Capstone Project
Email Campaign Effectiveness Prediction

By

Umesh, Ifraz, Shailendra, Akram

AGENDA

Problem Statement

Objective of Project:

- Most of the small to medium business owners are making effective use of Gmail-based Email marketing Strategies for offline targeting of converting their prospective customers into leads so that they stay with them in Business.

- The main objective is to create a machine learning model to characterize the mail and track the mail that is ignored; read; acknowledged by the reader.

- Develop the perceiving of various parameters involved in email campaigns.

Data Summary

The total no. of records in our dataset is 68353, There are 12 features in the given dataset, Email Status is target Variable.

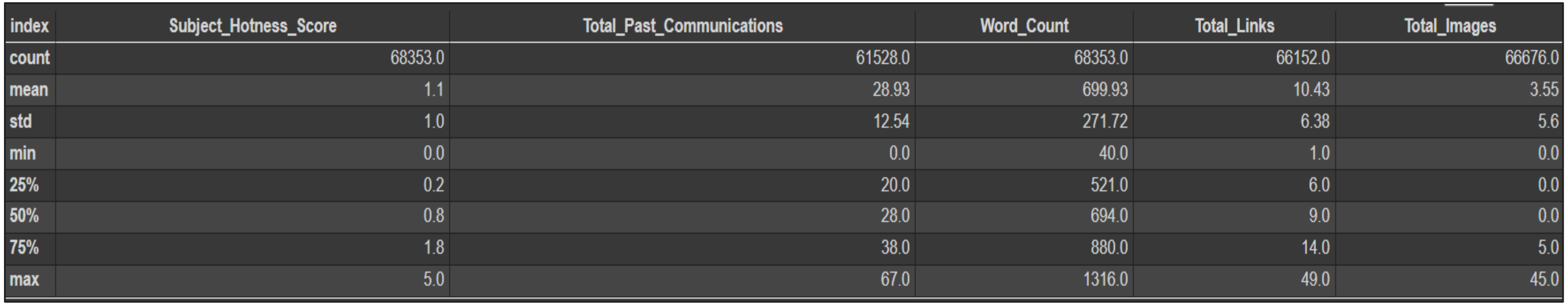
The 5 numerical variables were:

1. Word Count
2. Total Past Communications
3. Subject Hotness Score
4. Total Images
5. Total Links

The 7 categorical variables were:

1. Email Id
2. Email Type
3. Email Source Type
4. Customer Location
5. Email Campaign Type
6. Time Email Sent Category
7. Email Status

Descriptive Statistical Analysis



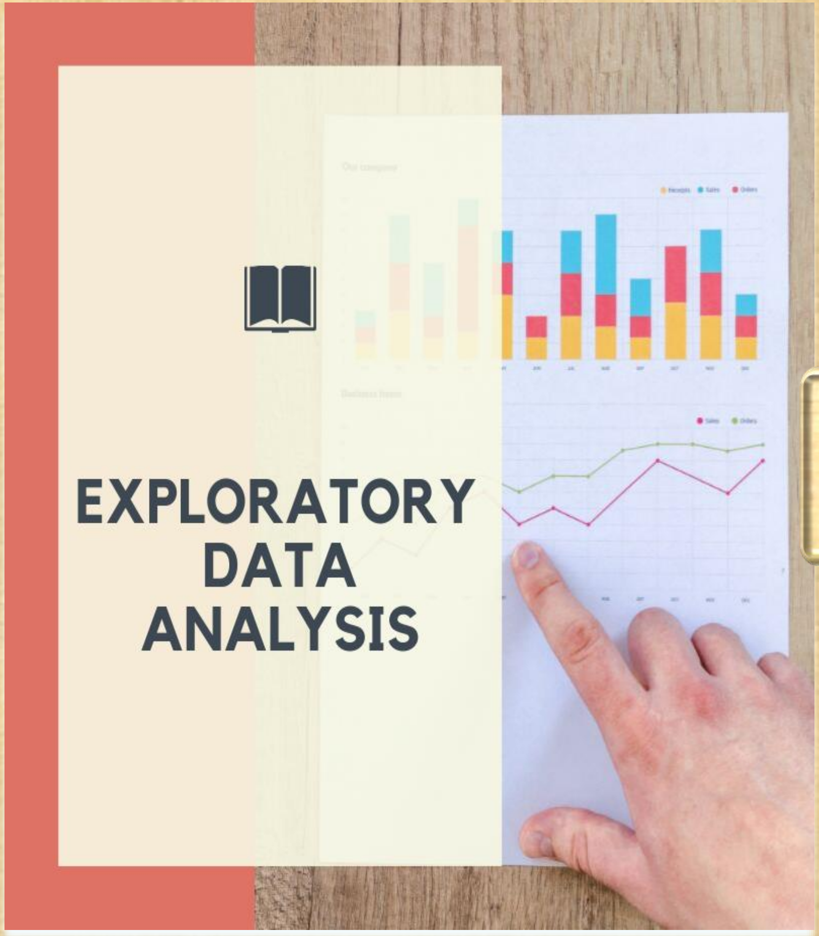
index	Subject_Hotness_Score	Total_Past_Communications	Word_Count	Total_Links	Total_Images
count	68353.0	61528.0	68353.0	66152.0	66676.0
mean	1.1	28.93	699.93	10.43	3.55
std	1.0	12.54	271.72	6.38	5.6
min	0.0	0.0	40.0	1.0	0.0
25%	0.2	20.0	521.0	6.0	0.0
50%	0.8	28.0	694.0	9.0	0.0
75%	1.8	38.0	880.0	14.0	5.0
max	5.0	67.0	1316.0	49.0	45.0

Fig 1: Descriptive Statistical Analysis (Numerical Features)

- The average subject hotness score for the given data set is around 1.10

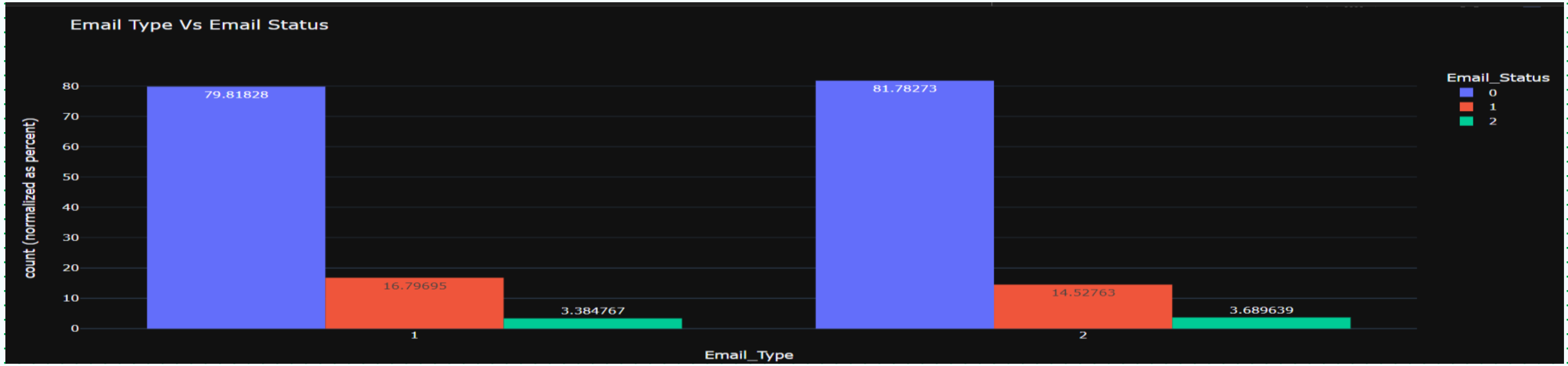
- The average total past communications are around 29. maximum total past communications are around 67 and the minimum is 0.

