Name: SAI DISHA .D

Email address: dishasai1997@gmail.com

Contact number: 9743362080

Anydesk address: 311845204

Years of Work Experience: Fresher

Date: 11 July 2022

Self Case Study - 1: Santander Customer Transaction Prediction

"After you have completed the document, please submit it in the classroom in the pdf format."

Please check this video before you get started:
https://www.youtube.com/watch?time_continue=1&v=LBGU1_JO3kg

Overview

*** Write an overview of the case study that you are working on. ***(MINIMUM 200 words)*** ***

Data: https://www.kaggle.com/competitions/santander-customer-transaction-prediction/data

Santander bank is the 16th largest banking institution in the world, based in Madrid and Santander in Spain. It has also extended its operations to parts of Asia, North and South America. They are always looking at ways to improve their customer experience, helping them understand their financial health and achieve their monetary goals with relevant products and services by using Artificial Intelligence. There are so many financial services that the bank offers, but not all of them will be useful to every customer, so for customer to have a smooth experience, providing customized service is important and their data science team is working to identify ways to solve some common challenges like: is a customer satisfied? Will a customer buy this product? Can a customer pay for this loan?

In any field, customer satisfaction is important and personalized service is the key to achieve this. This also saves cost by doing targeted advertising. Here we are trying to answer the question, "Can you identify who will make a transaction?". The goal here is to identify which customer will make a particular transaction in the future, irrespective of the amount of money made. This provides information to the bank about customers that they can analyze and expect their needs and improve their experience. This is a binary classification task with anonymized data containing over 400 columns of numerical features, a string ID_code, and a binary target column. The structure of the data is like the real world. They partition the provided dataset into two sets of 200,000 entries each for training and testing. 50% of data kept hidden on their server for evaluation of the submission. With the help of data exploration, analysis and experimenting with various machine learning algorithms, we can extract valuable insights and solve this task.

Research-Papers/ Solutions/ Architectures/ Kernels

*** Mention the urls of existing research-papers/solutions/kernels on your problem statement and in your own words write a detailed summary for each one of them. If needed, you can include images or explain with your own diagrams. It is mandatory to write a brief description of that paper. Without understanding of the resource please don't mention it***

1. URL: https://www.researchgate.net/publication/341361969_Customer_Transaction_Prediction_System

CUSTOMER TRANSACTION PREDICTON SYSTEM

Devendra Prakash Jaiswala, Srishti Kumara, Partha Mukherjee

- Nowadays, there is exploitation of customer level data to provide better services to their customers than their rivals using AI in almost every field. Here, the aim is to do the same using an anonymized customer dataset to predict their transactions using deep learning and gradient boost algorithms.
- There were many other research activities carried out on similar problems like classifying customer based on loyalty or churn rate and predicted customer transaction as derived outcome, but they faced their own challenges like high dimensionality and blurry understanding of an anonymized dataset.
- First, the key features identified from the data can improve customer transactions. Second, the probability of transactions can identify patterns in data or external bias that influence customer transactions. Third, the best model would help the organization in predicting whether any scenario would be ideal for a customer to commit a transaction or not.

- They are using the same Santander Bank dataset from Kaggle. Through data exploration, they found data to be skewed, and outliers (kurtosis), no correlation among features, no missing values, and there is a class imbalance.
- Random train-test split (80:20).
- After removing the Id column, box cox transformation normalizes the data, lowers skewness close to 0 and kurtosis close to 4.
- First, Logistic Regression model - returns 171 significant features out of 200, AUC and accuracy drops after 29 features (Baseline model). This model underfits.

