UNIVERSITY OF MARYLAND

ROBERT H. SMITH SCHOOL OF BUSINESS

ASSIGNMENT 3

Your objective is to develop models to predict the outcome variable “BadBuy”, which labels whether a car purchased at an auction was a “bad buy” (lemon). Your task is to build a model to guide auto dealerships in their decisions on whether to bid for and purchase a vehicle. You can also apply your learning from this analysis to make more data-informed car-buying decisions!

You will use **carvana.csv** which contains data from 10,062 car auctions as provided by Carvana. Auto dealers purchase used cars at auctions with a plan to sell them to consumers, but sometimes these auctioned vehicles can have severe issues that prevent them from being resold at a profit (hence, lemons). The data contains information about each auctioned vehicle.

Data Dictionary

Variable	**Definition**
Auction	Auction provider where vehicle was purchased
Age	The years elapsed since the manufacturer's year (how old is the vehicle)
Make	Vehicle manufacturer
Color	Vehicle color
WheelType	Vehicle wheel type description (Alloy, Covers)
Odo	Vehicle odometer reading
Size	Size category of the vehicle (Compact, SUV, etc.)
MMRAauction	Auction price for this vehicle (in average condition) at the time of purchase
MMRAretail	Retail price for this vehicle (in average condition) at the time of purchase
BadBuy	Whether the vehicle is a bad purchase / lemon (“YES”) or a good investment (“NO”)

Before you start:

- Load the following libraries in the given order: *tidyverse, tidymodels, plotly, skimr, caret*
- Load the Carvana data and call it *dfc*
- Explore the dataset using `skim()` etc.

Assignment Instructions

There are two main objectives. The first is to predict the variable BadBuy as a function of the other variables. The second is to build alternative models, measure, and improve performance.

1) **(~5 points) Data preparation**

   a) Load the dataset into R and call it *dfc*. Inspect and describe the data.

   →The data has 10 variables out of which Auction, make, color, wheeltype, size, badbuy are categorical and the dependent variable is badbuy. MMRAauction, odometer reading and age are numerical variables.
   There are null values in wheeltype and some colors are not available, which may have to be handled in the further analysis.

      i) Set the seed to **52156**. Randomly split the dataset into a training dataset and a test dataset. Use **65%** of the data for training and hold out the remaining **35%** for testing.

2) **(~10 points) Exploratory analysis of the *training* data set**

   a) Construct and report boxplots of the (1) auction prices for the cars, (2) ages of the cars, and (3) odometer of the cars broken out by whether cars are lemons or not. Does it appear that there is a relationship between either of these numerical variables and being a lemon? Describe your observations from the box plots. Please also pay attention to the outliers detected by the box plots and make sense of them.

   →For the auction price v/s being lemon boxplot: -
   The median auction price for the cars not lemon is higher as compared to the cars that are lemon, as expected.
   Some lemon cars also had very high auction prices(outliers), which means that these cars are not a good investment due to very high prices.

   → For the car-age v/s being lemon boxplot: -
   The more aged cars are classified as being a bad deal, which is shown by the higher median age of the cars which are considered a bad purchase.

   → For the odometer v/s being lemon boxplot: -
   It can be inferred that the cars that have less distance covered have undergone less wear and tear and are considered to be a good purchase.

But the outliers say that even though some cars have less distance covered, it does not necessarily mean that the cars are in a good shape. They may have some other defects or breakages which is why they are considered as lemon cars.

b) Construct and report a table for the count of good cars and lemons broken up by Size (i.e., How many vehicles of each size are lemons?).
Hint: Remember `tally()`? That's one way to do it. You may want to think more systematically and use a combination of summarize(), length(), mutate(), arrange()

i) Which size of vehicle contributes the most to the number of lemons? (That is, which vehicle size has the highest *percentage* of the total lemons?)

→Medium size cars have the highest percentage of total lemons.

ii) Because the vehicles of the size you identified in (i) contribute so much to the number of lemons, would you suggest the auto dealership stop purchasing vehicles of that size? Why or why not?

