Capstone Project - 3

Email Campaign Effectiveness Prediction

By-Dany Chitturi

Let's Catch The Defaulters

1. Defining the problem statement
2. Exploratory Data Analysis
3. Feature Selection
4. Handling Class Imbalance
5. Modeling
6. Model Building
7. Hyperparameter tuning
8. Model validation and Selection

Problem Statement

Most 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; or acknowledged by the reader.

Data Summary

- Email_ID: This column contains the email ids of individuals.
- Email_type: Email type contains 2 categories 1 and 2. We can assume that the types are promotional emails or important emails.
- Subject_Hotness_Score: It is the email effectiveness score.
- Email_Source: It represents the source of the email like sales or marketing or product type email.
- Email_Campaign_Type: Campaign type
- Total_Past_Communications: This column contains the previous emails from the same source.

Data Summary

- Customer_Location: Categorical data which explains the different demographics of the customers.
- Time_Email_sent_Category: It has three categories 1,2 and 3 which may give us morning, evening, and night time slots.
- Word_Count: It contains the number of words contained in the mail.
- Total_Links: Total links from the mail.
- Total_Images: The banner images from the promotional email.

Data Pipeline

- **Data processing-1:** In this first part we've checked and Imputed the null values and also checked for duplicated records
- **Data processing-2:** In this part, we performed outlier detection, encoded the categorical features
- **EDA:** In this part, we did exploratory data analysis(EDA) on all the features in our dataset to uncover the relationship between the dependent and independent variable(s).
- **Feature Selection:** In this part, we used VIF(Variance Inflation Factor) to check if there is any serious correlation between any two independent variables and based on the VIF values we selected the appropriate features.
- **Handling Imbalance:** we used Random Under Sampling and SMOTE (Synthetic Minority Oversampling Technique) to handle the imbalance in the data.
- **Model building :** Finally, In this part, we create 6 different models. We start with a simple model, then we use hyperparameter tuning to get the best optimal parameters.

Define Dependent Variable

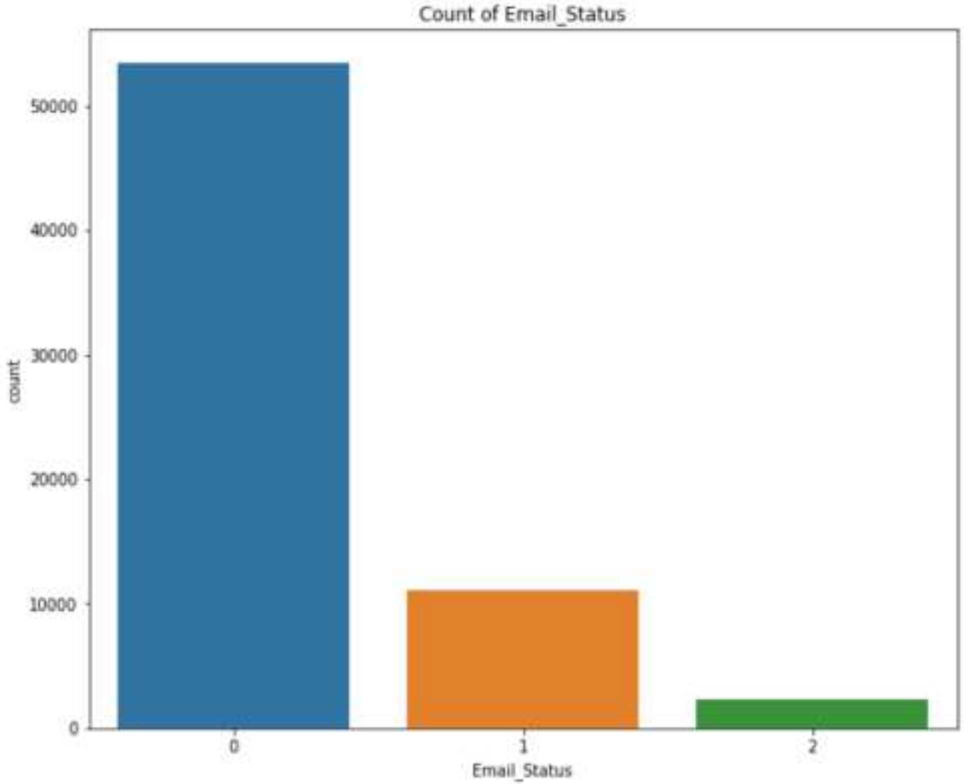
Email_Status: It contains the characterization of the mail that is ignored; read; or acknowledged by the reader.

