COMP9417 Project:

TracHack Challenge 22.2

**- Predicting Eligibility for the Emergency Broadband Benefit Program**

Group name: Machine Intelligence Unit

Members:

* Danver Zhao (z5317086)
* Fiona Wang (z5309988)
* Syed Hamza Warisi (z5259749)
* Mohammed Musa (z5284114)
* Arun Kumar Marndi (z5225111)

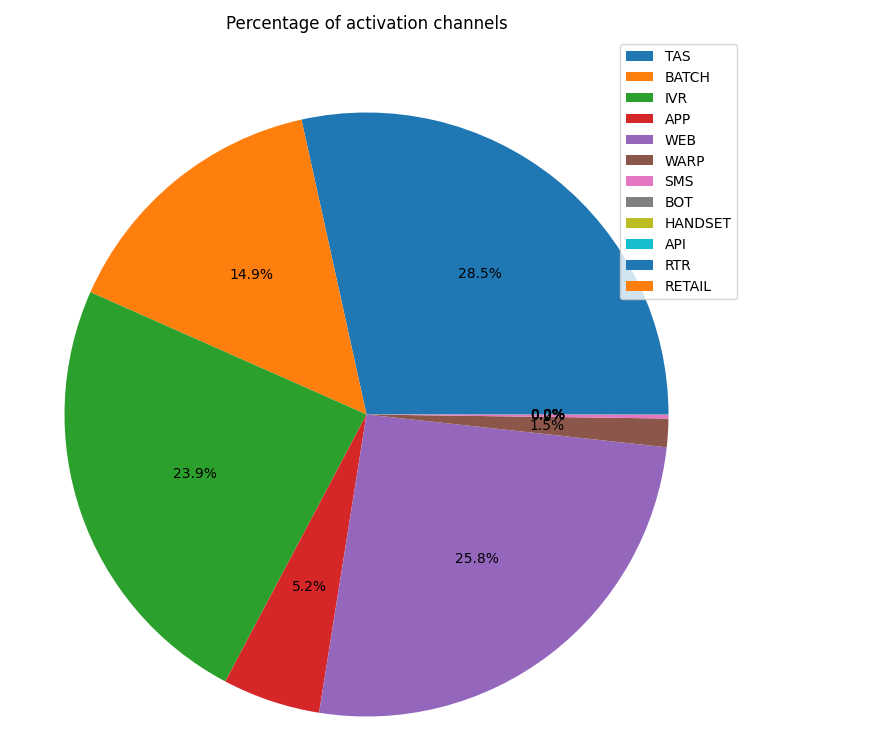
**1.1 Introduction**:

TracFone Wireless has the primary aim of providing phone coverage and broadband access for all. As part of the Consolidated Appropriations Act, the government established the Emergency Broadband Connectivity Fund aimed to help a dedicated subset of individuals afford internet services during the pandemic. The funds were to be used as part of the Emergency Broadband Connectivity Program (EBB Program) whereby eligible low-income households may be provided products and services at a discounted price.

The TracHack 22.2 challenge requires participants to use machine learning to make predictions on customers to determine if they are eligible for the Emergency Broadband Benefits Program. This challenge is a one-class classification problem, as the data being used for this problem is a combination of positively labeled and unlabeled data.

Data processing and cleansing were performed, followed by an exploration of four different one-class classification approaches. The best method was found to be Modified Logistic Regression, and this assessment was determined based on F1 scores of our predictions. Additional feature selection and hyperparameter tuning were executed to finetune the model.

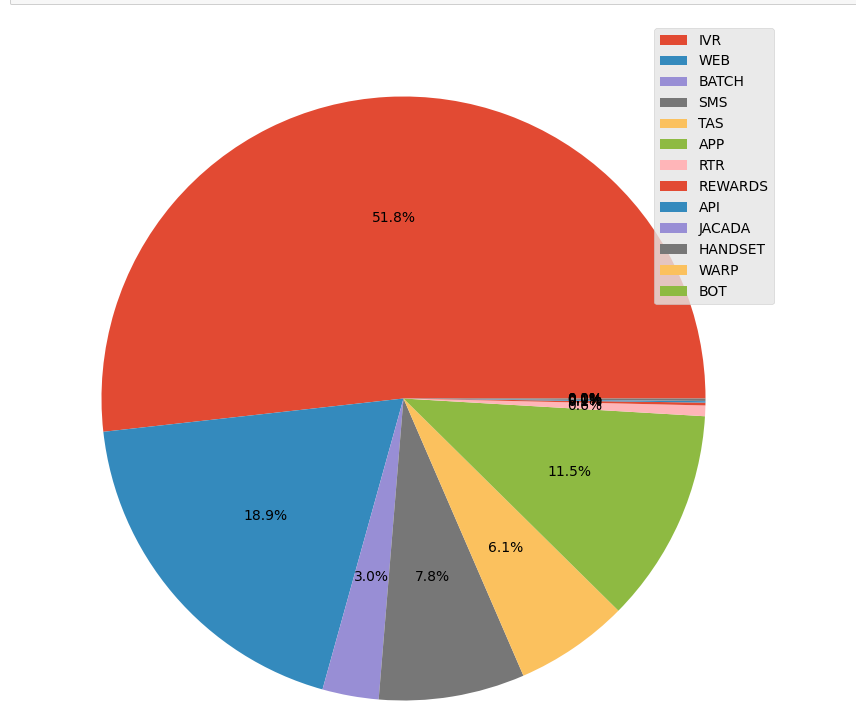
**2.1 Exploratory Data Analysis**:

Exploratory data analysis was performed to better understand the data at hand. Attempts were made to identify relevant features for making predictions on EBB eligibility

