Mario

CashFlow Management Regression Models

Project Details:

CashFlow Management using Regression Machine Learning Approaches.

1. Hypothesis

Building various regression models which would predict monthly cash flow for various business accounts for July 2019 based on data of previous few years using Regression Machine Learning Techniques.

2. The project aim(s)

Is to be as near to the ground truth values for “amout_usd” column for the July month of 2019.

3. Project plan

I have planned to work on the project by following the below-mentioned steps:-

- Requirement Gathering and Analysis
- Literature Survey of some of the existing state of art methods
- Select a state-of-art method and understanding it.
- Exploratory Data Analysis of training data (monthly and daily).
- Preprocessing and Cleaning data.
- Implementing Regression models.
- Comparison of results of proposed work and the existing state of art methods
- Conclusion

4. Approach followed:

The task can be divided majorly into 2 parts:

a. Exploratory Data Analysis of training data (monthly and daily)
b. Implementing Regression models

4a. Exploratory Data Analysis of training data (monthly and daily)

- Explored the data for understanding the traits of data flow.
    - We would be performing data analysis on data containing all monthly transactions to know the behaviour.
    -
- Here, I did Preprocessing and Cleaning of the training data (monthly and daily).
    - Remove noisy or unnecessary features/columns.
    - Here GL_account_description is not needed as it is represented by the GL_account feature.
    - Drop unnecessary columns.
    - aggregate data based on accounts.
    - Show the unique or distinct values in each column, from this we can make out that few rows are redundant and can be removed. For eg. company and company_name (we drop company_name)
    - Load detailed transaction data.
    - Remove NaN containing rows
    - count the number of missing elements (NaN) in each column.
    - remove the rows with missing elements if needed or drop the column (we choose to drop the column)
    - Mapping strings to integer values to be able to provide them to the training models.
    - See outliers in the data to eliminate them as they would hamper the analysis: Boxblot is used to get outliers if any.
    - I found a few outliers in amount usd, which can be removed. Removing outliers by taking top 99% quantile. (Top 99% data)
    - Saving this cleaned data for training.

4b. Implementing Regression models:

- Regression Models on Monthly aggregated Data:
  - Here I implemented 4 algorithms:
  1. Linear Regression
  2. Random Forest
  3. XGBoost
  4. LSTM

- Regression Models on Daily Transaction Data: (Aim is to predict every day transaction for July 2019 (201907) and then sum it to get the monthly prediction)
  - Here I implemented 2 algorithms which proved to give better results in the above monthly data:
  1. Random Forest
  2. XGBoost

- Common steps for Regression Models on Monthly aggregated Data:
  - Load Data

	account_id	partition_ledger_year_month	amount_usd
0	550385	201501	-390217.24
1	550385	201502	230944.09
2	550385	201503	367259.69
3	550385	201504	567962.85
4	550385	201505	753175.60

  -
  - I tried two ways after loading the data, 1. Normal string value for "partition_ledger_year_month" feature and 2. Datetime value for "partition_ledger_year_month" feature. But, the string value proved to give better results.

- Dropping two columns "GL_account" and "GL_account_description" as they are redundant and not required for the model to learn the amound_usd series.

- Described the data grouping them by account_id, which shows the statistic values of the feature "amount_usd":

account_id	count	mean	std	min	25%	50%	75%	max
550385	54.0	-7.019257e+06	1.161007e+07	-5.506018e+07	-7.659706e+06	-4.302384e+06	-4.127188e+05	5.159147e+06
550406	54.0	-7.849639e+07	9.242308e+07	-4.380654e+08	-1.425767e+08	-7.239205e+07	-7.645546e+05	8.774756e+07
661926	54.0	-3.624034e+06	3.491522e+07	-1.600823e+08	-2.238820e+07	4.256758e+06	1.637850e+07	4.627042e+07
693819	50.0	2.602236e+02	2.351533e+03	-1.374500e+03	-2.962825e+02	-8.944000e+01	2.277825e+02	1.631881e+04
751280	54.0	-2.156726e+07	1.055403e+08	-2.318708e+08	-8.325770e+07	-1.566982e+07	2.441177e+07	5.277015e+08
...	...	...	...	...	...	...	...	...
3169158	16.0	-8.641250e+00	3.456500e+01	-1.382600e+02	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
3274874	10.0	-5.062899e+04	1.657069e+05	-5.210531e+05	-5.374150e+03	-1.022400e+02	2.792755e+03	2.290221e+04
3283940	7.0	2.905001e+05	1.342318e+06	-8.994371e+05	-8.186797e+05	-1.206540e+05	1.135920e+06	2.419111e+06
3291608	7.0	2.819717e+03	4.820303e+03	0.000000e+00	0.000000e+00	0.000000e+00	4.749835e+03	1.023835e+04
3301816	4.0	8.559565e+04	2.846643e+05	-1.494705e+05	-4.347781e+04	-4.073460e+03	1.250000e+05	5.000000e+05

