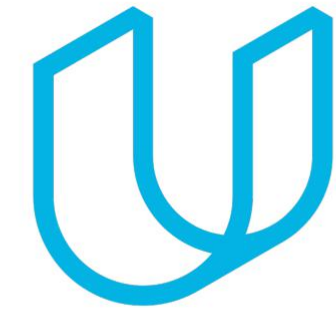
UDACITY MACHINE LEARNING ENGINEER NANODEGREE

CAPSTONE PROJECT

Xente Purchase Prediction

David Popoola

April 24, 2020



UDACITY

I. Definition

Project Overview

Xente is an e-payments, e-commerce, and financial services company in Uganda offering various products and services that can be paid for using Mobile Money (Airtel Money, MTN Mobile Money), Bank Card (Visa Card, Master Card), Xente wallet and on credit (Pay Later). Some of the products consumers can buy include airtime, data bundles, event tickets, movie tickets, bus tickets, and more. They can also pay water and electricity bills and TV subscription services.

Xente is a Ugandan e-commerce startup that makes it easy for consumers to make payments, get loans, and shop using simply a mobile phone.

Problem Statement

The goal of this project is to Create a machine learning model to predict what and when individuals will purchase next, based on their purchase history. I will be tackling this as a prediction problem and more specifically binary classification to determine if customers will purchase certain products in certain days in the future. Doing this will provide Xente with models and solutions that can help build better experiences for customers. These better experiences can come in the form of better target marketing and catering to their customers' needs better. This can help in the bottom line of the business leading to improved profitability, financial sustainability, better retention of customers due to tailored app experiences and also business longevity in the big picture.

The resulting models and solutions will help Xente with target marketing and catering to their customers' needs better. For Xente, this may result in improved profitability and financial sustainability; while customers receive an app that is tailored to them.

Metrics

Prediction results are evaluated using the F1 score of values obtained and the ground truth. The ground truth is the set of labels that have been given in the training data set. The ground truth

labels are definitive, as they depict products purchased by customers at several days in the past as given in the time range for the train dataset.

Given this is a Zindi competition project, I will use the leaderboard score for submissions as my evaluation criteria. The leaderboard score is actually gotten as the F1 score of the predicted values of the test data, which are unseen to the public. However, for my training process, I will also use the model accuracy and precision as the screening metric to select the best training model, and then use F1 to fine tune the parameters. The F1 is a great metric for this project because we are predicting the probability of label being either 0 or 1. Also there is a class imbalance, so false positives and false negatives can serve in improving the model.

II. Analysis

Exploratory Data Analysis (EDA)

The training data and test data from this problem are provided on Zindi website:
https://zindi.africa/hackathons/umojahack-2-xente-purchase-prediction-challenge/data

The train dataset is given in the Train.csv file. PID_Categories.csv file contains supplementary data to identify categories of products Xente offers. The SampleSubmission.csv file comprises some of the data points in the test data set.

There are 3 data fields in the training data:

- PID – the unique id given to a particular product.
- date – Previous dates of purchases made by customers
- acc – Unique ID given to customers on the platform

Let's look at the first 5 lines of training data:

date	acc	PID
2019-12-13 11:38:33.226	256787571627	80G0
2019-11-12 11:46:23.755	256789598703	WLIW

2019-11-12 11:46:23.755	256753510223	U1DD
2019-12-20 13:34:17.453	256702652564	T0LZ
2019-11-15 09:16:06.591	256704005298	80G0

