International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 06, June-2021

Predicting Customers' Next Order

1st Sami M K
Department of Computer Engineering
Indira College of Engineering and Management
Pune, India

2nd Radhika Gupta
Department of Computer Engineering
Indira College of Engineering and Management
Pune, India

3rd Kratika Gupta
Software Development Engineer
Amazon Web Services
Seattle, USA

***Abstract*—the popularity of targeted marketing has grown over the past few years. The trend of online shopping has become a new normal during the Covid pandemic. Customers buying products often leave behind a trail that helps us to predict the future. Understanding the costumers demand and their shopping pattern is the key to targeted marketing and is of immense value to companies. Using Machine Learning we can recognize the predictive patterns of the customers' behavioral data. This can be used to automatically add products to the shopping cart. Thereafter, the user can review the products in the cart before ordering it.**

Keywords*— Component; formatting; style; styling; insert (key words)

I. INTRODUCTION

The fact is, technology is collecting data with every single click. With this information, it becomes extremely easy for companies to improve their marketing strategies. Predicting customers' demands gives the company information to strategize and act accordingly. It also helps customers by automatically adding products to their cart. Such a model has competitive advantage over traditional methods. We introduce a model which uses a combination of person's previous order and the time interval between consecutive orders to predict their next order.

II. PROBLEM STATEMENT

To create a Machine Learning model that will help the user to determine their next order based on their previous ordering history. The model should determine the product, the interval after which the product will be ordered and the quantity of the products to be ordered.

III. DATASET

The dataset was released by Instacart by the name "The Instacart Online Grocery Shopping Dataset 2017". It is a set of files that has customers' order history. The dataset holds 3 million anonymous orders of nearly 2 lakh Instacart users. It supplies the history of products between 4 and 100 bought by the customer.
The dataset is divided into 3 parts, prior, Train and Test. The data does not include data about "reordered" products and the number of orders of products are not same.
The dataset consists of five tables: products, aisles, departments, orders, and products and it has a relational structure. Moreover, it supplies the week and hour of day the order was placed, and a relative measure of time between orders. In order to use this data we have used MySQL.

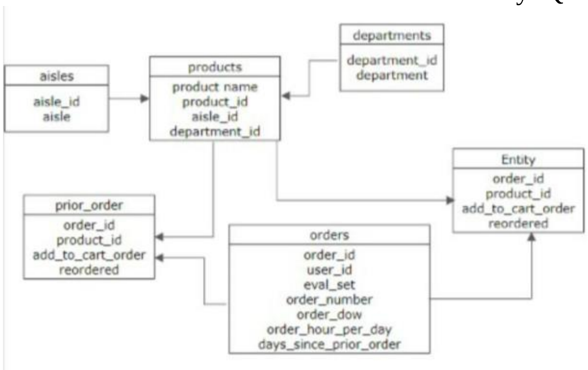

Fig.1. Relation diagram of dataset

IV. DATA PREPARATION

We have generated five tables from the dataset which are Products, Aisles, Department, Orders and Other Products. Then we joined these above tables into Productscombined (Department, Aisle, Products), Order combined (order_products_prior and orders). The Productscombined table contains all the details of each product and received 73,575 product ids. The order combined table has 73,000 records contains all the details of each order.

V. PROCESS

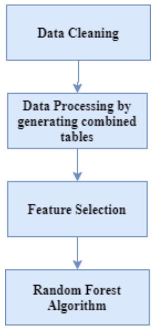

Fig.2. Systematic process

We have predicted the products that will be reordered based on the number of days since the last order, the day of the week, the time of the day and the products that the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 06, June-2021

customer adds first to the cart from 10,931 products. Later we merged the data from the combined tables for exploratory analysis. First, we merged the 'productscombined' table (has information related to products) and 'ordercombined1' table (has details about previous orders) and named this table as 'prioralldata'. Second, we merged the 'productscombined' table (has information related to products) and 'ordercombined2' (having details about trained orders) and named this table as 'trainalldata'. We also created the top 10 most popular products by all product_name, within the department and within the isle.

We calculated the distribution of reorders based on the factors reorders each day of the week, each hour of the day, frequency distribution by days since prior order, distribution of orders vs reorders orders_prior table, distribution of orders vs reorders orders_train table, distribution of top-10 products orders_prior table, distribution of top-10 aisles orders_prior table.

After observing the distribution of orders on different days of the week per hour we found that most of the purchases were made between 9 am and 7 pm i.e., during office hours, however on the weekends the scenario was slightly different. On Saturdays, the number of orders increased steadily from 9 am and dropped sharply after 4 pm. On the other hand, on Sundays the orders peaked at 10 am and dropped every hour till 5 pm. Then we calculated which products were popular purchases on weekends (with respect to orders_prior table). This showed us that mostly people bought organic fruits and veggies on the weekends.

```
Time for productscombined 0.749714

Top5_productscombined
    index  product_id                                        product_name  \
0       1           1                          Chocolate Sandwich Cookies
1       2           2                                    All-Seasons Salt
2       3           3                Robust Golden Unsweetened Oolong Tea
3       4           4   Smart Ones Classic Favorites Mini Rigatoni Wit...
4       5           5                           Green Chile Anytime Sauce

   aisle_id  department_id department                       aisle
0        61             19     snacks               cookies cakes
1       104             13     pantry          spices seasonings
2        94              7  beverages                         tea
3        38              1     frozen                frozen meals
4         5             13     pantry  marinades meat preparation

Number of products in database
    COUNT (product_id)
0               49688
```