***Activation Channels:***

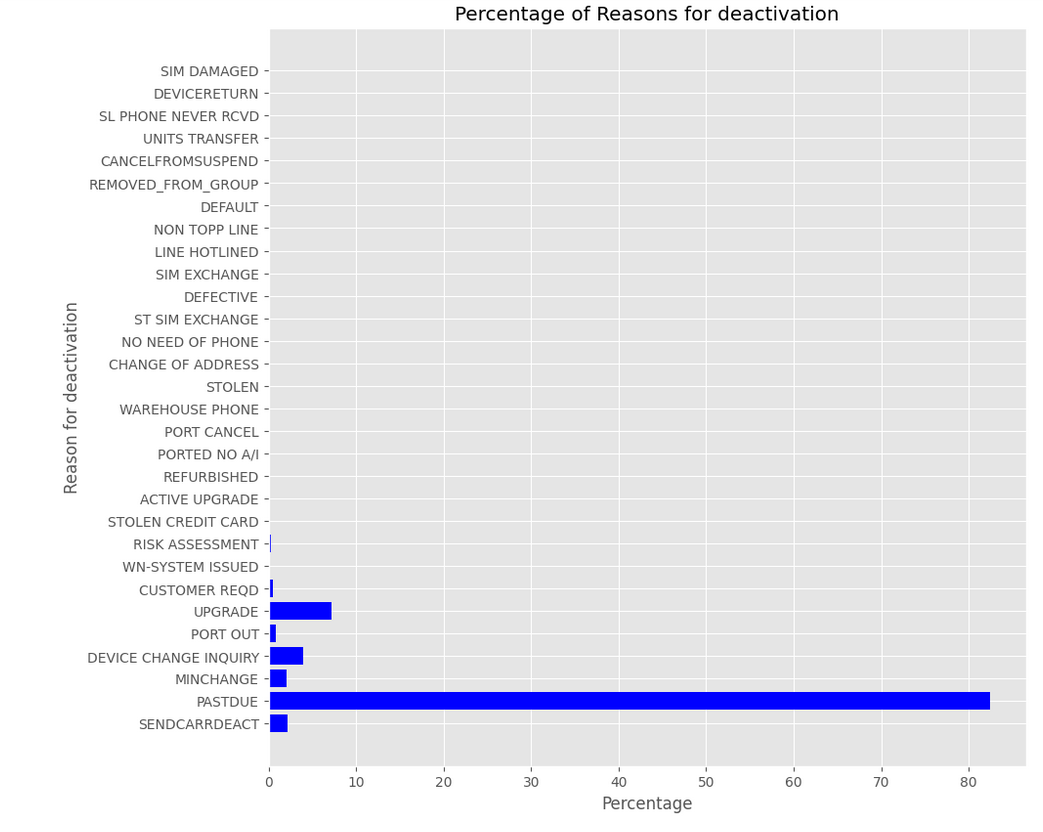
We first look into the activation data set that consists of the customers who use various activation channels. Below is the pie chart plot for various activation channels.

It can be seen that 28.5% from TAS, 25.8% of the activation request came from the web channel,23.9% from IVR(Interactive Voice Response),14.9% from retail and 5.2%from application.



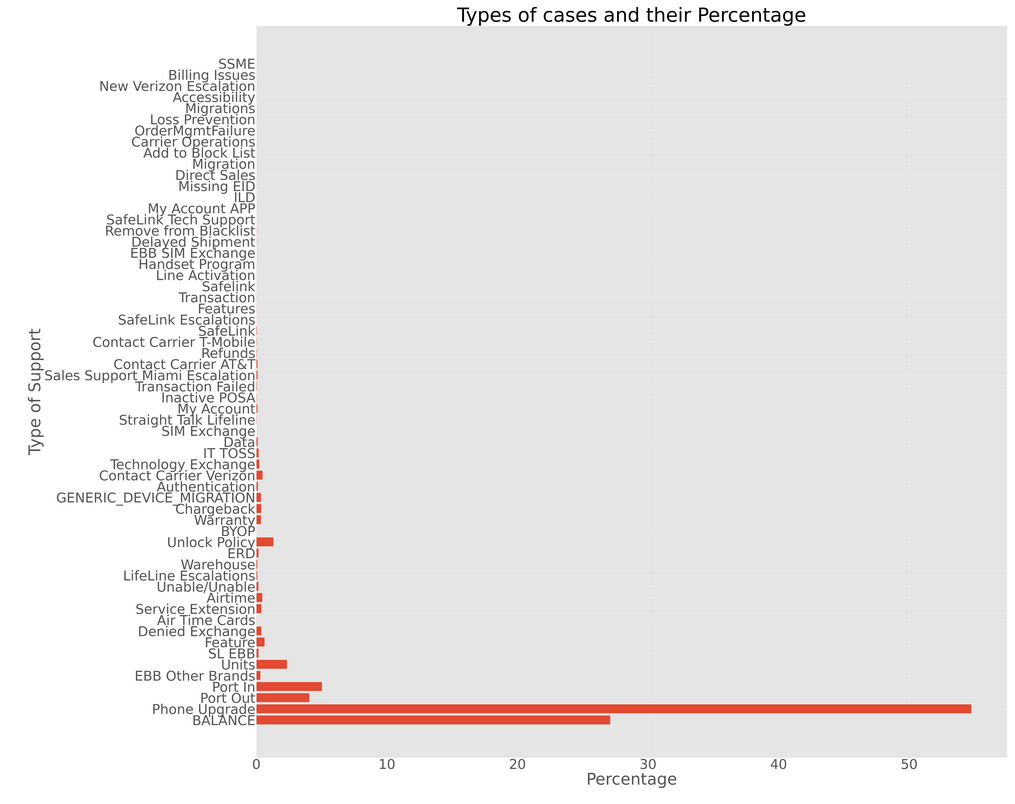
***Percentage of Reactivations***

51.8% of Reactivations came from IVR,18.9% from the company’s website, and the remaining percentages from others.



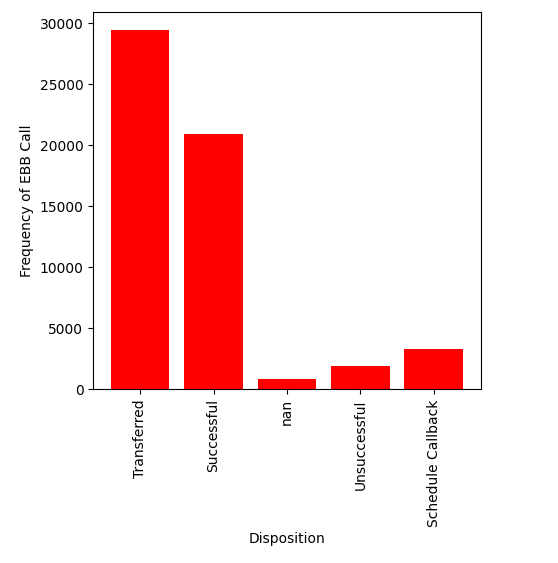
***Percentage of Reasons for Deactivation***

The major reason customers are not paying their rent is that they can’t pay their bills past the due date. This could be one of the contributing factors that they were not able to pay their bills on time or they came from financially weaker sections of the society. Other reasons included SendCardDeactivation and others.

