INSTACART MARKET BASKET ANALYSIS

Pratik Parija (parija.p@husky.neu.edu)
Rishika Dawkar (dawkar.r@husky.neu.edu)
Shreyansh Singh (singh.s@husky.neu.edu)

INFO7390 Advances in Data Sciences and Architecture
Spring 2018 Northeastern University

1. Abstract

Market basket analysis [1]has been an elementary part of quantitative decision support in retail marketing for many years and it is regularly cited as a prime application area of data mining. Analytics[2] often involves studying past historical data to research potential trends, to analyze the effects of certain decisions or events, or to evaluate the performance of a given tool or scenario. This is basically what we did in this project, we analyzed historical data and predicted a certain outcome based on that data. In this paper, we will discuss various methods to perform predictive analytics and try to improve the accuracy of the prediction. Understanding of basic Python packages (numpy, panda, ski-kit learn etc.) and elementary statistics would be a bonus. We have implemented this project in Jupyter notebook in Anaconda.

2. Introduction

The project named 'Instacart Market Basket Analysis' [3]is about predicting the products which have been purchased previously and the user is most likely to buy same products in their next order. Instacart, a grocery ordering and delivery app, aims to make it easy to fill your refrigerator and pantry with your personal favorites and staples when you need them. After selecting products through the Instacart app, personal shoppers review your order and do the in-store shopping and delivery for you.

It is a Kaggle competition and the participants were supposed to predict the next purchase item. Apart from this, we have applied Word2Vec to cluster items as per some features.

Predictive Analytics is not new in this era of technological advancement. Using certain machine learning algorithms and Tensor Flows we have performed certain exploratory data analytics to figure out simple answers about the customer behavior.
A lot has been done on the kernels in Kaggle. We have tried to tweak and extend on some kernels to find out more from the dataset.

3. About the Data

There are 6 tables provided by Instacart as below:

- **Aisles.csv** – This contains the names of the aisles based on the products in them.
- **Departments.csv** – It has the names of department categorized by products types.
- **Order_Product__Prior.csv** – It has details of all the previous customer orders.
- **Order_Product__Train.csv** – This is the dataset which will be used to train the test dataset explained next.
- **Orders.csv** – It is the main table containing details about the customer orders and tells which record belongs to which table, train, prior or test.
- **Products.csv** – This contain detail of all the products sold by Instacart along with their ProductID.

4. Background and Research

While scrolling through a lot of project ideas, we stumbled across this Kaggle competition by Instacart, which is an online grocery and delivery app. All of us liked the idea of the competition and looked through almost every kernel available on Kaggle. Since EDA was the most popular operation, we performed it. Apart from EDA, we tried to apply Random Forests which did not prove to be fruitful because the decision trees did not provide any meaningful output. The model was getting over-fitted and this is not what was desired.

[1] https://link.springer.com/content/pdf/10.1057/palgrave.jt.5740092.pdf
[2] http://www.businessdictionary.com/definition/analytics.html
[3] https://www.kaggle.com/c/instacart-market-basket-analysis

Next, we also tried to perform Logistic Regression, but this was also in vain since there was not Boolean type data in the dataset. Also, there was no meaningful correlation between the columns, so we could not find out anything useful out of it.

While performing EDA, we were able to figure out a lot of useful information which usually would have been not in plain sight. We were able to find out the busiest days and hours of the week, most bought products etc.

We tried to figure out some important features from each table and combine them together to perform some analysis and figure out more information about the data. We further tried to develop more problem statements and find out answers by analysis the data.

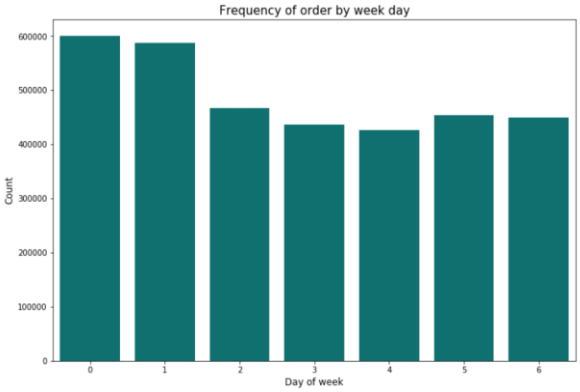
After that we tried to implement XG-Boost and LGBM followed by Logistic Regression. We figured out that LGBM was the most accurate with an accuracy of 90.46%.

