Machine Learning based Product Demand Analysis with Purchase Prediction for Black Friday Sales

Sanskar Tewatia[1], Ankit Singh Patel[2], Manmeet Singh[3,4] and Sukhpal Singh Gill[2]

[1]Electrical Engineering Department, Shiv Nadar University, Greater Noida, India
[2]School of Electronic Engineering and Computer Science, Queen Mary University of London, UK
[3]Centre for Climate Change Research, Indian Institute of Tropical Meteorology (IITM), Pune, India
[4]Interdisciplinary Programme (IDP) in Climate Studies, Indian Institute of Technology (IIT), Bombay, India
tsanskar@gmail.com, ankit.a.patel@hotmail.com, manmeet.cat@tropmet.res.in, s.s.gill@qmul.ac.uk

Abstract – In the United States, every year the fourth Thursday of November is celebrated as the Thanksgiving Day. The next day which is a Friday is known as the Black Friday. This day is the busiest day in terms of shopping because all major retailers and ecommerce websites offer massive amounts of discounts and deals. Hence, this sale is termed as the Black Friday Sale. There is a lot of potential to make profit even after such discounts, if the sales patterns from previous years' data are analysed properly. Looking at various demographics of customers and analysing the Purchase Amount spent by each Customer on various products, there is a need to find out patterns for this behaviour. Therefore, we utilized classical and modern machine learning techniques such as Linear Regression, Neural Networks, Gradient Boosting Trees and AutoML, to make predictions on the available test data to find out the best model for accurate predictions. Since the dataset contains information about various demographics and background of the customers, we encoded the data in an easy-to-understand format for each algorithm. Furthermore, various techniques of feature engineering are used, and new features are generated from existing features by grouping the target variable Purchase, for each sub-category of that feature. Erroneous predictions are also handled and ultimately the model performed well on unseen test data. Finally, this study can help researchers to find the best model for more accurate predictions.

Impact Statement **–** In this word of online shopping, ecommerce websites have a lot of information about their customers which the offline retail stores might not have. Looking at these demographics and history of sales of products by customers, it is possible to analyse this data and apply machine learning techniques so as to predict future sales, so that the retailer can stock their products and price them appropriately. In this paper, data from Black Friday Sales of 2016 is thoroughly analysed and numerous feature engineering techniques are used to give this data to machine learning models, which can make predictions on future sales. These techniques can be used for a wide variety of problems by the private sector and the choice of model provides deep insights into the robustness and speed of modern-day algorithms used by researchers all-round the globe.

Keywords - Sales Analysis, Machine Learning, Gradient Boosting Trees, Deep Learning, Artificial Intelligence.

I. INTRODUCTION

Earlier, the sales used to start on Thanksgiving Day (Thursday), but over time, more and more retailers began starting their sales at a later time of the day. Nowadays, the sale has been termed as Black Friday Sales because all sellers start the sales on Friday only [1] [2].

In our paper, first we meticulously analyse the dataset and observe the trends of purchase amount depending upon the available demographics of customers and the product categories. We then present various data pre-processing techniques that can be utilized for other datasets as well. These techniques immensely help in improving the accuracy of machine learning models. One more technique utilized was feature augmentation. Since only 11 features were available to us, new features were generated from these 11 features so as to extract valuable information about each customer, product, and the product categories [4]. Such feature generation techniques can be employed in other machine learning where very limited data is available [5].

Hence, after careful analysis of customer and product data, machine learning models were used to predict the purchase amount for old or new customers and/or Product. It was observed that the best results were obtained from the extensions of Gradient Boosting Trees – CatBoost and XGBoost [6] [7]. To further improve the score, weighted ensembles can be taken from our own previously generated predictions.

The main outcome from the prediction of such a problem is that the retailers can now stock the products appropriately and provide deals accordingly so as to maximise the profit earned. Hence, this solution is not a specific solution for Black Friday sales but can also be generally used for purchase prediction or demand analysis problems as well as other market related problems.

1.1 Motivation and Our Contributions

The fact that retail sales are one of the most crucial fields for research and exploitation for the domain of data analytics is common knowledge. Black Friday Sale is one of the most impactful sales on a global level. There are multiple preconceived notions in the minds of the general public, such as women being more interested when it comes to shopping. There can be many such notions, but how do such claims compare to the sales data gathered from retailers or ecommerce websites? Which city from a particular state spends the most on the sales? What is the age group of people that spends the most? What effect does marriage have on shopping? How do these factors help in prediction of sales amount for an upcoming sale? What can the retailers do so as to maximize their profits? How can they make predictions on sale and stock their shops accordingly? Data analysis on the available data and machine learning techniques can be utilized to find the solutions to such problems [4].

The ***main contributions*** for this research work are:

1. Black Friday sales and product demand analysis along with purchase prediction enabling us to forecast the demand of products that are required in the sales.
2. To identify best machine learning model for Black Friday sale prediction by comparing the Root Mean Square Error (RMSE) .

1.2 Article Organisation

This paper is organized as follows: Section 2 presents the related work. Section 3 presents the background of the problem. Section 4 explains the methodology. Section 5 presents results and analysis. Section 6 concludes the paper and presents the future directions.