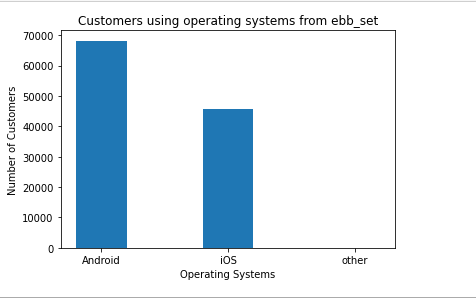


***Call support Percentages***

The major reason for contacting the support centers mostly consisted of Phone Upgrades, Balance, and others.

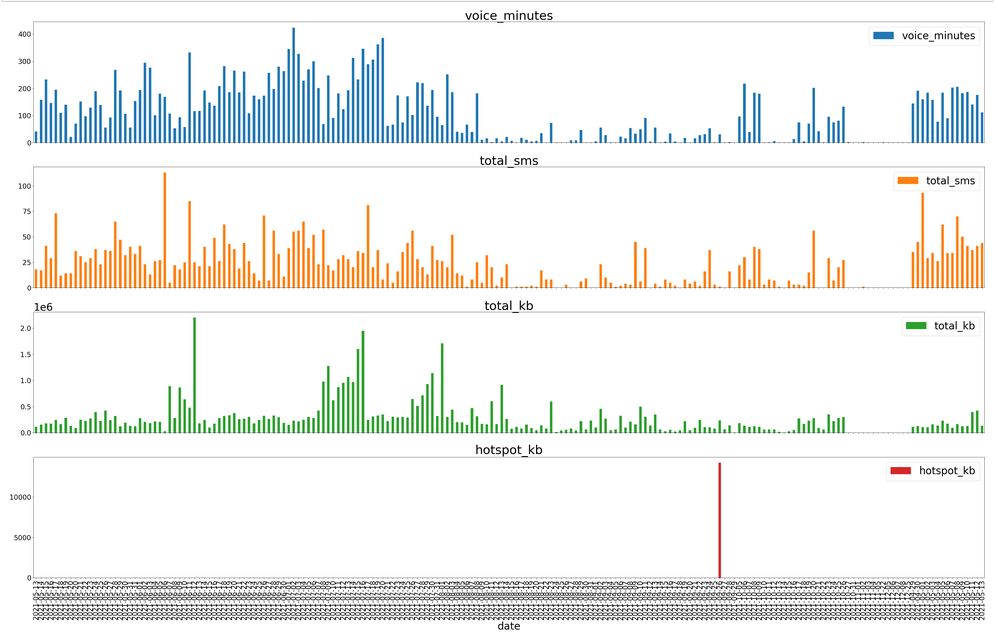
***Disposition of the Percentage of Calls for Emergency Broadband Benefit Plan***

Based on the interactions of the customers, of all the reasons for interaction, the main focus was on “Emergency Broadband Benefit Plan”. The majority of the calls were transferred or were successful. This indicates that the majority of the customers were in fact interested in the Emergency Broadband Benefit Plan.



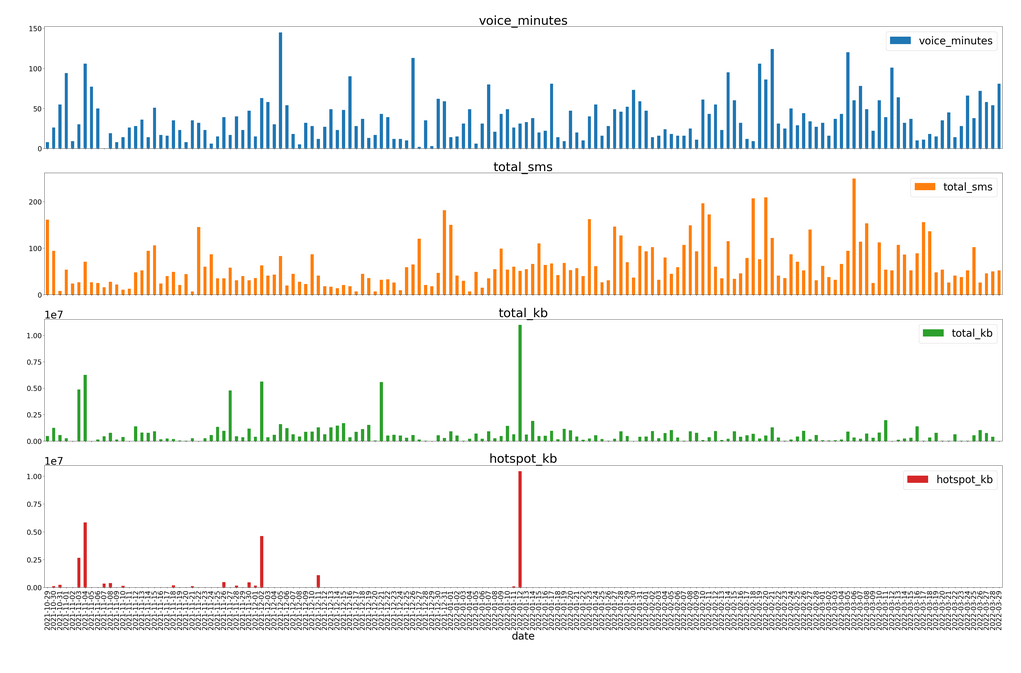
***Operating Systems Used by the Customer***

There were more customers using the Android Operating System than those using iOS. This could also be one of the indicators that Android phones are relatively cheaper and could be indicating that customers could be from low-income backgrounds.



