Capstone Project - 3
Supervised ML - Classification
Email Campaign Effectiveness Prediction

By
Subhajit Ganguly
Data Science Trainee, Almabetter

Problem Statement

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.

The Dataset

Independent Features

Discrete Features :

1. Email_Type : Type of email encoded as 1 and 2
2. Email_Source_Type : Source of email encoded as 1 and 2
3. Customer_Location : Location of customer encoded as A,B,C,D,E,F,G
4. Email_Campaign_Type : Type of campaign encoded as 1, 2 and 3
5. Time_Email_sent_Category : Time at which email email was sent encoded as 1, 2 and 3

Continuous Features :

1. Subject_Hotness_Score : A score between 0 to 5 for hotness of the email topic
2. Total_Past_Communications : Number of past communications
3. Word_Count : Words in the email
4. Total_Links : Number of links in the email
5. Total_Images : Number of images in the email

Email_ID : Unique identifier of emails sent

Dependent Feature

Email_Status : Email status encoded as 0 : ignored, 1 : read, 2 : acknowledged

Tackling the Problem

It is a multi-class classification problem with 3 classes in it.

1. Basic EDA :
In this step, I want to do some exploration on the data. First, I shall check for null values and try to replace or remove them. Then, I shall check for outliers using boxplots and try to replace or remove them. Thirdly, I shall get some visualizations to get an idea of the variables in hand.

2. Model training and testing :
In this step, I shall get a train-test pair from the given dataset and fit 4 classification models to the train set, make predictions on the test set using them and calculate various evaluation metrics. The models are namely : Decision Trees, Random Forests, Gradient Boosting Machine, Naive-Bayes Classifier.

3. Model Evaluation :
As the last step, I shall compare all the models and try to come up with a conclusion about which model might be the best choice here. And I'll talk about variable importance too.

EDA - Null Values?

```
Index: 68353 entries, EMA00081000034500 to EMA00089999316900
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Email_Type                   68353 non-null  int64
 1   Subject_Hotness_Score        68353 non-null  float64
 2   Email_Source_Type            68353 non-null  int64
 3   Customer_Location            56758 non-null  object
 4   Email_Campaign_Type          68353 non-null  int64
 5   Total_Past_Communications    61528 non-null  float64
 6   Time_Email_sent_Category     68353 non-null  int64
 7   Word_Count                   68353 non-null  int64
 8   Total_Links                  66152 non-null  float64
 9   Total_Images                 66676 non-null  float64
 10  Email_Status                 68353 non-null  int64
```

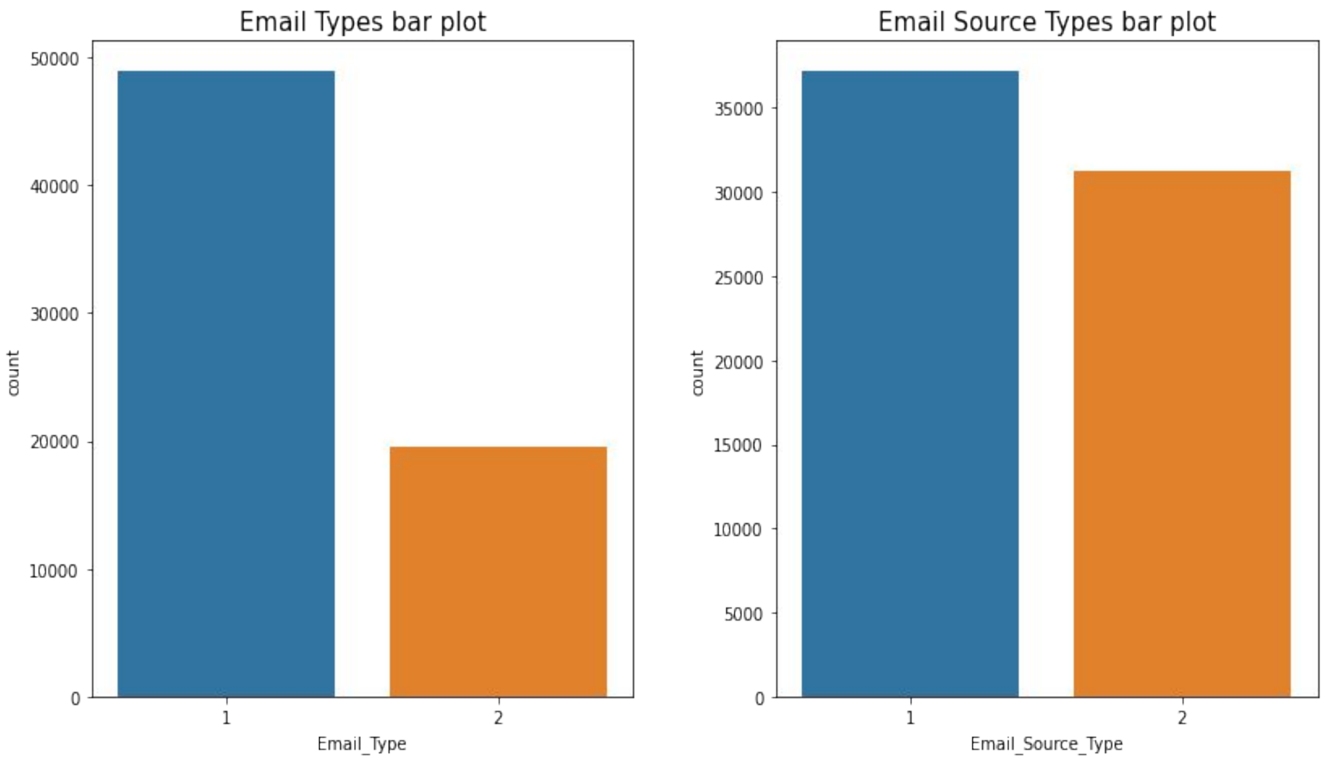
The columns with missing values are :