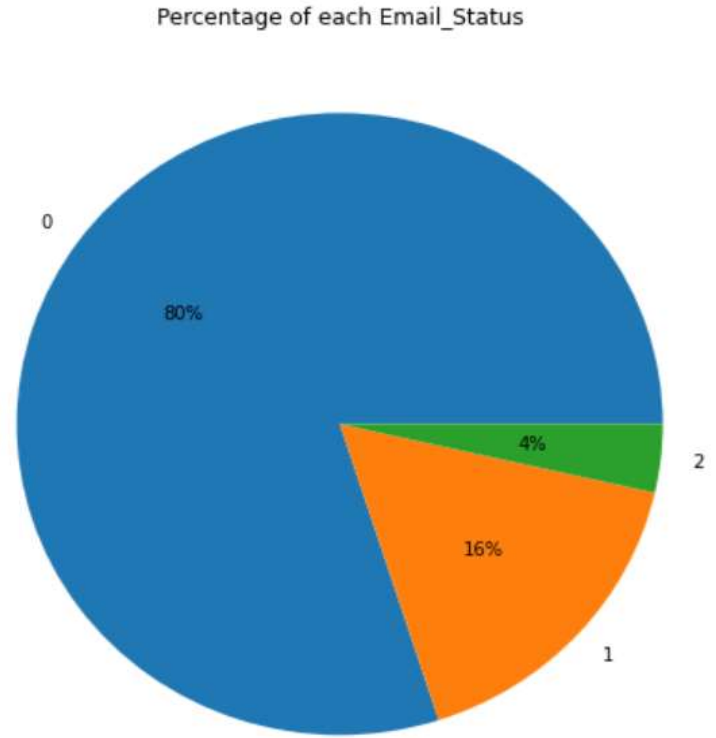
Class ‘0’: Ignored emails

Class ‘1’: Read emails

Class ‘2’: Acknowledged emails

Define Dependent Variable

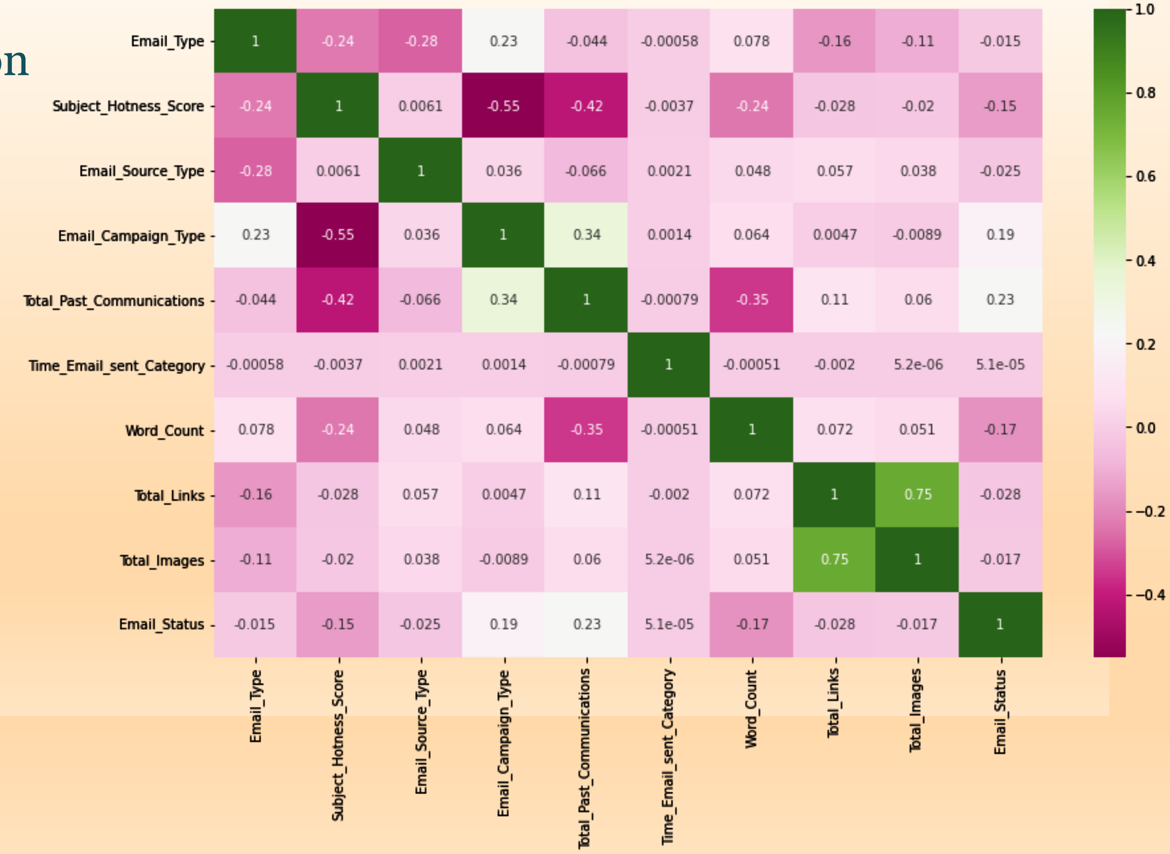




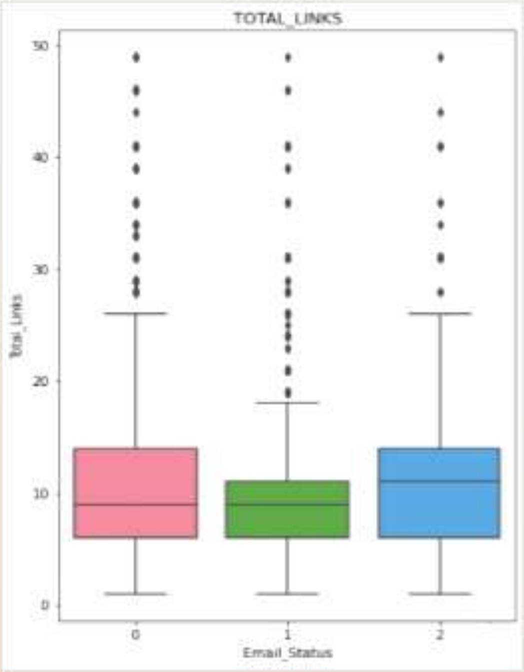
EDA

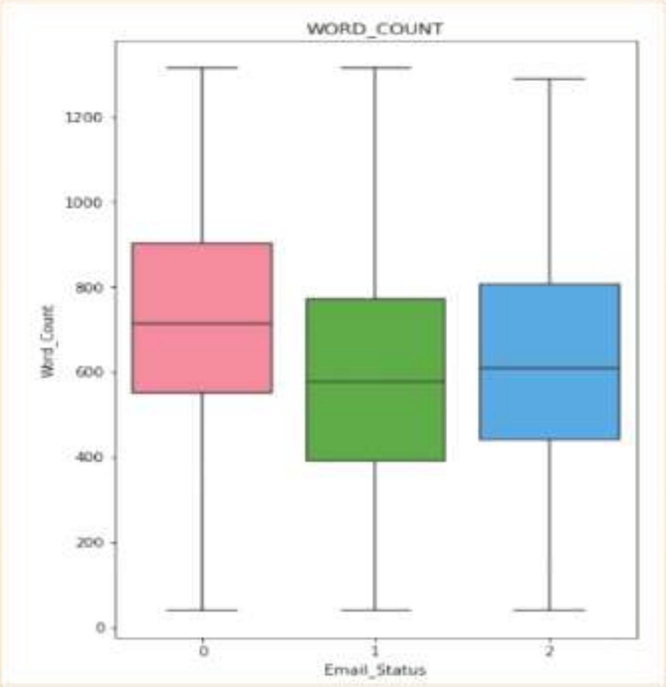
- We can observe that the correlation between Total Links and Total Images is 0.75

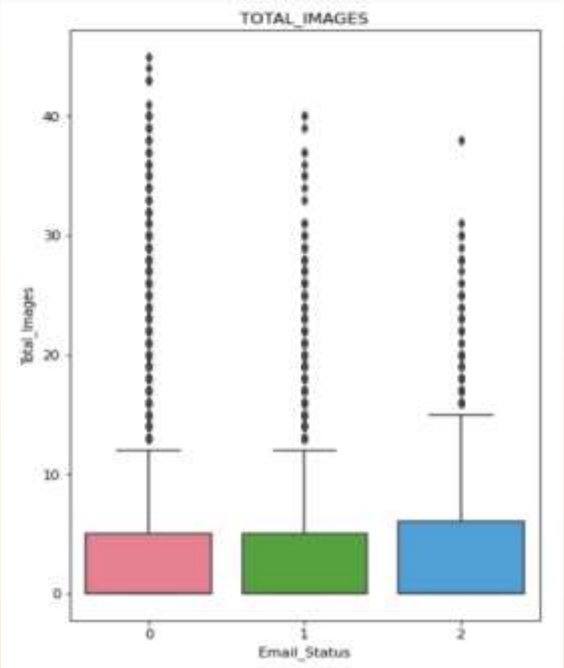