- The average word count is around 700 words. An email was sent with maximum words of around 1316 words.



$$\begin{tikzcd}
 & \text{Numerical Variables} \\
\text{For} \arrow[ur] \arrow[dr] & \\
 & \text{Categorical Variables}
\end{tikzcd}$$

Analysis of Categorical Features



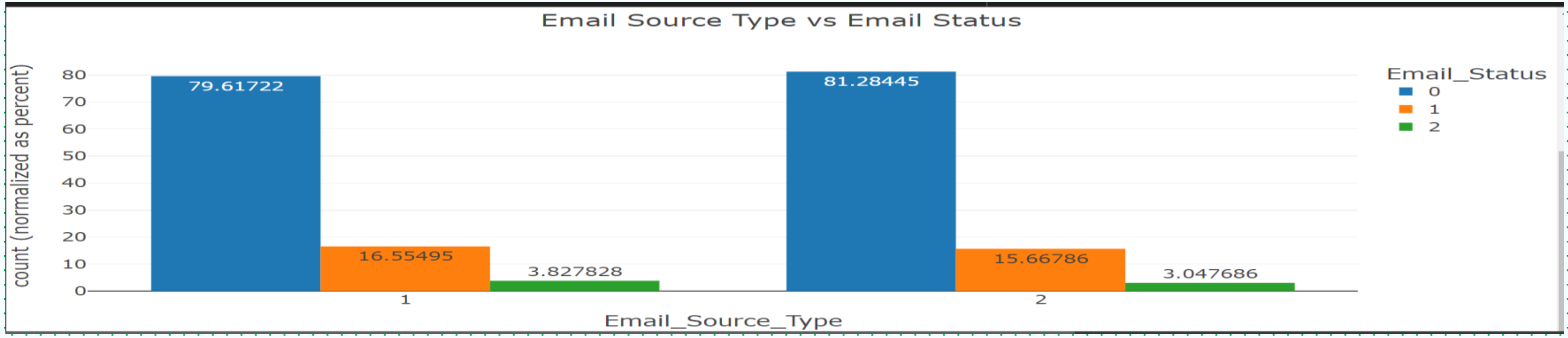
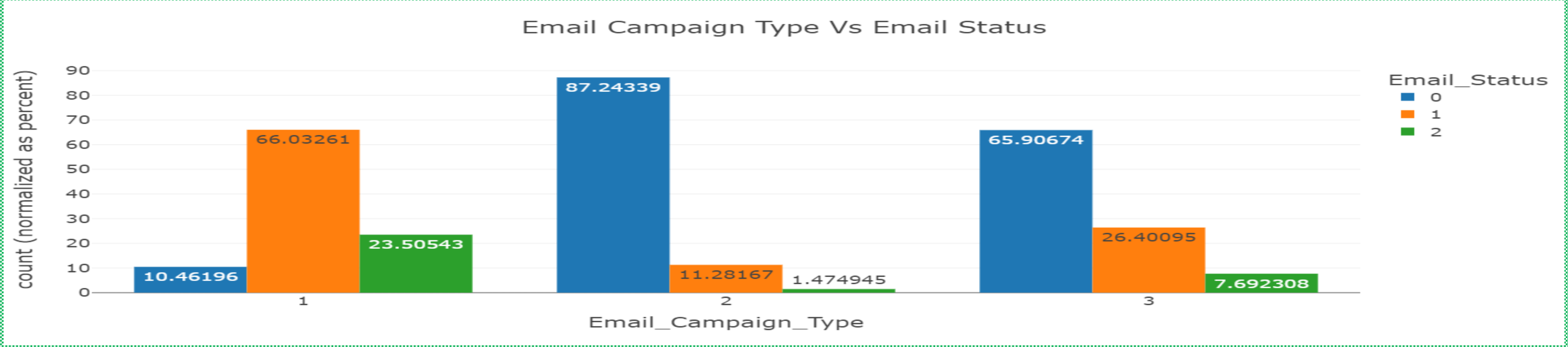


Fig 2. Email Type Vs Email Status (First)

Fig 3. Email Source Type Vs Email Status (Second)

Analysis of Categorical Features (contd)



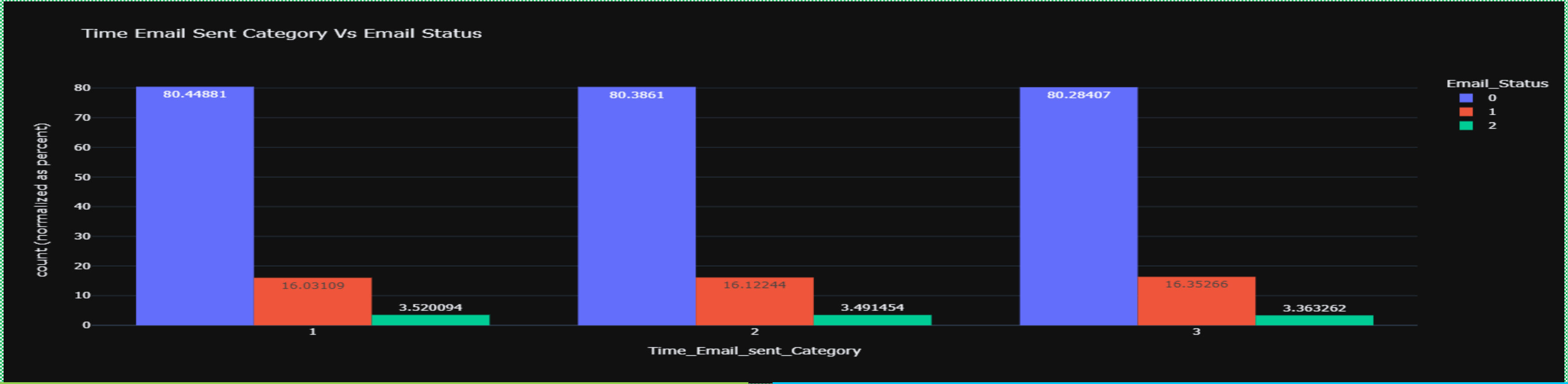


Fig 4: Email Campaign Vs Email Status (Above)

Fig 5: Time Email Sent Category Vs Email Status (Below)

Analysis of Numerical Features

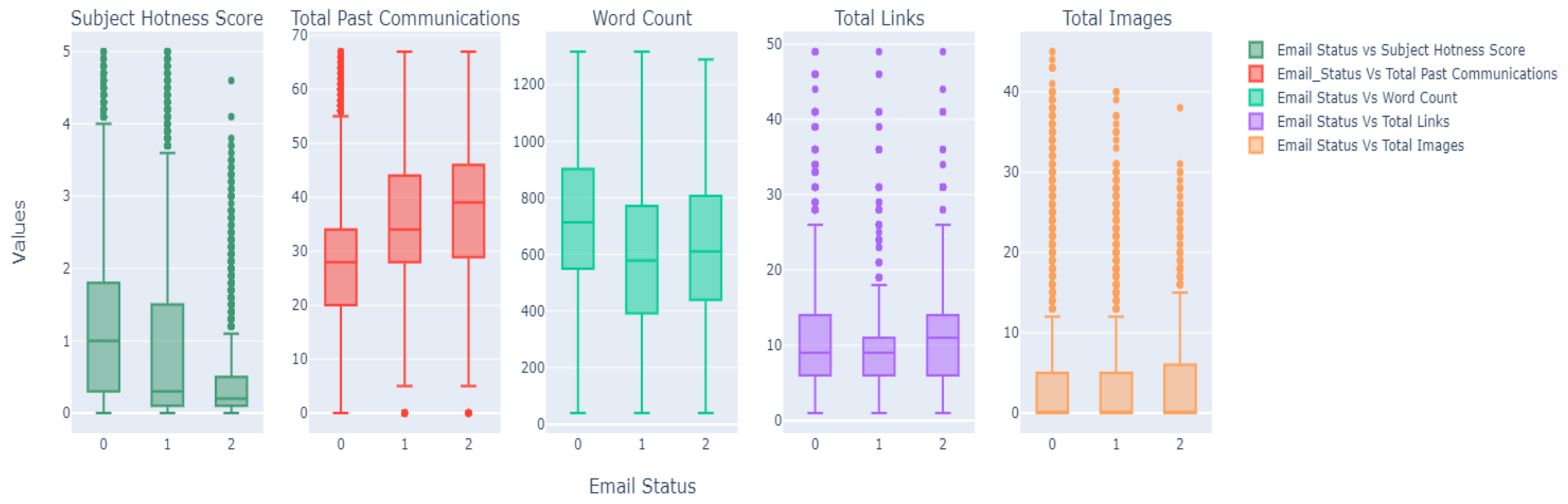


Fig 6 : EDA (Numerical Features)