1. **Customer_Location (Categorical)**
2. **Total_Past_Communications (Numerical)**
3. **Total_Links (Numerical)**
4. **Total_Images (Numerical)**

Handled Using :

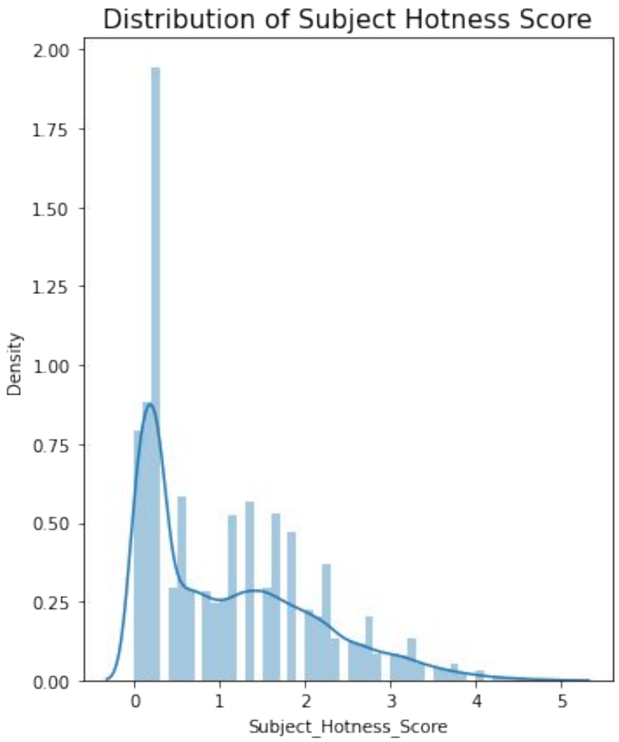
1. **Used KNN Imputer to impute values of Numerical Columns.**
2. **Predicted missing values of Customer_Location using other features and trained the model using rows with available Customer_Location.**

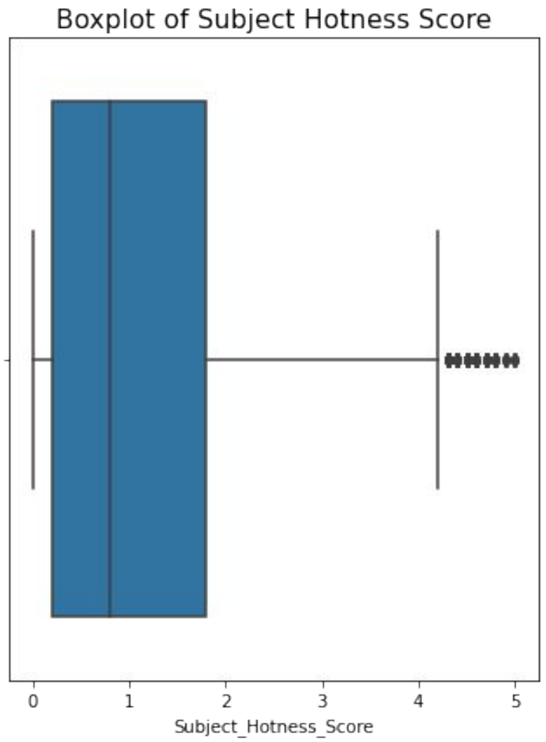
EDA - Visualizations - 1



- Most of the emails were of type 1.
- Emails were sent from both the sources with almost equal probability.

