APSTA-GE 2047 - Fall 2018
Instructor: Ravi Shroff

HOMEWORK ASSIGNMENT 5

Instructions:

- Please submit this assignment on NYU classes by **11:00am on Thursday, 12/6/2018** (the beginning of Week 13 lecture). It is worth 100 pts.
- Work in groups of 2-3, turn in one assignment per group, and indicate group members. *Work together on all parts of the assignment.*
- A submission should include either:
    - a pdf document of all written work and any R scripts you wrote, or
    - one or more RMarkdown files.
- All scripts should be clearly written, commented, and self-contained so that the grader can easily run them to reproduce your analysis.
- You will be graded on completeness, writing and visualization quality, and effort/creativity.
- You have three weeks for this assignment, so please start early, and talk to me if you need help!

1. [20 pts] Choosing an Appropriate Validation Set

In this question, you will work with the same stop-and-frisk dataset from Homework 3. Please submit one R script called rf_script.R with all your work (and any library scripts you use), along with a pdf of any written answers.

Load the same 'sqf_08_16.csv' file that you created in Homework 3, Question 1. Filter to stops where the suspected.crime is 'cpw', and select just the following columns:

- stop id, year, found.weapon;
- precinct;
- whether the stop occurred in transit, housing, or on the street;
- the ten primary stop circumstances (stopped.bc.*);
- the ten additional stop circumstances (additional.*);
- suspect age, build, sex, height, and weight;
- whether the stop occurred inside, whether the stop was the result of a radio call, and whether the officer was in uniform;
- length of observation period;
- day, month, and time of day.

Let's call the resulting dataset **sqf** (for "Stop, Question, Frisk").

A. **[5 pts]** Create one training set and two validation sets in the following manner. Restrict **sqf** to years 2013-2014, randomly shuffle the data, and split it in half. Call one half **train_half**, and the other half **test_half**. Next, restrict **sqf** to just the year 2015, and call

this **test_later**. Remove the stop id and year columns from **train_half**, **test_half**, and **test_later**.

B. **[5 pts]** Fit a random forest model on **train_half** using the *randomForest* package in R, predicting found.weapon as a function of all features. Use 200 trees, but all other options for the model can be the default options.

C. **[10 pts]** Generate predicted *probabilities* using the model from part B for both **test_half** and **test_later**. Compute the AUC of the model on each test set. Write a paragraph describing and interpreting your results. In particular, discuss the following three questions:
    a. Why do you think the AUC on **test_half** is noticeably higher than the AUC on **test_later**?
    b. If you were planning to use this model to guide how officers make stops in the future (e.g., by having officers use the model to compute the probability that an individual suspected of criminal possession of a weapon will have a weapon, and then only making a stop if the model-estimated probability is sufficiently high), would the AUC on **test_half** or **test_later** be a better estimate of performance on unseen data?
    c. More generally, when evaluating a model using a simple training/validation split approach, should you always do the split by shuffling and splitting randomly?

Hints:

1) The syntax for fitting a randomForest model is similar to the syntax for fitting other types of models that you've seen in previous homework assignments.
2) randomForest likes variables to be factors, so convert them if necessary. Note that the outcome found.weapon should be coded as a factor, so that randomForest performs classification and not regression.
3) You can deal with missing values by just restricting to complete cases. Don't worry about standardizing real-valued variables.
4) randomForest has trouble dealing with categorical features with many values (e.g., precinct). You should deal with this by spreading your data (using the spread() command), creating a binary indicator variable for each precinct.
5) You can feel free to parallelize the fitting of your randomForest model using the *doParallel* and *foreach* packages, although it shouldn't take overly long to fit the model without parallelizing. Feel free to look online to figure out how to do this, or to ask me if you get stuck and would like to know how.

2. [80 pts] Predicting Restaurant Violations

In this question, you will assume the role of a data scientist working for New York City's Department of Health and Mental Hygiene. Your supervisor is in charge of restaurant inspections across the city, and due to hiring restrictions has been forced to reassign inspectors to other tasks. She has tasked you with building a predictive model to help prioritize where the

remaining inspectors go, and would like you to use open data to help identify which restaurants are likely to have a high “inspection score” (a higher score is usually worse!).

You will use the file ‘DOHMH_New_York_City_Restaurant_Inspection_Results.csv', and turn in one R script called restaurant_analysis.R which does all parts of the analysis. Include written answers and the plot in a separate pdf file. You should use the *tidyverse*, *lubridate*, *ROCR*, and *randomForest* libraries, and the *doParallel* and *foreach* libraries if you would like to parallelize your random forest model fitting (this is optional; see the hints for Question 1).

First, read the following links to get a sense of how the restaurant inspection process works in New York City:
-How restaurants are scored and graded.
-A description of all the columns in the dataset.
-(optional) links to reports about the grading system.
Note that the dataset consists of one row for each violation (so there are often multiple rows for each restaurant inspection, which we can generally characterize by a restaurant id and date)

A. **[5 pts]** Import and clean the entire dataset as a tibble called **all_data**.
    a. This is optional, but I *strongly* suggest you rename columns and values to make these data easier to work with. Here are some ideas:
        i. Drop the following columns: BUILDING, STREET, PHONE, DBA, ZIPCODE, RECORD DATE, VIOLATION DESCRIPTION, GRADE DATE
        ii. Make INSPECTION DATE a date object called inspection_date, and extract the year as inspection_year
        iii. Rename the appropriate columns ‘id’, ‘borough’, ‘cuisine’, ‘action’, ‘code’, ‘critical’, ‘score’, ‘grade’, ‘inspection_type’, and rename the values in the ‘action’ and ‘inspection_type’ column with shorter, simpler values.
    b. Deal with ‘Missing’ borough information, remove restaurants that haven’t been inspected,[1] remove rows without a score or with a negative score, and remove any of the the following six inspection types:
        i. 'Calorie Posting / Re-inspection',
        ii. 'Inter-Agency Task Force / Re-inspection',
        iii. 'Smoke-Free Air Act / Re-inspection',
        iv. 'Administrative Miscellaneous / Re-inspection',
        v. 'Trans Fat / Re-inspection',
        vi. 'Inter-Agency Task Force / Initial Inspection'
    c. Some restaurants received different scores for the same inspection on the same day; replace all scores for any inspection for a given restaurant on a given day by the maximum score.
B. **[10 pts]** Create the sample of data that you will use for prediction as a tibble called **restaurant_data**. We will restrict our attention to all initial cycle inspections that took