- Email Campaign Type, Total Past Communications and word count shows a good correlation with the Email status



EDA



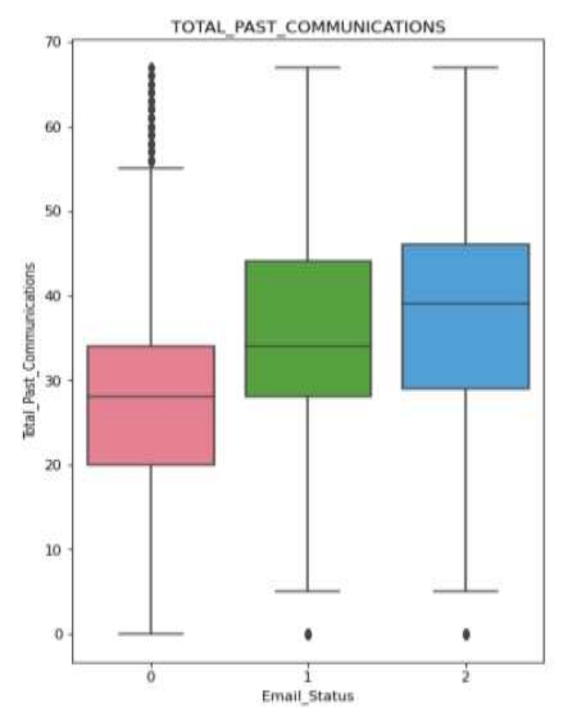


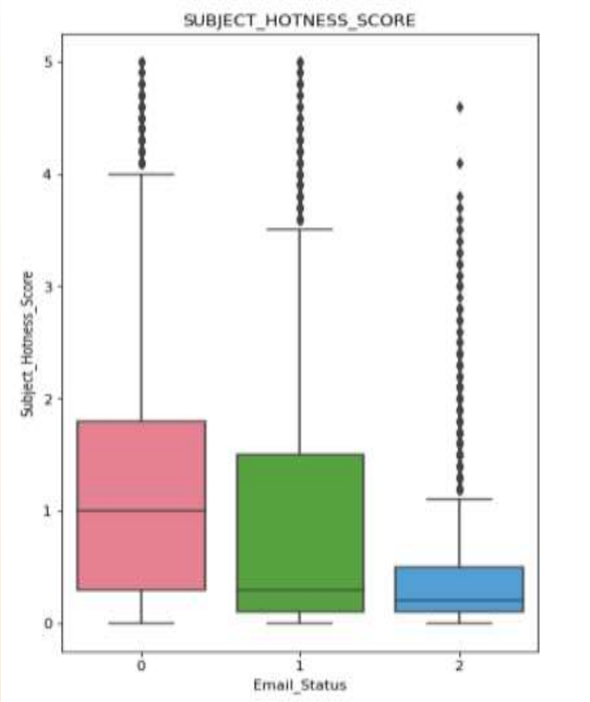


- As the word count increases beyond 600, we can see that there is a high possibility of the email being ignored. The more words in an email, the more it has a tendency to get ignored.