II. RELATED WORK

This problem has been available on Analytics Vidhya since July 2016. Trung et al.[3] have discussed about the implementation of bagging and boosting algorithms for this problem. Ching et al.[8] applied various regression-based algorithms namely – Linear, Logistic, Polynomial, Stepwise, Ridge, Lasso & Elasticnet Regression; along with MLK Classifiers, Decision Trees along with Deep Learning using Keras for this problem. Ching et al.[8] concluded that models like neural networks were too complex for this kind of a problem. Kalra et al.[9] studied the effects of modifying the ratio of training and testing data so as to find the model with the best score on a common metric – Root Mean Squared Error. The results from this experimentation helped them identify models which were overfitting and those which were underfitting.

2.1 Critical Analysis

Table 1 shows the critical analysis s carried out as compared to related work. A new set of features were created and added pertaining to unique entries of User_ID, Product_ID or Product_Category. A specific set of data encoding techniques were used for each of the categorical variables in the dataset. Finally, a common technique of handling erroneous predictions was utilized for all the models. All these techniques helped improve the performance of the models.

Table 1 - Comparison with Related Work.

Components of Experimentation		[3]	[8]	[9]	Our Study
Pre-processing	Outlier Removal				✔
	OneHot Encoding				✔
Features	Product Category	✔	✔	✔	✔
	Marital Status	✔	✔	✔	✔
	City Category	✔	✔	✔	✔
	Occupation	✔	✔	✔	✔
	Age	✔	✔	✔	✔
	Purchasing Power				✔
	Count Variables				✔
	Min, Max & Mean Variables				✔
	25, 50 & 75 Percentile Values				✔
Handling Negative Predictions					✔

III. BACKGROUND

This problem is listed in the form of a global contest by Analytics Vidya, where anyone can participate and check their standings with respect to people from all over the world. The dataset is also provided by Analytics Vidhya for the purpose of this contest [4].

When it comes to sales prediction, many factors such as age, gender, marital status, occupation, type of product, city, etc can immensely impact the purchase amount. In order to make personalized predictions for each customer, these demographics were given in the form of categorical data in this dataset. Using data pre-processing and machine learning techniques, the contest required the participants to submit their predictions on a test set on their contest page.

The evaluation metric for this paper and the public ranking used was Root Mean Squared Error (RMSE). RMSE is an extremely popular metric when training machine learning models and deep neural networks. RMSE is essentially the standard deviation of the prediction errors. Prediction errors are the difference between theoretical values and predicted/experimental values. RMSE is obtained by squaring the prediction errors, taking their mean, and finally taking their squared root as defined in Eq. (1) (Notation from Barnston, 1992).

$$RMSE_{fo} = \left[\sum_{i=1}^{N} (Z_{fi} - Z_{oi})^2 / N\right]^{1/2} \qquad (1)$$

- $\sum$ - Summation for all data entries
- $(Z_{fi} - Z_{oi})^2$ - Square of Prediction Error
- N – Sample Size

For this contest, 30% of their test data has been posted for the general public in order to get the current global ranking. At the end of the contest, ranking will be based on the RMSE score obtained on the remaining 70% data, which is at present, not public. As of the date of writing this paper, 22,693 people have taken part in this contest by uploading their submissions on this available 30% test data.

IV. METHODOLOGY

In this section the dataset is described, and various techniques are applied for feature engineering and data augmentation. After pre-processing, this data is fed to various Machine Learning Models and then these algorithms are ranked depending on their Root Mean Squared Error (RMSE).

4.1 Dataset

The "ABC Private Limited" is a retail company that wants to understand customer purchase pattern (Purchase amount) regarding various products of different categories. They have decided to share the purchase summary of various customers for a few categories of high-volume products from the previous month. This summary reveals various customer demographics. The ultimate aim is to apply machine learning models for prediction of purchases for the upcoming Sale, so that the company can stock the products appropriately and consequently provide adequate discounts for a better profit margin [4].

For the purpose of this global contest, this dataset was provided by Analytics Vidhya. The dataset consists of 11 features (Table 2), a mix of categorical and numerical features and one target variable – *Purchase*. There are 550068 rows of information.

Table 2 – Dataset Details

Variable	Definition
User_ID	User ID
Product_ID	Product ID
Gender	Sex of user
Age	Age Group
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 another category also (Masked)
Product_Category_3	Product may belong to another category also (Masked)
Purchase	Purchase Amount (Target Variable)

4.2 Data Exploration

First, we find out the total number of unique values of each feature. This will tell us how discrete each feature is, because same *User_ID* can buy multiple products and also the fact that the same *Product_ID* can be bought by multiple users.

```
User_ID                      5891
Product_ID                   3631
Gender                          2
Age                             7
Occupation                     21
City_Category                   3
Stay_In_Current_City_Years      5
Marital_Status                  2
Product_Category_1             20
Product_Category_2             17
Product_Category_3             15
Purchase                    18105
```

Figure 1 - Number of Unique Entries across each feature.

Fig. 1 gives us information about the number of *User_IDs & Product_IDs* that are repeated. We can then construct data from this existing dataset based on buying power per *User_ID* or mean, maximum, minimum expenditure for each unique *User_ID*, etc.

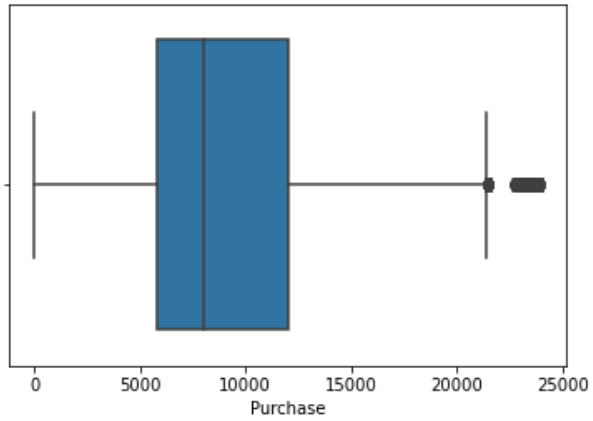

Figure 2 - Boxplot of the target variable – Purchase.

