DMG

Assignment 2

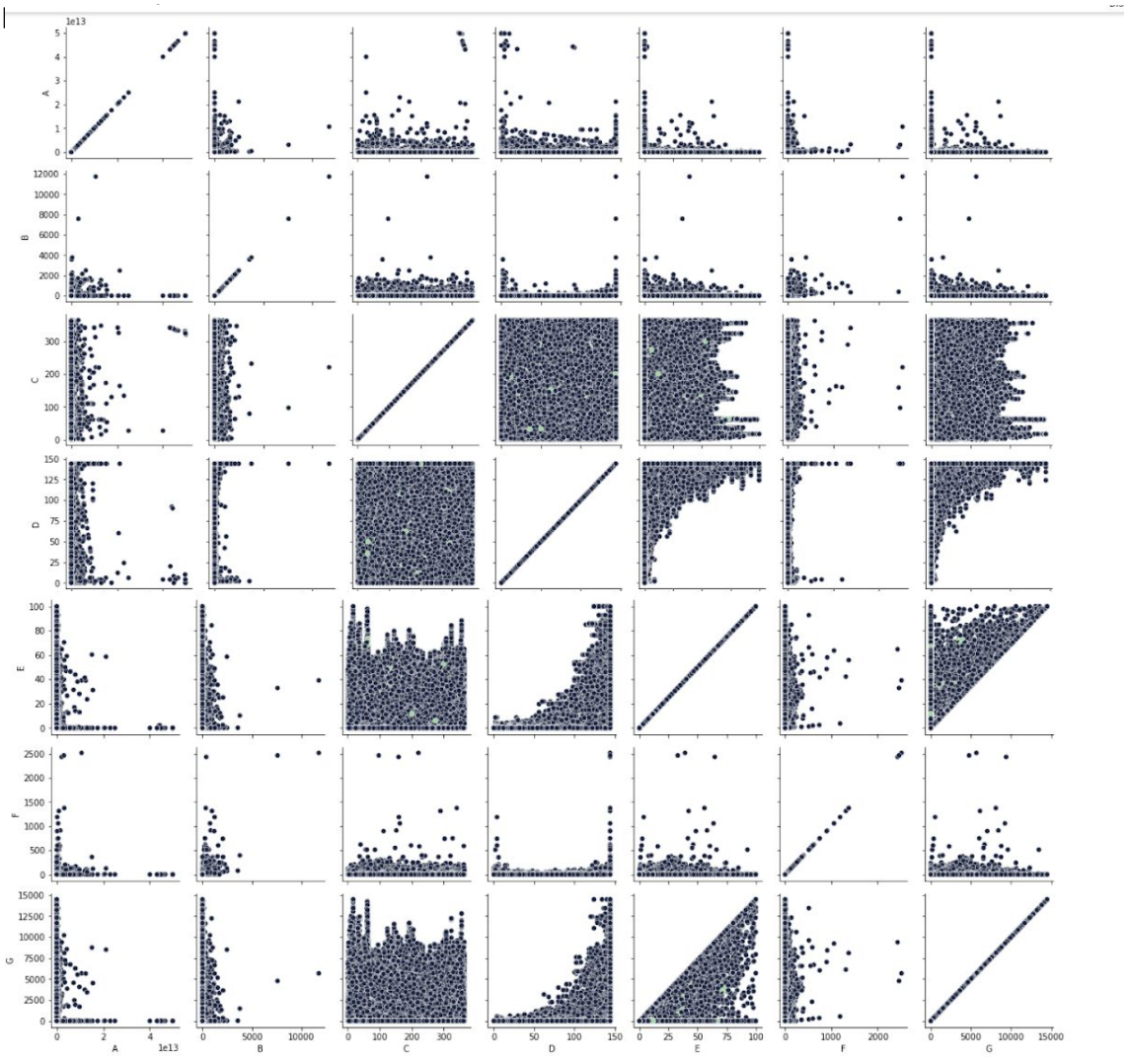
Anuj Verma (2017026)

Sakshi Saini (2017092)

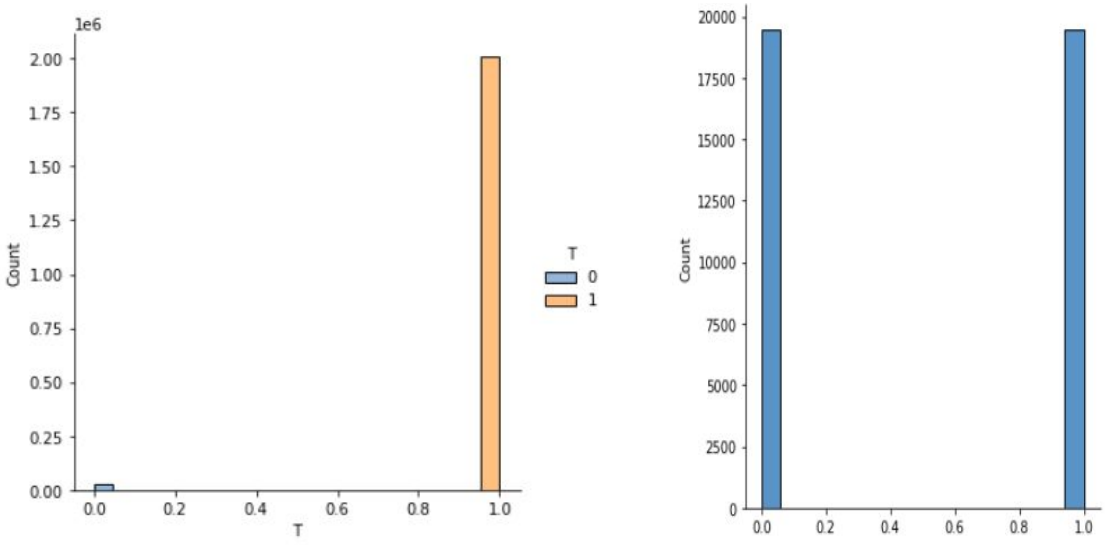
Assumptions:

