Assignment 1

Please complete in groups of 4 or less.

1.) Download the Acquire Valued Shoppers Challenge data from Kaggle:
https://www.kaggle.com/c/acquire-valued-shoppers-challenge/data

--

You may need to create a Kaggle profile to do so. You can read more about this data on the Kaggle competition data page linked above.

2.) Decompress the contest data and read it into a data manipulation tool. Consider the amount of RAM you have available and how different tools handle data in-memory and/or on-disk. This assignment will provide guidance for using SAS assuming most students will not have large amounts of RAM available.

--

The gzipped files can be unzipped into normal CSV files using a standard application such as 7Zip (http://www.7-zip.org/) or the unix command line application gunzip. Just storing the *.csv files may require more than 22 GB of disk space. You may need much more disk space for intermediate operations!

Reference lines 335-349 of https://github.com/jphall663/GWU_data_mining/blob/master/01_basic_data_prep/src/raw/sas/SAS_Part_0_Base_SAS_PROC_SGPLOT.sas for more information on loading files into SAS. (You may also use File -> Import.)

Hint: once the files are read into SAS, you may delete the original CSVs, but the SAS data sets will not be stored permanently, i.e. after shutting down SAS, unless you place them in a permanent SAS library such as the sasuser library or a library you create. To save disk space, you can delete large files as you finish parts of the assignment.

--

You should now have 4 data sets:

- *offers, a data set of different types of coupons – 6 columns, 37 rows*
- *testHistory, one shopper per row, used to test a predictive model – 5 columns, 151484 rows*
- *trainHistory, one shopper per row, used to train a predictive model – 7 columns, 160057 rows*
- *transactions, a detailed data set of shopper's transactions – 11 columns, 349655789 rows*

3.) The target variable for the contest was repeater. Remove repeattrips from the trainHistory set.

--

In the Acquire Valued Shopper Challenge data the trainHistory and testHistory data sets contain a large number of customers, one customer per row. This is the appropriate data format for predicting whether a shopper will make a repeat trip based on an offer they received. Each customer is uniquely identified

by the value of the `id` variable. In the contest, the labeled training data was used to build a model to predict whether each customer in the test data would be a repeat shopper based on the offer they received. The two extra columns in the training data not in the test data are the target variables used for training a supervised predictive model. There are two possible target variables in the training set:

repeater - a binary variable which signifies whether a shopper is a repeat shopper.
repeattrips - an ordinal variable name which indicates the number of times a shopper repeats.

Reference lines 195-199 of https://github.com/jphall663/GWU_data_mining/blob/master/01_basic_data_prep/src/raw/sas/SAS_Part_0_Base_SAS_PROC_SGPLOT.sas for more information on keeping a list of variables in a data set using SAS. It is more convenient to use the `drop=` statement in this case.

--

After dropping the repeattrips variable, the trainHistory set should now have 6 columns and 160057 rows.

4.) Join the offers set onto the trainHistory and testHistory sets.

--

Predicting repeat shoppers is difficult. As much extra data as possible from both the offers set and the transaction set should be added to the training and test sets. The type of offers a customer has received may influence their repeating behavior and their past transactions may give insight into their future transactions.

The training and test sets are in the appropriate format for making predictions on a per-shopper basis, i.e. one shopper per row. The offers and transactions set are not in an appropriate format for making predictions on a per-shopper basis, but information in each of these sets is helpful and should be joined onto the training and test data sets.

As a rule in predictive modeling, **any transformation that is applied to the training data must also be applied to the test data**. For part 4 of the assignment, join the information in the offers table onto the training data and onto the test data using a common variable in all three sets. After completing this join, you will be able to see the offers a customer has received for each row in the training and test sets.

Reference lines 301-311 of https://github.com/jphall663/GWU_data_mining/blob/master/01_basic_data_prep/src/raw/sas/SAS_Part_0_Base_SAS_PROC_SGPLOT.sas for more information of joining data sets using SAS.

--

The new training set should contain 11 columns and 160057 rows. The new test set should contain 10 columns and 151484 rows.

5.) Determine the maximum number of items and the maximum dollar amount a shopper has spent on items in the same category as the item for which they received an offer in the trainHistory or testHistory sets.

--

Another important type of information is customers past behavior. It is logical to conclude that a customer who has bought an item many times in the past may continue to buy that item, or a similar item, in the future. To include this information in the training and test data, you must summarize the transactions set. Group the transaction set by `id` and category and summarize the groups by taking the maximum of the purchasequantity and purchaseamount variables. Then left join the summarized set onto the trainHistory and testHistory sets by `id` and category.

This operation took about 5 minutes on my laptop.

Reference lines 124-135 of https://github.com/jphall663/GWU_data_mining/blob/master/01_basic_data_prep/src/raw/sas/SAS_Part_1_PROC_SQL.sas for more information on summarizing variables by group using SAS. It is a very simple syntax change to summarize by multiple groups.

After the join, the new training and test sets may contain missing values. Replace these missing values with 0, as missingness in this case simply indicates the customer has purchased 0 items, costing 0 dollars.

Reference lines 264-270 of https://github.com/jphall663/GWU_data_mining/blob/master/01_basic_data_prep/src/raw/sas/SAS_Part_0_Base_SAS_PROC_SGPLOT.sas for using SAS to sort data sets by more than one variable.

Reference lines 286-296 of https://github.com/jphall663/GWU_data_mining/blob/master/01_basic_data_prep/src/raw/sas/SAS_Part_0_Base_SAS_PROC_SGPLOT.sas for more information on using a SAS DATA step for merging data sets. It is a simple syntax change to join/merge a data sets on more than one variable.

Hint: It's easier to use a SAS DATA step to merge and update values simultaneously than using PROC SQL for the same tasks. The SAS syntax:

```
data new;
      merge left(in=in_left) right;
      by key;
      if x = . then x = 0;
      if in_left;
run;
```

creates a table called `new` by left joining the table `right` onto the table `left` by a variable named `key` while also setting a numeric variable named `x` to zero when it is missing. The `in=` option and the variable named `in_left` make this a left join instead of a standard SAS data step merge by tracking when a row of data comes in from `left` and only keeping those rows in the `new` set.

--

After the join operation, the new training set should contain 13 columns and 160057 rows. The new test set should now contain 12 columns and 151484 rows.

6.) Determine if the customer has previously purchased the exact item on offer.

Use information from the contest data description to engineer a binary indicator feature describing whether a customer has ever bought the exact item for which they received an offer and add this feature to the train and test sets. This variable should be 1 if the customer has bought the exact item before and 0 otherwise. This operation may require multiple steps.

--

After the join operation, the new training set should contain 14 columns and 160057 rows. The new test set should now contain 13 columns and 151484 rows.

7.) Create small subsets of the training and test data, export them to CSV, and turn them in.

Subset the augmented training data so that it contains only `id` numbers less than 14000000. Export this small file to CSV format.

Subset the augmented test data so that it contains only `id` values greater than 4810000000. Export this small file to CSV format.

Reference lines 322 -333 of https://github.com/jphall663/GWU_data_mining/blob/master/01_basic_data_prep/src/raw/sas/SAS_Part_0_Base_SAS_PROC_SGPLOT.sas for more information on exporting files from SAS. (You may also use File -> Export.)

--

Zip your code and your CSVs into a single archive file and submit that to blackboard with your group member's names.