GRN Coding and Modeling Exercise

WorldQuant Predictive

May 4, 2020

Contents

1 Problem description

XYZ Gaming Company makes a freemium gaming app called "QWERTY League" that generates revenue through in-app purchases of "AZERTY jewels". "AZERTY jewels" have a list price of $4 per jewel, but they occasionally go on sale. XYZ Gaming Company also regularly engages in public marketing events to promote the app in order to draw in new users.

XYZ Gaming Company would like you to create two models:

1. A revenue prediction model for their finance team that can help predict weekly revenue up to `3` months (i.e., one quarter) forward.

2. A classification model for their marketing team that can help identify high-value new users for targeted marketing campaigns.

2 Data

To facilitate model development, training data has been provided consisting of daily historical user activity and daily in-app purchase transaction data from the app launch on `20150403` through `20160702` as well as well the daily pricing schedule and marketing event schedule for `20150403` through `20161001`.

The following data are provided:

1. `activity.csv` which is the log of user account activity on the app for the `20150403` to `20160702` timeframe.

2. `marketing.csv` which is the list of major and minor marketing events in the `20150403` to `20161001` timeframe.

3. `pricing.csv` which is the daily schedule of prices for the in-app purchase for the `20150403` to `20161001` timeframe.

4. `transaction.csv` which is the log of user in-app purchases for the `20150403` to `20160702` timeframe.

Table 1: `activity.csv` contains the log of user activity for the app

Column	Description
`DATE`	Date in `YYYYMMDD` format
`USERID`	User ID (integer)
`ACTIVITY`	User activity category

Table 2: `marketing.csv` contains the schedule of marketing events for the app

Column	Description
`DATE`	Date of marketing event in `YYYYMMDD` format
`MARKETINGEVENT`	Description of marketing event

Table 3: `price.csv` contains the price schedule for the app

Column	Description
`DATE`	Date in `YYYYMMDD` format
`PRICE`	IAP unit price, in dollars
`PRICETYPE`	Price policy for the day

Table 4: `transaction.csv` contains the log of user in-app purchases

Column	Description
`DATE`	Date in `YYYYMMDD` format
`USERID`	User ID (integer)
`PRICE`	IAP unit price, in dollars
`UNITS`	Number of IAP units purchased for the day by the user
`TOTAL`	Total expenditure for the day by the user, in dollars

3 Instructions

In this exercise, your goal is to develop the two models mentioned above using the provided data, and to make some relevant forward predictions using those models.

3.1 Predicting weekly revenue

In this problem, the goal is to predict the weekly revenue of the app for the weeks in `problem-one-forecast-weeks.csv` (these are the weeks in the `20160703` to `201601001` timeframe).

Each line in `problem-one-forecast-weeks.csv` has two entries of the format `YYYYMMDD` that together representing a business week. The first entry is the first day in the week (a Sunday) and the second entry is the last day in the week (a Saturday). For example, the entry `20160703,20160709` represents the week Sunday `20160703` through Saturday `20160709`.

Your task is to generate a new CSV file `problem-one-answer.csv` containing rows with three values:

- The first value is week start date.
- The second value is the week end date.
- The third value is the predicted revenue for the week, in dollars.

where the (week start date, week end date) values are from the lines in `problem-one-forecast-weeks.csv`. *Please make sure that the file does* **not** *have a header line.*

A sample solution file, `problem-one-sample-answer.csv`, is provided for reference.

3.2 Predicting which new users will be high-value

In this problem, the goal is to classify whether each of the given users in `problem-two-new-users.csv` will be *high-value*.

Each line of `problem-two-new-users.csv` is a new user, defined as a user who created an account in the app on or after `20160605` and has logged in on at least `10` days in the `20160605` to `20160702` timeframe.

For the purpose of this exercise, a user is considered *high-value* if that user spends at least $100.0 in total between `20160703` and `20161001`.

Your task is to generate a new CSV file `problem-two-answer.csv` containing rows with two values:

- The first value is a user id from `problem-two-new-users.csv`.
- The second value is a `0/1` binary value where the value is `1` if your model predicts the user to be *high-value* (spend at least $100 in `20160703-20161001` timeframe) and `0` otherwise.

Please make sure that the file does **not** *have a header line.*

A sample solution file, `problem-two-sample-answer.csv`, is provided for reference.

3.3 Validating your solutions

Please make sure your solution files are valid, i.e.

1. Your solution file must be named correctly (i.e. `problem-one-answer.csv` for the first problem, and `problem-two-answer.csv` for the second problem).
2. The number of rows in your solution file must match the number of rows in the sample file.
3. The number of columns in your solution file must match the number of columns in the sample file.
4. The inputs (all but the last column) of your solution file must match that of the sample file. These have to match **exactly**.

5. Your predictions (the last column) must parse as a valid float for problem one, and must parse as a valid integer for problem two.

We have provided a Python 3.5+ script `validate.py` that can be used to check the above for your solution file. Make sure you are in the same directory as where `validate.py` and the sample files are located, and run the following commands in the terminal (replace `/path/to/your/` as appropriate):

```
$ python validate.py /path/to/your/problem-one-answer.csv 1
$ python validate.py /path/to/your/problem-two-answer.csv 2
```

The script will output if your files validated cleanly or if it found issues.

4 Deliverables

Please submit the following:

1. Your `problem-one-answer.csv` file. *Please make sure your solution file checks cleanly against the solution validator.*
2. Your `problem-two-answer.csv` file. *Please make sure your solution file checks cleanly against the solution validator.*
3. Your model code and supporting scripts (e.g. for generating the CSV files), for both problems.
4. A short description of your modeling approach for both problems.