After looking at the boxplot of the target variable in Fig. 2 i.e., the Purchase Column, we observe that most of the data is concentrated around 8000 units, whereas there are very few entries where the value of these Purchases is more than 22000 units. Hence, these rows need to be taken care of, during data pre-processing.

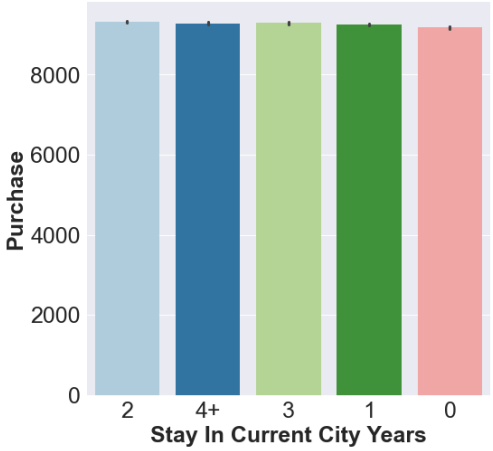

Figure 3 - Average Purchase across Native and Immigrant Residents.

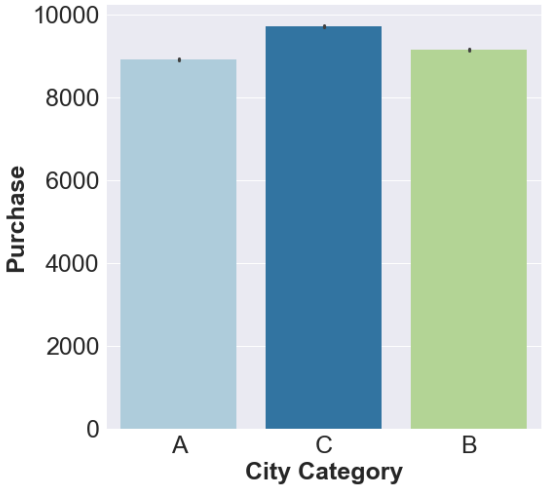

Figure 4 - Average Purchase across each City.

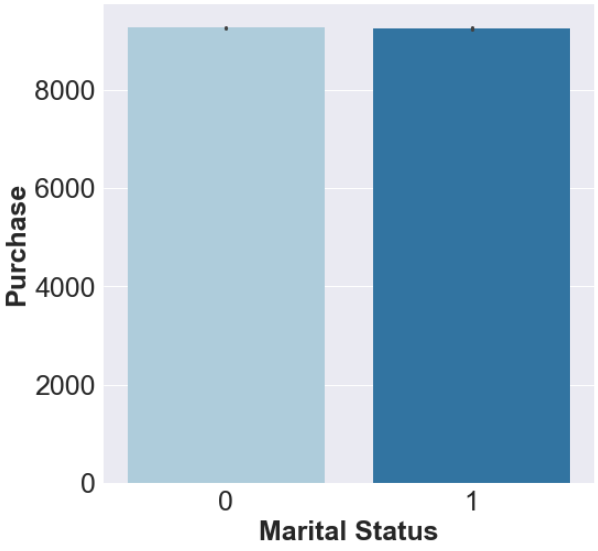

Figure 5 - Average Purchase by Married and Unmarried Customers.

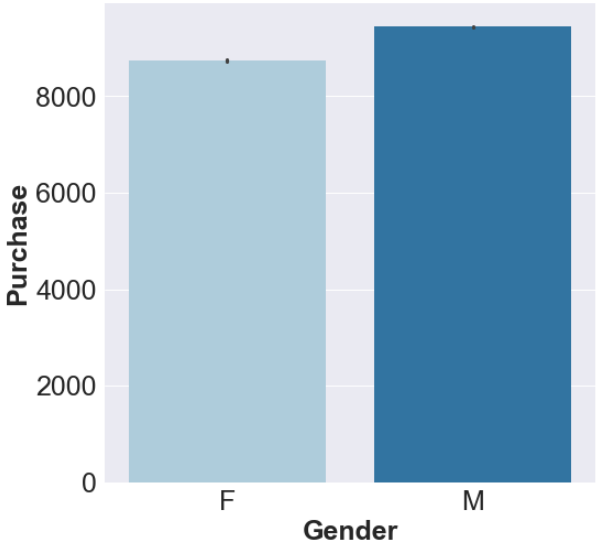

Figure 6 - Average Purchase by each Gender.

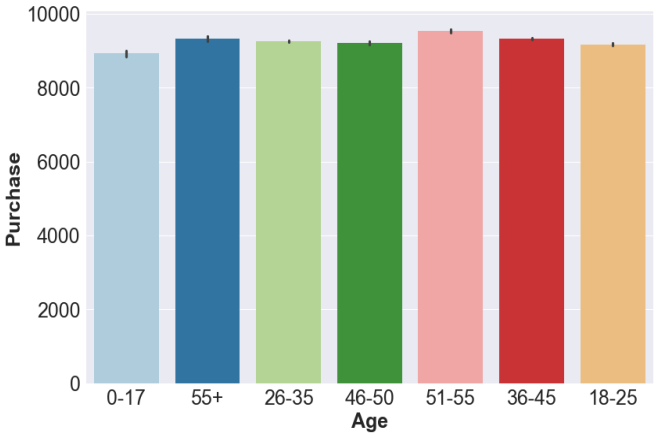

Figure 7 - Average Purchase across each age group.