Outlier Treatment

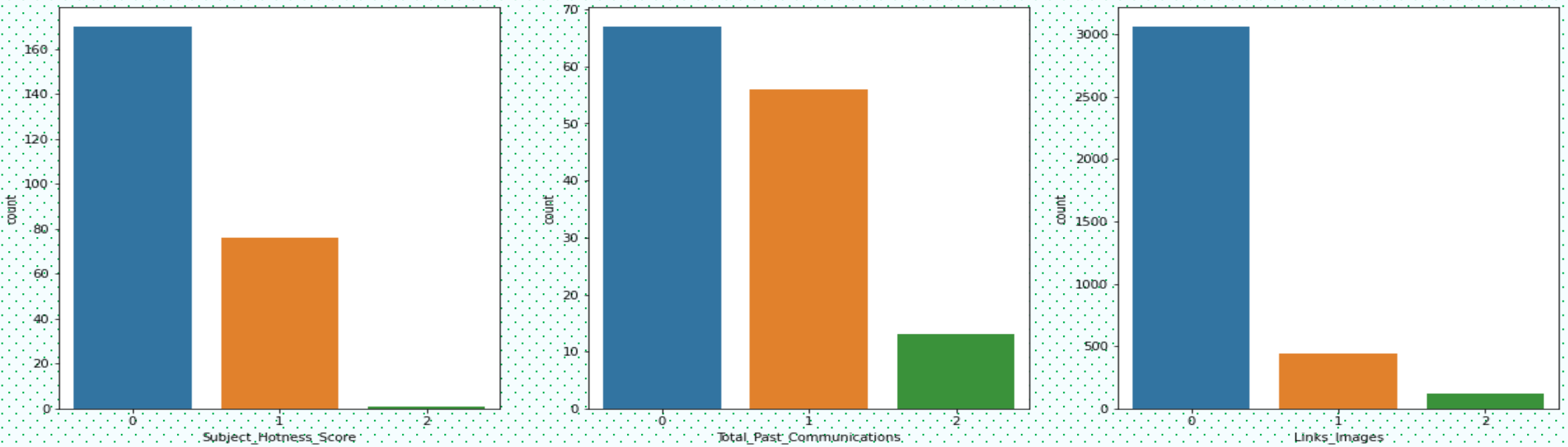
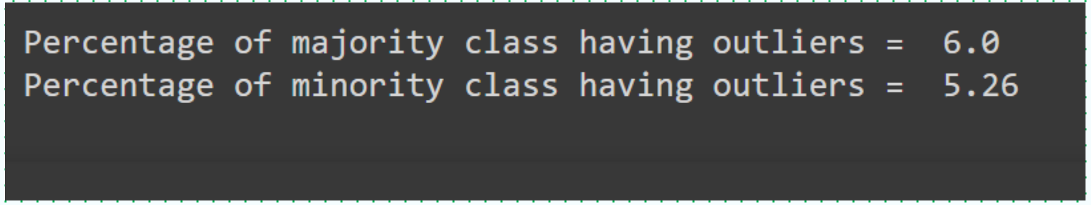
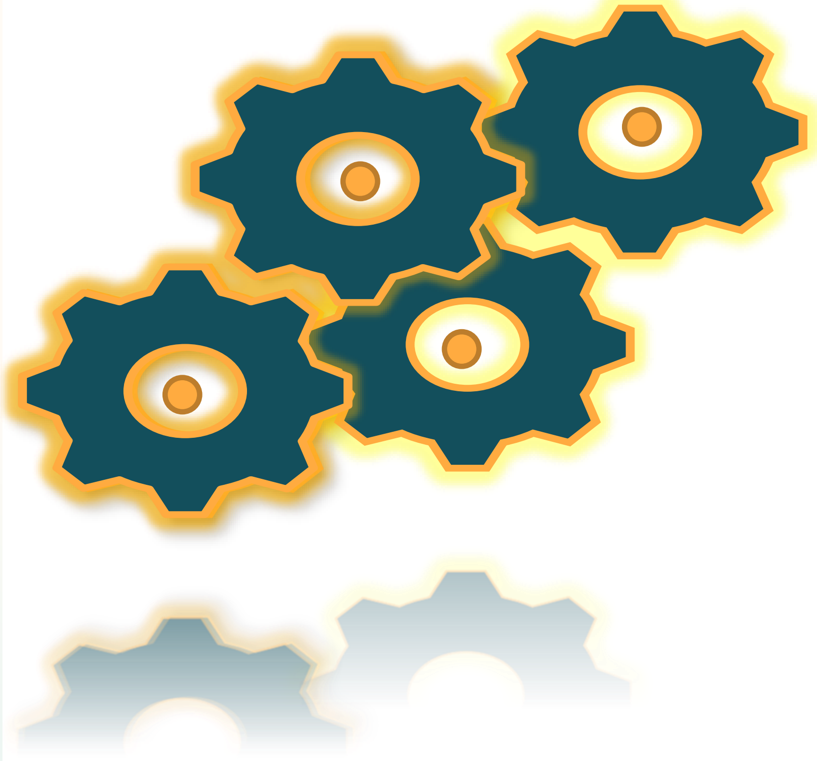


Fig 7: Outlier Treatment



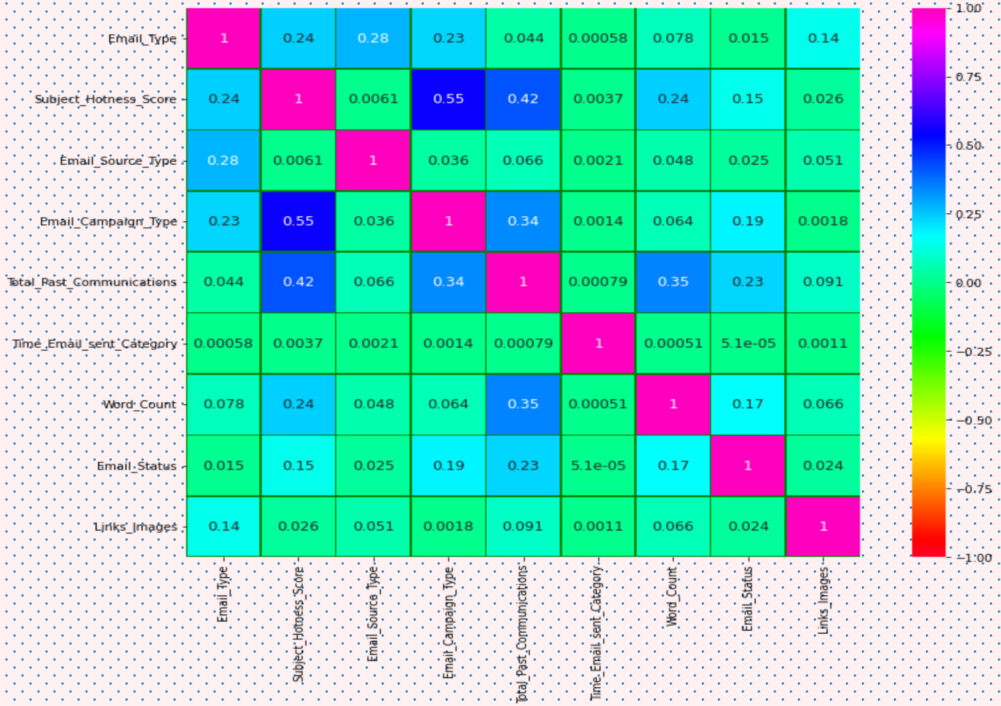
```
Percentage of majority class having outliers =  6.0
Percentage of minority class having outliers =  5.26
```

