7191614 0.7203447
```

```r
all_adj_r_squared
```

```
## [1] 0.6628494 0.7131177 0.7153038 0.7158768 0.7162441
```

```r
all_aic
```

```
## [1] 6928.129 6873.096 6871.431 6871.719 6872.254
```

```r
all_bic
```

```
## [1] 6939.677 6888.493 6890.678 6894.815 6899.199
```

```r
# print the degree selected by each model selection method using the which.max() or which.min() functio

which.max(all_r_squared)
```

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

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

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

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

```
## [1] 3
```

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

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

Answer

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

Answer

Since the AIC statistic leads to Model choice 3, we can predict that it is the statistic that leads to the best model choice. Also since the cross-validation and BIC lead to Model choice 2, we can predict that those statistics also might lead to the best model choice. The fifth model seems to be overfitting, so we can conclude that the

$$R^2$$

and

$$R^2_{adj}$$

do not lead to the best model choice.

Part 1.3 (15 points):

As we also discussed in class, we can use cross-validation to assess model fit: building 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:

$MSPE = \frac{1}{m}\Sigma_{i=1}^{m}(y_i - \hat{y}_i)^2$ 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. Add to the table above which model has the minimum MPSE by putting an x in the appropriate column and print out the MPSE values you get to show your work.

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

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

all_MSPE<-NULL
for(iDegree in 1:5){

current_model<-lm(price_bought~poly(mileage_bought, degree = iDegree), data = training_data)
test_predictions = predict(current_model, newdata = test_data)

all_MSPE[iDegree]<-mean((test_data$price_bought-test_predictions)^2)
}

# 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
```

```
## [1]     32627245     24946068     80971405    828584727 209208623673
```

```r
which.min(all_MSPE)
```

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

Part 1.4 (15 points):

Now rerun parts 1.1 to 1.3 but using only Mazda 3's instead of BMW 3 series (e.g., filter for "MAZDA3"). If you have written your code in a flexible way you should really only have to change the line "3 Series" to "Mazda 3" in part 1.1 and the rest of the code should run. Then fill out the table below for Mazda 3's, and answer all the following questions for the Mazda 3 data:

a. From looking at the models fits, which degree model seems to fit the data best?
b. Which statistics do you think are leading to the best model choice?
c. Overall (for both car brands), which model selection method do you think is working best and why?

Answer

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

a. From looking at the model fits, Degree 2 seems to fit the data the best. Degree 1 does not make sense since plotting a straight line to the data does not make sense since the data is not linear. Degree 3 seems to be overfitting since our data is not supposed to be concave down in the latter quarter of the graph. Degree 4 overfits by being concave up at the end (it also does not make sense that the price increases after a certain amount of mileage. Degree 5 also overfits since it is illogical that price alternate between increasing and decreasing after certain amounts of mileage.

b. Cross-validation seems to lead to the best model choice since it coincides with our preferred model choice (model 2).

c. Since cross validation leads us to infer that Degree 2 is the best model choice for the BMW 3 Series and the Mazda 3 (although Degree 2 might not be the optimal model choice for the BMW, it still is a relatively good fit), we can infer that the best model selection method is cross-validation.

Bonus question (0 points):

Run three-fold cross-validation where you split the data into 3 parts and you:

a. Learn the model parameter estimates on two of the data splits (2/3rds of the data).
b. Make predictions on the last data split (1/3 of the data).
c. Repeat steps *a* and *b* leaving a different test set out each time and training on the other 2 splits.
d. Average the MSPE results over the 3 cross-validation splits.
e. Repeat this procedure for models of degree 1-5 (i.e., by having nested for loops).

Does the three-fold cross-validation results lead to the same conclusions as using only 1 training and test split?