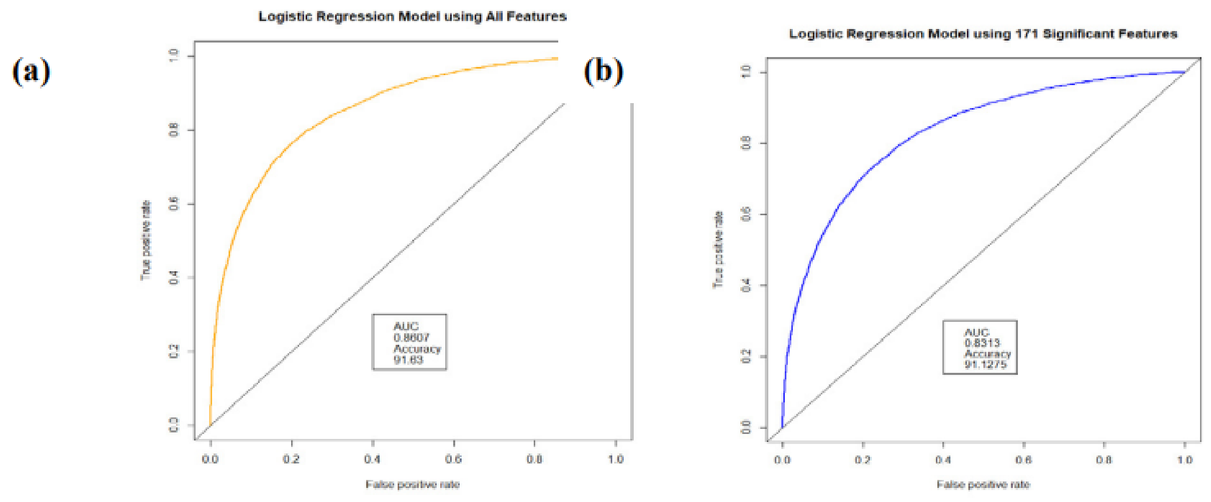

Figure 3:(a) ROC Plot with 200 features. (b) ROC Plot with 171 features

- Second, using Keras, they implemented NN with the following features:
  - Combination of 1D convolutional layers and dense layers
  - Activation: Sigmoid (output) and ReLU (hidden layers)
  - Optimizer: RMSProp
  - Regularization: BN layer, dropout of 0.2 (hidden layers)
  - Loss function: Binary cross entropy function.
  - Good AUC and accuracy values observed after 50 epochs without over-fitting.
  - Model with best results among the three.

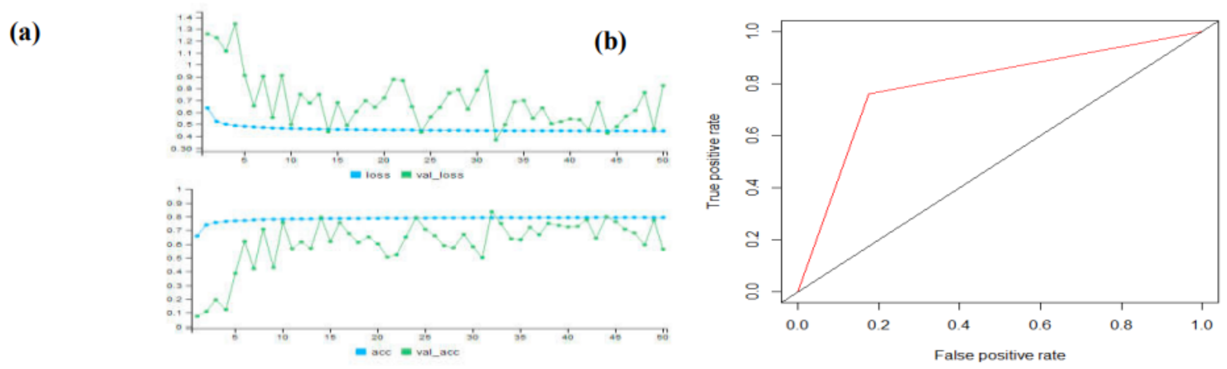

Figure 4: (a) Accuracy and Loss Plot for ANN training. (b) ROC Plot of the predictions

- Third, boosting algorithm (XGBoost) with the following features:
  - over-sampling
  - regularization on weights is 5
  - ratio of training data per tree is 0.7 and the maximum tree depth is 100
  - After 500 iterations, highest accuracy and AUC among all models.
  - Inferior to ANN, but produces the best results despite over-fitting.