5. Data Analysis

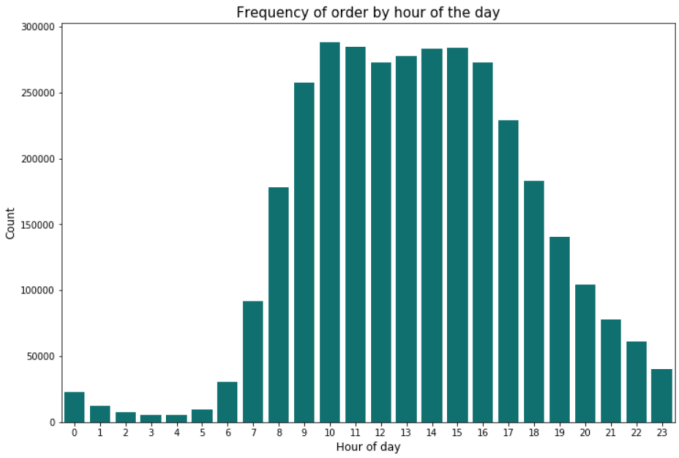
Exploratory Data Analysis:

As per our findings after performing exploratory data analysis, we figured out a lot of things about the data.

For example, we were able to know that weekend is the busiest part of the week. People tend to order during the weekends. Maybe because it is difficult to order during the weekdays since there is a lot of work.



To figure this out we used the orders.csv file which contains the data about the order id and the user id, i.e. related to the customer. Also, there is data about day of the week and, also hour of the day on which an order was placed. We were also able to figure out that the afternoon was the busiest time of the day. As we can see, as per the graph below the time of 10:00 A.M to 3:00 P.M witnessed the maximum number of days. So, using both the graphs, we can conclude that afternoon hours during the weekend would be busiest.



XG-Boost

The accuracy using this came to around 90.43%

```
In [32]: X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.33, random_state=7)

In [33]: model = XGBClassifier()

         Step 11: Train the model using XGBClassifier

In [34]: model.fit(X_train, y_train)

Out[34]: XGBClassifier(base_score=0.5, colsample_bylevel=1, colsample_bytree=1,
                       gamma=0, learning_rate=0.1, max_delta_step=0, max_depth=3,
                       min_child_weight=1, missing=None, n_estimators=100, nthread=-1,
                       objective='binary:logistic', reg_alpha=0, reg_lambda=1,
                       scale_pos_weight=1, seed=0, silent=True, subsample=1)

In [35]: y_pred = model.predict(X_test)

In [36]: predictions = [round(value) for value in y_pred]

In [37]: accuracy = accuracy_score(y_test, predictions)

         Accuracy for the XGBoost Classifier

In [38]: print("Accuracy: %.2f%%" % (accuracy * 100.0))

         Accuracy: 90.43%

In [39]: # target variable for train set #
         train_y = train_df.reordered.values
```

LGBM

There was slight increase in the accuracy with 90.46% with LGBM.

```
         Light GBM

In [56]: import lightgbm as lgb

In [57]: train_X = lgb.Dataset(X_train)
         train_y = lgb.Dataset(y_train)
         test_X = lgb.Dataset(X_test)
         test_y = lgb.Dataset(y_test)

In [58]: model= lgb.LGBMClassifier()

In [59]: model.fit(X_train, y_train)

Out[59]: LGBMClassifier(boosting_type='gbdt', class_weight=None, colsample_bytree=1.0,
                        learning_rate=0.1, max_depth=-1, min_child_samples=20,
                        min_child_weight=0.001, min_split_gain=0.0, n_estimators=100,
                        n_jobs=-1, num_leaves=31, objective=None, random_state=None,
                        reg_alpha=0.0, reg_lambda=0.0, silent=True, subsample=1.0,
                        subsample_for_bin=200000, subsample_freq=1)

In [60]: pred=model.predict(X_test)

In [61]: #accuracy score of Light GBM
         print(accuracy_score(pred, y_test))

         0.90463624372
```

Logistic Regression

The accuracy with this was around 90.16%.

Logistic Regression

```
In [45]:
from sklearn.linear_model import LogisticRegression

In [46]:
#Logistic Regression model
clf=(LogisticRegression(C=0.02))

In [47]:
#fitting the model
clf.fit(X_train, y_train)

Out[47]:
LogisticRegression(C=0.02, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)

In [48]:
#predictions
pred=clf.predict(X_test)

In [49]:
#accuracy score of Logistic Regression Model
print(accuracy_score(pred, y_test))

0.901983416522
```

The rest of the code can be found at the link below:

https://github.com/pratikparija93/Advances-In-Data-Science

6. Acknowledgement

We would like to show our gratitude to professor Nik Bear Brown for guiding us and encouraging us during this project.

7. References

[1] https://www.kaggle.com/c/instacart-market-basket-analysis/kernels

[2] https://towardsdatascience.com/a-gentle-introduction-on-market-basket-analysis-association-rules-fa4b986a40ce

[3] https://en.wikipedia.org/wiki/Xgboost

[4] https://en.wikipedia.org/wiki/Random_forest