Answer

Part 2: Logistic regression

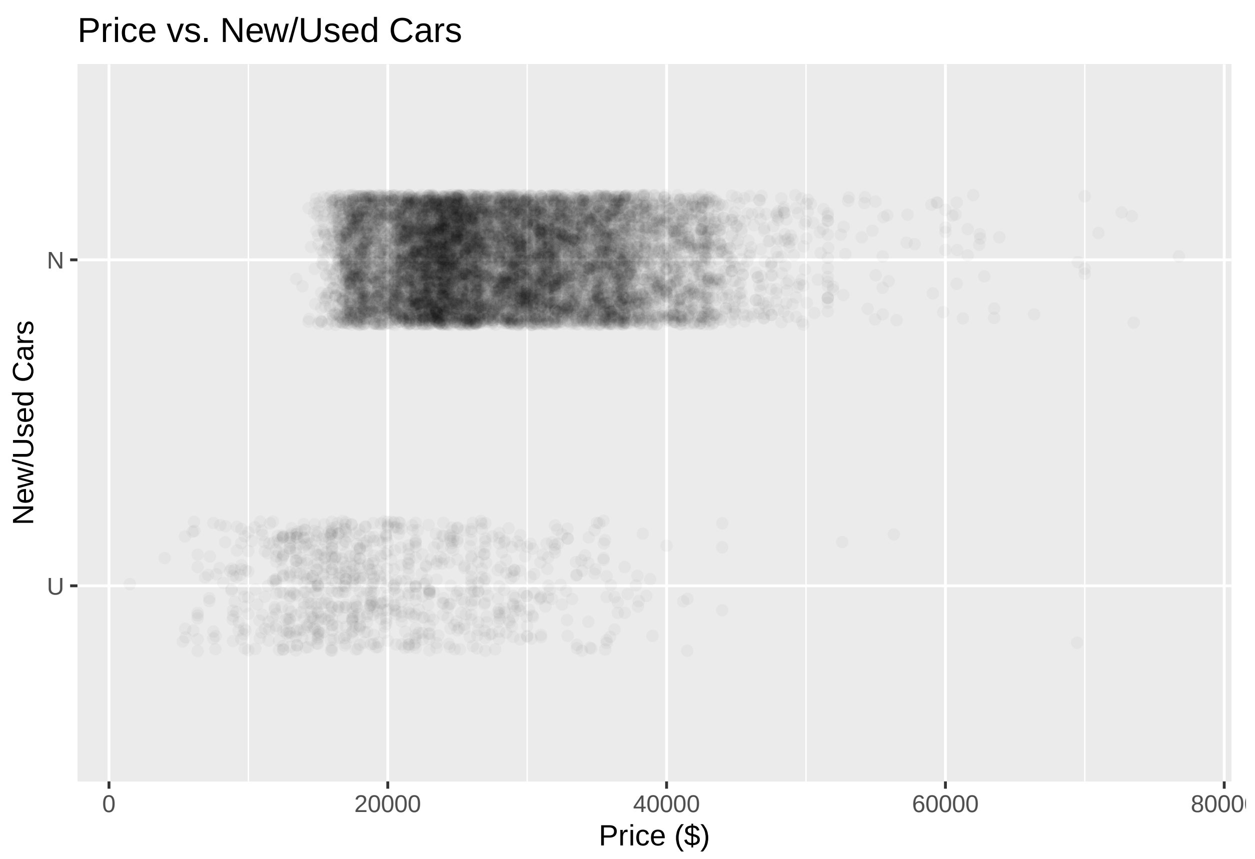
In these exercises you will examine logistic regression by fitting models that can give the probability that a car is new or used based on other predictors including the car's price.

The exercises are straightforward extensions of the material covered in class, so if you get stuck I suggest you watch the class videos. Also, make sure you understand what all the code does in order to be prepared for the final exam.

Part 2.1 (10 points): The code below creates a data frame called `toyota_data` that has sales information on Toyota cars. It also changes the order of the levels on the categorical variable `new_or_used_bought` which will make the plotting and modeling work better in these exercises.

To start, create a visualization of the data using ggplot where price is on the x-axis and whether the car is new or used is on the y-axis. Use a `geom_jitter()` to plot points that have less overlap, and set the alpha transparency appropriately to also help with over-plotting. Finally, report how many of the cars in the `toyota_data` are new and how many are used.

```
# get toyota cars and change the order of the levels of the new_or_used_bought variable
toyota_data <- filter(car_transactions, make_bought == "Toyota") %>%
  mutate(new_or_used_bought = factor(as.character(new_or_used_bought, levels = c("U", "N")))) %>%
  mutate(new_or_used_bought = relevel(new_or_used_bought, "U")) %>%
  na.omit()


# visualize the data

toyota_data%>%
  ggplot(aes(x=price_bought, y=new_or_used_bought))+
  geom_jitter(alpha = 0.03, position = position_jitter(height = .2))+
  ggtitle("Price vs. New/Used Cars")+
  xlab("Price ($)")+
  ylab("New/Used Cars")
```



```r
table(toyota_data$new_or_used_bought)
```

```
##
##     U     N
##   915 10335
```

Answer:

In the 'toyota_data', there are 915 Used and 10335 New cars.