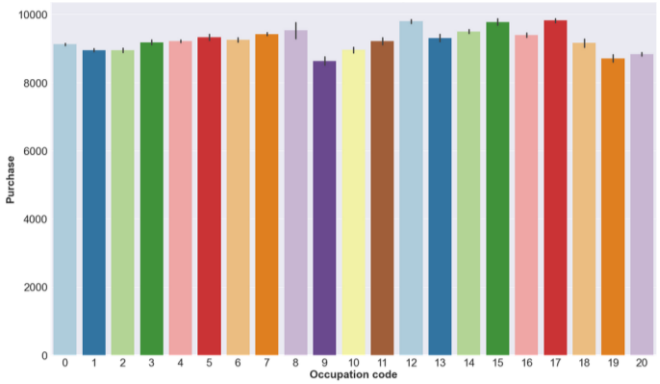

Figure 8 - Average Purchase by each Occupation.

It is evident from these plots in Fig 3 - Fig 8 that there is no significant difference between the value of mean *Purchase* across various different categories of features. Therefore, it can be inferred that on average, people spend very similar amounts of money irrespective of their background, hence the Purchase Column can be considered to be uniformly distributed.

4.3 Data-Preprocessing

In this section, various techniques of data pre-processing are applied to the dataset so that it is converted into numeric format.

4.3.1 Null Values

Since the customers can purchase products from any category they wish, there is a need to fill these rows with some data. If we drop all such rows, we will lose most of our training data and will be unable to build unbiased models.

User_ID	0.000000
Product_ID	0.000000
Gender	0.000000
Age	0.000000
Occupation	0.000000
City_Category	0.000000
Stay_In_Current_City_Years	0.000000
Marital_Status	0.000000
Product_Category_1	0.000000
Product_Category_2	31.566643
Product_Category_3	69.672659
Purchase	0.000000

Figure 9 - Percent of total training data that is missing.

In order to fill the missing data (Fig 9), two approaches were tried - filling these rows with Zero and filling them with -999. Two separate models were trained using the same linear regression technique. The evaluation metric used for comparison was Root Mean Squared Error on the Test Data. It was evident that filling the missing rows with -999 showed a slightly better score compared to the score obtained when filling them with zero (Table 2).

Table 3 - Filling Null Values.

Approach	Root Mean Squared Error
Filling with 0	4623.9775
Filling with -999	4610.7056

4.3.2 Removing Outliers

In machine learning, outliers are extreme values that are outside the range of expected observations, not akin to the remaining data. Hence, it is important to ensure the absence of outliers so as to improve model fitting and further help towards prediction of more accurate values. Printing boxplots of various percentiles of data (Fig 10, Fig 11 & Fig 12), it was found that taking only 99.5 percentile of the train data helped get rid of outliers and the new dataset consisted of rows where the value of *Purchase* remained in this range.

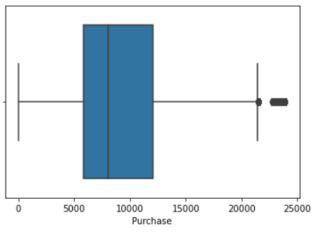

Figure 10 - 100 Percentile.

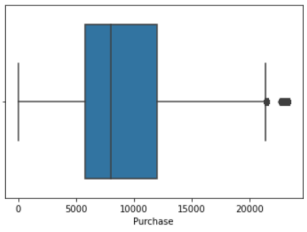

Figure 11 - 99.75 Percentile.

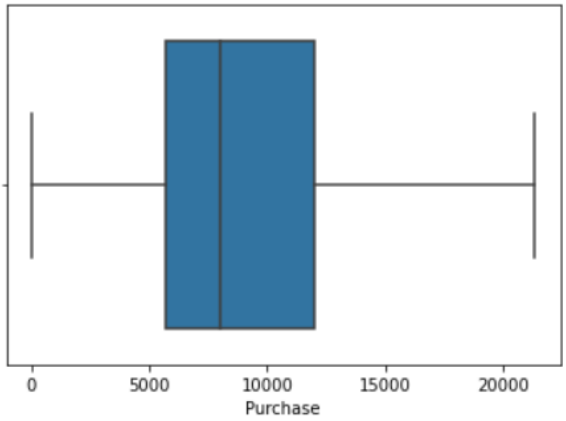

Figure 12 - 99.5 Percentile.

4.3.3 Feature Generation

Since there were only 11 features for the target variable, new features need to be generated from these existing features so as to improve model performance.

It was evident from Fig. 1 that multiple User_IDs were repeating, and so was the case with Product_ID. One feature that was created is called "Purchasing_Power". Grouping the *Purchase* column values by User_IDs and taking sum for every User_ID, a Pandas Series was created with this information. Taking percentiles (in multiples of 10) of the total sum of this series, a number from 1 to 10 was assigned in a dictionary, to each unique *User_ID*, based on the comparison of each *User_ID's* expenditure and this percentile of the total sum. Then for each *User_ID* in the train and test set, this dictionary was mapped to a new feature named *Purchasing_Power*. If a new *User_ID* comes up, absent in the dictionary, a mean value of 5 will be assigned as the Purchasing_Power to that new *User_ID*.