Fig.3. productcombined

```
Time for orderscombined1 1.89819

Top5_orderscombined1 Table
    index  order_id  user_id eval_set  order_number  order_dow  \
0      71     23391        7    prior            17          0
1      71     23391        7    prior            17          0
2      71     23391        7    prior            17          0
3      71     23391        7    prior            17          0
4      71     23391        7    prior            17          0

   order_hour_of_day  days_since_prior_order  product_id  add_to_cart_order  \
0                 10                    28.0       13198                  1
1                 10                    28.0       42803                  2
2                 10                    28.0        8277                  3
3                 10                    28.0       37602                  4
4                 10                    28.0       40852                  5

   reordered
0          1
1          1
2          1
3          1
4          1


Total number of products in orders in database
    COUNT (product_id)
0               72909
```

Fig.4. ordercombined1

```
Time for orderscombined2 2.931859

Top5_orderscombined2 Table
    index  order_id  user_id eval_set  order_number  order_dow  \
0      11   1187899        1    train            11          4
1      11   1187899        1    train            11          4
2      11   1187899        1    train            11          4
3      11   1187899        1    train            11          4
4      11   1187899        1    train            11          4

   order_hour_of_day  days_since_prior_order  product_id  add_to_cart_order  \
0                  8                    14.0         196                  1
1                  8                    14.0       25133                  2
2                  8                    14.0       38928                  3
3                  8                    14.0       26405                  4
4                  8                    14.0       39657                  5

   reordered
0          1
1          1
2          1
3          1
4          1


Total number of products in orders in database
    COUNT (user_id)
0            73575
```

Fig.5. ordercombined2

VI. FEATURE SELECTION

A. LASSO Regression

Lasso regression is a linear regression with L1 regularization. If we see the red point, it is deviated from the original deviation, this point is called outlier. Outlier could be because of human or experimental error or variability during the observation of data. Because of outlier we could not get an almost straight line. The predicted value is far from the actual value and it is because of gradient descent or cost function but because of the data.

LASSO involves a penalty factor that decides how many features are kept; using cross-validation to choose the penalty factor helps assure that the model will generalize well to future data samples. It automates feature selection based on standard linear regression by stepwise selection or choosing features with the lowest *p*-values.

We have used the LASSO regression algorithm to choose the first six features that will help in deciding the products that will be reordered.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 06, June-2021

```
Ranking of LASSO features is
1 . Score   order_number  is  0.00523130112728
2 . Score   add_to_cart_order  is  -0.0
3 . Score   days_since_prior_order  is  -0.0
4 . Score   order_hour_of_day  is  -0.0
5 . Score   product_id  is  -3.53015330177e-07
6 . Score   order_id  is  2.56075609104e-07
7 . Score   order_dow  is  -0.0
['order_id' 'order_dow' 'days_since_prior_order' 'order_hour_of_day'
 'order_number' 'add_to_cart_order' 'product_id' 'reordered']
[  2.56075609e-07  -0.00000000e+00  -0.00000000e+00  -0.00000000e+00
   5.23130113e-03  -0.00000000e+00  -3.53015330e-07]
```

Fig.6. Top 6 features selected using LASSO

B. *SelectKBest Algorithm*

If we see the redpoint, it is deviated from the original deviation, this point is called outlier. Outlier could be because of human or experimental error or variability during the observation of data. Because of outlier we could not get an almost straight line. The predicted value is far from the actual value and it is because of gradient descent or cost function but because of the data.

A penalty factor in LASSO decides how many features are to be kept; the penalty factor is chosen using cross-validation which makes sure that the model generalizes to future data samples. It automates feature selection based on standard linear regression by stepwise selection or choosing features with the lowest *p*-values.

We have used the LASSO regression algorithm to choose the first six features that will help in deciding the products that will be reordered.

```
Ranking of features is
1 . Score   order_number  is  7436.76339123
2 . Score   add_to_cart_order  is  1230.39796882
3 . Score   days_since_prior_order  is  1204.20175736
4 . Score   order_hour_of_day  is  48.7814693703
5 . Score   product_id  is  9.06752313378
6 . Score   order_id  is  5.11306880726
7 . Score   order_dow  is  0.389505663797
['order_id' 'order_dow' 'days_since_prior_order' 'order_hour_of_day'
 'order_number' 'add_to_cart_order' 'product_id' 'reordered']
```

Fig.7. Top 6 features chosen

We added a new feature based on reorders in relation to total number of products and found that around 60% of all the products have been reordered.

We performed feature selection with SelectKBest and LASSO. Both the algorithms gave almost similar results, so we decided to choose first six features that are ‘order_number’, 'add_to_cart_order', 'days_since_prior_order', 'order_hour_of_day', 'product_id', 'order_id' to predict which products will be reordered.