- We can see that increase in Total Images increases the chance of the email being ignored.

EDA

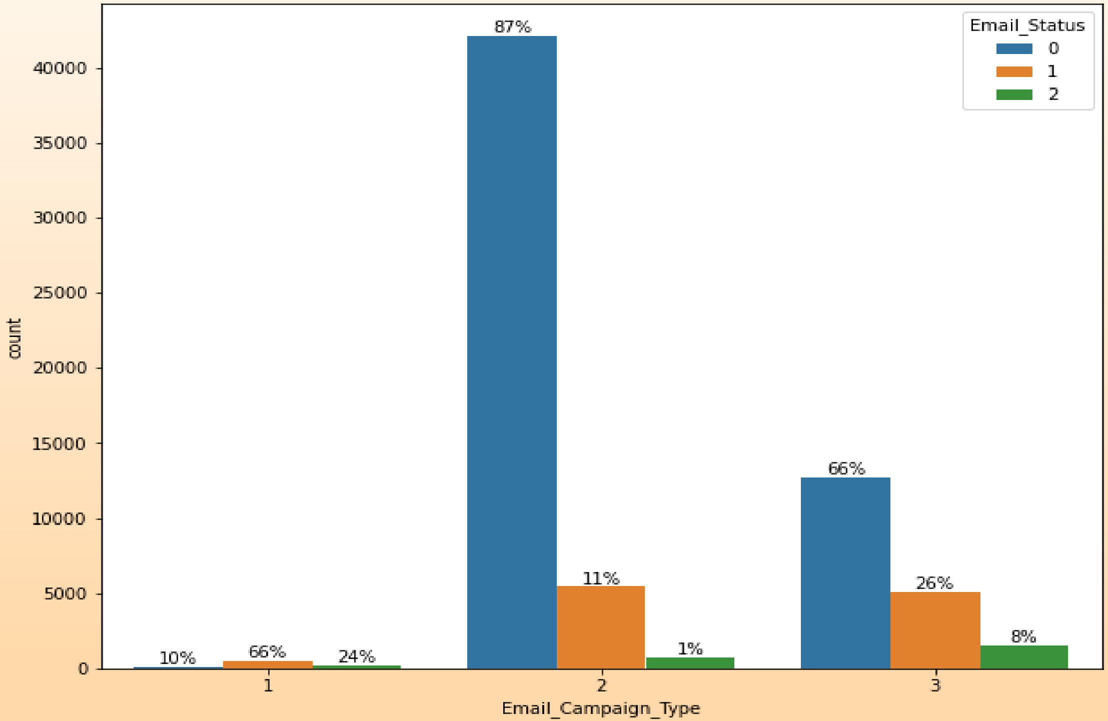




- We can see that the increase in the total past communications increases the chance of the email being read or acknowledged. This is just about making connections with customers.

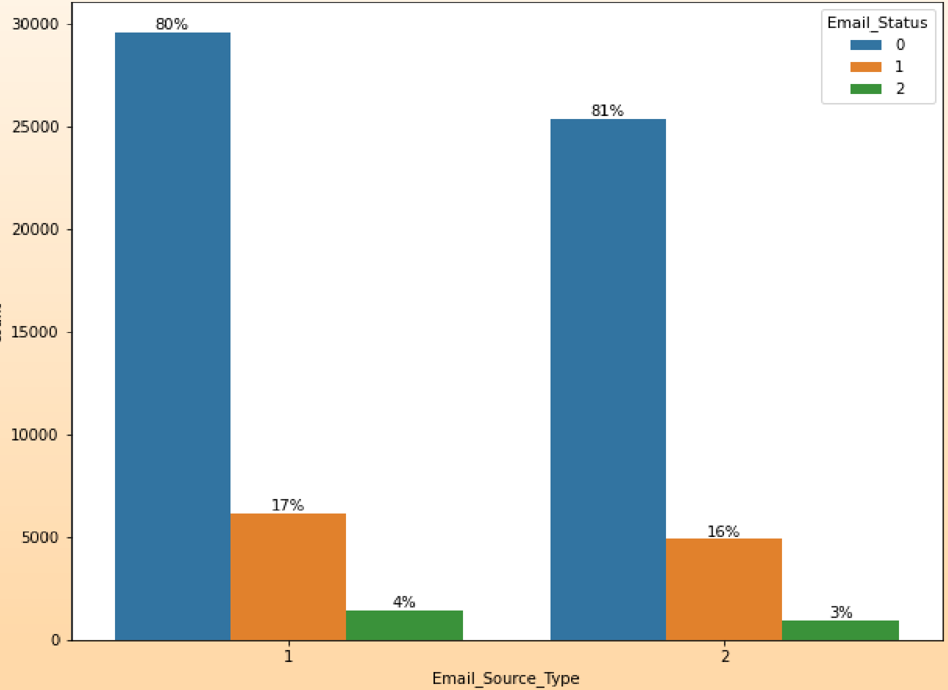
- We can see that the increase in subject hotness score increases the chance of an email being ignored and the less the subject hotness score the more the emails get read or acknowledged