- Model training and prediction methods are implemented.
- Loading the submission file with our ouput ledger year month to be predicted for. (i.e. 201907)

	partition_ledger_year_month
0	201907
1	201907
2	201907
3	201907
4	201907

-
- Function for Linear Regression model with returning predicted values for "201907" partition_ledger_year_month. This function is called once for each account id as it is fired on groupby "account_id".
- Saving results to the submission csv.

	account_id	amount_usd
0	550385	-7.620617e+06
1	550406	-1.619195e+08
2	661926	2.715227e+06
3	693819	-5.260680e+02
4	751280	-1.033526e+08

-
-

- For RandomForest approach I tried 2 different models:
    1. **RF Model 1: n_estimators=100, max_depth=4 (n_estimators means no. of trees and max depth of each tree)**
    2. **RF Model 2: n_estimators=300, max_depth=30 (n_estimators means no. of trees and max depth of each tree)**

- For XGBoost approach I tried 2 different models:
    1. **XGBoost Model 1: n_estimators=100, learning_rate=0.2 (n_estimators means no. of trees and learning rate implies the rate at which the new values needs to be updated to the next training epoch/iteration)**
    2. **XGBoost Model 2: n_estimators=1000, learning_rate=0.3 (n_estimators means no. of trees and learning rate implies the rate at which the new values needs to be updated to the next training epoch/iteration)**

- The LSTM model didn't give proper results.

- Common steps for Regression Models on Daily Transaction Data:
    - Loading Common libraries and input data for both RF and XGBoost methods.
    - loading the cleaned training data from the EDA part:

```
(176718, 4)
```

	account_id	partition_ledger_year_month	date_general_ledger	amount_usd
0	751280	201807	20180717	4700449.77
1	751280	201808	20180830	5284250.81
2	751280	201510	20150910	3644851.76
3	751280	201902	20190213	4331.25
4	751280	201511	20151118	0.00

    - For RandomForest approach I tried 2 different models:
        1. **RF Model 1: n_estimators=10, max_depth=3 (n_estimators means no. of trees and max depth of each tree)**
        2. **RF Model 2: n_estimators=100, max_depth=4 (n_estimators means no. of trees and max depth of each tree)**

    - For XGBoost approach I tried 2 different models:
        1. **XGBoost Model 1: n_estimators=10, learning_rate=0.3 (n_estimators means no. of trees and learning rate implies the rate at which the new values needs to be updated to the next training epoch/iteration)**
        2. **XGBoost Model 2: n_estimators=100, learning_rate=0.2 (n_estimators means no. of trees and learning rate implies the rate at which the new values needs to be updated to the next training epoch/iteration)**

- Loading the submission file with our output ledger year month to be predicted for. (i.e. 201907) on a daily basis.
- Dropping columns ["account_id","amount_usd"] as we only require "partition_ledger_year_month" for X_test
- Implemented a function for Random Forest model with returning predicted values for "201907" partition_ledger_year_month. This function is called once for each account id as it is fired on groupby "account_id". Function to apply model function on each group and passing columns ['partition_ledger_year_month'], ['amount_usd'] as X and Y to fit the model
- The function to apply the model function on each group. Grouping the training data by "account_id" and "partition_ledger_year_month"
- Grouping daily predicted cashflow by sum of each account each month.
- Saving results to the submission csv.

	account_id	amount_usd
0	550385	-2.085322e+07
1	550406	-2.491265e+08
2	661926	2.825468e+08
3	693819	-1.520052e+04
4	751280	-3.111376e+07

• Conclusion:

- After submission of both the approaches for Linear Regression, we found that Approach 1 using string value for "partition_ledger_year_month" is better, so for further approaches we would be only using one approach that is Approach 1 string value.
- Random Forest proved to give best results out of all the implemented models.
- The LSTM model didn't give proper results.