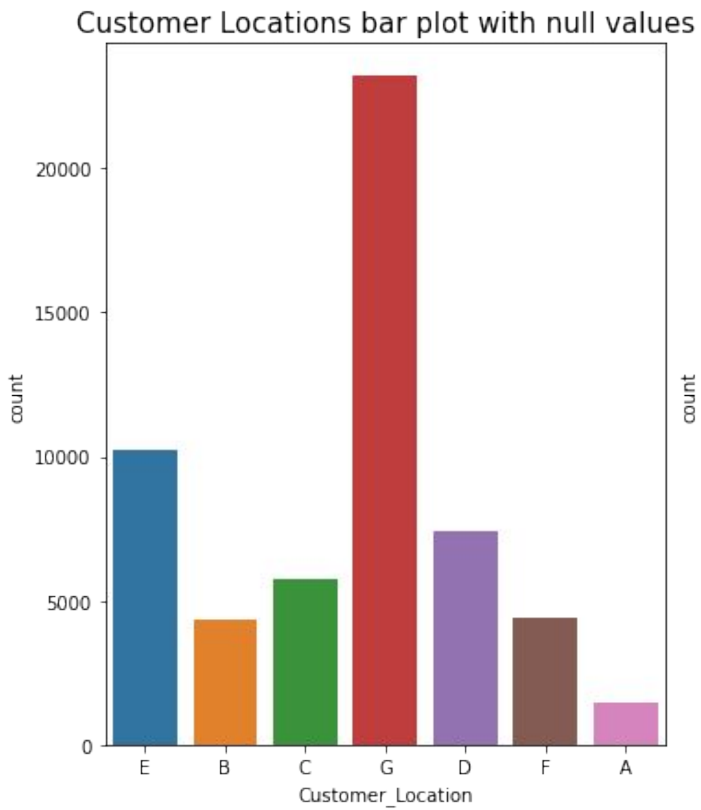
EDA - Visualizations - 2

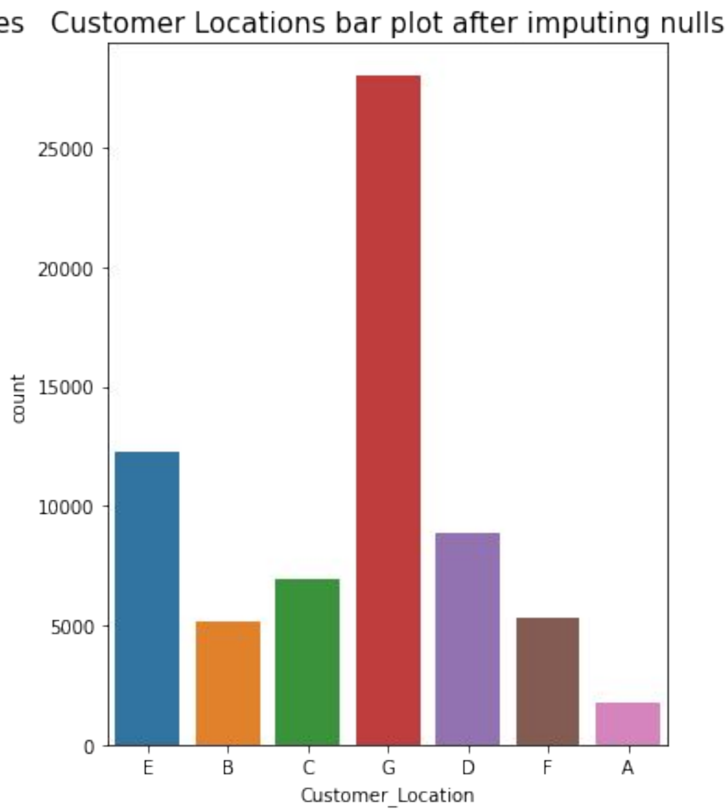




- **Emails with lower subject hotness score is higher in numbers. There are some outliers too.**