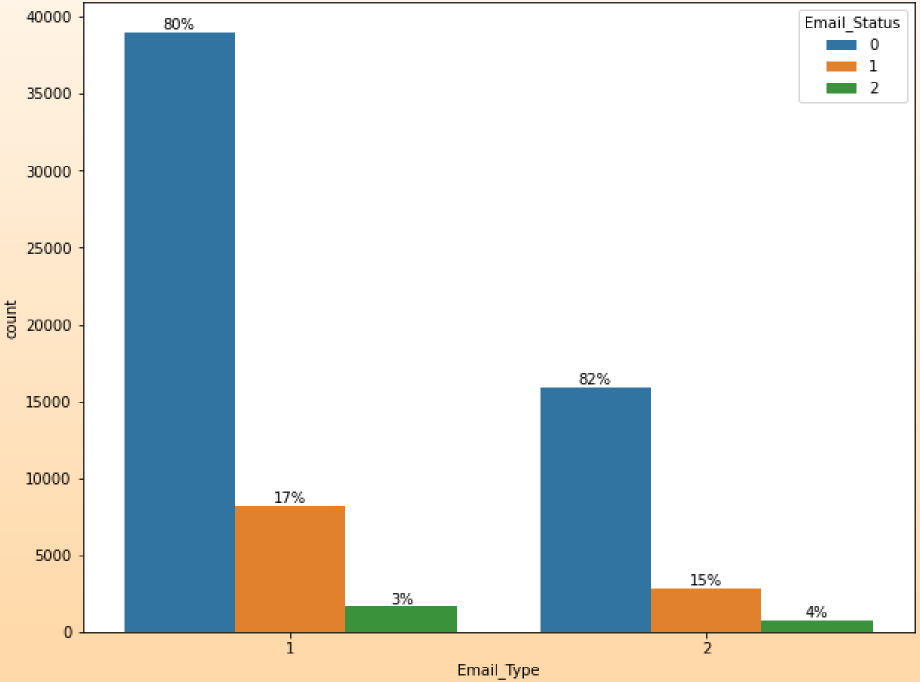
EDA (continued)



•In the Email campaign type feature, even though the number of emails sent through campaign type 1 is very few they have a high possibility of getting read. Most emails were sent from campaign type 2 and 80% of them are ignored. Seems like campaign 3 was a success because even when less number of emails were sent under campaign 3, more emails were read and acknowledged.

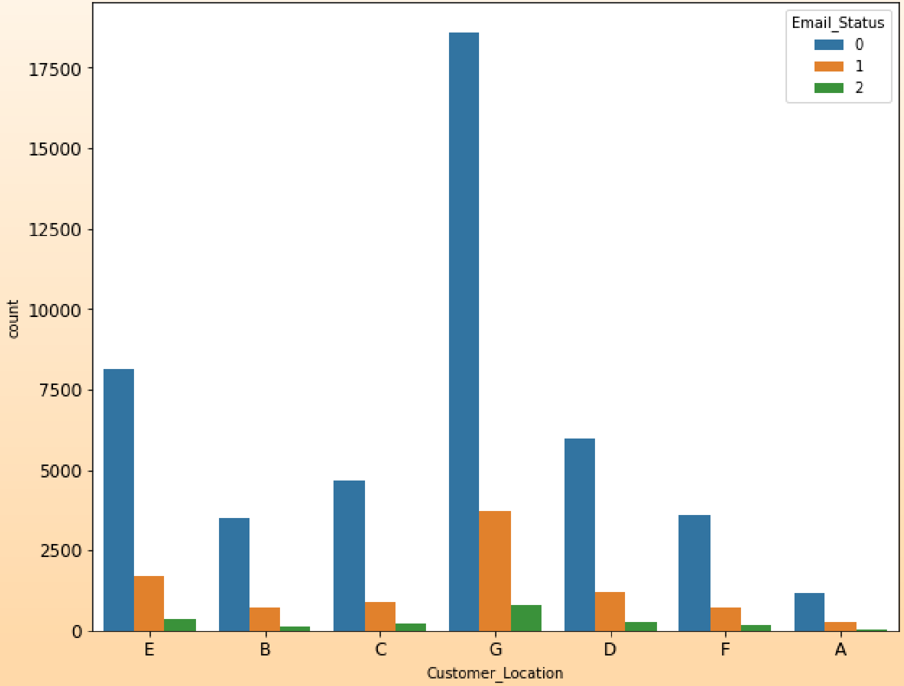
EDA (continued)