→No, because the percentage of the good purchase cars is not known. Moreover, the purchase decision should also be based on the other factors such as odometer values, auction prices etc.

3) (~20 points) Run a linear probability model to predict a lemon using all other variables.

a) Compute and report the RMSE using your model for both the training and the test data sets. Use the predicted values from the regression equation. **Do not** do any classifications yet.

b) For which dataset is the error smaller? Does this surprise you? Why or why not?

→The error is smaller for Training data-set which is not surprising as the model is Built on the training data so it captures its information better as compared to the test data set.

c) Use a cutoff of 0.5 and do the classification in the test data. Compute and report the confusion matrix (recall to convert BadBuy into a factor for the confusion matrix).

i) Which type of errors (false positives and false negatives) occur more here?

→False negatives errors occur more here.

(1) For this problem, do you think a false positive or a false negative is a more serious error? Based on your answer, which metric makes a better objective?

→False negative errors are more serious because even if the car is lemon, the prediction suggests the auto dealers to purchase it.
Since False negative error carries a more significant cost, sensitivity is better metric.

d) What is the testing accuracy of your model? Based on accuracy, does the model perform better than using a random classifier (i.e., the baseline accuracy)?
Hint 1: Calculate manually if you like, or use the `confusionMatrix()` function.
Hint 2: The baseline accuracy is the accuracy you would achieve if you classified every single class as a member of the most frequent class in the actual test dataset.

→Accuracy = (TP+TN)/(TP+TN+FP+FN) = 0.673
truth class0 = 1782
truth class1 = 1739
Baseline accuracy= 1782/ (1782+1739) = 0.5061
Calculated accuracy> baseline accuracy, implies, our model has a better accuracy.

e) Compute and report the predicted "probability" that the following car is a lemon:

Auction="ADESA"	Age=1	Make="HONDA"	Color="SILVER"
WheelType="Covers"	Odo=10000	Size="LARGE"	
MMRAauction=8000	MMRAretail=10000		

Does the probability your model calculates make sense? Why or why not?

→ The predicted probability =-0.1410712
It does not make sense as probability cannot be negative.

4) (~25 points) Run a logistic regression model to predict a lemon using all other variables.
Hint 1: Don't forget to convert your dependent variable BadBuy to a factor in both datasets.

Hint 2: If you haven't yet, switch to using *caret* at this point.

a) Did you receive a rank-deficient fit error? Why do you think so? Figure out the variables causing the problem by running tally() for all your factor variables, and recode them in a way to prevent the error.

→ Yes, rank-deficient error occurs because our data contains insufficient information as there are null values in the color and wheeltype variables.

Hints: You will need to recode two factor variables:

1. *Color* has two redundant levels that need to be combined.
2. Create a new category for *Make*, call it OTHER, and recode any of the makes with less than 10 observations as OTHER.

Make sure to make the changes in the full dataset, convert BadBuy to a factor, repeat the process of setting the seed to 52156 and splitting the data.
Run your logistic regression again to confirm the rank-deficient fit error is gone.

b) What is the coefficient for Age? Provide an exact numerical interpretation of this coefficient.

→The coefficient for Age is 2.785e-01.
Holding everything else constant, one-year increase in age is associated with increase in the odds of car being lemon by a factor of 1.0690784440 (about 27.85%).

c) What is the coefficient for SizeVAN? Provide an exact numerical interpretation of this coefficient.

→The coefficient for SizeVAN is -5.982e-01.
Holding everything else constant, the odds of a car of size van being lemon is 5.497877e-01 (59.82%) lower than that of any other car with same other features.

d) Use a cutoff of 0.5 and do the classification in the test data. Compute and report the confusion matrix for your test data predictions.

→A total of 3521 predictions were made, out of which the model predicts 2062 cases to be a good purchase and 1459 cars to be lemon i.e. a bad purchase.

e) Compute and report the predicted probability using your logistic model for the same car from 3(e). What does the resulting value tell you about this particular car now? Does the result make more sense than the result in Question 3(e)? Why or why not?

→The predicted probability =0.04152115. Yes, it makes more sense as it is positive.

Pro tip: Pipe a confusion matrix (from any model) into tidy() and see what happens!