VII. DATA CLEANING

To clean the data, we replaced all the NaN and infinity with the mean value from enron_df. We dropped categorical data as only numeric data goes in not machine learning algorithms.

VIII. SELECTION OF MODEL

Classifier	Accuracy	Precision	Recall
Logistic Regression	0.6439598287081486	0.5601015873015873	0.7849578967290638
SVM_rbf	0.6982428869687135	0.7238603174603174	0.7614877582575289
SVM_sigmoid	0.5778661668114418	0.5758730158730159	0.6733021460203243
Gaussian Naive Bayes	0.6448132400676633	0.7484952380952382	0.6873790057401367
SVM_linear	0.6370411009006537	0.5414349206349206	0.7872649635405348
Decision Tree	0.7155854249531387	0.7363047619047619	0.7778481343055672
Random Forest	0.7654185525533762	0.8372571428571428	0.7858539594296855
KNN	0.7433822520916197	0.8233142857142856	0.7663866110063716

Fig.8. Analysis of each model

Based on the above table we selected Random Forest Algorithm since it provided highest accuracy.

A. *Random Forest Model*

Random Forest is a tree-based Machine Learning algorithm. Multiple decision trees are constructed and trained on sample drawn from the original dataset. An average of the individual from each decision tree and a majority class vote in a classification task are the result in case of regression task. Higher the number of trees in the forest higher the accuracy.

Random Forest Algorithm:

- Select random k points from the training set.
- Build the decision tree with the selected data points.
- Choose the number of decision trees that you want to build.
- Repeat steps 1 and 2.
- For each data point find the prediction of each tree and make the final prediction based on majority votes.

1) We needed to decide the number of trees. Though greater numbers of trees improve the quality of classification, it makes the code work slower. We checked the accuracy, precision and recalled for number of trees equal to 120, 300, 500, 800 and 1200. Based on the output we built the Random Forest Classifier model with default parameter of n_estimators = 1200. So, we used 1200 decision-trees to build the model.

To increase the accuracy, we altered few parameters like max_depth, max_sample_split, max_leaf_nodes and max _features.

2) The maximum depth i.e., the nodes are expanded until all leaves are pure or until all leaves have less than min_samples_split samples. We tested for max_depth equal to 5, 8, 25, 30, and none. We selected Max_depth = 25 as it gave us the best result.

3) max_sample_split is the minimum number of samples needed to split an internal node. Its default value is 2. We checked value against 2, 5, 10, 15, and 100. Max_sample_split value equal to 2 gave us the best answer.

4) max_leaf_nodes are the maximum number of leaves in the tree. We checked it against the value equal to 2, 5, 10 and none. The choice none gave us the best answer.

5) max_feature is the number of features to consider when looking for the best split.

6) At the end we tested the max_features. The search for the split stopped when we got at least one valid partition of node samples. Now we could finalize all the parameters for Random Forest. We compared the results with data that had

no 'add_to_cart_order' and 'product_id' because we did not have this information in our test data set.

Classifier	Accuracy	Precision	Recall
Random Forest finalized params	0.9203553848732837	0.9433650793650793	0.8902311899357788

Fig.9. Analysis of the model

IX. FUTURE SCOPE

This Machine Learning model could be used by target strategists to increase the market value of the supermarkets and online grocery stores. The algorithm could further be extended on other data sets. For example, it could be trained on pharmaceutical stores dataset to automatically order medicines of regular customers for example patients suffering from diabetes, low blood pressure etc. The accuracy of the model can further be increased by deploying other models in place of the Random Forest or LASSO regression models.

X. CONCLUSION

Using various Machine learning algorithms like LASSO, SelectKBest, and Random Forest classifiers we have predicted the date, time and the products for the next order of the customer. After testing out the model we received an accuracy of 89%. This ensures that there are endless possibilities to which this model can be expanded. Moreover, using this prediction, the supply chain industries can enhance their marketing strategies. This also provides a platform for the users where they must do minimal work.

REFERENCES

[1] Niu, X., Li, C., & Yu, X. (2017). Predicitive analytics of e-commerce search behavior for conversion.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68-73.

[2] Python. (2017). Python website. Retrieved from https://www.python.org/

[3] Randomforestclassifier. (2017). Retrieved from http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[4] Russell, S., & Norvig, P. (1995). Artificial intelligence - a modern approach. PrenticeHall, Englewood Cliffs: Artificial Intelligence.

[5] Lee, M., Ha, T., Han, J., Rha, J., & Kwon, T. (2015). Online footsteps to purchase:Exploring consumer behaviors on online shopping sites. In Proceedings of the ACM Web Science Conference.

[6] Manning, C., Raghavan, P., & Schuetze, H. (2008). Introduction to information retrieval. Cambridge University Press.

[7] Kaggle – the home of data science and machine learning. (2017). Retrieved from https://www.kaggle.com/

[8] Alsanad, Ahmed. "Forecasting Daily Demand of Orders Using Random Forest Classifier." INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND NETWORK SECURITY 18, no. 4 (2018): 79-83.