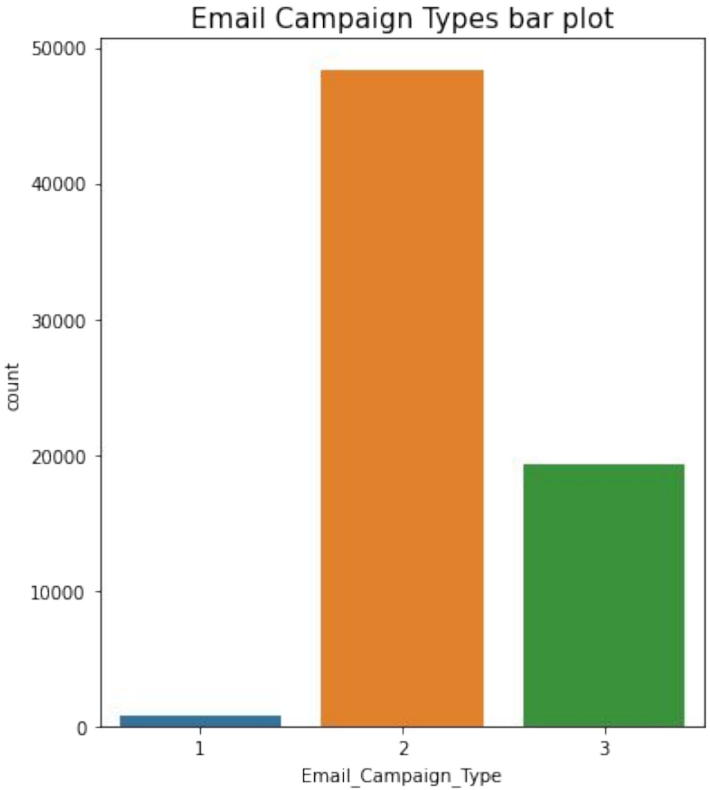
EDA - Visualizations - 3

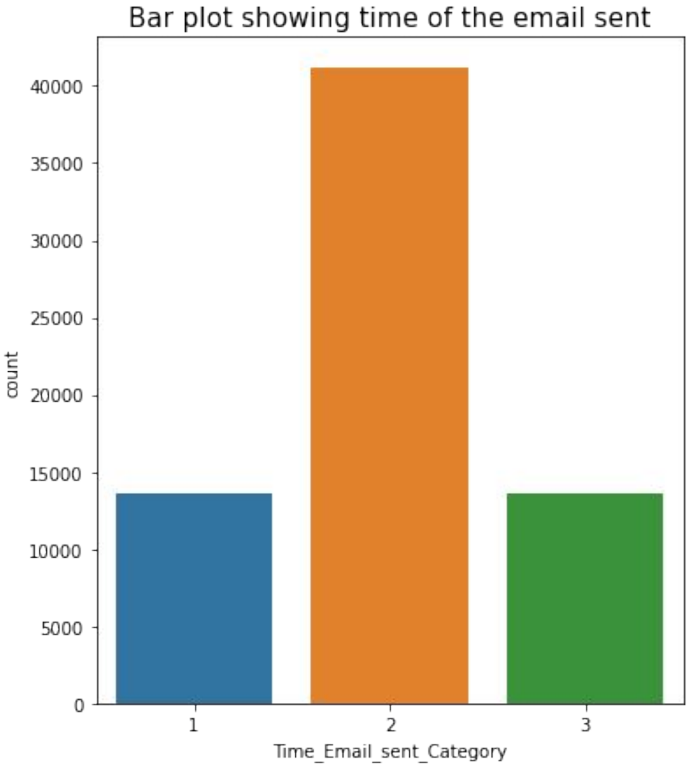




- **Area G has most number of customers and area A has least number of customers.**
- **Also the bars from before and after missing value imputation are in conjunction which implies the method of predicting missing values has worked quite well.**

