Machine Learning Engineer Nanodegree

Capstone Project

Name of Project: Black Friday Sales Prediction

Sreyashi Dutta

March 23rd, 2019

Reference: https://datahack.analyticsvidhya.com/contest/black-friday/

Definition

Project Overview

Domain: Retail

Retail industry is one of the blooming industries where Machine Learning is heavily used these days. It is also a highly revenue generating industry and hence companies are investing heavily on machine learning to increase the sales of their products.

Black Friday is celebrated a day after the Thanksgiving in USA. Typically, retail companies and small shops offer black Friday sales to their customers and it has been observed as highly popular amongst the customers which in turn brings in a lot of revenue for the companies. As such, companies are now using machine learning algorithms to come up with recommendations of products for the customer in order to increase the sales.

Problem Statement

A retail company “ABC Private Limited” wants to understand the customer purchase behavior (specifically, purchase amount) against various products of different categories. They have shared purchase summary of various customers for selected high volume products from last month.
The data set also contains customer demographics (age, gender, marital status, city_type, stay_in_current_city), product details (product_id and product category) and Total purchase_amount from last month.

Now, they want to build a model to predict the purchase amount of customer against various products which will help them to create personalized offer for customers against different products.

This project is available on AnalyticsVidhya.com

Metrics

Each machine learning model is trying to solve a problem with a different objective using a different dataset and hence, it is important to understand the context before choosing a metric. Usually, the answers to the following question help us choose the appropriate metric:

Type of task: Regression or Classification

Business goal?

What is the distribution of the target variable?

Since our target is to predict the purchase amount of customer against various products, it qualifies for Regression task.

We have a number of methods in Regression technique which can be used as a measure to evaluate our model such as:

Mean Squared Error (MSE)

Root Mean Squared Error (RMSE)

Mean Absolute Error (MAE)

R Squared (R^2)

Adjusted R Squared (R^2)

Mean Square Percentage Error (MSPE)

Mean Absolute Percentage Error (MAPE)

Root Mean Squared Logarithmic Error (RMSLE)

RMSE is usually the default metric of many models because loss function defined in terms of RMSE is smoothly differentiable and makes it easier to perform mathematical operations. Compared to Mean Absolute Error, RMSE punishes larger errors.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Where y hat is the predicted value and y is the original value.

Analysis

Data Exploration

Datasets are provided in the form of train and test sets in CSV files.

The variables and their definition in the dataset are mentioned below:

Variable	Definition
User_ID	User ID

Product_ID	Product ID
Gender	Sex of User
Age	Age in bins
Occupation	Occupation (Masked)
City_Category	Category of the City (A,B,C)
Stay_In_Current_City_Years	Number of years stay in current city
Marital_Status	Marital Status
Product_Category_1	Product Category (Masked)
Product_Category_2	Product may belong to other category also (Masked)
Product_Category_3	Product may belong to other category also (Masked)
Purchase	Purchase Amount (Target Variable)

The dataset consists of

Dataset	Total	Variable count
Train	550068	12
Test	233599	11

We need to mark User_ID against Product_ID and predict the purchase amount for the data in test set and provide it in a CSV as a submission file.

Some of the variables provided are interesting.

Age which is usually a continuous variable here has been provided as a categorized variable – putting it in different brackets: '0-17','18-25','26-35','36-45','46-50','51-55','55+'

Occupation has been also provided as a masked variable. Meaning the actual occupation is not provided, rather the occupation has been mapped to integers and marked against each user, thereby making it not only categorized variable but also numerical which will be easy for our model to compute.

City_Category again has been masked and categorized as A,B,C.

Stay_In_Current_City_Years is a categorized variable with categories – 0,1,2,3,4+

Marital Status has been provided in the form of numerical values and categorized as binary values i.e 0 and 1.

There are three Product_Category variables which are provided as numerical values, meaning product categories in this case have been mapped to numbers thereby reducing the pain of handling textual data.

Each Product_ID (product basically) can be part of multiple product categories and accordingly has been mapped in the datasets.

Last but not the least is the Purchase variable in the train set which is continuous variable giving us the amount spent by customers against product purchases. This is the target variable and hence is missing from the test set.