***Time Series Visualization of Network data Customer Using the Tracfone network***

It can be observed that the customer rarely used a hot spot. However, he used the phone data and voice minutes quite frequently.

***Time Series Visualization of Customer’s Phone Data***

The customer used a lot of phone data and frequently used voice call services.

**3 Methodology:**

**3.1 Data pre-processing:**

For all datasets, categorical values are transformed into numerical values by one-hot encoding and multi hot encoding depending on the features, for example, manufacturer, operating system, and state use one-hot encoding, whereas for support feature where customers can have multiple categories each multi hot encoding was used to better represent the data.

All string dates such as the last redemption date need to be turned into numerical values as well, this can be done by first turning it into a DateTime object and then into a timestamp.

New features are also created to better summarize the compressed dataset, where there exist multiple rows with the same customer\_id. For example, after compressing all the same customer\_id into one in the activations datasets, a new feature called “number of activations” was created to replace the deleted data. For some datasets like suspensions, which only have 2 features - a start date and an end date. A new feature - total\_suspension\_time - was created as the original data was not relevant or able to represent the income of that individual. Relevant features from other .csv files were joined to the corresponding main files (ebb\_set1.csv, ebb\_set2.csv, and eval\_set.csv) to produce three large .csv files containing all relevant data used to train models.

**3.2 Isolation Forest**

Given that the data provided for this project consists of a combination of positively labeled and unlabelled data, research was taken into the more popular approaches and algorithms designed for one-class classification problems. Certain approaches take the stance that the data points given fit a particular distribution or pattern, and these constitute one class (such as the positive class), whilst any outliers or anomalies of this pattern are then considered as the other class (such as the negative class).

One such method that is commonly used for outlier detection is known as Isolation Forest, which utilizes tree structures. In decision trees, nodes that are further away from the root would have required more steps and separations to be carried out in order to be finalized, than nodes that are placed very close to the root. Nodes that require more separation are therefore more in alignment with the majority of the data, and so these can be considered as inliers, whilst nodes that are situated closer to the root are far easier to separate from the general distribution of the data, and can be seen as outliers.

There are a few main considerations to be made when dealing with the Isolation Forest algorithm. One of the most crucial hyperparameters for this algorithm is the level of contamination, which is essentially the expected level of outliers to inliers in our data. It is preferred that there is some level of pre-existing knowledge of the distribution, though this can also be determined through repeated evaluation and testing.

It is most optimal to train the model on data that excludes any outliers, and so the model was trained on only the positive labeled data first. The model was fitted to two datasets - one being the original eval\_set, with the other being applied to an aggregated set containing multiple features obtained from various other given datasets. Once the algorithm was fitted to the training data, it was used to predict any outliers based on varying contamination rates from 0.1 to 0.5, with our best performance being obtained at a contamination rate of 0.1, as determined by the given F1 score.

