**APPROACH – JOBATHON (AUG’22)**

**Problem Statement:**

Most organizations today rely on email campaigns for effective communication with users. Email communication is one of the popular ways to pitch products to users and build trustworthy relationships with them.

Email campaigns contain different types of CTA (Call To Action). The ultimate goal of email campaigns is to maximize the Click Through Rate (CTR).   
CTR is a measure of success for email campaigns. The higher the click rate, the better your email marketing campaign is. CTR is calculated by the no. of users who clicked on at least one of the CTA divided by the total no. of users the email was delivered to.

The task at hand is to build a machine learning-based approach to predict the CTR of an email campaign.

**Dataset**

The train dataset contains 1888 rows and 21 columns (including target). The test set contains 921 rows and 20 columns

The following features are provided in the training & test set: -

*campaign\_id, sender, subject\_len, body\_len, mean\_paragraph\_len, day\_of\_week, is\_weekend, times\_of\_day, category, product, no\_of\_CTA, mean\_CTA\_len, is\_image, is\_personalised, is\_quote, is\_timer, is\_emoticons, is\_discount, is\_price, is\_urgency, target\_audience*

Target Variable is *click\_rate* (for the email campaign)

**Approach:**

The inspection of the dataset was carried out to determine the main features of the dataset. It was found to have 8 numerical & 13 categorical variables (out of which 5 are Boolean flags). Some of the columns like sender, product, category & target\_audience were found to have some classes which are present only in the train/test set. This reduces the predictive power of these columns. On the other hand, numerical column like subject\_len, body\_len & mean\_paragraph\_len were found to be positive skewed with only a weak correlation with the target.

For the purpose of clean up, I removed the campaign\_id column (which is unique for each row) and is\_timer (which had only one value – 0). The non-zero values in “is\_price” column (only 13 in number) were converted to 1. Apart from this, I also tried feature scaling & transformation were tried for the numerical columns, but they had no effect on the performance of the final model. This dataset doesn’t give much scope for new feature creation and although I tried to create some new features (both interaction features & polynomial features) from the subject\_len, body\_len, mean\_paragraph\_len & no\_CTA\_len, they didn’t increase the performance finally.

The cross validation technique I used for model evaluation was K-Fold & Stratified K-Fold with K=3 to K=5. Since Stratified K-Fold is expected to give a better assessment of model performance, I have used the concept of target bins (1-5) to implement it.

I divided the models into two categories – single leaner models (Linear Regression, Ridge& Lasso Regression, SVR & KNN) and ensemble learners (RandomForest, XGBoost, Gradient Boost, LGBM & Catboost). The ensemble models outperformed the single models in all cases by a large margin (in R square metric & RMSE). Feature scaling, transformation & addition of interaction features had some effect in increasing the performance of linear models (R^2 upto 0.25), however there was no significant change in performance for ensemble models.

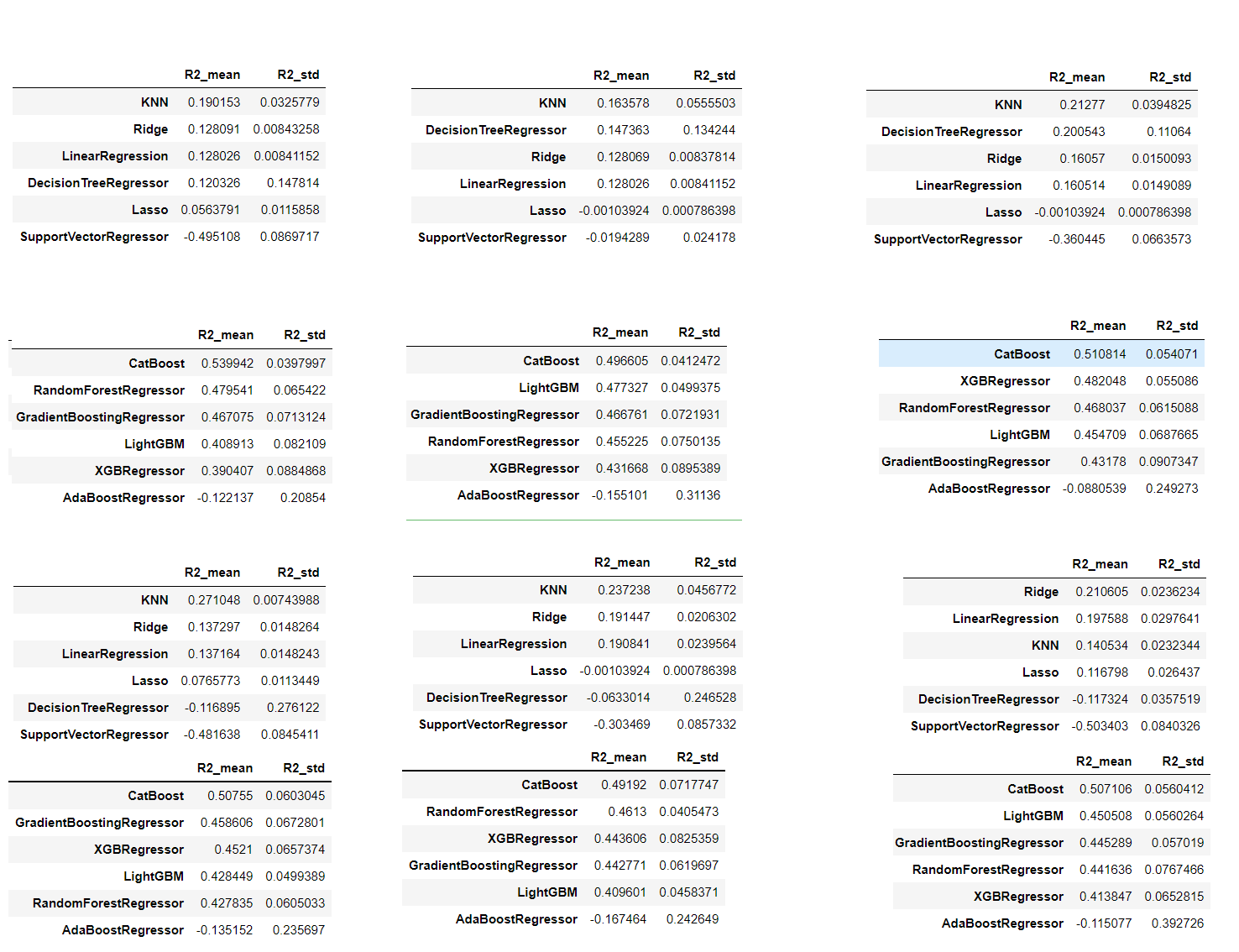
Among the Ensemble models, Catboost & Random Forest had the best performance. However during the various cross validation runs, there was a large variation in scores obtained which could be due to the small size of the data set. The highest test score that could be obtained was 0.65 using CatBoost.

In order to reduce the variance of scores & overfitting, I decided to try the Stacking & Voting Regressors available in the sklearn library. These work by using two or more base learners & combining the predictions to give a better & more consistent performance. I used the 4 ensemble models (RF, XGB, Catboost & LGBM) in Stacking & Voting Regressors & this had some effect in increasing the test score.

For the final model, I fine-tuned all the ensemble learners & again aggregated them using the VotingRegressor of sklearn. With this I was able

**Conclusion**

Of all the models, the voting ensemble classifier gave the best performance & R2 score of 0.67. This maybe because it aggregates the result of multiple model, hence it makes better predictions on average. The most significant features which predict the *click\_rate* are body length (*body\_len*), no of calls to action (*no\_of\_CTA*) and mean length of CTA (*mean\_CTA\_len*).

****