Feature Engineering



1. Combining Total Images and Total Links:

High positive correlation observed and hence Links Images = Total Images + Total Links



	Email_Type	Subject_Hotness_Score	Email_Source_Type	Email_Campaign_Type	Total_Past_Communications	Time_Email_sent_Category	Word_Count	Email_Status	Links_Images
Email_Type	1	0.24	0.28	0.23	0.044	0.00058	0.078	0.015	0.14
Subject_Hotness_Score	0.24	1	0.0061	0.55	0.42	0.0037	0.24	0.15	0.026
Email_Source_Type	0.28	0.0061	1	0.036	0.066	0.0021	0.048	0.025	0.051
Email_Campaign_Type	0.23	0.55	0.036	1	0.34	0.0014	0.064	0.19	0.0018
Total_Past_Communications	0.044	0.42	0.066	0.34	1	0.00079	0.35	0.23	0.091
Time_Email_sent_Category	0.00058	0.0037	0.0021	0.0014	0.00079	1	0.00051	5.1e-05	0.0011
Word_Count	0.078	0.24	0.048	0.064	0.35	0.00051	1	0.17	0.066
Email_Status	0.015	0.15	0.025	0.19	0.23	5.1e-05	0.17	1	0.024
Links_Images	0.14	0.026	0.051	0.0018	0.091	0.0011	0.066	0.024	1

Fig 8: Heatmap Correlation

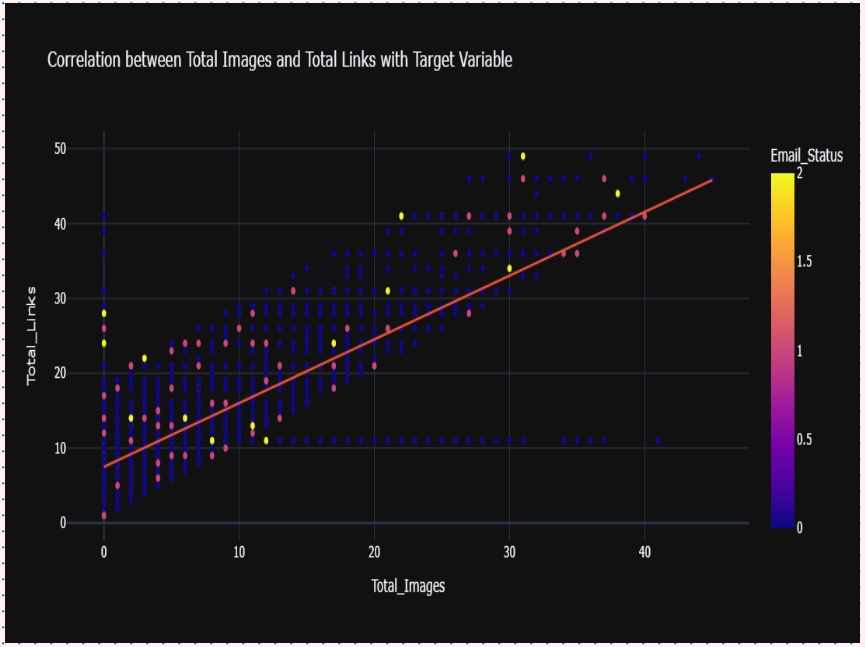
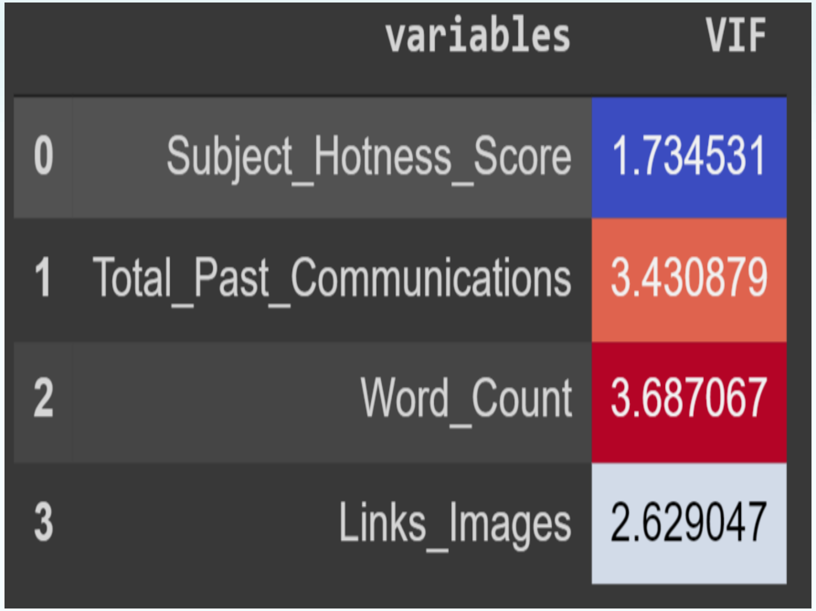


Fig 9: Linear Relationship Between Total Images & Total Links With Target Variable

2. Multicollinearity checked using VIF Factor Why?

- Variables with high multicollinearity can adversely affect the model and removing highly correlated independent variables can help in reducing the curse of dimensionality as well

- We can observe that all numerical variables are within the threshold (i.e., 5)



	variables	VIF
0	Subject_Hotness_Score	1.734531
1	Total_Past_Communications	3.430879
2	Word_Count	3.687067
3	Links_Images	2.629047