Count Variables were generated for *Age, Occupation, User_ID, Product_ID, Product_Category_1 and Product_Catgory_2, Product_Category_3*. These features would help identify repeating customers, from a particular age group, occupation, or their unique *User_IDs* or the *Product_IDs* that were bought multiple times.

Furthermore, 5 more features were generated for each of the following features – *User_ID, Product_ID, Product_Category_1, Product_Category_2 and Product_Category_3 (Table 4)*. This is done by grouping the *Purchase* Column by these 5 features and operating on them. All of these new features were of numerical type. Hence, separate encoding was not required.

Table 4 - New features created from existing features.

MinPrice	Minimum value of *Purchase* column for various categories of that feature
MaxPrice	Maximum value of *Purchase* column for various categories of that feature
MeanPrice	Mean value of *Purchase* column for various categories of that feature
25_Percentile	25 Percentile value of *Purchase* Column for various categories of that feature
50_Percentile	50 Percentile value of *Purchase* Column for various categories of that feature
75_Percentile	75 Percentile value of *Purchase* Column for various categories of that feature

4.3.4 Data Encoding

Categorical Variables are handled differently by every algorithm. In total, 6 approaches (Table 5) were tried and in two of them, it was possible to specify the categorical variables to the algorithm beforehand [10]. In others, the categorical variables were converted to numerical format using various kinds of encoding.

Table 5 - Type of input for each algorithm.

Linear Regression	Only Numerical input
Neural Networks	Only Numerical input
XGBoost	Only Numerical input
Catboost	Both Numerical & String input
AutoML	Only Numerical input
Ensemble Model	Both Numerical & String input

The columns titled *User_ID* & *Product_ID* might contain crucial information about customers and specific products hence dropping these columns would have negatively impacted the performance and robustness of the models. Hence, these columns were Label-Encoded to convert them to *int* format. In *Label-Encoding*, each label is assigned a distinctive integer value based on alphabetical ordering. Hence, for each model, 5891 unique integer values were assigned for each *User_ID*, and 3631 unique integer values for *Product_ID*.

One-Hot Encoding creates additional features based on various categories of values in that specific feature. Every unique value in the category will be added as a new feature column. For example, in our dataset, the feature *City_Category* is One-Hot Encoded for all approaches. For different approaches, the categorical features were converted to numerical form either using *One-Hot Encoding* or *Label Encoding*.

4.3.5 Handling Erroneous Predictions

After making predictions on the test set, it was observed that some values of *Purchase* were predicted as negative, which is not theoretically possible since the minimum value of *Purchase* in the train dataset was always positive. Hence, in the predicted values of *Purchase*, all negative predictions from all models were replaced with zero.

4.4 Model Building

Various algorithms were analysed and used for training the dataset and making predictions on the test dataset.

4.4.1 Linear Regression

Linear regression is the study of *linear*, *additive* relationships between variables. Let Y denote the "dependent" variable whose values needs to be predicted, and let $X_1, \ldots, X_k$ denote the "independent" variables from which one wishes to predict it, with the value of variable X_i in period t (or in row t of the data set) denoted by X_{it}. Then the equation for computing the predicted value of Y_t is denoted in Eq. (2):

$$Y = b_0 + b_1 X_{1t} + b_2 X_{2t} + \cdots . b_k X_{kt} \qquad (2)$$

For our dataset, this variable Y is *Purchase*. Since this model can only take numerical input, the input features – *Age*, *Gender*, *Stay_In_Current_City* were label-encoded. Rest of the features were already numerical.

4.4.2 XGBoost

Extreme Gradient Boosting is an implementation of *gradient boosting decision tree algorithm*. This software library is mainly focused on computational speed and model performance. Boosting is an ensemble-based technique in which new models are trained sequentially, so as to correct the errors made by existing/old models. In this approach, new models are added sequentially, until no further improvements are noticed [6].

The main advantages of this algorithm are – support for parallel processing, inbuilt cross-validation, has variety of regularization techniques to reduce over-fitting. Gradient boosting assists in the prediction of optimal gradient for additive models, unlike classical gradient descent techniques where output errors are reduced at each iteration. Since this model can only take numerical input [6], the input features – *Age*, *Gender*, *Stay_In_Current_City* were label-encoded. Rest of the features were already numerical.

Table 6 - Hyperparameter Values for XGBoost

Parameter	Value
eta	0.03
min_child_weight	10
subsample	0.8
Colsample_bytree	0.7
Max_depth	10
n_estimators	750
Num_rounds	1500

GridSearchCV is a library function that helps in looping through predefined hyperparameters and fitting of estimators (models) on the training set so as to select the most optimum values of parameters from a predefined list of hyperparameters [11]. Table 6 lists the best value of these hypermarameters for XGBoost, attained after extensive experimentation.