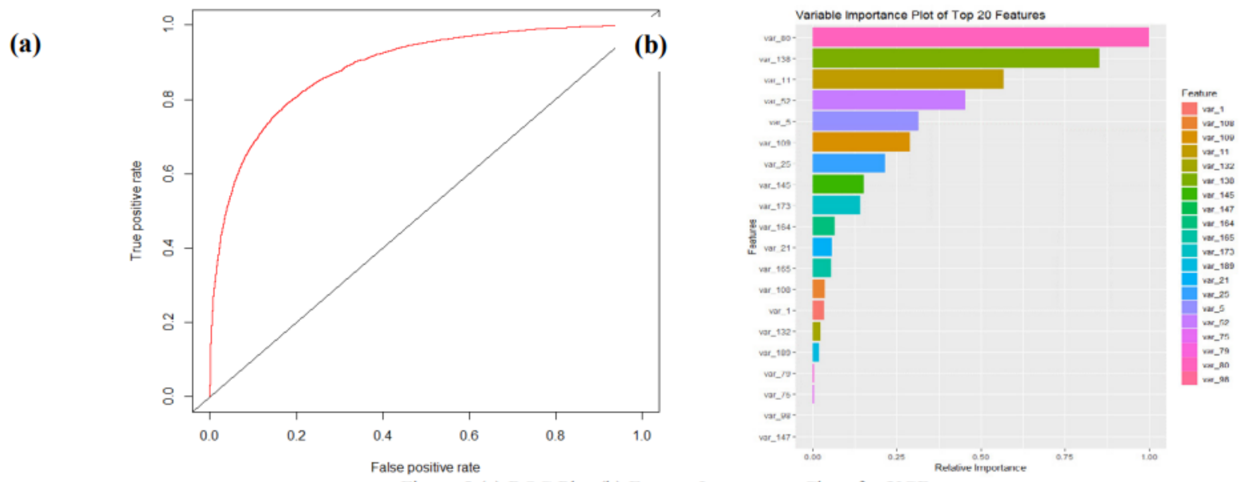

Figure 5 (a) ROC Plot (b) Feature Importance Chart for XGBoost

- ANN is the best in predicting instances where a customer would make the transaction and XGBoost, where a customer would not.

2. URL:

   [(PDF) Predicting the Future Transaction from Large and Imbalanced Banking Dataset (researchgate.net)]

Predicting the Future Transaction from Large and Imbalanced Banking Dataset

Sadaf Ilyas , Sultan Zia Zaib un Nisa, Umair Muneer Butt, Sukumar Letchmunan

- With e-banking, it is possible to make transactions wherever we want and whenever we want and transactions have become faster too. This has led to an exponential increase in customer transactions data. Also, making it difficult to manage or monitor these transactions, using ML decisions are made on the spot on behalf of humans, even when humans are not supervising them.
- Supply and Demand Factors of ML in financial institutions:

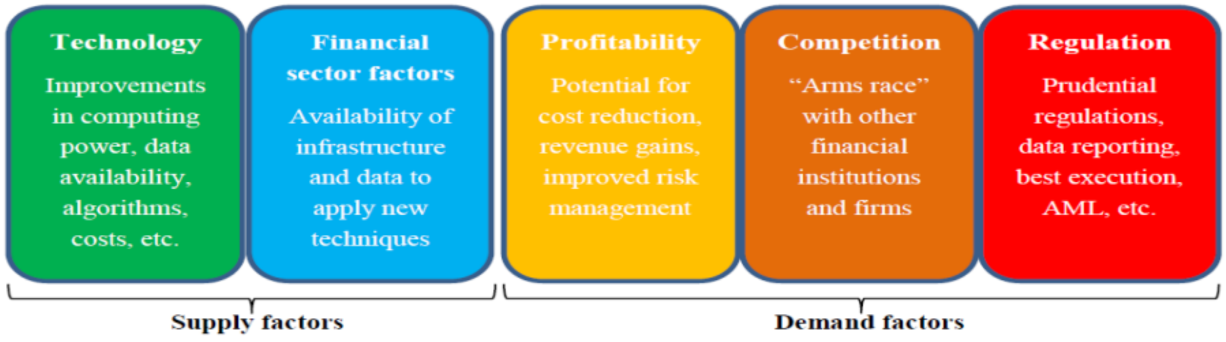
Technology
Improvements in computing power, data availability, algorithms, costs, etc.
Financial sector factors
Availability of infrastructure and data to apply new techniques
Profitability
Potential for cost reduction, revenue gains, improved risk management
Competition
"Arms race" with other financial institutions and firms
Regulation
Prudential regulations, data reporting, best execution, AML, etc.
Supply factors
Demand factors