Part 2.2 (12 points): Now fit a logistic model to predict the probability a car is new or used based on the price that was paid for the car using the `glm()` function. Then extract the offset and slope coefficients and save them to objects called `b0` and `b1`. Using these `b0` and `b1` coefficients, write a function `get_prob_new()` that takes a price a car was sold for as an input argument and returns the probability the car was new using this equation:

P(new|price) = $\frac{e^{\hat{\beta}_0+\hat{\beta}_1 \cdot price}}{1+e^{\hat{\beta}_0+\hat{\beta}_1 \cdot price}}$

Finally, use the `get_prob_new()` function to predict the probability that a car that sold for $10,000 was a new car and report this probability. Comment on whether you would expect a probability this high based on the visualization of the data you created in part 2.1.

```r
# build the logistic regression function

log_reg_fit<-glm(new_or_used_bought~price_bought, data = toyota_data, family = 'binomial')

# extract the coefficients
b0<-coefficients(log_reg_fit)[1]
b1<-coefficients(log_reg_fit)[2]


# create the prediction function

get_prob_new<-function(the_price){
  prob_new<-exp(b0+b1*the_price)/(1+exp(b0+b1*the_price))
  names(prob_new)<-NULL
  return(prob_new)

}





# what is the probability that a car that costs 10,000 is new?


get_prob_new(10000)
```