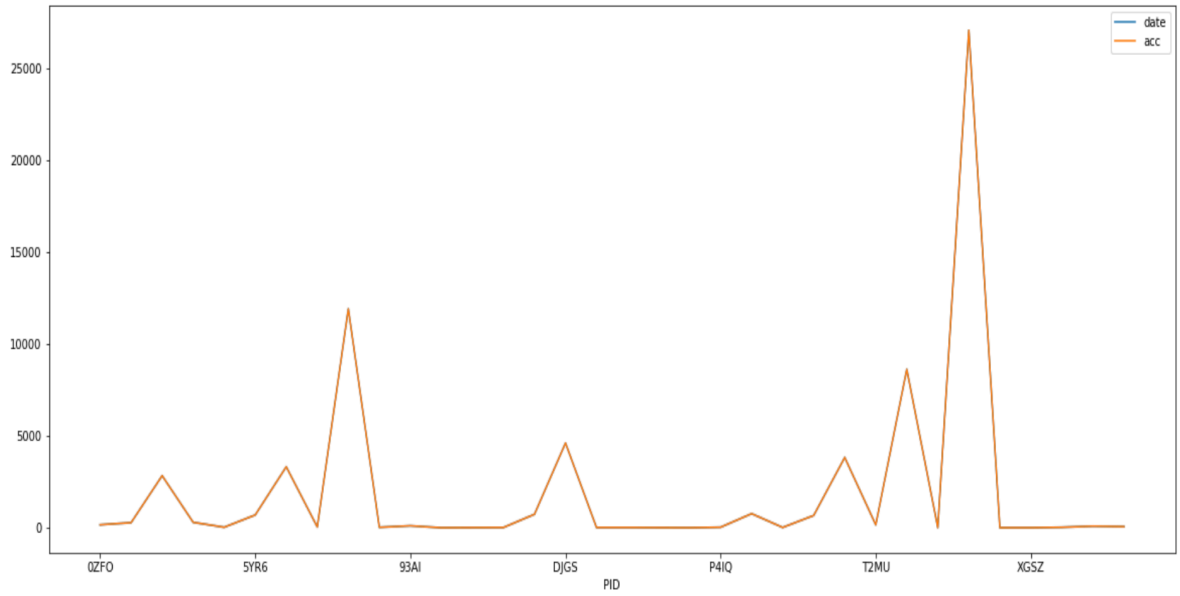
Everything looks fine at this point. On further analysis of the training data, no NaN or missing values across records and columns. So need to clean missing or empty data in the pre-processing step.

Some overview statistics of this training data set are as follows:

- Number of training data points: 66, 514
- Number of testing data points: 111, 048

Exploratory Visualization

The first plot shows several PIDs by count. We can see that some products are purchased by more frequently than others. Obtaining more graphs proved difficult eg graphing individual products purchase etc.



Graph showing the visualization of certain product categories sold by Xente by count.

	datetime	80G0	WLIW	U1DD	T0LZ	6Q4Z	5YR6	P7R7	Q9SJ	DJGS
0	2019-11-01	1	1	1	1	1	1	1	1	1
1	2019-11-02	1	1	1	1	1	1	1	1	1
2	2019-11-03	1	1	1	1	1	1	1	0	1
3	2019-11-04	1	1	1	1	1	1	1	0	1
4	2019-11-05	1	1	1	1	1	1	1	0	1

5 rows × 35 columns

Sample Data wrangling and pre-processing on PIDs and dates as the index column

The distribution of products as a result is not equal. Hence, we expect that certain products should have positive prediction more than others.

Algorithms and Techniques

Since this is a binary classification problem, determining if a customer purchases a product on a certain day or not, I will be using supervised learning algorithms. Unsupervised techniques like Principal Component Analysis(PCA) will not aid much as the training data only contains 3 features. First I checked the entire train sets for missing values, and NaN values that could occur in records and columns, but none existed. Then I will extract and develop any necessary features from the columns in the given train data set. These features include:

1. Datetime
2. Day
3. Date x PID

The algorithms I will be implementing are:

1. Random forest
2. Logistic regression
3. Decision trees
4. K nearest neighbours
5. Naive Bayes
6. Support vector machine
7. Gradient boosting (CatBoost).

I will use a cross validation data set to compare these initial algorithms and find the best one, then do fine tuning on the best algorithm to get optimal accuracy. This fine tuning will be in the form of hyper-parameter tuning and also potentially new features the model needs to make better predictions.

Logistic regression is a regression model where the outcome is binary, and the model predicts the probability (0 to 1) of the outcome.

Decision trees is a model which uses a tree to do binary classification at each node (feature), and at the leave of the tree a label prediction is returned.

Random forest is by using an ensemble of decision trees to reduce overfitting of one tree model. The training algorithm for random forest applies the general technique of bagging, which means selecting a random sample of training data and fitting trees to these samples. Finally, we average the result of all trees to make prediction.

K-nearest neighbour model is using the average of k nearest data points to predict the testing data value. The “distance” her is usually 2D Euclidean distance.

Naive Bayes classifier is a family of simple probabilistic classifiers based on applying Bayes’ theorem with naive independence assumptions between the features.

Support vector machine is an algorithm which outputs an optimal hyperplane to separate labelled training data.

Finally, gradient boosting model is similar to random forests, with the difference being how we train them. For gradient boosting, we assign prediction score in each leaf instead of a binary value, and during training we add one new tree to the ensemble at a time and do the optimizing.