1. In the runner code, I have assumed that the file "`to_predict.csv`" and the model (taken as input) exists in the same directory as the runner.py.

Data Visualization:



Data Skewness



Before / After Under-Sampling

Methodology

For Pre-Processing

1. Used undersampling

   Reason - Data Imbalance problem

```
from imblearn.under_sampling import RandomUnderSampler
rus = RandomUnderSampler(sampling_strategy='majority')
X_res, y_res = rus.fit_resample(Xtrain, Ytrain)
```

2. OverSampling

   Reason - Data Imbalance problem

```
sm = SMOTE(random_state=27, ratio=1)
Xtraine, Ytrain = sm.fit_sample(Xtrain, Ytrain)
```

After Both of the pre-processing techniques mentioned above, models were tuned.

1. Decision Tree

   Reason - Classification Problem and data is large so it will take less time to train and classify, robust to outliers and outliers. Computationally Effective.

Parameters tuned to reduce overfitting:

```
decision_tree = DecisionTreeClassifier( max_leaf_nodes = 25, max_depth = 15, class_weight='balanced', max_features=7, random_state=20, criterion='gini', min_samples_leaf=20)
```

Accuracy was around 76% after tuning.

2. Random Forest

   Reason - Decision tree can lead to overfit also and may give a local optima but when we use ensemble method we consider the best of all the local optima values so this can be helpful. Since it does not do sampling of data and it only takes sqrt(n) features it does not get the actual distribution of data which is in the minority, so it is giving a lesser AUC score than the decision tree.

```
rfclass = RandomForestClassifier(random_state=0, max_depth = 20,
class_weight={0: 100, 1: 1},
                    n_jobs=4, n_estimators= 101,verbose = 1)
rfclass.fit(X_res, y_res)
```

Accuracy was around 70%

3. Bagging

   Reason - Decision tree can lead to overfit also and may give a local optima but when we use ensemble method we consider the best of all the local optima values so this can be helpful. Since it is using the sampling of data technique along with feature selection so it is giving better AUC score than the decision tree. Since data is highly imbalanced so after oversampling or undersampling the decision tree can lead to overfit but bagging does less overfitting than the decision tree.

```
bag_tree = BaggingClassifier(
        DecisionTreeClassifier( max_leaf_nodes = 25, max_depth = 15, class_weight='balanced',max_features=7, random_state=20, criterion='gini', min_samples_leaf=20), oob_score= True, verbose = 1, bootstrap_features = True, n_jobs = 4, n_estimators = 100 )
```

Bagging is done with a Decision Tree as Base Estimator.
Accuracy was 79% after tuning.

Used, BalancedBaggingClassifier

```
from imblearn.ensemble import BalancedBaggingClassifier
bal_tree = BalancedBaggingClassifier(
                        DecisionTreeClassifier(max_depth=25,    random_state=20,
min_samples_leaf=20),
            oob_score= False, verbose = 1, bootstrap_features = True, n_jobs = 8,
n_estimators = 51, warm_start = False)
```

Tried BalancedRandomForestClassifier

```
from imblearn.ensemble import BalancedRandomForestClassifier
balencedRF = BalancedRandomForestClassifier(max_depth=50,random_state=0)
balencedRF.fit(Xtrain, Ytrain)
```

4. Boosting

Also used, predict_proba to get the prediction in terms of probabilities instead of only discrete values. This also had a significant change in accuracy.
Same reason as bagging.

```
gradboost = GradientBoostingClassifier(max_depth = 8, n_estimators = 250,
validation_fraction= 0.2, subsample= 0.8,max_features = 'auto',warm_start = True,
min_samples_splitz= 3)
```

5. Regressors

Regressors were used to get the probabilistic values, they increased the accuracy significantly.

```
from sklearn.ensemble import GradientBoostingRegressor
gradboostreg = GradientBoostingRegressor(max_depth = 8, n_estimators =
```