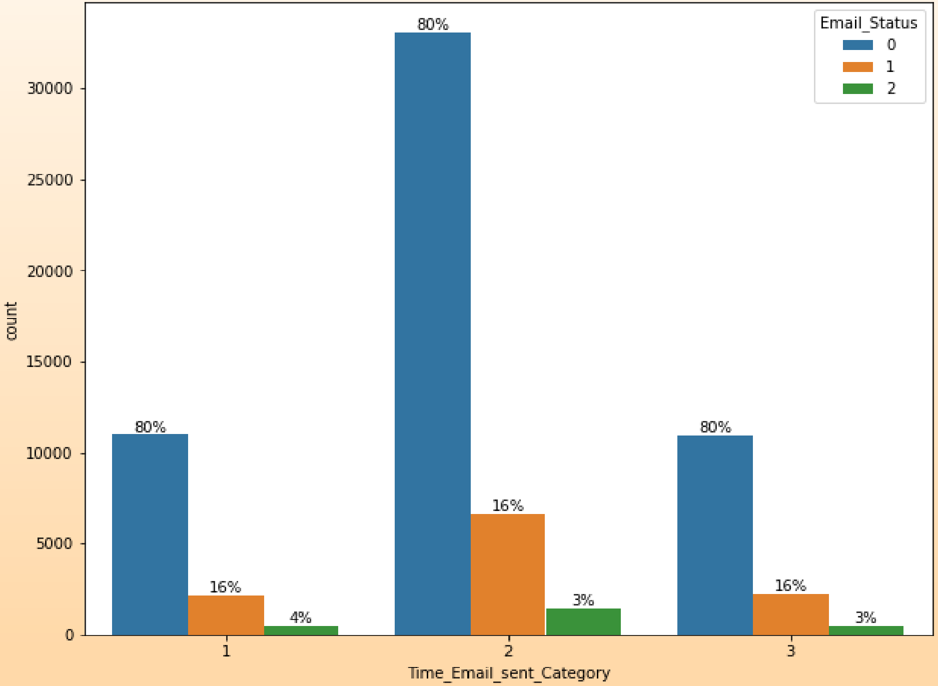




- The emails of type 1 are sent more than the email of type 2, but the proportion of ignored, read, and acknowledged emails are same for the both email types.
- Both the email source types have a similar proportion of ignored, read and acknowledge emails This shows that the email source type has nothing to do with email status.

EDA (continued)





- We found that the Customer_location and Time_email_sent_category have nothing to do with Email_Status. we came to the conclusion that the email being Ignored, Read, or Acknowledged is the same irrespective of the customer’s location and the time at which the email was sent.

Preparing dataset for modelling

ype_1	Email_Campaign_Type_1	Email_Campaign_Type_2	Email_Campaign_Type_3	Total_Past_Communications	Word_Count	Links_Images	Time_Email_sent_
0	0	1	0	33.00000	440	8.0	
1	0	1	0	15.00000	504	5.0	
1	0	0	1	36.00000	962	5.0	
0	0	1	0	25.00000	610	16.0	
0	0	0	1	18.00000	947	4.0	
1	0	1	0	28.93325	416	11.0	
1	0	1	0	34.00000	116	4.0	
0	0	1	0	21.00000	1241	8.0	
1	0	1	0	28.93325	655	15.0	
1	0	1	0	40.00000	655	11.0	

Task:- Multi-Class Classification
Train Set:- (128406, 12)
Test Set:- (13383, 12)
Response:- 0,1 or 2

Model Building (Baseline Model)

	KNeighborsClassifier_SMOTE
Accuracy_train	0.884694
Accuracy_test	0.601509
Train_recall	0.884694
Test_recall	0.601509
Train_precision	0.892543
Test_precision	0.751687
Train_f1	0.882310
Test_f1	0.656714
Train_auc	0.984006
Test_auc	0.684688

Model Validation and Selection(SMOTE)

	Accuracy_train	Accuracy_test	Train_recall	Test_recall	Train_precision	Test_precision	Train_f1	Test_f1	Train_auc	Test_auc
KNeighborsClassifier_SMOTE	0.884593	0.600239	0.884593	0.600239	0.892496	0.750208	0.882200	0.655831	0.984025	0.682467
LogisticRegression_SMOTE	0.534414	0.621983	0.534414	0.621983	0.517731	0.771898	0.508988	0.678570	0.722921	0.768609
DecisionTreeClassifier_SMOTE	0.999393	0.700889	0.999393	0.700889	0.999393	0.732509	0.999393	0.715420	1.000000	0.616592
RandomForestClassifier_SMOTE	0.611225	0.682059	0.611225	0.682059	0.603058	0.777285	0.597563	0.720829	0.800027	0.773785
XGBClassifier_SMOTE	0.616171	0.701711	0.616171	0.701711	0.605183	0.774642	0.602516	0.732155	0.807768	0.776387
DecisionTreeClassifier_Tuned_SMOTE	0.784597	0.714414	0.784597	0.714414	0.782897	0.746030	0.783300	0.728953	0.931905	0.719412
RandomForestClassifier_Tuned_SMOTE	0.603274	0.681237	0.603274	0.681237	0.594396	0.777370	0.589227	0.720340	0.795756	0.773608
XGBClassifier_Tuned_SMOTE	0.790571	0.774639	0.790571	0.774639	0.790439	0.760084	0.783933	0.766052	0.924655	0.779019
DecisionTreeClassifier_Grid_Tuned_SMOTE	0.787697	0.717702	0.787697	0.717702	0.785985	0.746862	0.786329	0.731111	0.933401	0.720217
RandomForestClassifier_Grid_Tuned_SMOTE	0.605260	0.678473	0.605260	0.678473	0.596344	0.775819	0.590479	0.717988	0.795463	0.772858

Model Validation and Selection (continued)

Observation 1: K-nearest neighbor(KNN) is performing well, but it's not giving good results when compared to the remaining models.

Observation 2: Previously the decision tree is overfitting the data but after the hyperparameter tuning, we have avoided the problem of overfitting

Observation 3: Logistic Regression, Random Forest, and XGBoost were performing well and gave better results when compared to previous models.

Model Validation and Selection (continued)

Observation 4: After the hyperparameter tuning, Random Forest and XGB have the best F1 and AUC scores.