Benchmark

The benchmark model I decided upon is the random forest model. The reason being it is an easy to implement model in terms of complexity and naive enough to be compared with other algorithms. There is no published/stated result for this project, but since it is a Zindi competition, I can get a sense of how well the model is performing by looking at the competition leaderboard score and ranking.

III. Methodology

Data Pre-processing

First I create a new index column using the dates from the train dataset. I parse the dates and use the dates as the index. The problem can also be seen from the angle of time series data forecasting where we try to forecast and make predictions as well based on a given history (context length).

There are a total of 4 features. Acc for unique customer IDs, PID for unique product IDs, datetime and day which are derived from from the date index column.

The next step is to split the data into training and validation sets. I split based on the dates and gave a larger timeframe to the train data set. Only transactions on dates in 2020 were in the validation/ local test set.

Implementation

I have tried 6 supervised learning models excluding the benchmark model. I also discovered they were not producing good results, unlike the Catboost algorithm/model. I decided to continue all development with the Catboost model since it yielded much better results than the other models.

After exploratory data analysis and splitting into training and validation sets. I added extra features for the model that could prove to be potentially useful.

Other actions were carried out as well such as reshaping, obtaining unique PIDs etc.

One challenge in the implementation phase was the limitation in the feature engineering that could be done. Other metrics, data and information could have been used to generate insights and patterns that could help predict purchases by customers on different days with a better likelihood.

Refinement

Based on the Zindi leaderboard score for the different models, Catboost gives the best result.

Therefore, I did some fine tuning and got slightly incremental results by increasing iterations from 20 to 150.
Since Catboost is built to prevent overfitting, there's no worry of the model overfitting.

IV. Results

Model Evaluation and Validation

The final model I used is CatBoost (Gradient Boosting), which is similar to random forest, but with a better performance in the given problem. I have used validation data set to do make sure there is no overfitting and to find the best parameters within reasonable training time. The model is pretty robust because a boosted tree uses many trees to do the training, and unlike random forest, it uses scores in the leaves instead of binary classification. Also, the Zindi competition score is based on unseen data that are not accessible to anyone. If I can get similar F1 socre in my training and the "unseen" testing data in the competition, I am confident that the model provides acceptable results.

Justification

Yes the final result is significantly better than the benchmark model. Although the idea behind random forest and boosted trees is the same, the training process is different. I think boosted trees is a more robust model and hence should give better result. In my project, benchmark model has F1 score of 0.06 while CatBoost model has F1 score of 0.12. I believe the final result is good enough for solving this problem. It is not 100% done or production ready, but should be in the top 25% solution space.

V. Conclusion

Reflection

At first this problem seems to be a simple linear regression classification problem which would also serve in predicting purchase of products by different customers on different days. There are some oversimplifications and as a result tricky components to this problem. For example, the lack of adequate features that could help a model learn about behaviours, patterns and trends in particular users as well as purchase trends of particular products. The tricky part was generating new features that could help from the limited feature set provided in the train dataset. I first thought about the days of the week, or periods of the year and trying to associate purchase of some of the products to these certain time frames. But almost all product categories by intuition cannot be split to be products purchased in certain timeframes, so there is not much of a seasonality or trend there. But this still proved a bit useful. I tried 7 different supervised machine learning algorithms. Gradient boosting gave the best result. After that I fine tuned the parameter and got 0.11 score on the leaderboard (ranked top 25% on the Zindi competition leaderboard). As mentioned before and below, more features could not be engineered or created, due to limited possible features in the first place. My take on this is: adding features may not always be better as they may not be useful, necessary or contribute to improve the model in the first place. A better approach would be to get access to more data as discussed in the improvement section which can aid in the development of a model with better prediction capabilities. This would significantly help in getting much better F1 scores than tiny incremental improvements on the current score

Improvement

To improve this project in the future, I will do more experiments with different techniques such as time series analysis techniques. Ranging from ETS (Error, Trends, Seasonality) decomposition techniques, to other means of forecasting and predicting personalised to each account for each unique customer. Given more data, and company metrics like customer life time value (LTV), churn etc, I could also try out more methods and see how creative I can get in improving the model and having more accurate predictions. Deep Learning frameworks can also be utilized to make sense of the data and try to draw out patterns, trends and even more accurate predictions.

Different forms of deep learning neural networks may be able to help build better models. However, this as all neural networks would need extremely more data, computational power (GPUs), also inner/hidden layers in the network.