**3.3 OneClass SVM**

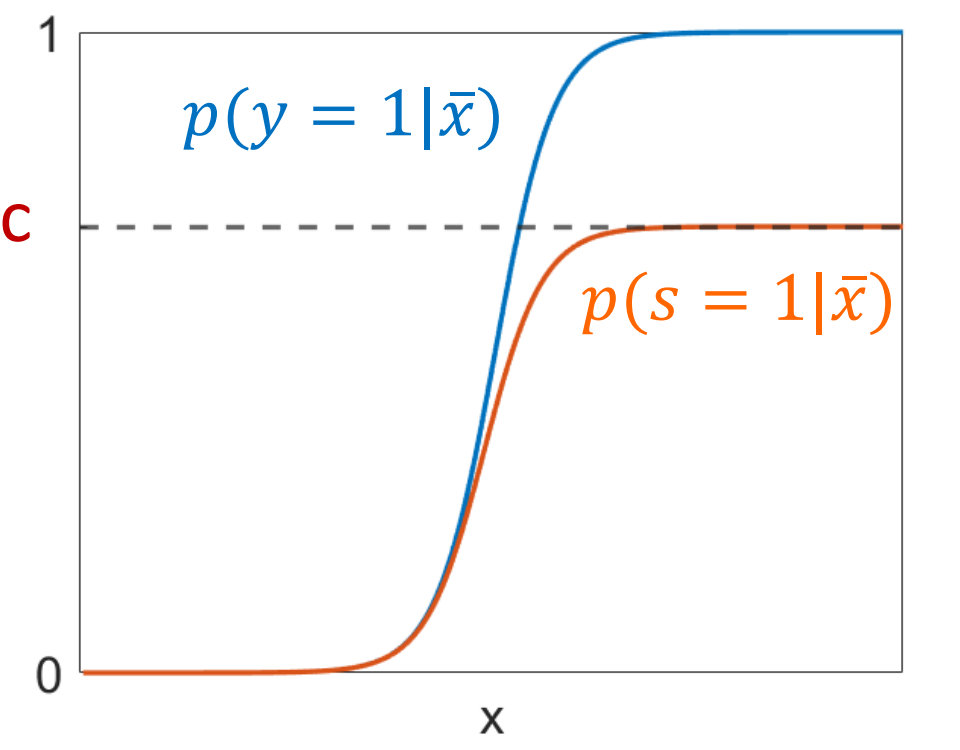
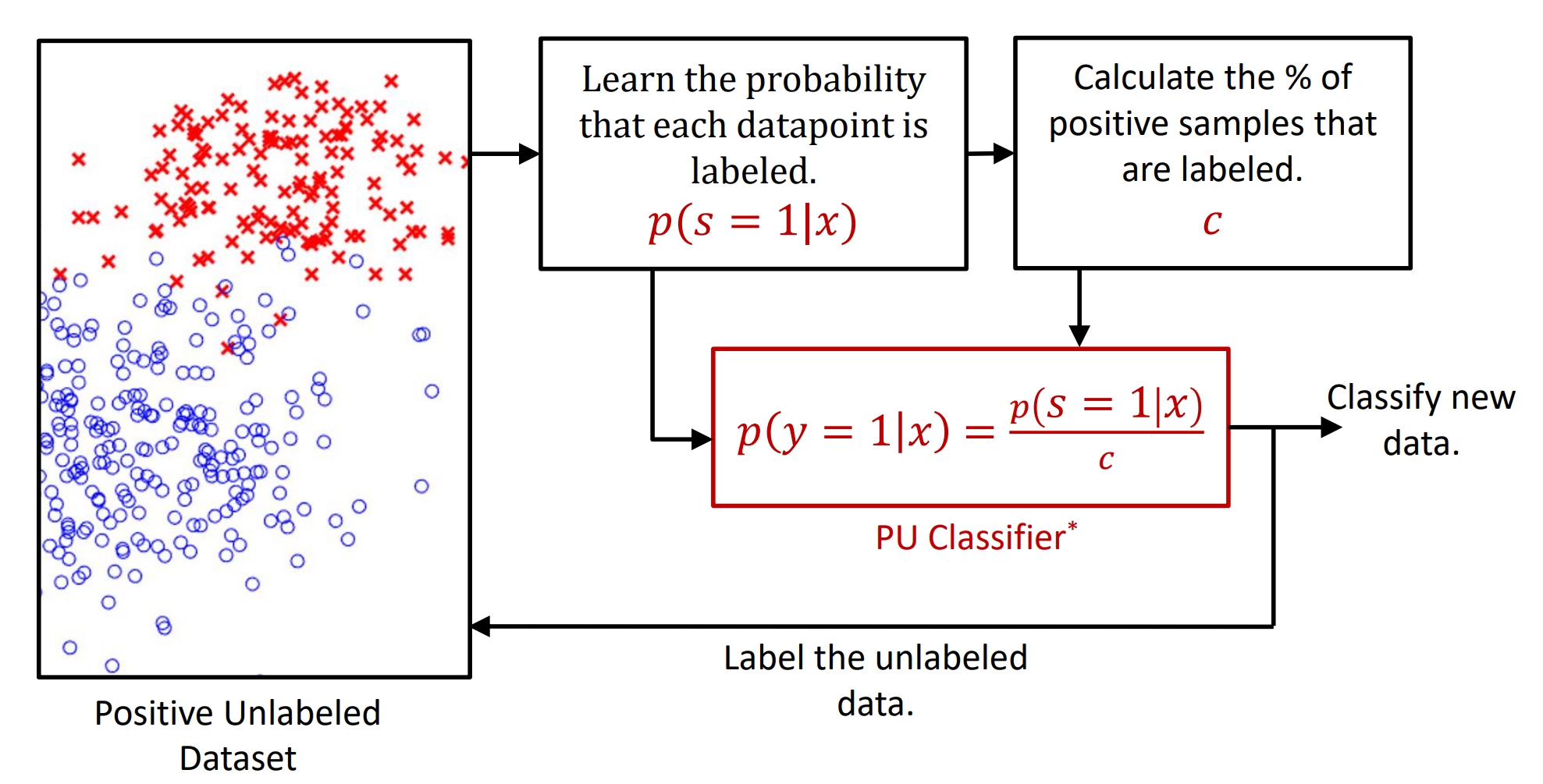
As stated before, the data for the TracHack competition only contains positively labeled data making it a one-class classification or PU problem. There aren’t many models which can deal with one class classification and the model which appeared the most during some exploration was OneClassSVM, the model was easy to implement, just fit the training data, having a relatively small amount of arguments to adjust from, therefore it looks like a great place to start off for experimenting with models.

For feature selection, the features were selected by inspection and intuition. For example, in the TracHack description, the EBB program was supposed to help low-income individuals to have a stable use of network service. Therefore features that related to a person’s status were chosen for example the total time of suspensions shows how stable a person’s income was to be able to maintain buying redemptions.

**3.4 Linear Discriminant Analysis**

Linear Discriminant Analysis was a model that was considered in getting an optimal result in the challenge. This model has several uses such as classification, dimension reduction, and data visualization. It often produces robust, decent, and interpretable classification results. It is usually a benchmark method for real-world classification problems before other more complicated and flexible ones are employed. The approach of LDA has some assumptions of interest which are; multivariate normality, homogeneity of variance/covariance, multicollinearity and independence.(further reading <https://en.wikipedia.org/wiki/Linear_discriminant_analysis>). The performance obtained showed that LDA was not a suitable model for handling a one-classification problem.