Observation 5: From the above observations, we have concluded that we would choose our model from Random Forest Classifier or XGB Classifier

Model Validation and Selection (continued)

We had chosen Random Forest Classifier for our prediction and the best hyperparameters obtained are as below.

criterion='entropy',
max_depth= 8,
min_samples_leaf=20,
min_samples_split=10,
min_child_weight=1,
max_features='auto'
n_estimators=50,
max_leaf_nodes = None
max_samples = None
min_impurity_decrease = 0.0
min_weight_fraction_leaf = 0.0
Min_impurity_split=None

Model Validation and Selection (continued)

We had chosen Extreme Gradient Boosting Classifier(XGBoost) for our prediction and the best hyperparameters obtained are as below.

colsample_bylevel= 1,
colsample_bynode=1
colsample_bytree=1, gamma=0,
learning_rate=0.1, max_delta_step=0,
max_depth= 8, min_samples_leaf=20,
Min_samples_split=25,min_child_weight=1,
missing=None,
n_estimators=100,
n_jobs=1,
nthread=None, num_boost_round=10
objective='multi:softprob
random_state=0, reg_alpha=0,
reg_lambda=1, scale_pos_weight=1

Model Evaluation(Hyperparameter tuned)

XGBoost

Classification Report

	precision	recall	f1-score	support
0	0.86	0.90	0.88	10700
1	0.40	0.31	0.35	2208
2	0.13	0.14	0.13	475
accuracy			0.77	13383
macro avg	0.46	0.45	0.45	13383
weighted avg	0.76	0.77	0.77	13383

RandomForest

Classification Report

	precision	recall	f1-score	support
0	0.90	0.77	0.83	10700
1	0.31	0.32	0.31	2208
2	0.12	0.51	0.19	475
accuracy			0.68	13383
macro avg	0.44	0.53	0.44	13383
weighted avg	0.78	0.68	0.72	13383

Model Evaluation(Hyperparameter tuned)

XGBoost

```
Confusion matrix

[[9620  832  248]
 [1313  680  215]
 [ 221  187   67]]
```

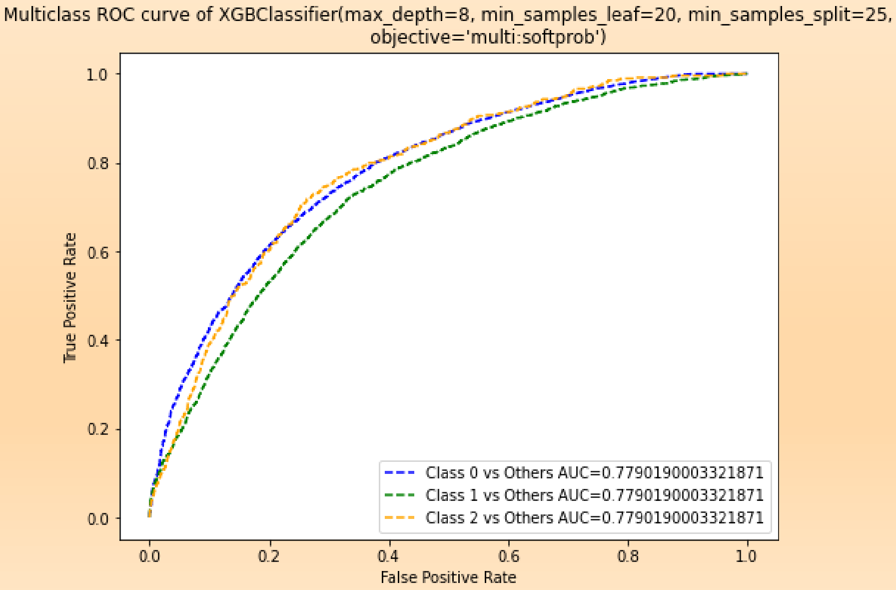
RandomForest

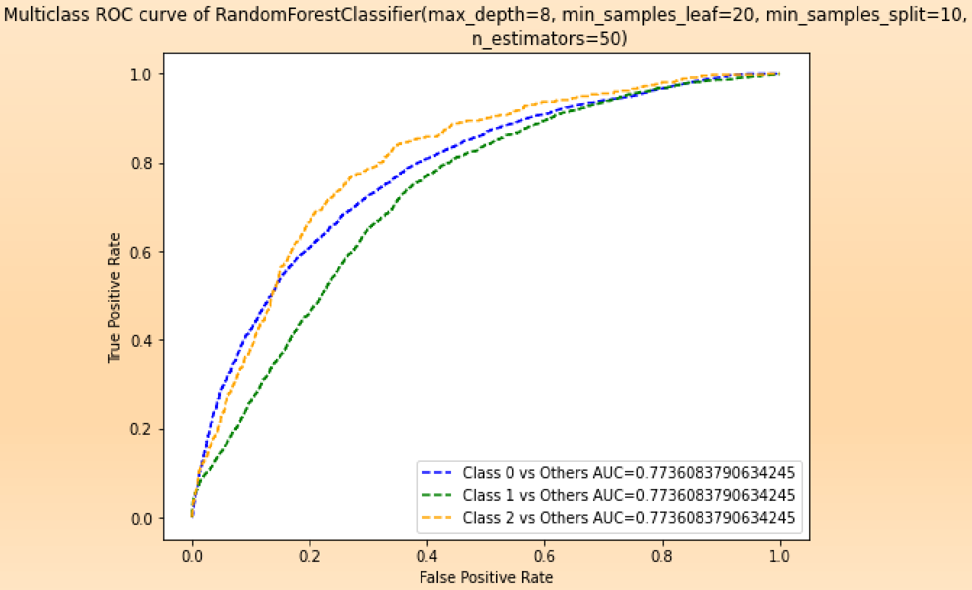
```
Confusion matrix

[[8168 1399 1133]
 [ 785  709  714]
 [ 104  131  240]]
```

Model Evaluation(ROC Curve)