Fig. 1. AI and Machine Learning Factors for Financial Adoption [1].

- They discuss various applications of ML in the banking sector, like fraud detection using SVM, RF, etc
- Here also, they use the Santander Bank dataset from Kaggle. They did all the implementations using python 3.7.1 on anaconda3 & Jupiter notebook 5.7.4. They took all the algorithms from sci-kit learn.

- During data preprocessing, they found the dataset to be clean with no missing values or noise. All attributes have floating-point values.
- They changed the data type from float64 to float32 to reduce time and memory cost (50% reduction in space occupied).
- During EDA, they explore - min, max, mean, standard deviation, skewness and kurtosis, distribution of train and test dataset (turned out to be identical). They found the data to be imbalanced as well.
- They found no correlation among feature variables.
- They used var_81 as root node to build decision tree, as it was the most important feature. Through random forest, they found var_110 was the most useful feature and var_86 least important feature.
- Since PCA works only if there is redundancy in data, it failed to separate class 0 and class 1 label data points. Similar to PCA, they try another technique called LDA - linear discriminant analysis, which is like PCA, except it tackles over-fitting. But, even this technique cannot perform well.
- Because of class imbalance, they use AUC and confusion matrix for evaluation instead of accuracy.
- Every classifier cannot perform best on every type of dataset. It does not mean a classifier outperforming others on a dataset that this classifier can outperform the others on any other datasets too. So they have tried many algorithms here - Logistic Regression, Random Forests, Decision Tree, Multi layer perceptrons, Gradient boosting (XGBoost, CatBoost, LightBGM).
- LightBGM gave remarkably higher AUC than all other classifiers. Also, as the predictions did not classify the data in 0’s or 1’s, instead, it gave the floating-point. After careful analysis, they chose a threshold and converted the predictions to classes 0’s and 1’s and out the precision, accuracy, recall, and f1 score, but the AUC dropped to 0.761222. The least performing model was random forests.

3. URL: https://www.kaggle.com/code/yag320/list-of-fake-samples-and-public-private-lb-split/notebook

List of Fake Samples and Public/Private LB split

- They found that the test set had both unique data points (real samples) and data points sampled from the real samples. They count the number of real and synthetic samples and extract magic features from them.
- These magic features were of many types. One was multiplying the column values with the count got above.

First Cut Approach

*** Explain in steps about how you want to approach this problem and the initial experiments that you want to do. ***(MINIMUM 200 words)*** ***

*** When you are doing the basic EDA and building the First Cut Approach you should not refer any blogs or papers ***

Data Source:
https://www.kaggle.com/competitions/santander-customer-transaction-prediction/data

1. Exploratory data analysis
    - Since they have given anonymized dataset with mostly numerical columns, I cannot come up with much questions to find answers by exploring the data.
    - I want to check the distribution of target variables. Already from the research papers above, I know the dataset is imbalanced; I want to try both over-sampling and under-sampling and see which gives better performance.
    - Check for missing values, duplicate rows and outliers.
    - Check for correlation of features with target variables.
    - Check basic statistics - like mean, median, standard deviation, min, max, kurtosis, percentile, skewness, etc.
    - Observe distribution of features - histograms, box plots or violin plots.
    - Check the data-type of all features.
    - Plot the dataset, colored according to their class labels. Use t-SNE for visualization.
2. Performance metrics: AUC & Confusion Matrix Accuracy is not suitable since the dataset is highly imbalanced and the TPR should be as high as possible. The model should correctly predict if a customer makes a transaction, so these two metrics are good to monitor.
3. Splitting of data into train and cross-validation: I would like to use stratified k fold (better than random k fold since data is imbalanced).