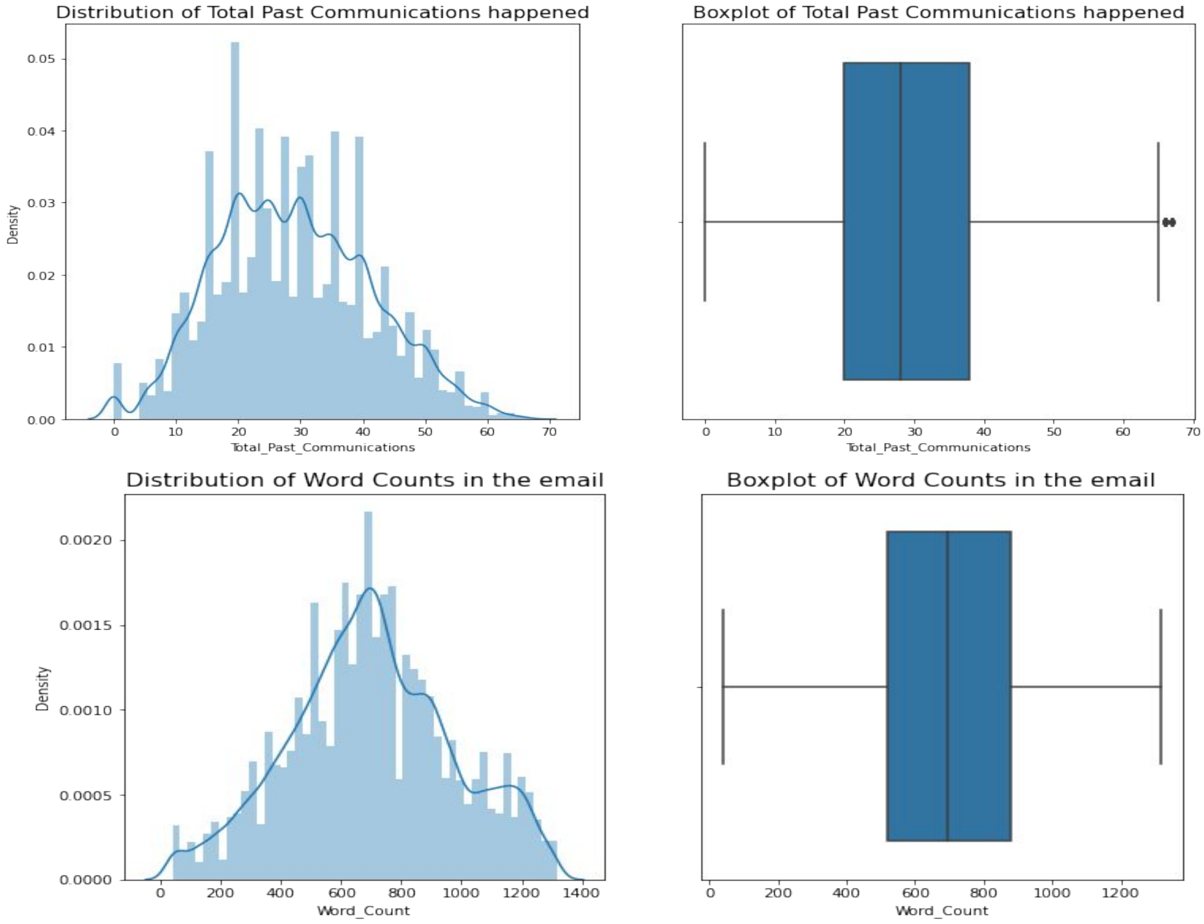
EDA - Visualizations - 4





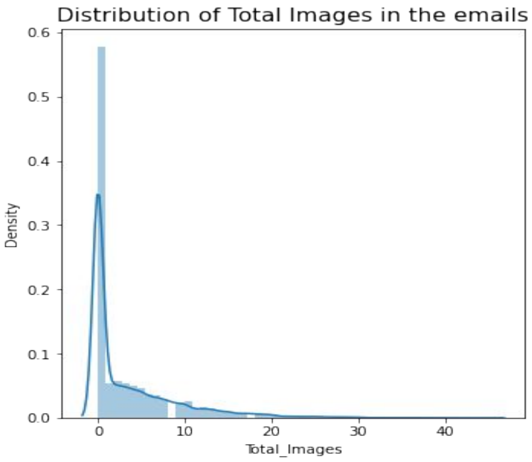
- Most of the emails were sent as a part of 2nd type of campaign.
- Most of the emails were sent at the time bucket 2.

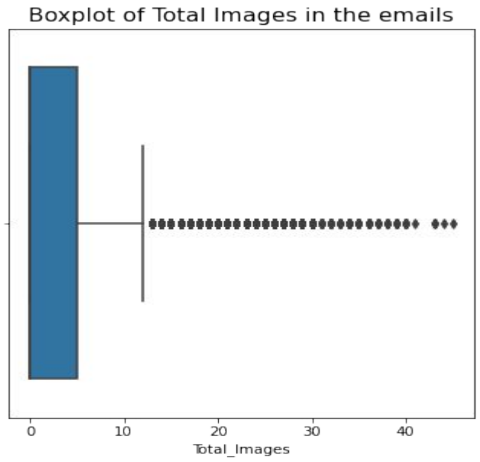
EDA - Visualizations - 5

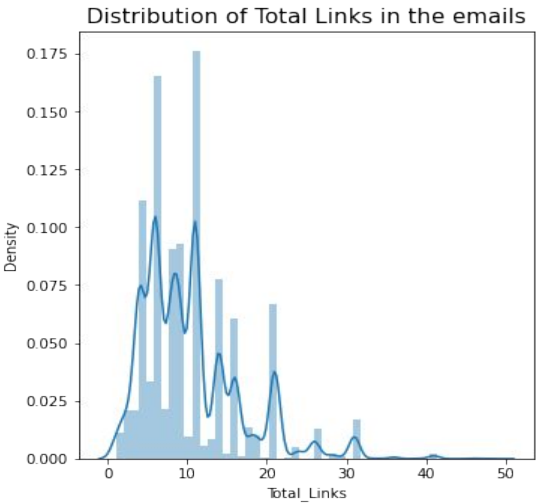


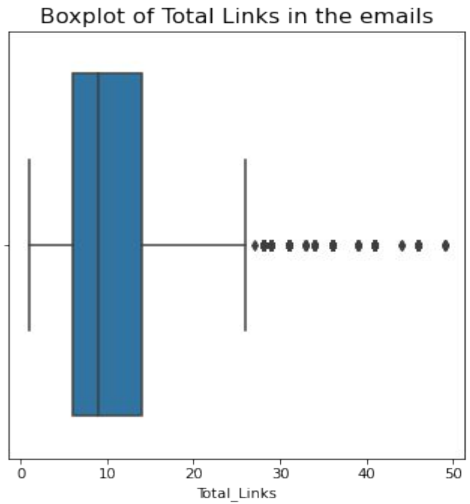
- Number of Total Past Communications is somehow normally distributed with average number of communications around 30. There are 2 outlier points too.
- Email word counts is somehow normally distributed with mean number of words around 700.