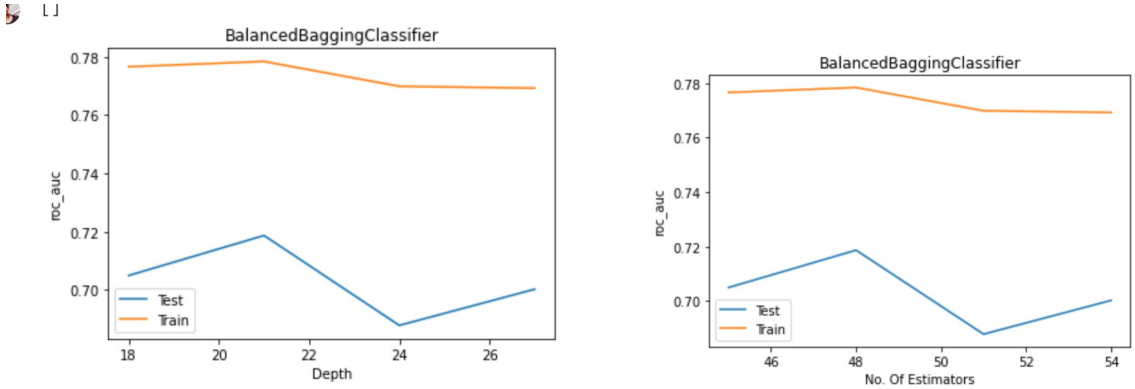
```
350,validation_fraction= 0.25,subsample = 0.9 )
```

```
model=XGBRegressor(n_jobs=4,max_depth=5,objective='reg:logistic',learning_rate
=0.01,subsample=0.9,n_estimators=5000,reg_alpha=0.005,
min_child_weight=6,colsample_bytree=0.9,seed=27)
```

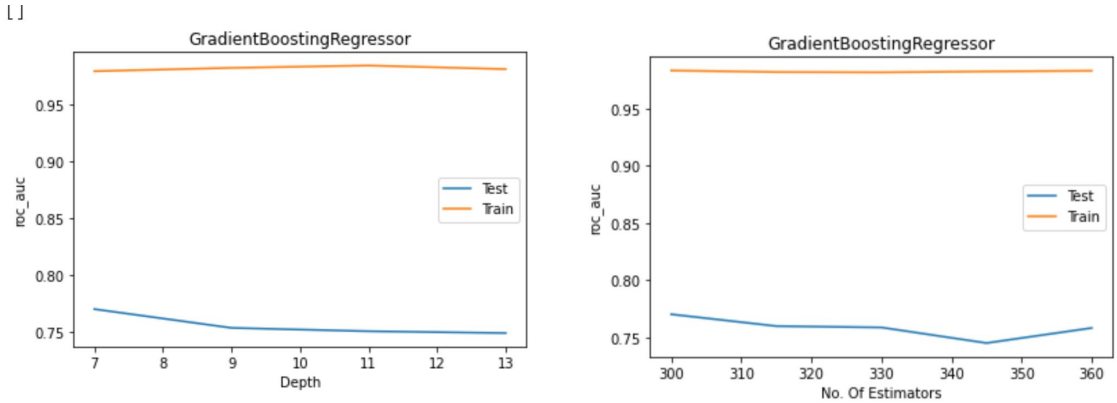
Best Models

Plotted the effect of Depth and Number of Estimators with the roc_auc.

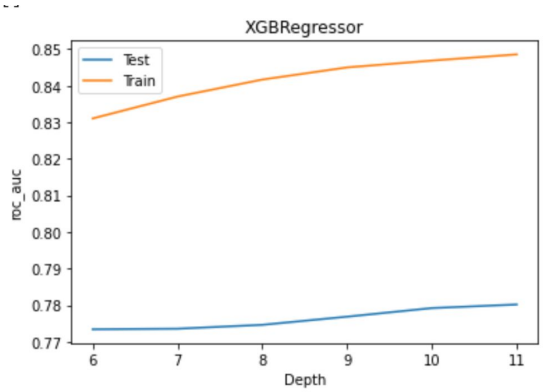
1. BalancedBaggingClassifier

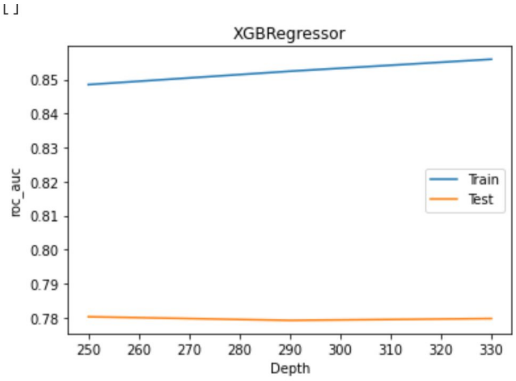


2. GradientBoostingRegressor



3. XGBRegressor





Drive Link of the saved models and csv files :
https://drive.google.com/drive/folders/1Mgj2WPaEZAe-rQ-idhSRb1IzwI6-KLit?usp=sharing

Learning

1) How to do data visualization and find the significant relations between attributes by observing the distribution of the data set.
2) How to handle the imbalance problem using different preprocessing techniques .
3) How to select the model for such imbalance data.
4) Accuracy is not important in the case of imbalance problems; we have to depend on the AUC score and f1-score for these types of data.
5) Importance of probability for calculating the AUC.
6) Sampling techniques have significant effect on performance when dataset is imbalanced.