**3.5 Modified Logistic Regression**



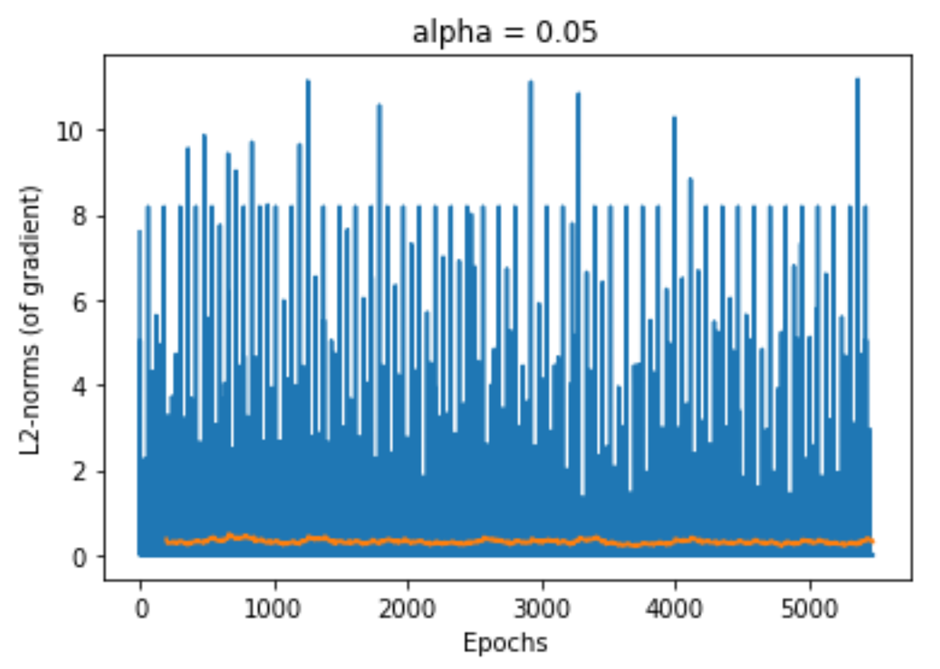
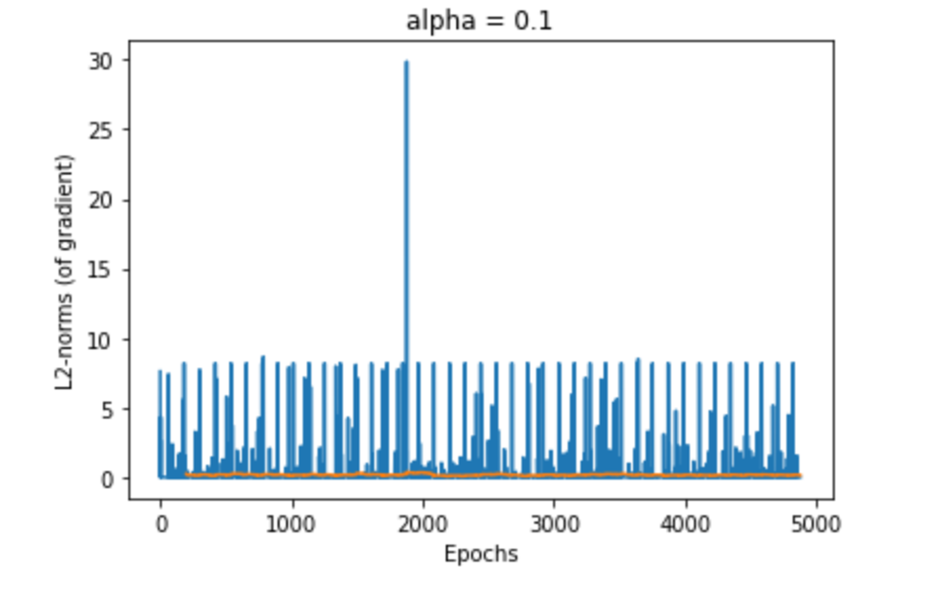
This Modified Logistic Regression method is a non-traditional positive-unlabeled classifier. It combines traditional Logistic Regression with a probabilistic approach to estimate parameter ***C***, which represents the percentage of positive data that are labeled. This parameter is then used to scale the sigmoid curve, which allows the model to estimate the probability of a data point belonging to the positive class, which in turn provides insight into how many data points belong to the negative class.

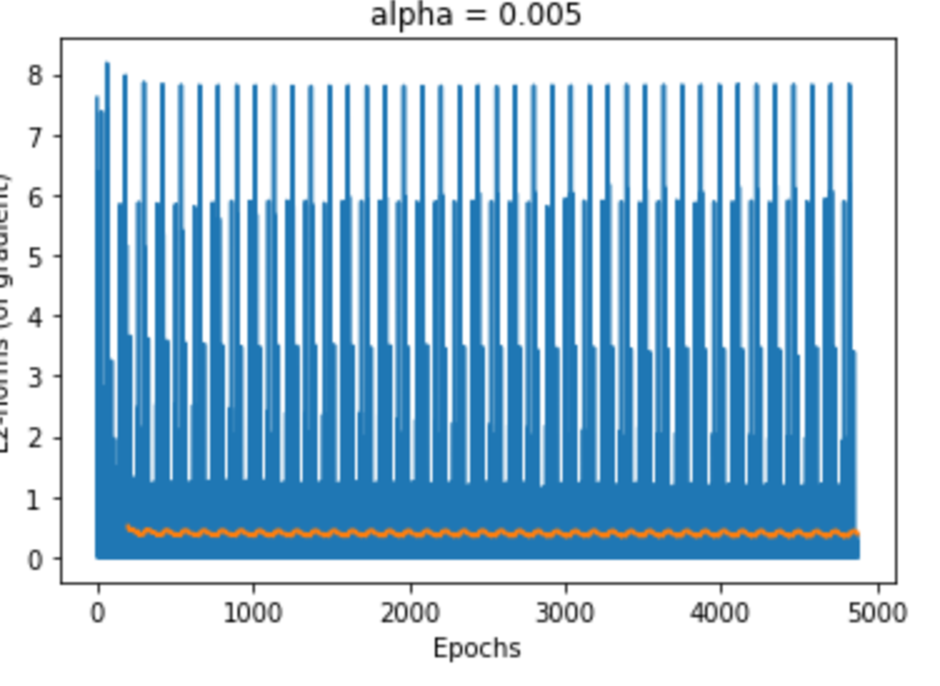
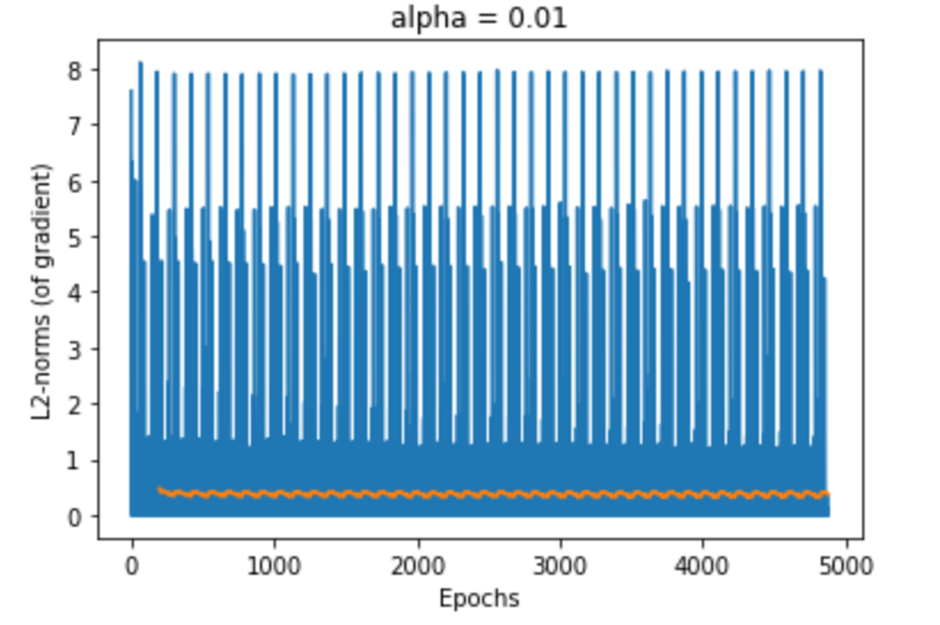
This model was considered because it has been designed specifically for positive-unlabeled datasets, just like the TracHack dataset that was provided. It has also shown promising results on a subset of the MINST dataset, where the model was able to classify 3s and 5s with an F1 score of over 0.96.

