Approaches for Credict Card Lead predictions case:

**Approach on High level**

* Data pre-processing, analysis and model building.
* Tried various classical machine learning algorithms like Logistic Regression, SVM, Random Forest and XGBOOST, in which XGBOOST outperform.
* XGBOOST with parameters {'subsample': 0.75, 'n\_estimators': 100, 'max\_depth': 5, 'learning\_rate': 0.05, 'colsample\_bytree': 0.75} being used.
* Tried A Neural Network too, with 3 hidden dense layer, in which all activation units are relu and optimizer is Adam, perform very similar results like XGBOOST.

**FEATURE ENGINEERING:**

* The are NaN values in the Credit\_Product feature in both train and test data. When relationship of NaN compared with the target, I find that missing values has a good correlation with targets, so I create a new category in credit\_product as missing.
* Convert AGE into month.
* Create a new feature with AVG\_ACCOUNT\_BALANCE / AGE, which shows relationship of balance with a person age.
* Create a new feature with AVG\_ACCOUNT\_BALANCE / VINTAGE, which shows relationship of balance with a how old the account of a person with a bank.
* Create a new feature with AVG\_ACCOUNT\_BALANCE / AGE, which shows relationship of balance with a person age.
* Create a new feature with binning the AGE feature in to three bins, Young, Middle, Older, Youngs are below 30, Middles are between 31 to 50, Older are above 50.
* Create a new feature with binning the Vintage feature in to three bins, New, Trust, Faith, New are the accounts which are opened in less than 12-month, Trust are the accounts which are between 12 months to 60 months, Faith are the oldest accounts above 60 months.
* Create a new feature with binning the AVG\_ACCOUNT \_BALANCE feature in to five lower, middle, upper-middle, upper, rich. The reason behind common understanding of account balances a person averagely maintain (assumed the data is of Indian consumer). This help to categories the customer into segments as per their account balance. So, I binned

lower 🡪 0 to 2 lakhs,

Middle 🡪 2 lakhs to 5 lakhs

Upper-middle 🡪 5 lakhs to 9 lakhs

Upper 🡪 9 lakhs to 15 lakhs

Rich 🡪 Above 15 lakhs.

* All categorical feature encoded with one hot encoding.
* Tried Boxcox transformation for numerical dataset, but it does not impact well,

MinMaxScalar perform very well so I used it to standardised it.

**Model & Score:**

* XGBOOST with parameters {'subsample': 0.75, 'n\_estimators': 100, 'max\_depth': 5, 'learning\_rate': 0.05, 'colsample\_bytree': 0.75} being used.
* ROC\_AUC\_SCORE for my model is 0.8630787202.

Submitted By:

Zishaan Khan

Email: [zkzshan@gmail.com](mailto:zkzshan@gmail.com),

Phone / whatsapp: 9522228922