0	PID_MeanP	46.78	26	User_ID	0.52
1	PID_50Perc	10.87	27	Age_Count	0.51
2	PID_75Perc	9.30	28	Pc3_25Perc	0.51
3	PID_25Perc	5.16	29	City_Category_C	0.51
4	UID_75Perc	1.26	30	Pc3_75Perc	0.51
5	UID_MeanP	1.22	31	Product_Category_2_Count	0.50
6	UID_25Perc	1.01	32	Pc2_MinP	0.50
7	UID_50Perc	0.97	33	Occupation_Count	0.50
8	Pc1_MaxP	0.90	34	Pc2_50Perc	0.49
9	Pc1_50Perc	0.78	35	Product_Category_3	0.49
10	PID_MaxP	0.74	36	Pc2_75Perc	0.49
11	Pc1_MinP	0.74	37	Pc3_MeanP	0.49
12	Gender	0.71	38	Occupation	0.48
13	Pc1_25Perc	0.69	39	UID_Min	0.48
14	Pc1_75Perc	0.69	40	Product_Category_1_Count	0.47
15	Pc1_MeanP	0.66	41	PID_MinP	0.47
16	UID_MaxP	0.65	42	Pc2_25Perc	0.47
17	User_ID_Count	0.63	43	Stay_In_Current_City_Years	0.46
18	Purchasing_power	0.56	44	Pc2_MeanP	0.46
19	Age	0.54	45	Product_ID_Count	0.46
20	Product_Category_3_Count	0.54	46	Pc2_MaxP	0.46
21	Pc3_50Perc	0.54	47	Product_Category_2	0.44
22	Pc3_MinP	0.53	48	Product_ID	0.44
23	City_Category_A	0.53	49	City_Category_B	0.43
24	Pc3_MaxP	0.52	50	Marital_Status	0.41
25	Product_Category_1	0.52			

Figure 13 - Feature Importance for XGBoost.

It is clearly evident from Figure 13 that the feature importance of newly generated models is much higher compared to that of the original features. Hence, it can be concluded that these features helped the model immensely.

4.4.3 Neural Networks

Keras is a powerful and open-source Python library for developement and evaluation of deep neural networks. It wraps the efficient numerical computation libraries for simple implementation [12]. For our model, *Gender* Column is label-encoded and *Age*, *Gender*, *Stay_In_Current_City_Years* were encoded using One-Hot Encoding. Entire training data was scaled using Sklearn's MinMaxScaler, which essentially scales each value in the dataset within a range of [0,1].