Fig 10: Variance Inflation Factor

3. Feature Importance

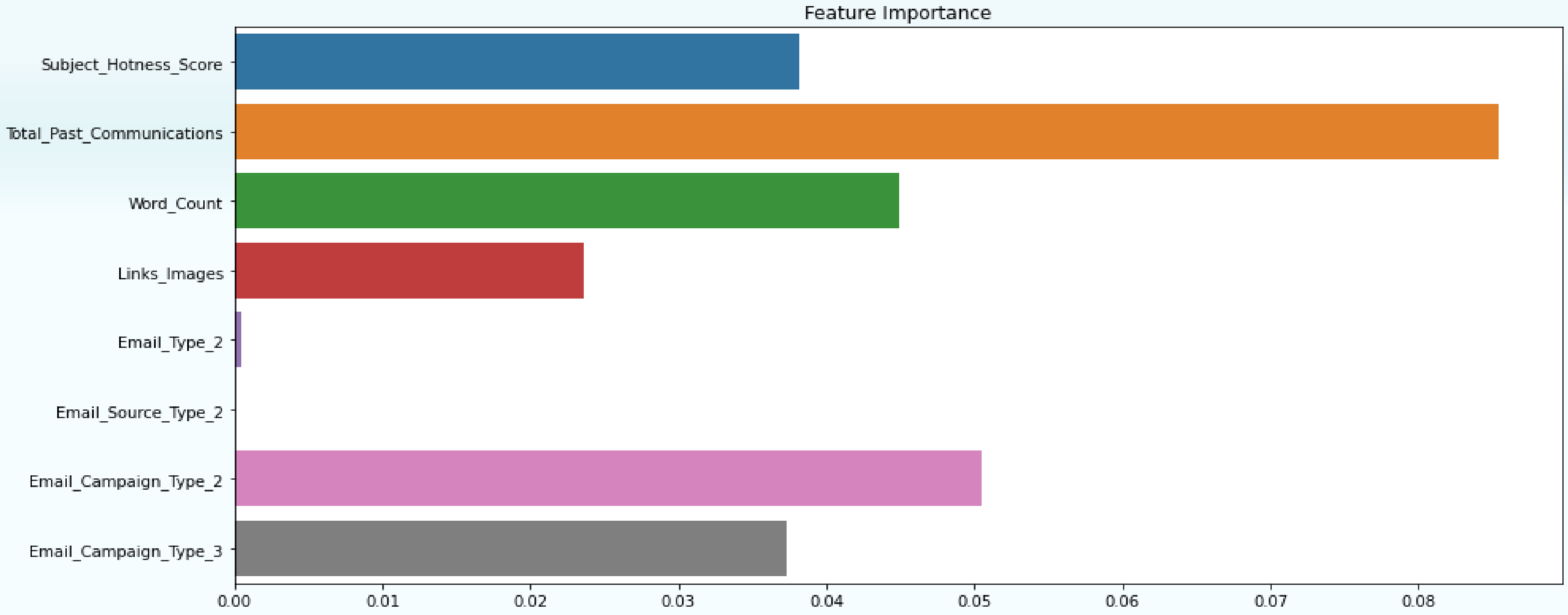


Fig 11: Feature Importance

3. Understanding Feature Importance

The concept used to understand feature importance is Information Gain.

Why?

- It explains which feature has maximum impact in classification, based on the notion of Entropy.
- It works well for numeric as well as categorical data.
- From the graph, we understand that Total_Past_Comunication and Email_Campaign_Type have high importance.
- Time_Email_Sent_Category and Customer_Location are not important and hence we decide to drop the feature.

4. Feature Scaling & OneHotEncoding

Numerical variables were scaled using **MinMaxScaler**.

Why?
The numerical features of the dataset do not have a certain range and they differ from each other.

Categorical variables were encoded using **One-Hot Encoding.**

Why?
This method changes categorical data to a numerical format and enables you to group your categorical data without losing any information

Target Variable

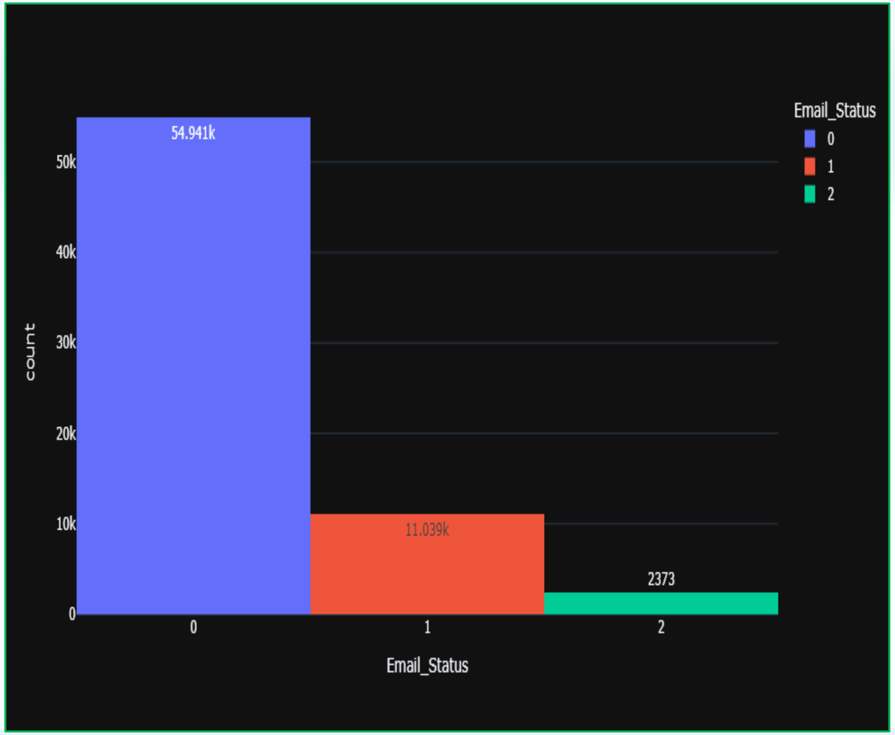


Fig 11: Email Status (Target variable) Count

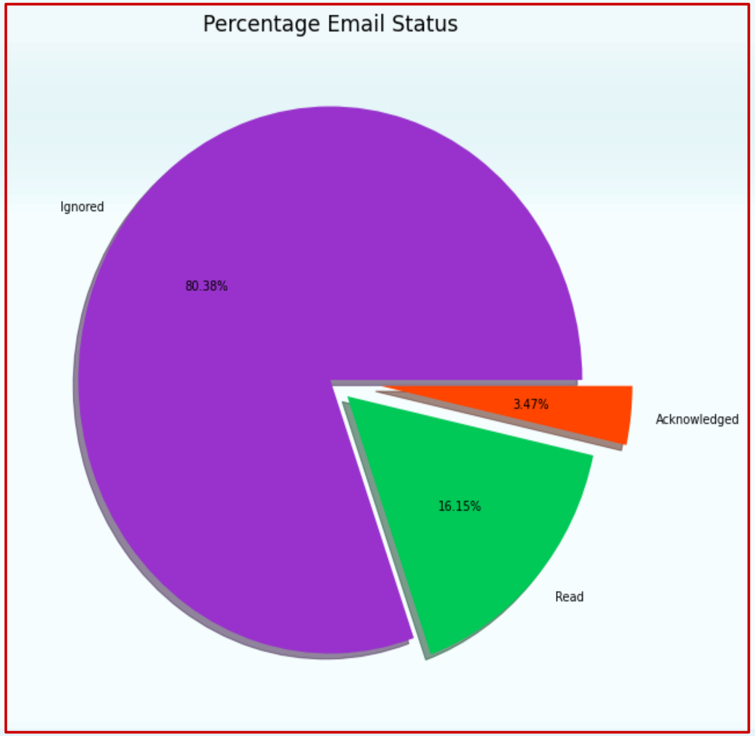


Fig 12: Email Status Percentage Count

Understanding Target Variable

The target variable consists of 3 classes:

- ❑ 0 - ignored - 54941
- ❑ 1 - read - 11039
- ❑ 2 - acknowledged - 2373
- ❑ The target Variable was highly imbalanced.

Why would class imbalance get in the way of training your model?

Class imbalance is when the number of samples is different for the different classes in the data. Most models trained on imbalanced data will have a bias towards predicting the larger class(es) and, in many cases, may ignore the smaller class(es) altogether.

What can we do?

