Capstone Project Submission

Instructions:

i) Please fill in all the required information.

ii) Avoid grammatical errors.

Team Member's Name, Email and Contribution:

Yogesh. K,
Yogeshiaf399@gmail.com
Individual Project.

Please paste the GitHub Repo link.

Github Link:- https://github.com/Yogeshkrishn/Email-Campaign-Effectiveness-Prediction.git

Please write a short summary of your Capstone project and its components. Describe the problem statement, your approaches and your conclusions. (200-400 words)

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.

SUMMARY

- 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.
- In addition to the ML Model prediction, we also analyzed what all features can help us in getting the Email status to be not ignored by the customers.

APPROACHES INVOLVED

- Data collection
- Data preparation
- Exploratory data analysis
- Feature Engineering
- Handling Imbalanced dataset
- Working different models and Evaluating model

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 modelling, it was observed that for imbalance handling Oversampling i.e. SMOTE worked way better than undersampling 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 89% and test score of 81% for F1 score.