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_7 (Dense)              (None, 256)               11520
_________________________________________________________________
dense_8 (Dense)              (None, 1024)              263168
_________________________________________________________________
dense_9 (Dense)              (None, 1024)              1049600
_________________________________________________________________
dense_10 (Dense)             (None, 512)               524800
_________________________________________________________________
dense_11 (Dense)             (None, 512)               262656
_________________________________________________________________
dense_12 (Dense)             (None, 32)                16416
_________________________________________________________________
dense_13 (Dense)             (None, 1)                 33
=================================================================
Total params: 2,128,193
Trainable params: 2,128,193
Non-trainable params: 0
```

Figure 14 - Keras Model Summary

Count variables were generated for each unique entry of *User_ID*, *Product_ID*, *Product_Category_1*, *Product_Category_2 and Product_Category_3. Additionally*, Min, Max, Mean, 25_Percentile, 50_Percentile and

75_Percentile Variables are generated for *User_ID* and *Product_ID*. Lastly, Meanprice variables were generated for each of the three Product Categories. The structure of this model is as shown in Figure 14. Adam Optimizer was used for our model, with Learning_Rate set to 0.001 and loss set to mean_squared_error. Early_stopping [13] helped in stopping the training after 58 Epochs and restoring the best weights.

4.4.4 AutoML

H2O's AutoML is an extremely helpful tool for providing a simple wrapper function that performs a large number of modelling-related tasks, automating the machine learning workflow, which includes automatic training and tuning of multiple models bound by a user-specified limit of number of models. Stacked Ensembles based on all previously trained models are automatically trained on collections of individual models to produce highly predictive ensemble models. Usually, these ensemble models are the top performing models in the AutoML Leaderboard [14].

For our problem, *Age, Gender, Marital_Status, City_Category & Stay_In_Current_City_Years* were converted to string type for appropriate encoding by AutoML. Train and test data was converted to H2OFrame. A local H2O instance was setup on our personal device and 8GB RAM was allocated to it. The only parameter was max_models, which was set to 12. AutoML's Leaderboard as displayed in Figure 15 provides valuable information about each prediction model [15].

model_id	mean_residual_deviance	rmse	mse	mae
StackedEnsemble_AllModels_AutoML_20201228_143416	5.70948e+06	2389.45	5.70948e+06	1745.77
StackedEnsemble_BestOfFamily_AutoML_20201228_143416	5.74758e+06	2397.41	5.74758e+06	1751.79
GBM_5_AutoML_20201228_143416	5.75964e+06	2399.93	5.75964e+06	1753.37
GBM_4_AutoML_20201228_143416	5.76707e+06	2401.47	5.76707e+06	1756.52
GBM_grid__1_AutoML_20201228_143416_model_2	5.79899e+06	2408.11	5.79899e+06	1756.79
GBM_3_AutoML_20201228_143416	5.8116e+06	2410.73	5.8116e+06	1765.6
GBM_2_AutoML_20201228_143416	5.83748e+06	2416.09	5.83748e+06	1771.08
GBM_1_AutoML_20201228_143416	5.86851e+06	2422.5	5.86851e+06	1776.57
GBM_grid__1_AutoML_20201228_143416_model_1	5.88586e+06	2426.08	5.88586e+06	1779.41
DRF_1_AutoML_20201228_143416	5.89499e+06	2427.96	5.89499e+06	1775.21
XRT_1_AutoML_20201228_143416	5.90009e+06	2429.01	5.90009e+06	1776.42
DeepLearning_1_AutoML_20201228_143416	6.17407e+06	2484.77	6.17407e+06	1828.57
DeepLearning_grid__1_AutoML_20201228_143416_model_1	6.54403e+06	2558.13	6.54403e+06	1894.34
GLM_1_AutoML_20201228_143416	2.43716e+07	4936.76	2.43716e+07	4005.39

Figure 15 - AutoML's Leaderboard Summary

4.4.5 Catboost

CatBoost is a recently open-sourced machine learning algorithm (from Yandex) which has the capability of integrating with existing deep learning frameworks. Similar to XGBoost, CatBoost is based on the gradient boosting library, which is a powerful machine learning algorithm that is widely applied to various sorts of problems in various domains. It is extremely robust as the data essentially needs little to no pre-processing at all. One implication is that it is possible to either specify categorical columns before training, or to just leave them in string format itself.

State of the art results have been demonstrated by this library on multiple Categories of data, such as audio, text, image including historical data. Catboost uses a technique where weighted sampling happens in the tree-level and not in the split-level. The main aim is to maximize the split-scoring accuracy [7].

This model has an argument named *cat_features* where one can specify the categorical features before training [16]. A list of categorical variables was prepared and passed to this argument. This list had the following entries – *User_ID, Product_ID, Gender, Age, Occupation, City_Category, Marital_Status, Stay_In_Current_City_Years.* Numerous parameters were tested using GridSearchCV, and the most optimal ones were – Learning_rate=0.04, max_depth=10 and iterations=4000.

0	PID_MeanP	12.875408	25	Pc2_MaxP	1.074566
1	PID_75Perc	6.350930	26	Product_Category_1_Count	1.022687
2	UID_25Perc	5.294287	27	Pc3_MinP	0.789584
3	PID_50Perc	5.038760	28	Product_Category_3_Count	0.787685
4	UID_MeanP	4.845355	29	Pc1_MaxP	0.748969
5	UID_75Perc	4.667984	30	Pc1_MeanP	0.729837
6	PID_25Perc	4.663049	31	Age_Count	0.727354
7	User_ID	4.597544	32	Pc3_MaxP	0.720994
8	UID_50Perc	3.986670	33	Pc1_50Perc	0.716456
9	UID_MaxP	3.732204	34	Gender	0.698784
10	User_ID_Count	3.188741	35	Pc1_75Perc	0.676638
11	UID_Min	2.962162	36	Pc1_25Perc	0.595421
12	Product_Category_1	2.742421	37	Occupation_Count	0.466051
13	Occupation	2.646217	38	Product_Category_2	0.400199
14	Purchasing_power	2.476389	39	Product_Category_3	0.376655
15	PID_MinP	2.459930	40	Marital_Status	0.293158
16	Product_ID	2.362142	41	Pc2_25Perc	0.242846
17	Product_ID_Count	2.255900	42	Pc3_25Perc	0.182340
18	Age	2.231589	43	Pc3_75Perc	0.179048
19	City_Category	1.780540	44	Pc3_50Perc	0.170634
20	PID_MaxP	1.630509	45	Pc3_MeanP	0.154758
21	Stay_In_Current_City_Years	1.494760	46	Pc2_50Perc	0.138978
22	Product_Category_2_Count	1.234950	47	Pc2_75Perc	0.126565
23	Pc1_MinP	1.210996	48	Pc2_MeanP	0.125618
24	Pc2_MinP	1.124740			

Figure 16 - Feature Importance for Catboost Model

Similar to the XGBoost Model, the feature importance (Figure 16) of the newly generated features is generally higher than that of the original features. Hence, it can be concluded that these data engineering techniques helped build a much more accurate model.

4.4.6 Ensembles

From the previously generated models, it is possible to make new models by taking weighted averages of old models. It was observed that the best score was obtained by taking the ratio of 57:43 weighted average of CatBoost and XGBoost models respectively. Adding any other model only degraded the performance (Table 7).

V. Performance Evaluation

This section discusses the configuration settings and experimental results.

5.1 Configuration Settings