There are techniques by which this problem can be curated

1. Under Sampling 2. Over Sampling

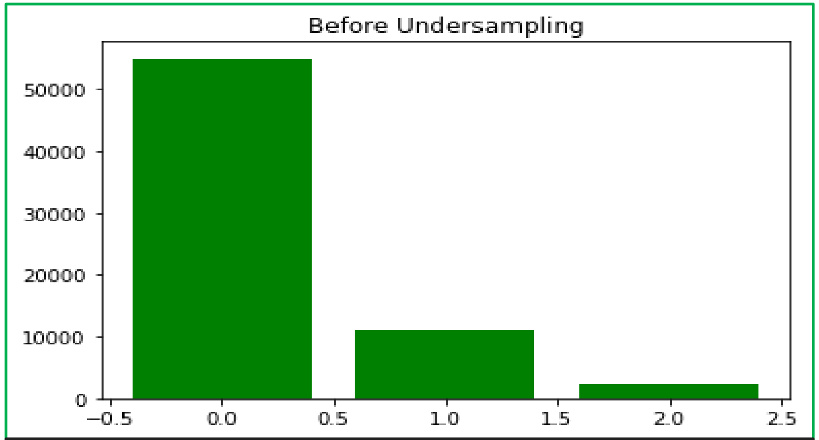
Handling Imbalanced data

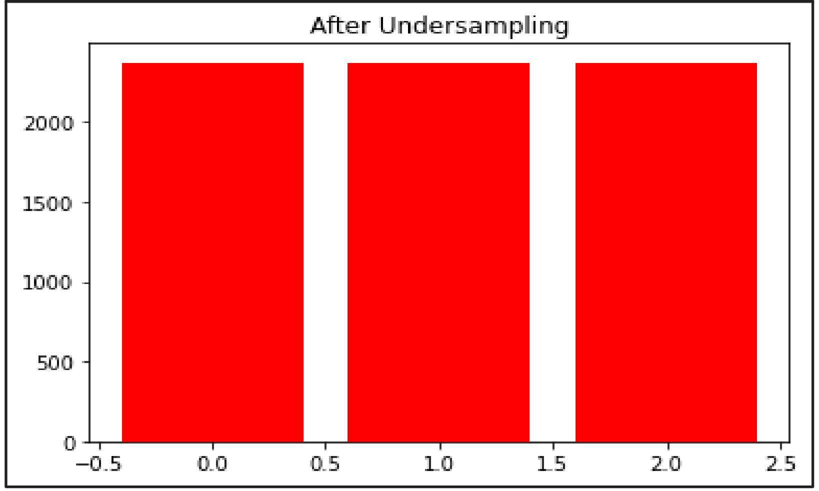
1. **Under-sampling Technique:**

- ❑ The technique used was Random Under Sampler.
- ❑ Created balanced data with 2373 records for each class.
- ❑ Refer to **figure 13** (Before under-sampling) **figure 14** (After Under-sampling)

Why didn't it work?

Created baseline models with under sampled data and it was observed that they underperformed primarily due to loss of information.





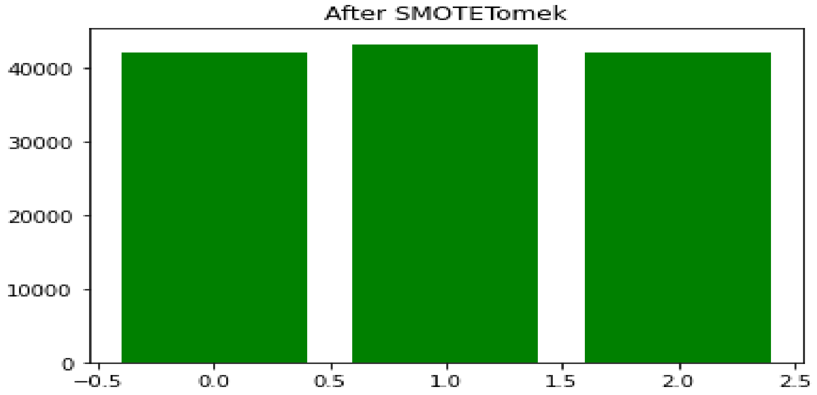
Handling Imbalanced data

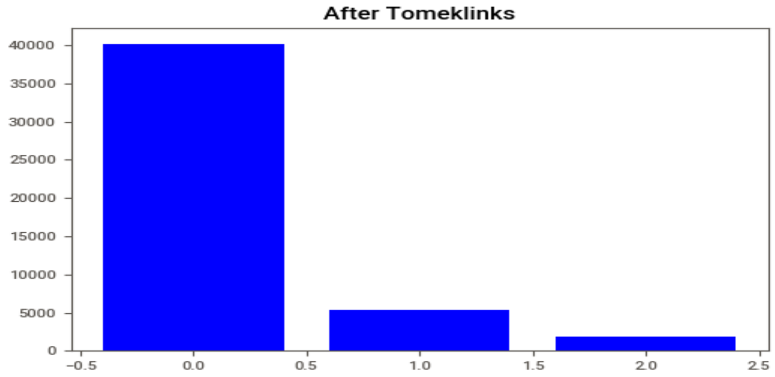
2. Oversampling Technique:

The technique used was **SMOTETomek & Tomeklink**

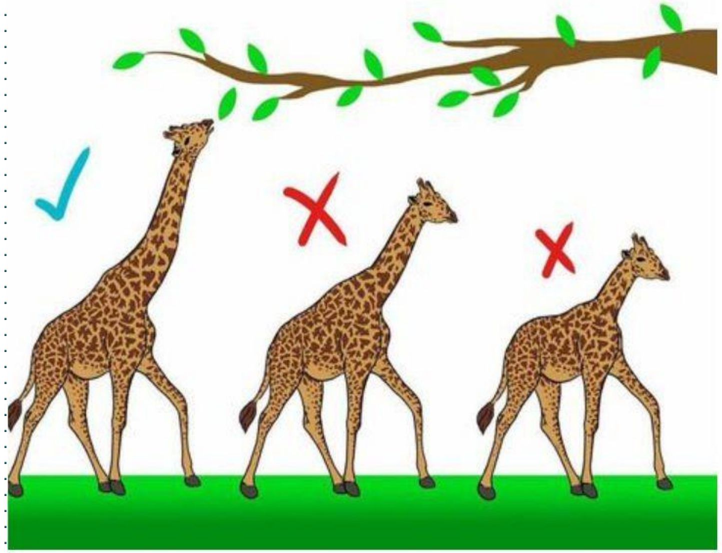
- ❑ Frequency of unique values of the Email Status: [[0, 1, 2] [42018, 42147, 43166]] using **SMOTETomek**

- ❑ Frequency of unique values of the Email Status: [[0 1 2] [40204, 5321, 1898]] using **Tomeklink**