[1] Hint: look at the information about INSPECTION DATE in the description link above

place in 2015, 2016, or 2017 (i.e., **restaurant_data** should have *one row* for each initial cycle inspection of a restaurant that took place in those years). For example, suppose that there are 30 rows in the data for "Ravi's Pizza," corresponding to five different inspections, each of which had 6 violations per inspection. Suppose that one initial cycle inspection occurred in 2015, one in 2016, and one in 2017 (the other two inspections were either in other years or weren't initial cycle inspections). Then there should be exactly *three* rows in **restaurant_data** corresponding to the three initial cycle inspections of "Ravi's Pizza."

   a. Create a binary outcome variable called 'outcome', defined by whether the score for that initial cycle inspection was 28 or higher, or not.
   b. For each initial cycle inspection, just keep the following features: borough, cuisine, outcome, and inspection_year.

C. **[15 pts]** Perform some feature engineering. We will only create features that could be known *before* a given initial cycle inspection takes place.
   a. Add month and weekday to **restaurant_data**
   b. Add four features constructed from historical inspection records:[2]
      i. The number of previous inspections with score < 14 (call this num_previous_low_inspections)
      ii. The number of previous inspections with score >= 14 and < 28 (call this num_previous_med_inspections)
      iii. The number of previous inspections with score >= 28 (call this num_previous_high_inspections)
      iv. The number of previous inspections which resulted in closing or re-closing (call this num_previous_closings)
   c. Restrict **restaurant_data** to only the top 50 most common cuisines[3].

D. **[10 pts]** Create a training set of all initial cycle inspections in 2015 and 2016 (**train**), and a testing set of all initial cycle inspections in 2017 (**test**). Fit a standard logistic regression model on the training set, predicting outcome as a function of only cuisine, borough, month, and weekday. Compute the AUC of this model on the **test** dataset.

E. **[10 pts]** Fit a random forest model on **train**, predicting outcome as a function of cuisine, borough, month, weekday, and the four historical features created in Step C. Use 1000 trees, but other settings can have default values.[4] Compute the AUC of this model on the **test** dataset. How does the AUC of the random forest compare with the AUC of the logistic regression model?

F. **[10 pts]** Generate a precision plot that compares the performance of the logistic regression and random forest models on just the highest ranked inspections. Specifically,
   a. create a plot where the x-axis is the number of restaurants, ranked from highest model-estimated probability of the outcome to lowest model-estimated probability

[2] Consult the hints at the end of this assignment if necessary.
[3] This is to make the random forest model easier to fit (specifically, so you don't have to "spread" the cuisine variable). It has nothing to do with the popularity of various cuisines!
[4] Some of the hints from Question 1 (about fitting random forest models) might help here as well.

of the outcome, and the y-axis displays the corresponding model precision (e.g., if you were to use the random forest model to rank all restaurants from most likely to have the outcome to least likely to have the outcome, then a point like (100, 0.3) would indicate that among the 100 highest-ranked restaurants, 30 of them had the outcome). This is just like the performance plot you made in Homework 3, except the x-axis should be on the absolute scale (not percent scale), and the y-axis should display precision instead of recall.

   b. There should be two curves on this plot, one for logistic regression and one for random forest. Restrict the x-axis to be a reasonable range, say from 100 to 2000 inspections.

G. **[10 pts]** Write a paragraph or two about your results. Based on the AUCs you computed in steps D) and E), would you choose one model over the other? What about based on the precision plots? How can two models both have fairly low AUC, but one has much higher precision on the highest ranked inspections?

H. **[10 pts]** Finally, write a paragraph or two (or more!) about any possible ethical issues involved in using such a predictive model to prioritize restaurant inspections. Here are a few open-ended questions to guide you, although you should feel free to explore your own questions:

   a. Do you think the fields in the data are accurately recorded? If some might be more prone to human bias (implicit or explicit) in how they are recorded, which might they be, and why? In particular, do you think that the outcome we are trying to predict is objectively well-measured?
   b. Does it make sense to prioritize inspections based on whether or not they have a high score? What about whether or not they are likely to close, or to have a critical health violation?
   c. Can you think of other data that could be collected or other oversight processes that could be put into place to make sure that you are prioritizing restaurant inspections in an “accurate and fair” manner?

Hints:

1) There are many ways to create the historical features in Step C), but the easiest might be to create a new tibble by joining **restaurant_data** with **all_data** (restricted to just id, score, action, and inspection_date) using the merge() command with all.x==T and allow.cartesian==T. Then take this joined dataset, filter so that inspection_date.y < inspection_date.x (so you are only looking at historical investigations), then group by id and inspection_date.x, then create the historical features.
2) Make sure to replace NA values with zeros for the historical features for restaurants that have no prior inspections.
3) Constantly look at the data after you create new variables to make sure that your commands work as expected. Restrict to a small sample, if necessary, and check by hand that your commands are doing what you think!
4) If there are cuisines in the test data that don’t appear in the training data, throw out those rows.