The entire experimentation has been conducted on a windows 10 machine with the following specifications- Processor – Intel® Core i7-10750H, RAM – 16GB 3200MHz DDR4, GPU- Nvidia RTX 2060 (6GB VRAM) and System Type – 64 bits. The Python3 version is 3.6.3 and Jupyter Notebook has been used for writing the python codes.

5.2 Results and Discussion

It can be observed from Table 7 that XGBoost & Catboost models made much more accurate predictions than the other models. It should also be noted that these models trained much faster than the others and required lesser RAM as well. As of the date of writing this article, **the rank of our ensemble model is 116 out of a total of 22,693 submissions,** which proves the effectiveness and robustness of gradient boosting algorithms, along with the necessity of feature engineering and feature augmentation.

Table 7 - RMSE Score on Test Data

Approach	RMSE
Linear Regression (Baseline)	4623.9775
Linear Regression (Feature-engineered)	2610.3038
Neural-Network	2534.5756
AutoML	2484.0706
XGBoost	2461.0996
Catboost	2458.1701
Ensemble – (0.57*CatBoost)+(0.43*XGBoost)	2447.3112

VI. Conclusions and Future Work

In this article, the dataset was thoroughly analysed, and it was found out that there was no direct correlation between any specific demographic and the Purchase price. Hence, new features were generated according to the importance of each existing feature, and the existing categorical data was appropriately converted into numerical format so as to be used by the algorithm. Extremely robust and modern techniques were applied [6] [7], along with handling of outliers and erroneous predictions. All these techniques led to a respectable rank of 116 out of a total of 22,693 participants based on the Root Mean Squared Error on the public test data.

Future Work: This work may be prolonged in following ways-

1. This data is from 2016. Due to various factors like Covid-19 and the general boom in the Internet usage, shopping patterns have changed, and these techniques can similarly be applied to new and updated data from various ecommerce websites, etc.
2. The data from the 2020 Black Friday Sales can be studied and used to make sales predictions in case of any other future pandemic.
3. A detailed dataset with more information about the actual selling products can help build a much better model.
4. Latest deep learning models can be utilized in the future to improve predictions.
5. The exact same techniques of feature engineering and data augmentation can be applied to other problems (e.g. for healthcare, agriculture or weather forecasting) with limited data.

SOFTWARE AVAILABILITY

We released Black Friday Analysis and Prediction as open source. The implementation code with experiment scripts can be found at the GitHub repository : " https://github.com/sanskartewatia/Sales-Prediction ".

REFERENCES

[1] Yi, David (November 23, 2010). "Black Friday Deals for Target, H&M, Forever21, Old Navy, Radio Shack, and More". Daily News. New York. Archived from the original on August 15, 2011.

[2] "Black Friday Moves to Thursday as Stores Woo Shoppers". Financially. Yahoo! Finance. November 23, 2010. Archived from the original on July 26, 2011. Retrieved January 2, 2012.

[3] Trung ND, Thien TD, Luu TD, Huynh HX." BLACK FRIDAY SALE PREDICTION VIA EXTREME GRADIENT BOOSTED TREES ". 08/6/2019 http://vap.ac.vn/Portals/0/TuyenTap/2020/4/18/fa7f0835392b41f4a4751e1593be09f5/8-132-Black-Friday.pdf

[4] Practice problem: Black Friday sales prediction | knowledge and learning," Jul 2016. [Online]. Available: https://datahack.analyticsvidhya.com/contest/black-friday/

[5] Aman Jain, "https://www.kaggle.com/amanacden/blackfriday-av/notebook".

[6] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pp. 785-794. 2016.

[7] Dorogush, Anna Veronika, Vasily Ershov, and Andrey Gulin. "CatBoost: gradient boosting with categorical features support." arXiv preprint arXiv:1810.11363 (2018).

[8] C. M. Wu, P. Patil and S. Gunaseelan, "Comparison of Different Machine Learning Algorithms for Multiple Regression on Black Friday Sales Data," *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, Beijing, China, 2018, pp. 16-20, doi: 10.1109/ICSESS.2018.8663760.

[9] S. Kalra, B. Perumal, S. Yadav and S. J. Narayanan, "Analysing and Predicting the purchases done on the day of Black Friday," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1-8, doi: 10.1109/ic-ETITE47903.2020.256.

[10] Catboost Documentation, " https://catboost.ai/docs/concepts/python-reference_catboostregressor.html ".

[11] GridSearchCV Documentation, " https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html ".

[12] Keras Documentation, "https://keras.io/api/".

[13] Keras EarlyStopping, "https://keras.io/api/callbacks/early_stopping".

[14] He, Xin, Kaiyong Zhao, and Xiaowen Chu. "AutoML: A Survey of the State-of-the-Art." Knowledge-Based Systems 212 (2021): 106622.

[15] AutoML Documentation, " http://docs.h2o.ai/h2o-tutorials/latest-stable/h2o-world-2017/automl/index.html "

[16] Catboost Documentation, " https://catboost.ai/docs/concepts/python-reference_catboostregressor.html ".

Sanskar Tewatia is a senior year Electronics and Communication Engineering student pursuing his studies from Shiv Nadar University, Greater Noida, India. His main research interests are in the field of Machine Learning and its application in solving problem. He is passionate about deep learning and has experience of working on robotics and microcontrollers. He also likes to write and edit technology-based articles for a blogging website in his free time.

Ankit Anil Patel is pursuing master's in computer science at School of Electronic Engineering and Computer Science, Queen Mary University of London, UK. Prior to this, Ankit held his bachelor's in computer engineering degree from St. John College of Engineering and Management, Palghar, India.

Manmeet Singh is a Scientist at the Centre for Climate Change Research at the Indian Institute of Tropical Meteorology (IITM), Pune. He is presently pursuing his PhD in Climate Studies from the Indian Institute of Technology Bombay. His main research interest is in the computational modelling of earth system and its processes. He is presently involved in studying the Monsoons using coupled general circulation models and non-linear methods. He also has experience in the development of Direct Numerical Simulation model and using it for the basic understanding of jets. He is passionate about the use of Deep Learning and Deep Reinforcement Learning as tools to solve the problems in Climate Science.

Sukhpal Singh Gill is a Lecturer (Assistant Professor) in Cloud Computing at School of Electronic Engineering and Computer Science , Queen Mary University of London, UK. Prior to this, Dr. Gill has held positions as a Research Associate at the School of Computing and Communications, Lancaster University, UK and also as a Postdoctoral Research Fellow at CLOUDS Laboratory, The University of Melbourne, Australia. His research interests include Cloud Computing, Fog Computing, Software Engineering, Internet of Things and Healthcare. For further information, please visit http://www.ssgill.me.