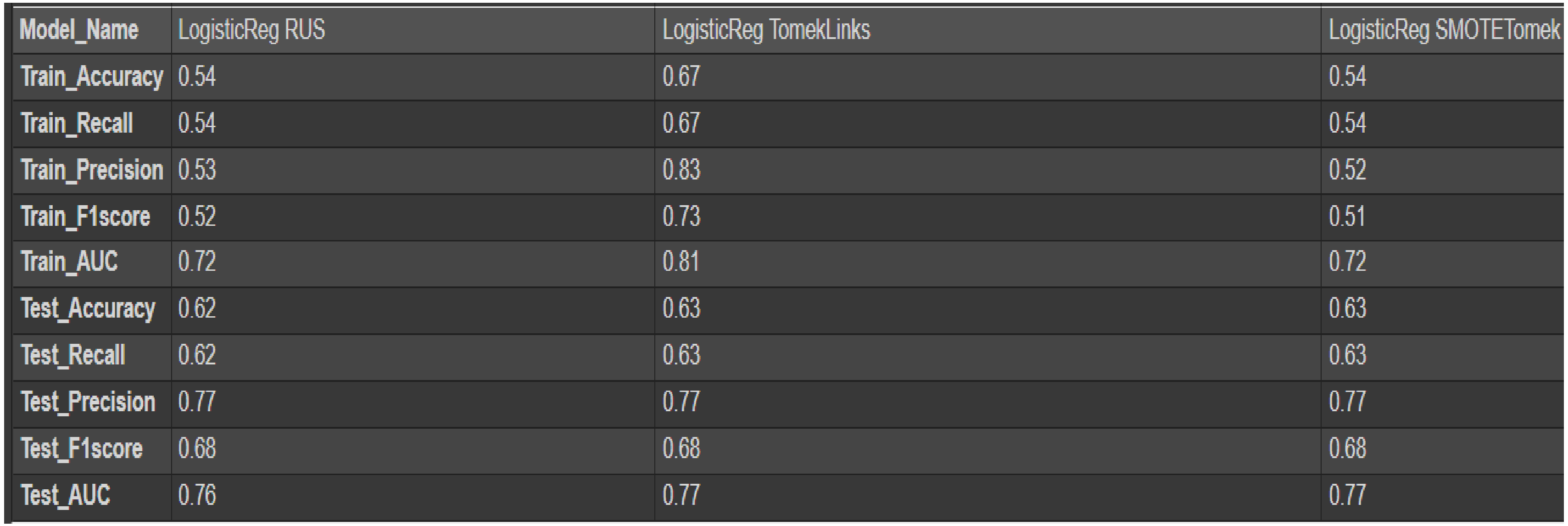




Model Training and Evaluation

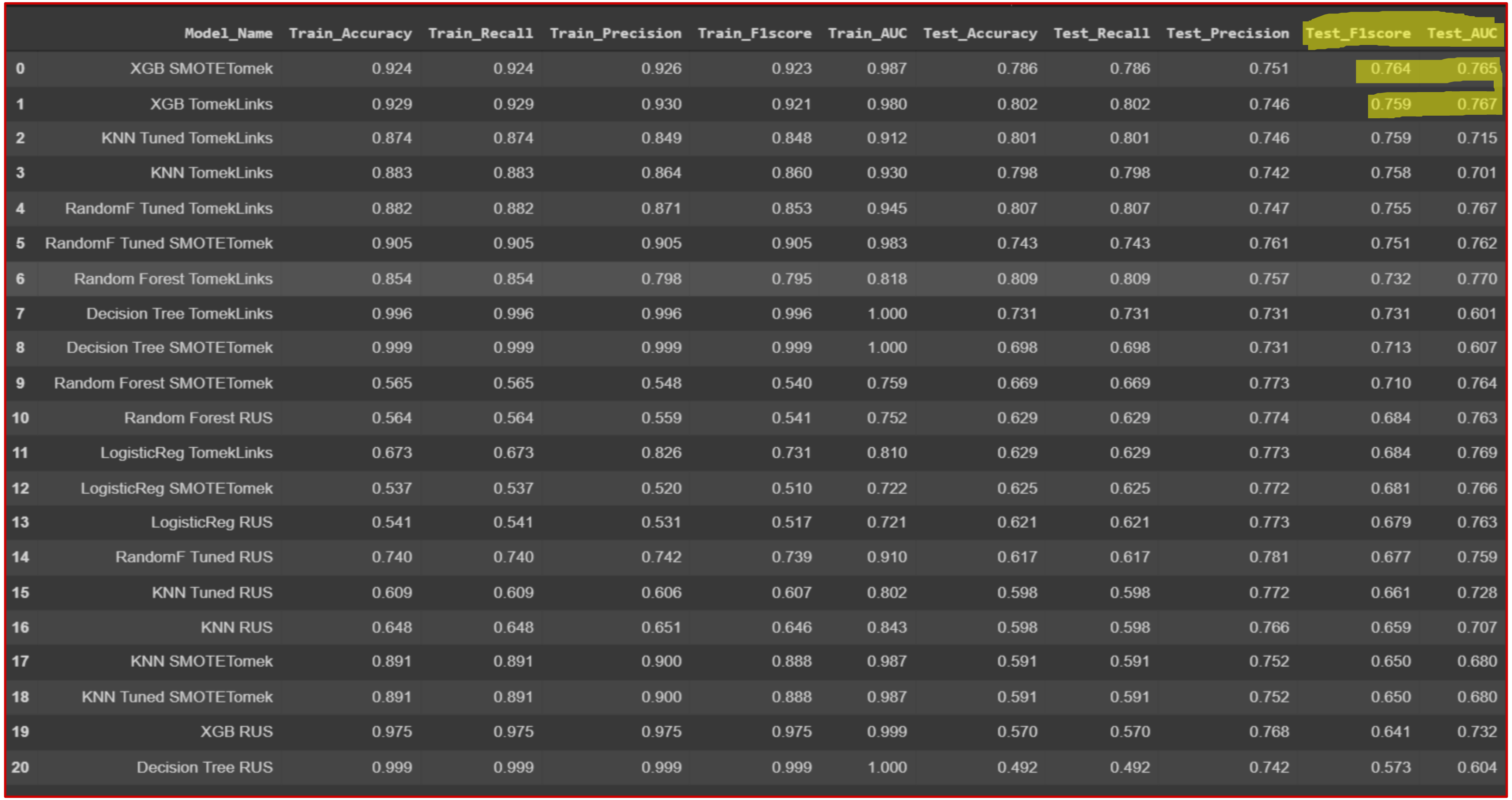


Applying Model (baseline model)



Model_Name	LogisticReg RUS	LogisticReg TomekLinks	LogisticReg SMOTETomek
Train_Accuracy	0.54	0.67	0.54
Train_Recall	0.54	0.67	0.54
Train_Precision	0.53	0.83	0.52
Train_F1score	0.52	0.73	0.51
Train_AUC	0.72	0.81	0.72
Test_Accuracy	0.62	0.63	0.63
Test_Recall	0.62	0.63	0.63
Test_Precision	0.77	0.77	0.77
Test_F1score	0.68	0.68	0.68
Test_AUC	0.76	0.77	0.77

Evaluation of Models



	Model_Name	Train_Accuracy	Train_Recall	Train_Precision	Train_F1score	Train_AUC	Test_Accuracy	Test_Recall	Test_Precision	Test_F1score	Test_AUC
0	XGB SMOTETomek	0.924	0.924	0.926	0.923	0.987	0.786	0.786	0.751	0.764	0.765
1	XGB TomekLinks	0.929	0.929	0.930	0.921	0.980	0.802	0.802	0.746	0.759	0.767
2	KNN Tuned TomekLinks	0.874	0.874	0.849	0.848	0.912	0.801	0.801	0.746	0.759	0.715
3	KNN TomekLinks	0.883	0.883	0.864	0.860	0.930	0.798	0.798	0.742	0.758	0.701
4	RandomF Tuned TomekLinks	0.882	0.882	0.871	0.853	0.945	0.807	0.807	0.747	0.755	0.767
5	RandomF Tuned SMOTETomek	0.905	0.905	0.905	0.905	0.983	0.743	0.743	0.761	0.751	0.762
6	Random Forest TomekLinks	0.854	0.854	0.798	0.795	0.818	0.809	0.809	0.757	0.732	0.770
7	Decision Tree TomekLinks	0.996	0.996	0.996	0.996	1.000	0.731	0.731	0.731	0.731	0.601
8	Decision Tree SMOTETomek	0.999	0.999	0.999	0.999	1.000	0.698	0.698	0.731	0.713	0.607
9	Random Forest SMOTETomek	0.565	0.565	0.548	0.540	0.759	0.669	0.669	0.773	0.710	0.764
10	Random Forest RUS	0.564	0.564	0.559	0.541	0.752	0.629	0.629	0.774	0.684	0.763
11	LogisticReg TomekLinks	0.673	0.673	0.826	0.731	0.810	0.629	0.629	0.773	0.684	0.769
12	LogisticReg SMOTETomek	0.537	0.537	0.520	0.510	0.722	0.625	0.625	0.772	0.681	0.766
13	LogisticReg RUS	0.541	0.541	0.531	0.517	0.721	0.621	0.621	0.773	0.679	0.763
14	RandomF Tuned RUS	0.740	0.740	0.742	0.739	0.910	0.617	0.617	0.781	0.677	0.759
15	KNN Tuned RUS	0.609	0.609	0.606	0.607	0.802	0.598	0.598	0.772	0.661	0.728
16	KNN RUS	0.648	0.648	0.651	0.646	0.843	0.598	0.598	0.766	0.659	0.707
17	KNN SMOTETomek	0.891	0.891	0.900	0.888	0.987	0.591	0.591	0.752	0.650	0.680
18	KNN Tuned SMOTETomek	0.891	0.891	0.900	0.888	0.987	0.591	0.591	0.752	0.650	0.680
19	XGB RUS	0.975	0.975	0.975	0.975	0.999	0.570	0.570	0.768	0.641	0.732
20	Decision Tree RUS	0.999	0.999	0.999	0.999	1.000	0.492	0.492	0.742	0.573	0.604