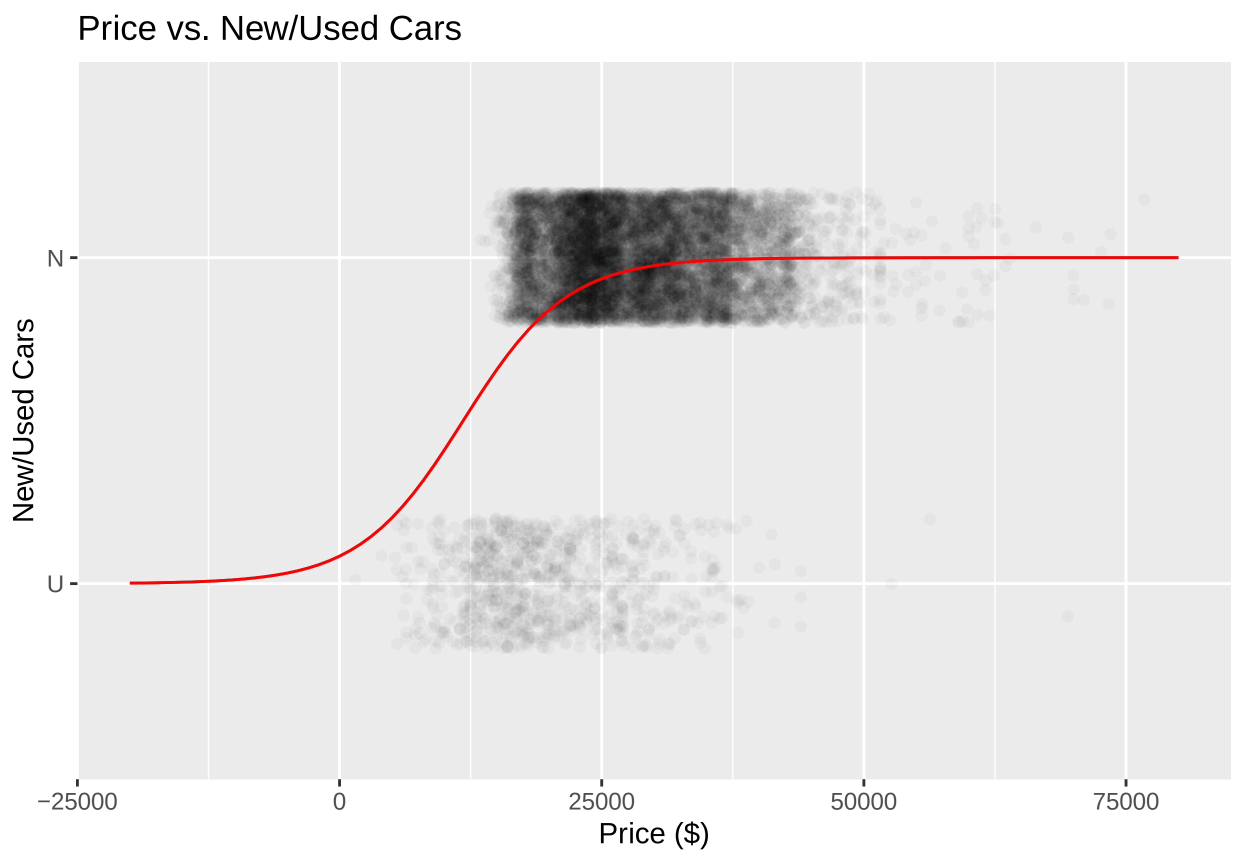
```
## [1] 0.4104008
```

Answer: The probability that a car that costs $10,000 is new is 41.04%

No, I would not expect a probability this high based on my visualization. There seem to Be no new car values at $10,000 since there is barely any data in my data-set and most of the data are new cars. The probability is probably high since the majority of cars are new cars and if you pick any car in the data set, it is most likely going to be a new car.

Part 2.3 (10 points): Now plot the data again as you did in part 2.1, but add to this plot a plot of the logistic regression function you created in part 2.2 as a red line.

```r
# plot the logistic regression function

plot_prob_line<-function(the_price){

  get_prob_new(the_price)+1
}


toyota_data%>%
  ggplot(aes(x=price_bought, y=new_or_used_bought))+
  geom_jitter(alpha = 0.03, position = position_jitter(height = .2))+
  ggtitle("Price vs. New/Used Cars")+
  xlab("Price ($)")+
  ylab("New/Used Cars")+
  stat_function(fun = plot_prob_line, color = "red")+
  xlim(-20000, 80000)
```



Part 2.4 (15 points): Let's now fit a multiple logistic regression to the data in order to get the probability a car is new given the price of the car (`price_bought`) and the number of miles driven (`mileage_bought`). Extract from this function the regression coefficient estimates and save them to objects called `b0`, `b1` and `b2`. The create a function called `get_prob_new2()` that uses the coefficients to predict a car is new using the equation:

P(new|price, mileage) $= \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 \cdot price + \hat{\beta}_2 \cdot mileage}}{1+e^{\hat{\beta}_0 + \hat{\beta}_1 \cdot price + \hat{\beta}_2 \cdot mileage}}$

Finally, use the `get_prob_new2()` function to predict the probability that a car that sold for 10,000 and had 500 miles was a new car, and the probability that a car that sold for 10,000 and had 5000 miles was a new car and report these probabilities in the answer section.

ln(p/1-p)=b0+b1*the_price +b2*the_mileage

```
# fit a multi-logistic regression model and create a function that returns probabilities a car is new
log_reg_fit_2<-glm(new_or_used_bought~price_bought+mileage_bought, data = toyota_data, family = 'binomia

b0<-coefficients(log_reg_fit_2)[1]
b1<-coefficients(log_reg_fit_2)[2]
b2<-coefficients(log_reg_fit_2)[3]
```

```
get_prob_new_2<-function(the_price, the_mileage){
  prob_new_2<-exp(b0+b1*the_price+b2*the_mileage)/(1+exp(b0+b1*the_price+b2*the_mileage))
  names(prob_new_2)<-NULL
  return(prob_new_2)
}

# predict $10,000 price and 500 miles driven
get_prob_new_2(10000, 500)
```

```
## [1] 0.9708602
```

```
# predict $10,000 price and 5000 miles driven
get_prob_new_2(10000, 5000)
```

```
## [1] 0.3771669
```

Answer:

The probability that a car sold for 10,000 and with 500 miles driven is new is 97.08% while the probability that a car sold for $10,000 with 5000 miles driven is 37.77%.

Part 2.5 Bonus problem (0 points): See if you can use ggplot to visualize how the price and mileage influence the probability estimates that a car is new, by creating a two dimensional "heat map" where a gradient of color corresponds to different probabilities (e.g., darker shades could correspond to lower probabilities and lighter shades could correspond to higher probabilities). To do this you can follow the steps we discussed in class which involved created a data frame that has columns with prices, miles sold, and the corresponding probability values. Then use ggplot with the `geom_raster` to visualize the heat map.

```
# create a 2D plot of the probability that a car is new as a function of price and mileage
```

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 9.