EDA - Visualizations - 6



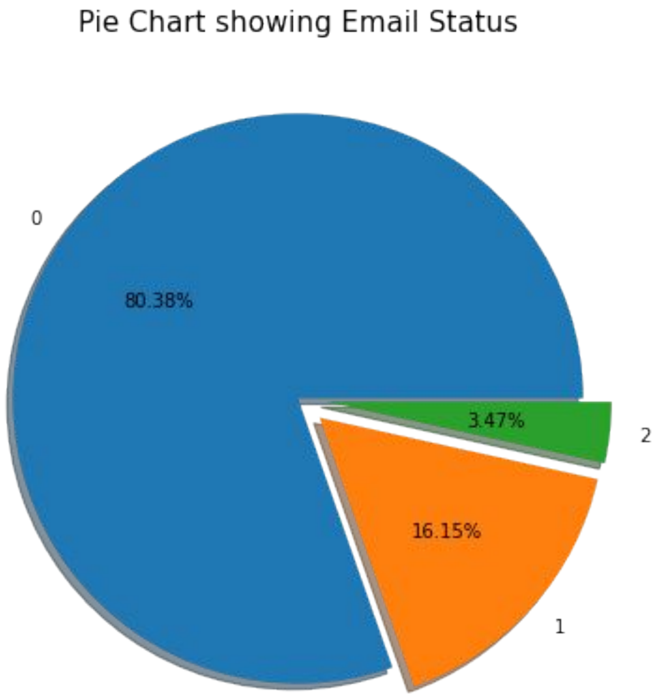






- Total link numbers has no distinct distribution. Emails have mean number of links around 10. Further investigation is required on this feature. There are a few outlier points.
- Most of emails had no photos in it. There are a few outlier points.

EDA - Visualizations - 7 & Further Analysis



Around 3.47% of the emails are actually hitting targets that is those customers are reading and acknowledging the mails. Around 16.15% of customers are reading the mails who might be potential customers given some offers or something else.

- Found that some Total_Past_Communications, Total_Links, Total_Images were in decimals which were erroneous entries. So I replaced them with nearest integers.
- Removed 0.1% of data as anomalies using Isolation Forest.

Data Lost

0.1%

Model Training and Testing - Baseline

AI

Decision Trees
(max_depth = 10)

```
Confusion Matrix for Training set :

[[36748  5211  1992]
 [ 9420 19928 14603]
 [ 3459 10390 30102]]
Confusion Matrix for Testing set :

[[9034 1402  552]
 [2432 4812 3744]
 [ 879 2748 7361]]
```

```
Classification report for Training set :

precision    recall  f1-score   support

           0       0.74      0.84      0.79     43951
           1       0.56      0.45      0.50     43951
           2       0.64      0.68      0.66     43951

    accuracy                           0.66    131853
   macro avg       0.65      0.66      0.65    131853
weighted avg       0.65      0.66      0.65    131853
```

```
Classification report for Testing set :

              precision    recall  f1-score   support

           0       0.73      0.82      0.77     10988
           1       0.54      0.44      0.48     10988
           2       0.63      0.67      0.65     10988

    accuracy                           0.64     32964
   macro avg       0.63      0.64      0.64     32964
weighted avg       0.63      0.64      0.64     32964
```

We can see that a baseline Decision Tree Model is giving similar kind of metrics across both train and test set which signifies that class imbalance is managed and we are good to go with training.

Model Training and Testing - Decision Trees

Decision Trees

max_depth = 15, min_samples_split = 3

```
Confusion Matrix for Training set :

[[40604  2534   813]
 [ 5585 25230 13136]
 [ 1366  6499 36086]]

Confusion Matrix for Testing set :

[[9332 1287  369]
 [1914 5434 3640]
 [ 479 1887 8622]]
```

The Decision Tree Classifier performed pretty well with around 77% accuracy for training set and around 71% accuracy for testing set.

Classification report for Training set :

	precision	recall	f1-score	support
0	0.85	0.92	0.89	43951
1	0.74	0.57	0.65	43951
2	0.72	0.82	0.77	43951
accuracy			0.77	131853
macro avg	0.77	0.77	0.77	131853
weighted avg	0.77	0.77	0.77	131853

Classification report for Testing set :

	precision	recall	f1-score	support
0	0.80	0.85	0.82	10988
1	0.63	0.49	0.55	10988
2	0.68	0.78	0.73	10988
accuracy			0.71	32964
macro avg	0.70	0.71	0.70	32964
weighted avg	0.70	0.71	0.70	32964

Model Training and Testing - Random Forest

Random Forest

max_depth = 15, min_samples_split = 3, n_estimators': 80

```
Confusion Matrix for Training set :

[[42074  1274   603]
 [ 5147 26286 12518]
 [ 1286  3857 38808]]

Confusion Matrix for Testing set :

[[10020   764   204]
 [ 1993  5504  3491]
 [  486  1246  9256]]
```

```
Classification report for Training set :

              precision    recall  f1-score   support

           0       0.87      0.96      0.91     43951
           1       0.84      0.60      0.70     43951
           2       0.75      0.88      0.81     43951

    accuracy                           0.81    131853
   macro avg       0.82      0.81      0.81    131853
weighted avg       0.82      0.81      0.81    131853
```

```
Classification report for Testing set :

              precision    recall  f1-score   support

           0       0.80      0.91      0.85     10988
           1       0.73      0.50      0.59     10988
           2       0.71      0.84      0.77     10988

    accuracy                           0.75     32964
   macro avg       0.75      0.75      0.74     32964
weighted avg       0.75      0.75      0.74     32964
```

The Random Forest Classifier gave around 81% accuracy on training set and around 75% accuracy on testing set. The precision and f1 score for all classes are pretty good, but the recall for class 1 is pretty low.