After observing the variable, we proceed on to check the list of NAs in our variables. Following capture depicts the list:

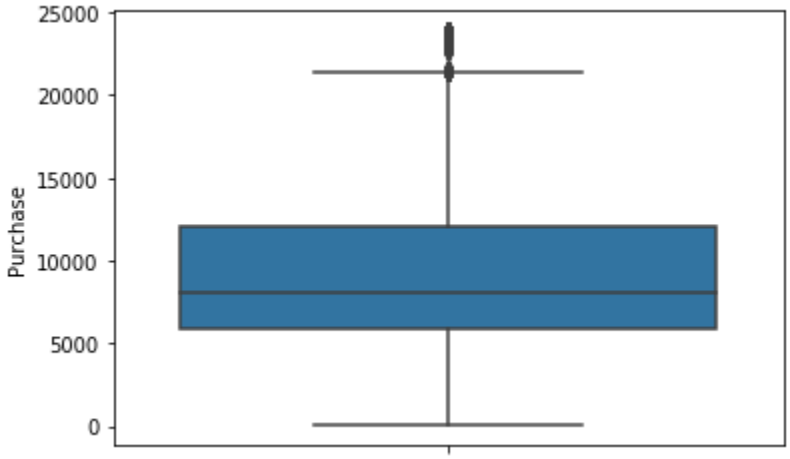
```
NA Values - Train DS
User_ID : 0
Product_ID : 0
Gender : 0
Age : 0
Occupation : 0
City_Category : 0
Stay_In_Current_City_Years : 0
Marital_Status : 0
Product_Category_1 : 0
Product_Category_2 : 173638
Product_Category_3 : 383247
Purchase : 0

NA Values - Test DS
User_ID : 0
Product_ID : 0
Gender : 0
Age : 0
Occupation : 0
City_Category : 0
Stay_In_Current_City_Years : 0
Marital_Status : 0
Product_Category_1 : 0
Product_Category_2 : 72344
Product_Category_3 : 162562
```

Next, we make an observation of the unique values in the train set. Following capture shows the list:

```
Unique values count
User_ID : 5891
Product_ID : 3631
Gender : 2
Age : 7
Occupation : 21
Stay_In_Current_City_Years : 5
City_Category : 3
Marital_Status : 2
Product_Category_1 : 20
Product_Category_2 : 18
Product_Category_3 : 16
```

Visualization of Purchase variable:

The Purchase variable from the train set is stored in a separate dataframe and further dropped from train set. Through boxplot, we visualize the behavior



As we can see from the plot there are some outliers. It is important that we remove the outliers since they have a tendency to adversely impact the output.

Algorithm and Techniques:

The algorithm chosen is XGBoost in this case.

A brief introduction and how do I think it will fit this use case:

XGBoost stands for eXtreme Gradient Boosting. It has gained immense popularity these days among Data Scientists.

The XGBoost algorithm is mainly used for two reasons – Computational Speed and Model performance.

The XGBoost library implements the gradient boosting decision tree algorithm.

This algorithm goes by lots of different names such as gradient boosting, multiple additive regression trees, stochastic gradient boosting or gradient boosting machines.

Boosting is an ensemble technique where new models are added to correct the errors made by existing models. Models are added sequentially until no further improvements can be made. Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

XGBoost hence can be said produces a model which has lesser errors compared to other techniques be it linear regression or any of the Classification methods such as decision trees, random forest etc. and as such I expect this technique to give optimum result with minimized error compared to other techniques.

Methodology

Data Pre-processing

The test set is then appended with the train set dataframe

Now we had observed previously that the data (train & test) had quite a few NAs i.e missing values. So we replace them with 0 so that it is easier for our algorithm to interpret the data.

Next we import the LabelEncoder library from scikit-learn. This is done to convert text or categorical data to numerical for our model to interpret it better.

Label Encoding: It assigns each unique category in a categorical variable with an integer. No new columns are created. The integer assignment is arbitrary.

We use the fit_transform() to transform the categorical variable Gender (M/F) to integer 1/0.

The age variable is converted into an integer set by introducing a dictionary with keys as the age group and values as a numerical value mapped to each age-group.

{'0-17':0,'18-25':1,'26-35':2,'36-45':3,'46-50':4,'51-55':5,'55+':6}

We also observed Stay_In_Current_City_Years contains values such as 0,1,2,3,4+. So we replace the value 4+ with 4 so that this variable is uniform numerically.