(5) (~40 points) Explore alternative classification methods to improve your predictions.

- In the models below, use a 10-fold cross validation to make the results consistent across.
- Use the same training and test data you created and used after recoding the data in Q4.
- Make all comparisons to the logistic model you have run in Q4 after recoding the data.

a) Set the seed to **123** and run a linear discriminant analysis (LDA) using all variables.

i) Compute the confusion matrix and performance measures for the test data, and compare them **with the logistic regression results**. Discuss your findings.

→The false negatives in logistic model= 721, and false negatives in LDA model= 749. Which implies that the logistic model is better.

b) Set the seed to **123** and run a kNN model using all variables.

i) Create a plot of the k vs. cross-validation accuracy.

ii) What is the optimal k? What else do you infer from the plot?

→The optimal k =19, we infer that if k value is increased beyond optimal value, the model accuracy will decrease.

Hint: To inspect the details of any model, you will need to train the model and store it before piping it into predict(). See the GitHub repository for guidance.

iii) Compute the confusion matrix and performance measures for the test data, and compare them **with the logistic regression and LDA model** results. Discuss your findings.

→knn performs worse than LDA and logistic model, which implies that the data is linear.

c) Set the seed to **123** and build a lasso model using all variables.

i) Set the seed to **123** and run a Lasso model using all variables. Report the table of variable importance in a tibble format and share your observations. **Hint:** See the Github repo for help. Use a 100-point grid between 10^{-5} and 10^{2}

→The variable WheelTypeNULL is the most important.

ii) Report the plot of variable importance for the 25 most important variables.

iii) What is the optimum lambda selected by the model? What does it mean that the algorithm chooses this particular lambda value?

→ optimum lambda = 0.0003053856, the algorithm chooses this value of lambda as it tries to find the 'sweet spot' on the bias-variance curve.

iv) Compute the confusion matrix and performance measures for the test data, and compare them **with the logistic regression, LDA, and kNN model** results. Discuss your findings.

→The number of false negative values of LR is least, hence it is better than LDA and knn

Comparing the confusion matrices, we see that the number of false negatives is the same but, the number of false positive cases more for Lasso than the LR model, hence, we can conclude that the LR model performs better.

d) Set the seed to **123** and build a (I) ridge and (II) elastic net[1] model using all variables.

i) Compute the confusion matrix and performance measures for the test data, and compare them **only with the lasso model** results. Discuss your findings. **Hint:** Use the same grid for lambda. Notice the different optimum value!

→The number of false negatives in Ridge model is less than both that of lasso and the elastic net models, hence Ridge model outperforms the others.

[1] Naive elastic net. Feel free to run a grid search but be careful not to hit the limits of your computational power!

e) Set the seed to **123** and run a quadratic discriminant analysis (QDA) with all variables

i) Have you received an error? What do you think the error you received means? Do some research and explain what you think it is about.

→Yes, we received an error as there is collinearity in the variables.

ii) Why is the rank deficiency a problem for QDA, but not for LDA?

→LDA assumes that the variance amongst the classes is equal, but QDA is less strict and allows different feature covariance matrices for different classes.

iii) Compute the confusion matrix and performance measures for the test data, and compare them **only with the LDA model** results. Discuss your findings.

→The sensitivity of LDA 0.569 is greater than that of QDA, for which it is 0.44. Hence, LDA model is better than QDA.

f) Among all the models you have studied, which model do you think is better for the given business case/problem? Discuss why you think it is better than the others. Also report the AUC curves for the models you have developed on the same chart.

→Amongst all the discussed models, Ridge performs the best as its sensitivity is the highest.

Bonus question: You may have noticed that lasso drops certain levels of Make and Color such as "Brown", keeping the other levels of the same variable ("Blue" etc.). This may not be helpful, so you may want to use a grouped lasso. Set the seed to 123 and try grouped lasso with the lambda values 50 and 100. Do the results make more sense now? Why or why not?
Hint: Run a plain lasso again with a lambda value of 0.01 and print the coefficients this time. Compare them with the coefficients from group lasso.