Model Training and Testing - Gradient Boosting Machine

Gradient Boosting Machine

max_depth = 15, min_samples_split = 5, n_estimators': 80

```
Confusion Matrix for Training set :

[[43934    16     1]
 [  226 43600   125]
 [   15    12 43924]]

Confusion Matrix for Testing set :

[[10267   670    51]
 [ 1411  8728   849]
 [  282   443 10263]]
```

- **This seems pretty good to me! Training accuracy is almost 100% and testing accuracy is almost 90%! But I am going to fit another GBM to investigate whether the accuracy of 100% is a result of overfitting or not.**
- **Also the recall for class 2 is pretty high, which signifies that the model is pretty good at predicting which customers might acknowledge the mails.**

```
Classification report for Training set :

              precision    recall  f1-score   support

           0       0.99      1.00      1.00     43951
           1       1.00      0.99      1.00     43951
           2       1.00      1.00      1.00     43951

    accuracy                           1.00    131853
   macro avg       1.00      1.00      1.00    131853
weighted avg       1.00      1.00      1.00    131853

Classification report for Testing set :

              precision    recall  f1-score   support

           0       0.86      0.93      0.89     10988
           1       0.89      0.79      0.84     10988
           2       0.92      0.93      0.93     10988

    accuracy                           0.89     32964
   macro avg       0.89      0.89      0.89     32964
weighted avg       0.89      0.89      0.89     32964
```

Model Training and Testing - Gradient Boosting Machine

AI

Gradient Boosting Machine

max_depth = 10, min_samples_split = 5, n_estimators': 60

Without GridSearchCV

```
Confusion Matrix for Training set :

[[42853  1003    95]
 [ 4905 33302  5744]
 [ 1035  1697 41219]]
Confusion Matrix for Testing set :

[[10354   584    50]
 [ 1678  7254  2056]
 [  335   874  9779]]
```

```
Classification report for Training set :

              precision    recall  f1-score   support

           0       0.88      0.98      0.92     43951
           1       0.93      0.76      0.83     43951
           2       0.88      0.94      0.91     43951

    accuracy                           0.89    131853
   macro avg       0.89      0.89      0.89    131853
weighted avg       0.89      0.89      0.89    131853
```

```
Classification report for Testing set :

              precision    recall  f1-score   support

           0       0.84      0.94      0.89     10988
           1       0.83      0.66      0.74     10988
           2       0.82      0.89      0.86     10988

    accuracy                           0.83     32964
   macro avg       0.83      0.83      0.83     32964
weighted avg       0.83      0.83      0.83     32964
```

- **It seems that the difference between Training and Testing set accuracy is consistent. But, it seems like the previous model is overfit as the difference between train and test set accuracy is lesser in this GBM model. So, this one is a better choice for GBM.**

Model Training and Testing - Naive Bayes Classifier - 1

GaussianNB

```
Confusion Matrix for Training set :

[[25196  5145 13610]
 [ 6723  6473 30754]
 [ 3148  1892 38910]]
Confusion Matrix for Testing set :

[[6213 1295 3479]
 [1656 1658 7674]
 [ 822  462 9704]]
```

```
Classification report for Training set :

              precision    recall  f1-score   support

           0       0.72      0.57      0.64     43951
           1       0.48      0.15      0.23     43950
           2       0.47      0.89      0.61     43950

    accuracy                           0.54    131851
   macro avg       0.55      0.54      0.49    131851
weighted avg       0.55      0.54      0.49    131851

Classification report for Testing set :

              precision    recall  f1-score   support

           0       0.71      0.57      0.63     10987
           1       0.49      0.15      0.23     10988
           2       0.47      0.88      0.61     10988

    accuracy                           0.53     32963
   macro avg       0.56      0.53      0.49     32963
weighted avg       0.56      0.53      0.49     32963
```

Model Training and Testing - Naive Bayes Classifier - 2

MultinomialNB

```
Confusion Matrix for Training set :

[[29227  9029  5695]
 [16138  9617 18195]
 [14172  4406 25372]]
Confusion Matrix for Testing set :

[[7186 2354 1447]
 [3946 2429 4613]
 [3485 1104 6399]]
```

- It seems like Naive-Bayes is doing a terrible job according to all metrics here. So, we have to discard the idea of using it.