The main consideration to be made with this method is that it requires the assumption that at least partial separation of the positive and negative classes exists. The evaluation of the result produced by this model proved that the data provided was indeed separable, and so this was not an issue during the analysis and execution.

Preliminary exploration was performed on this method, using only 3 features (revenue bucket, tenure, number of upgrades), which resulted in an F1 score of 0.66. Upon the addition of state and manufacturer data, the F1 score increased to 0.82.

Hyperparameter tuning was performed to select the learning rate (i.e. alpha).





The graphs above show the changes in the l2-norm of the gradient throughout the process of training the model. Note that the algorithm updates the weights at each individual data point, hence the gradients found will be stochastic in nature due to the existence of outliers. As seen in the graphs above, alpha values of 0.1 and 0.05 gave relatively unstable gradients, while the graph for alpha = 0.01 and alpha = 0.005 was relatively stable and also similar. Hence 0.01 was chosen as the learning rate, given its stability and reduced iterations to reach convergence.

There were more than 200 features after data processing. Some features were already excluded manually due to their apparent lack of relevance. However, in addition to this, a feature selection algorithm was performed. Feature selection in one-class classification models is challenging, as there are no ground truth labels for the negative class. However, the Modified Logistic Regression method was able to achieve an F1 score of 0.96, and the output of this model was treated as ground truth for feature selection. Using sklearn.linear\_model.LogisticRegression with L1 regularization, features with corresponding weights of 0 were identified and removed (see index for result).

Ultimately, applying feature selection had minimal impact on the F1 score of the models.

**4 Results**

**4.1 Isolation Forest**

The best F1 Score that was achieved by this model was 0.661336. This score was achieved when the Isolation Forest algorithm was fitted and executed on the original eval\_set provided. The chosen hyperparameters for this specific model are given by n\_iterations = 300, and contamination = 0.01.

With the addition of more features, along with a much higher contamination of 0.45, the F1 score had reduced to 0.343165. There are two main reasons behind the relatively poor results obtained above. Due to a lack of pre-defined knowledge on the distribution of the negative class to the positive class, determining the appropriate value for the contamination was a challenge. Secondly, the involvement of various features without extensive cross-validation and hyperparameter tuning ultimately meant that the algorithm may pick up on outliers based on certain features that have little influence or association with the eligibility of customers.

**4.2 OneClass SVM**

| OneClass SVM arguments | F1 score |
| --- | --- |
| kernel = rbf, fit all data in set1 and set2 | 0.657053 |
| kernel = rbf, fit only set1 labeled | 0.642208 |
| kernel = sigmoid | 0.556104 |
| kernel = linear | 0.561175 |
| kernel = poly, degree = 1 | 0.561175 |
| kernel = poly, degree = 4 | 0.439869 |
| kernel = poly, degree = 8 | 0.427811 |
| kernel = poly, degree = 10 | 0.425539 |
| kernel = poly, degree = 12 | 0.423901 |

OneClass SVM scored a mediocre score of 0.65, a reason why OneClass SVM might be hard to improve can be in the documentation which said it is used for novelty detection, the idea of novelty detection is to detect rare events. However, it can't be determined if the class of ebb-eligible = 0 is a rare instance. Therefore the OneClass SVM might have misclassified more data to ebb-eligible = 1.

Some insights into the OneClassSVM are that kernel = rbf performs the best out of {linear, poly, rbf, sigmoid} and is the default kernel. The kernel = poly performance gets worse as the degree increases. Also by increasing the number of training data it doesn't actually improve the model that much. The model with F1 score of 0.642208 used 24000 positive labeled data and the model with F1 score of 0.657053 used 49000 positive labeled data and 50000 mixed unlabeled data. By increasing the amount of data by 4 times the F1 score only improved by 0.01. This could mean the OneClassSVM is really good at determining the decision boundary with few data points, so more data won't affect the decision much. This also means that the number of labeled data doesn't matter as well, the OneClassSVM only needs a few to determine which group is the outlier so that it can output the prediction value correctly.

**4.3 Linear Discriminant Analysis**

The F1 score obtained was 0.54, This was a poor result compared to the previous models that were tested. This can be attributed to the method the algorithm was applied to predict the data and also Linear Discriminant Analysis is not an ideal model for a one-classification problem. Perhaps further fine-tuning of the hyperparameters would have given a better result but the optimum results will not be achieved.

**4.4 Modified Logistic Regression**

All Modified Logistic Regression Models were trained on the entire dataset of all customers (ebb\_set1, ebb\_set2, eval set). An additional parameter ***S*** was used, which equals 1 if a data point is labeled positive, and 0 if unlabeled.

Using 100 iterations through the whole dataset gives the following results.

| Alpha | F1 Score |
| --- | --- |
| 0.005 | 0.908076 |
| 0.01 | 0.961871 |
| 0.05 | 0.823567 |

At an Alpha rate of 0.01, the model was also tested with a different number of iterations through the dataset.

| Iterations | F1 Score |
| --- | --- |
| 100 | 0.961871 |
| 300 | 0.959069 |

Increasing the number of iterations from 100 to 300 did not improve the F1 score of the model. Hence 100 iterations with alpha = 0.01 gave the optimal results for Modified Logistic Regression.

**5.1 Discussion**:

Out of the 4 methods explored, Modified Logistic Regression gave the best results, with an F1 score of 0.96. Isolation Forest achieved a score of 0.66, One-Class SVM achieved a score of 0.65, and Linear Discriminate Analysis achieved a score of 0.54. Consequently, the Modified Logistic Regression method was chosen for the final submission due to its higher performance.