XGBoost

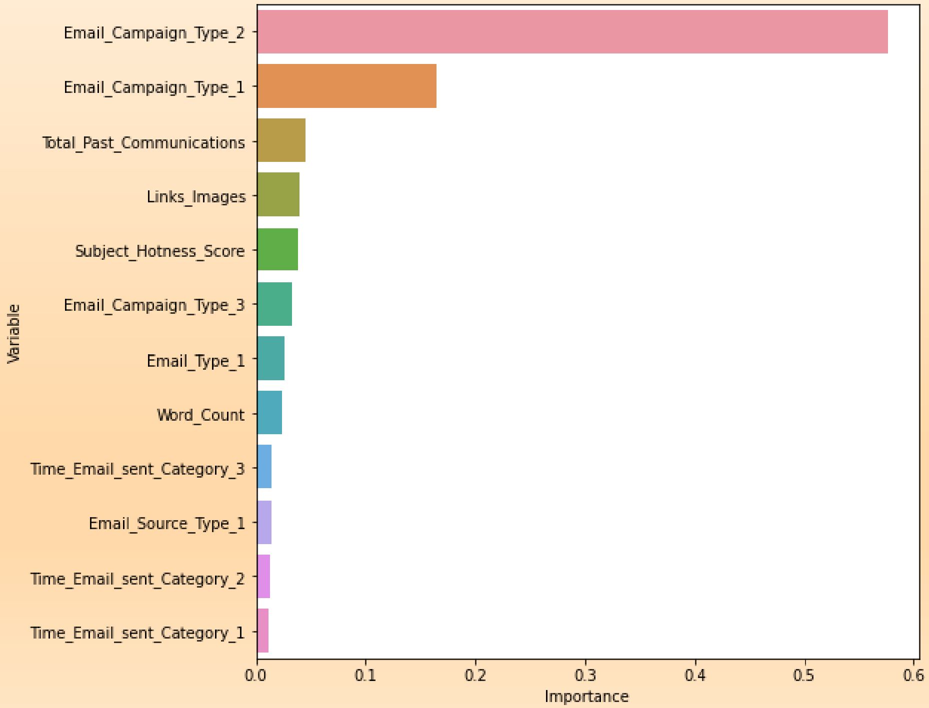
RandomForest



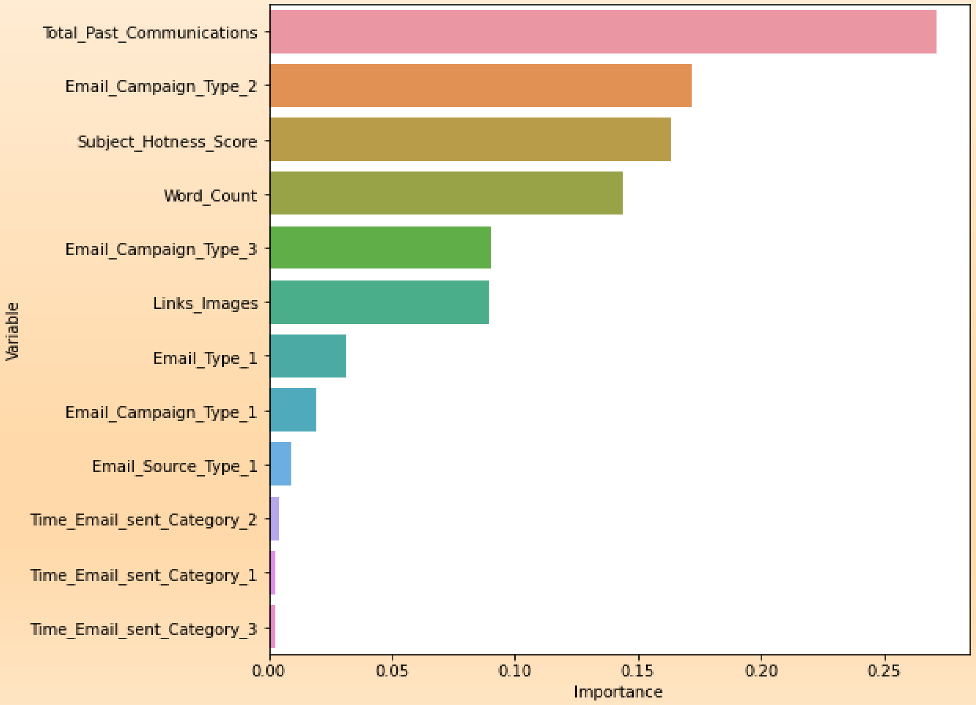


Model Evaluation(Feature Importance)

XGBoost



RandomForest



Conclusion

- In EDA, we observed that the Email campaign type was the most important feature. Even though there are very few emails sent from Email campaign type 1 there is a very high possibility of getting read.

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

- The more the number of Total past communications, the more it leads to reading and acknowledging em ails. Therefore, having a healthy relationship with customers is a big yes.

- More images were there in ignored emails.

Conclusion

- For modeling, it was observed that for imbalance handling Oversampling i.e. SMOTE worked considerably better than undersampling as the latter resulted in a lot of loss of information.

- The Decision tree is overfitting both UnderSampled and Smote data. It is working great on train data and worse on test data, but after the hyperparameter tuning, we avoided the problem of overfitting.

- The Email campaign type, Total past communications, and word count were found to be the most relevant features in predicting the status of the email. Based on the metrics, XGBoost Classifier and Random Forest Classifier worked the best, giving an F1 score of 77% and Auc score of 78%, an F1 score of 72%, and an Auc score of 77% respectively.

Challenges

- Choosing the appropriate technique to handle the imbalance in data was quite challenging as it was a tradeoff b/w information loss vs the risk of overfitting.
- Overfitting was another major challenge during the modeling process.
- Understanding what features are most important and what features to avoid was a difficult task.
- Decision-making on missing value imputations and outlier treatment was quite challenging as well.

Thank You