Classification report for Training set :

	precision	recall	f1-score	support
0	0.49	0.66	0.56	43951
1	0.42	0.22	0.29	43950
2	0.52	0.58	0.54	43950
accuracy			0.49	131851
macro avg	0.47	0.49	0.47	131851
weighted avg	0.47	0.49	0.47	131851

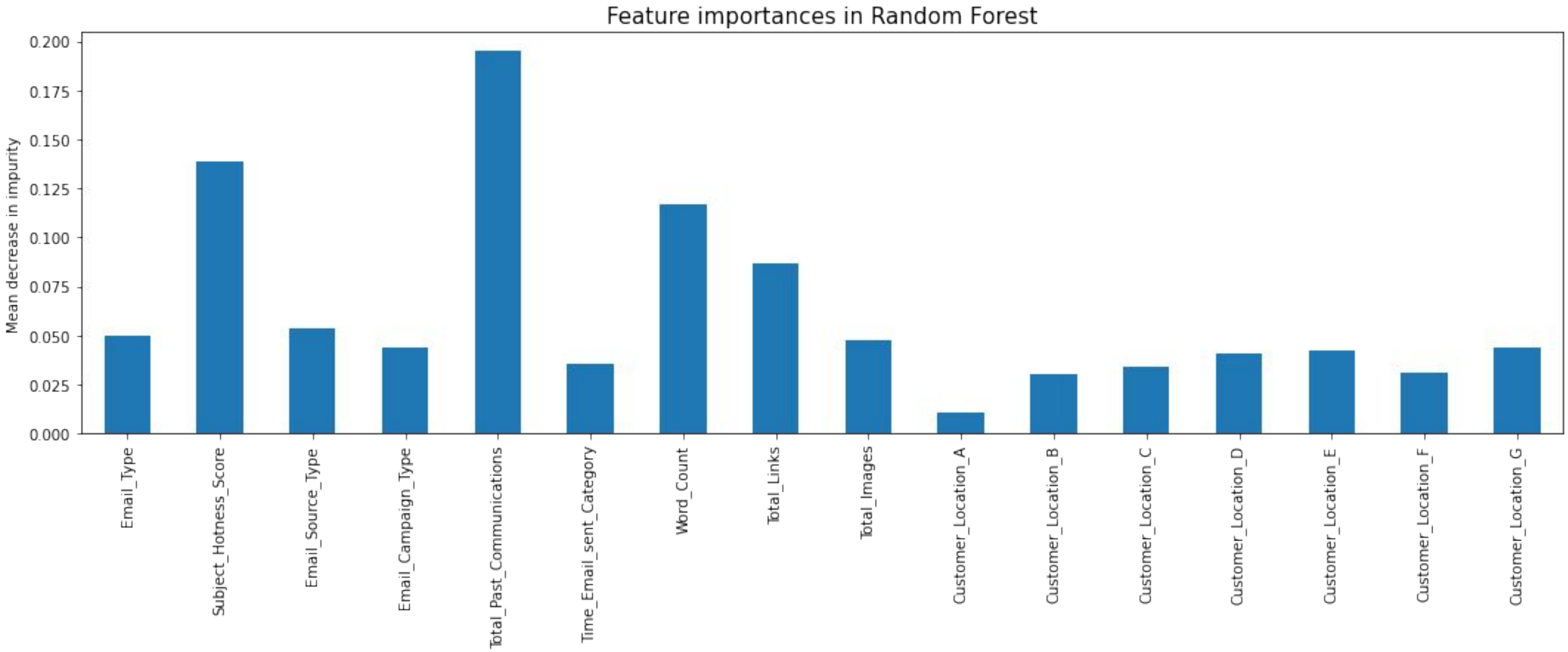
Classification report for Testing set :

	precision	recall	f1-score	support
0	0.49	0.65	0.56	10987
1	0.41	0.22	0.29	10988
2	0.51	0.58	0.55	10988
accuracy			0.49	32963
macro avg	0.47	0.49	0.47	32963
weighted avg	0.47	0.49	0.47	32963

Model Evaluation - The Best one?

Model Name	Performance Score	Speed Score	Final Score
Decision Trees	3	2	32
Random Forest	2	3	23
Gradient Boosting Machine	1	4	14
Multinomial Naive-Bayes Classifier	5	1	51
Gaussian Naive-Bayes Classifier	4	1	41

Model Evaluation - Feature Importance



We can see that Total Past Communications have the greatest effect in predicting email campaign effectiveness. And Customer Location A has the least importance. Subject Hotness Score and Word Count are also important features.

Final Verdicts

AI

1. Important Variables :

Total_Past_Communications, Subject_Hotness_Score and Word_Count are the 3 most important features in predicting effectiveness of the campaign.

2. Best Model :

Gradient Boosting Machine is the best choice here. Although the model is overfit or prone to overfit, no other model could get to the accuracy on test set it has provided.

3. Challenges faced :

I am listing some challenges faced by me :

- Huge data size.
- Too much training time for black box models.
- Choosing the best model due overfitting challenges.

4. Use cases :

Before discussing the use cases, let's understand the outputs first. I'm taking the GBM model as reference. If you see the precision of all 3 classes from the model, they are pretty good. But the recall and f1-score for class 1 is pretty bad. These observations signifies that this model is pretty good in predicting whether the customer will completely ignore or completely respond to the mail. If some company is thinking about somehow targeting the customers who have read the mails, that is choosing potential customers, the model won't give great results in that case.

Thank You!