The next variable City_Category has three values A,B and C. We use the get_dummies method in pandas library to convert the respective categories into individual categorical variables with values as 1s and 0s mapped accordingly.

Next step was to convert the variables - Stay_In_Current_City_Years, Product_Category_2, Product_Category_3 to integer type to have consistency and also model would perform better.

Now for simplification we do the following:

Take all the train-set count of data from the dataframe and store it in a dataframe X.

Take all train-set purchase amount data and store it in Y

Take all test-set count of data from the dataframe and store it in X_test

Take all test-set purchase amount data and store it in Y_test which is nothing but empty for now and we need to predict the purchase amount for given X_test.

Using LabelEncoder we transform the User_ID variable in both X and X_test for simplification purpose.

It is also done for Product_ID variable in both X and X_test.

However, during data analysis, it was found that there is a difference of 46 Product_IDs between train and test sets. This also means that there is no Purchase amount captured for these Product_IDs in the train set.

While transforming the Product_ID variable in test set, the ones which are present in the difference list (i.e Product_IDs not present in train but present in test set) have been assigned a value of -1.

We use a function loc available in Pandas library while transforming Product_ID variable in X_test.

pandas.DataFrame.loc : It access a group of rows or columns by labels

Implementation

Once all the above mentioned pre-processing techniques are completed, it is time for us to build a model using XGBoost algorithm.

The model is built using XGBRegressor and fine-tuned the parameters :

n_estimators (*int*) – Number of trees to fit.

max_depth (*int*) – Maximum tree depth for base learners

learning_rate (*float*) – Boosting learning rate

n_jobs (*int*) – Number of parallel threads used to run xgboost.

verbose (*int*) – The degree of verbosity. Default is 1 which indicates warning messages only.

The model is then fit to our train data – X,Y.

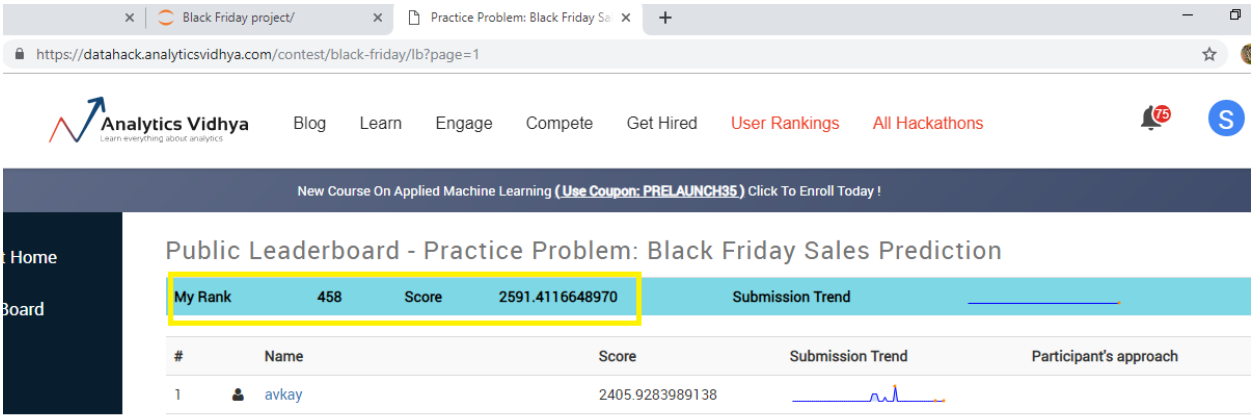
Once that is done, we predict this model on test data. So Purchase amount is predicted against each X_test entry.

With these we build a dataframe, which consists of the predicted Purchase amount and the respective original User_ID and Product_ID (two of which were previously stored in a dataframe called submission)

This dataframe is then stored in a CSV file which is used for submitting our predictions.

Result and Conclusion

Upon submitting it in the AnalyticsVidhya forum, the below score is achieved.



The score is not bad, however there is a scope of improvement. In my opinion, the Product_IDs which are missing in train set has been significant in the resulting score. Also, by fine-tuning parameters further, there could be a slight improvement (although I tried to lower the learning rate and result was a weaker model).