**Problem Statement:**

The client is Happy Customer Bank which deals with different kinds of banking products, like Savings accounts, Current accounts, investment products, credit products, among other offerings.The main purpose of the bank is that it can do the cross sales of their credit products to their existing customers.

So, as a Decision Scientist our objective is to find the probabilities of the customer interested in getting a credit card from the bank.

**Solution Methodology :**

* **Data Cleaning and data manipulation.**

1. **Check and Handle duplicate data**

**Approach :** No duplicate data was observed.

1. **Check and Handle *NAN*(Null) values.**

**Approach :** Replaced Null values in the categorical column Credit\_Product with text ‘N.A.’. The null value was not replaced by mode of this column as this might have lead to incorrect prediction. And also if we replace that with mode, then most of the predictions would have been trained wrong with the label ‘No’, even if they were meant to ‘Yes’.

* **Exploratory Data Analysis & Feature Engineering**

1. **Univariate data analysis :**
   * **Univariate analysis of some categorical variables :**

Chart, bar chart

Description automatically generated

**Observation :**

* We can clearly see number of people who did not opt for credit card is almost 3 times more than the number of people who did. There can be class imbalance issue here.
* Count of non-active people in last 3 months is more than the count of active people in the bank.
* No. of male members are more than the no. of female members.
  + **Univariate analysis of Numerical variables variables :**

Chart, histogram

Description automatically generated

**Observation :**

* We can clearly observe maximum number of people are in the approx. range 20+years upto 40 years. Minimum value of age is 23, while maximum value of age is 85.
* Average account balance(Avg\_Account\_Balance column) distribution plot is little right skewed. The minimum value is 20790 and maximum value is 10352009.

Also, we can see the distribution in this column is not a Normal Distribution. So, in order to correct that, applying a **log transformation** would be a great idea.

* The ‘Vintage’ column has the maximum values in the range 7 and around 40. The minimum and maximum vintage values are 7 and 135 respectively.

1. **Outlier analysis :**

* **Outlier analysis of Numerical variables :**

Chart, box and whisker chart

Description automatically generated

**Observation :**

* We can’t see any outliers in any variables, however there are some little extreme values in **Avg\_Account\_Balance\_log**(generated from ***Avg\_Account\_Balance***), specially at the left side of the boxplot of **Avg\_Account\_Balance\_log** column. Hence, we can do a lower capping to **01-percentile** in **Avg\_Account\_Balance\_log** column. as **Avg\_Account\_Balance\_log** has good influence on the **Target variable**.

1. **Bivariate/Multivariate data analysis :**

* **Bivariate analysis of some categorical variables :**
* **Then went ahead with countplots of different categorical columns with Target variable.**
* **Then with the pairplot and Heatmap to check the relation and correlation coefficients among different variables.**

1. **Class Imbalance Analysis :**

**Approach:**

Text

Description automatically generated

* So, we can see the customers not taking credit cards are 3 times more than the ones taking.

But in real world scenario, the data actually holds true values, as the **no. of people Not taking the credit cards will always be much more than people actually taking it**. So, performing any sampling techniques could lead to in inaccurate results.

1. **Label Encoding the categorical variables :**

* Label Encoding was done on the categorical variables in Train and Test dataframe.

1. **Feature Scaling :**

* **After the Train & Test split, StandardScaler(), scaling method was used.**
* **Classification Modelling Technique :**
* I have used four models, which are:
  + 1. Logistic Regression.
    2. Decision Tree Classifier.
    3. Random Forest Classifier.
    4. Extreme gradient boosting(XGBoost).
* Evaluated these models on the basis of roc\_auc score which is the best evaluation metric when Probability has to be predicted.

1. Logistic Regression :
   * Used simple logistic regression model with the params like class\_weight= ‘balanced’, which can take care of the class imbalance issue.
   * Achieved a roc\_auc score of 73% in Train as well as in Test data.
2. Decision Tree Classifier :
   * Used a model to improve the roc\_auc score, achieved the roc\_auc score of 100% in Train data and 70.5% in the Test data.
   * Clearly this is the case of overfitting in the model.
3. Random Forest Classifier :
   * Used a Random forest classifier to overcome the overfitting challenge from previous DT model.
   * Achieved a roc\_auc score of 99.9% on training data and 86% on the test data.
   * Cleary the score has improved from previous model but still this is also the case of overfitting model.
   * Went ahead with some Hyperparameter Tuning technique like GridSearchCV to find the best parameters for RandomForestClassifier().
   * When the new parameters were used the roc\_auc score changed to 88% and 87% in Train and Test data respectively, which is a great improvement in the model.
4. Extreme Gradient Boosting(XGBoost) Classifier :
   * Used a XGBoost Classifier(as this works well with the data where there is class imbalance issue) with class\_weight = ‘balanced’.
   * Achieved a roc\_auc score of 89% and 87% on Train and Test data.
   * For further improvement in the model, applied some hyperparameter tuning using GridSearchCV, to get the best parameters for this classifier.
   * After fitting the model again with new parameters, achieved a roc\_auc score of 87.8% and 87.3% on Train and Test data respectively, which is a quite decent accuracy.

* **Final Submission :**
* Submitting the final model with **Random Forest Classifier** which is **hyperparameter tuned**. As, it has descent **roc\_auc** score on the Test data, along with good **Recall score**.