Test F1 Score & Test Roc - AUC Score Comparison

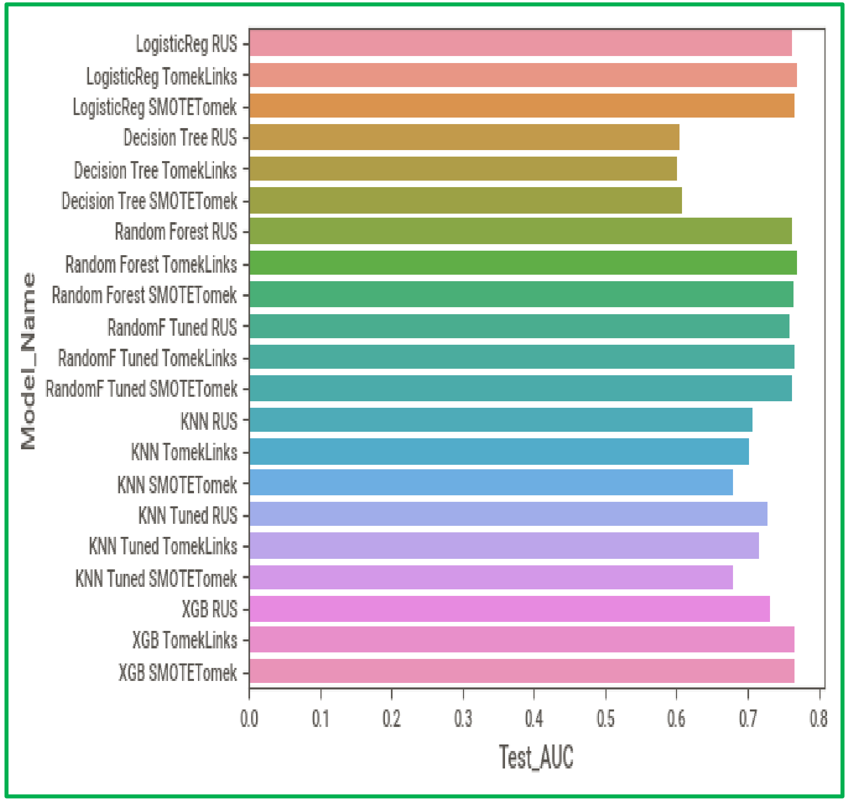


Fig: Test Roc- AUC Score

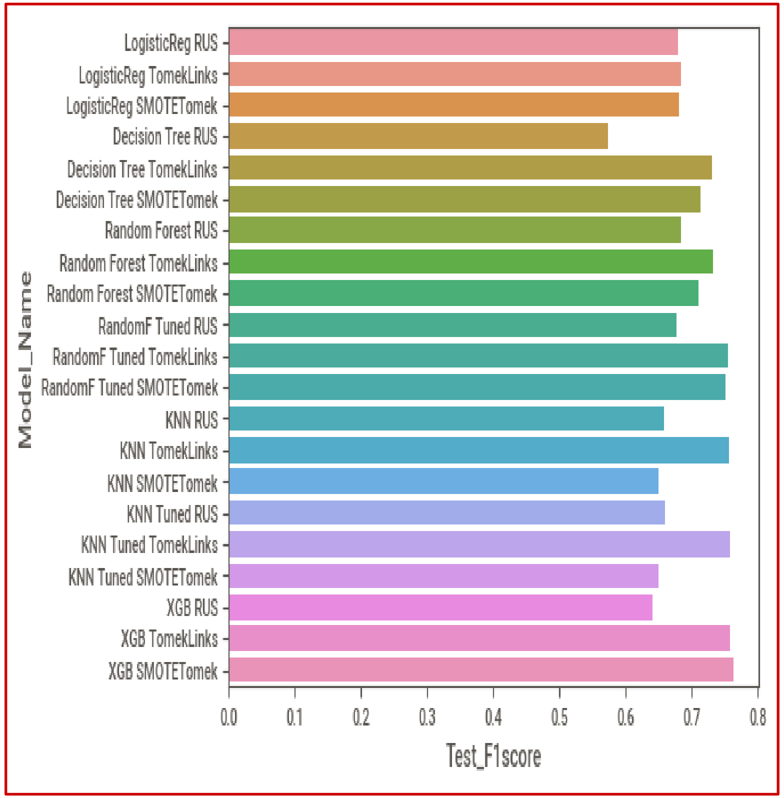


Fig: Test F1 Score

Best Performing Model

AI

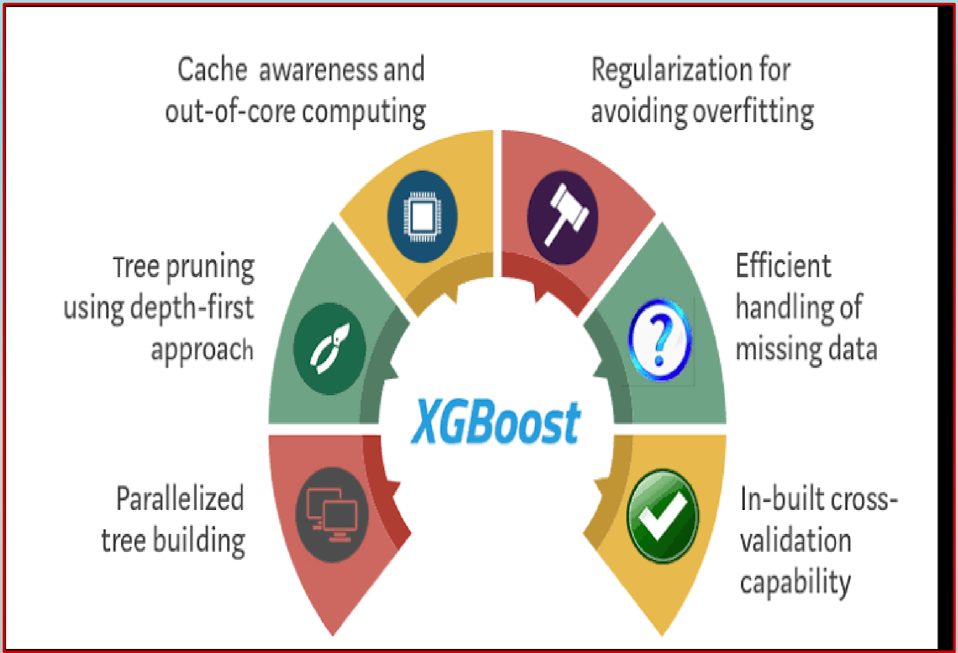
XGBoost:

01 Robust to outliers.

02 Supports regularization.

03 Works well on small to the medium dataset.

04 F1 score for train & test set were 90% & 77% respectively


Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

Conclusion

- In EDA, we observed that Email_Campaign_Type was the most important feature. If your Email_Campaign_Type was 1, there is a 90% likelihood of your Email to be read/acknowledged.

- It was observed that both Time_Email_Sent and Customer_Location were insignificant in determining the Email status. The ratio of the Email Status was the same irrespective of the demographic location or the time frame the emails were sent on.

- As the word_count increases beyond the 600 mark, we see that there is a high possibility of that email being ignored. The ideal mark is 400-600. No one is interested in reading long emails!

- For modeling, it was observed that for imbalance handling Oversampling i.e., SMOTELink & SMOTETomek worked way better than under-sampling as the latter resulted in a lot of loss of information.

- Based on the metrics, XGBoost Classifier worked the best giving a train score of 92% and a test score of 77% for F1 score.

Challenges

01 Choosing the appropriate technique to handle the imbalance in data was quite challenging as it was a tradeoff between information loss vs the risk of overfitting.

02 Overfitting was another major challenge during the modeling process.

03 Understanding what features are most important and what features to avoid was a difficult task.

04 Decision-making on missing value imputations and outlier treatment was quite challenging as well.

THANK
YOU