4. The first check with numeric data is to determine if magnitude matters. If there is a sizeable difference in magnitude of features, I want to perform standardization as models like logistic regression are sensitive to scales of features but it does not affect tree based algorithms like decision tree, gradient boosting, etc.

(for feature engineering, I referred to these two books:

➔ *Feature engineering and selection by Max Kuhn and Kjell Johnson*
➔ *Feature Engineering for Machine Learning by Alice Zheng & Amanda Casari)*

5. Feature Engineering - There are over 200 numeric features.
    - If distributions of these features are heavy tailed, will apply log transform or inverse values of these features (as models like neural networks are sensitive to skewed distribution)
    - Using K-means as a featurization technique - Here, we incorporate the target variable along with other input features before passing it to K-means model. We can scale the target values to get more or less attention from the algorithm. Input data augmented with k-means cluster features can then be passed to our models, like Logistic Regression, etc. There is a potential for data leakage here and to avoid this, only a small part of data can learn clusters, which will not be part of training the model. (Don't know if this is useful, just want to experiment this technique and see how it works)
    - Some other simple techniques - taking minimum, maximum, mean values from each row as new feature value.
    - Other techniques that I came across were using polynomial features and interaction variables, but these are more useful for linear models like logistic regression than tree-based algorithms, also they will increase the number of features drastically. So, I will keep them as my last choice in case of other techniques don't work.

6. Feature Selection:
    - Removing all low variance features
    - Removing highly correlated features
    - Fitting the model and selecting the best features based on model coefficients.
    - Choose features from one model and use another model to train, sklearn provides *SelectFromModel* for this.
    - Univariate feature selection: scoring of each feature against the target. Two ways to do this using sklearn - *SelectKBest & SelectPercentile*
    - I will keep greedy feature selection as the last choice as this will take a lot of time and also chances of over-fitting with this technique are high.
7. Model Implementation (along with regularization to avoid over-fitting) and Hyper-parameter optimization:
    - Logistic Regression
    - Tree based algorithms - Decision Trees, Random Forests. Gradient boosting.
    - Neural Networks
    - RandomSearch or hand-tuning.
8. Model deployment

Notes when you build your final notebook:

1. You should not train any model either it can be a ML model or DL model or Countvectorizer or even simple Standard scalar
2. You should not read train data files
3. The function1 takes only one argument "X" (a single data points i.e 1*d feature) and the inside the function you will preprocess data point similar to the process you did while you featurize your train data
    a. Ex: consider you are doing taxi demand prediction case study (problem definition: given a time and location predict the number of pickups that can happen)
    b. so in your final notebook, you need to pass only those two values
    c. def final(X):
    ```
        preprocess data i.e data cleaning, filling missing values etc
        compute features based on this X
        use pre trained model
        return predicted outputs
    final([time, location])
    ```
    d. in the instructions, we have mentioned two functions one with original values and one without it
    e. final([time, location])   # in this function you need to return the predictions, no need to compute the metric
    f. final(set of [time, location] values, corresponding Y values) # when you pass the Y values, we can compute the error metric(Y, y_predict)
4. After you have preprocessed the data point you will featurize it, with the help of trained vectorizers or methods you have followed for your train data
5. Assume this function is like you are productionizing the best model you have built, you need to measure the time for predicting and report the time. Make sure you keep the time as low as possible
6. Check this live session: https://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4148/hands-on-live-session-deploy-an-ml-model-using-apis-on-aws/5/module-5-feature-engineering-productionization-and-deployment-of-ml-models