The relatively poor performance of the other models can somewhat be attributed to the lack of understanding of the distribution of data. These models may also be better suited to regular classification problems and lack suitability for one-class classification problems. In contrast, Modified Logistic Regression accounts for the lack of understanding of the distribution of classes and attempts to accommodate for that in the model through the parameter **C**. The probabilist approach was effective in estimating this parameter, thus allowing the model to produce optimal results.

The only metrics used for evaluation were the F1 scores provided by the TracHack team through daily submissions. It is an appropriate metric for the problem, as it balances the importance of both recall and precision. The evaluation of models using any other metrics was not performed, as the dataset consists of a significant amount of unlabeled data. This prohibited the splitting of data into training and testing set, and the use of testing set to evaluate other metrics, such as accuracy.

If more time was given, data processing could be made more efficient with more preliminary data analysis. For example, 99% of the customers use either IOS or Android as their operating system. It is possible to just create 3 features IOS, Android, and others instead of 74 features created using one-hot encoding which will help the model to be more efficient. A more effective encoding method could be implemented to better represent the data of multiple categories. The best encoding method is multi-hot encoding however it does not represent the number of times one category has appeared.

Daily submissions can be better utilized to test different model parameters options, as on someday 10 submissions are made and only 1 or 2 are made on others.

These modifications could potentially improve the performance of all models. If there were more than one model with good performance, ensemble methods such as bagging and boosting could be used to create a model with optimal performance.

**6.1 Conclusion:**

The TracHack challenge required participants to make predictions on the eligibility of customers for the EBB program. Positive unlabeled data were given, which presented a one-class classification problem. Data processing was performed after the initial data analysis. Four machine learning models were explored and fine-tuned through hyper-parameter tuning and features selection. Modified Logistic Regression was chosen as the final model due to its high performance, and an F1 score of 0.96. The performance of all models were analysed, and possible improvements were stated for future exploration.

**7.1 References**:

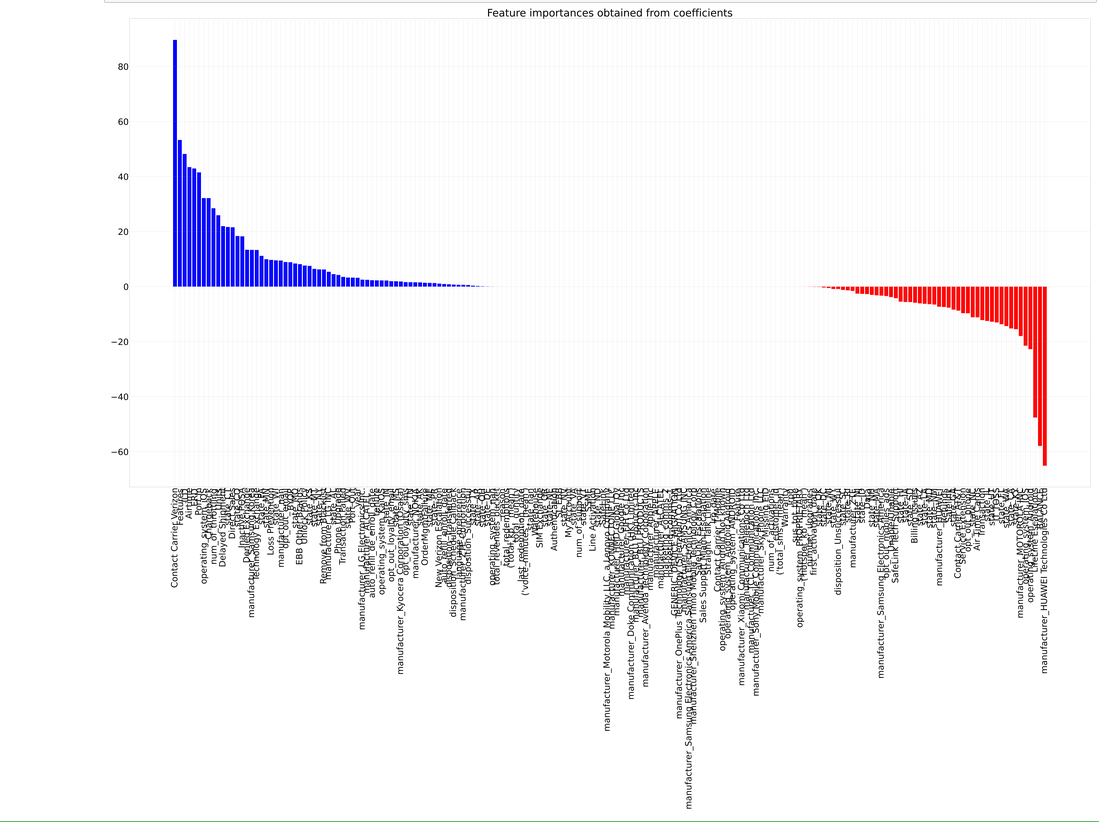
* Brownlee, J. (2020). One-Class Classification Algorithms for Imbalanced Datasets. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/one-class-classification-algorithms/> [Accessed 20 April 2022].
* Krishnan (2019). Anomaly Detection with Isolation Forest & Visualization. [online] Medium. Available at: <https://towardsdatascience.com/anomaly-detection-with-isolation-forest-visualization-23cd75c281e2> [Accessed 20 April 2022].
* Paperspace Blog. (2020). Anomaly Detection Using Isolation Forest in Python. [online] Available at: <https://blog.paperspace.com/anomaly-detection-isolation-forest/> [Accessed 20 April 2022].
* Yang Xiaozhuo (2020). Linear Discriminant Analysis, Explained. [online]

Available at:

<https://towardsdatascience.com/linear-discriminant-analysis-explained-f88be6c1e00b> [Accessed 20 April 2022].

* Jaskie, K., Elken, C. and Spanias, A. (2019). *A Modified Logistic Regression for Positive and Unlabeled Learning*. [online] Research Gate. Available at: <https://www.researchgate.net/publication/340306999_A_Modified_Logistic_Regression_for_Positive_and_Unlabeled_Learning> [Accessed 20 April 2022].

**8.1 Appendix**:

****