CELL PHONE CHURN PREDICTION PROBLEM

The Machine Learning HW5 is a classification problem where we must predict whether churn is 0 or 1.

**EDA and Preprocessing**

Shape of training data - (1695, 24)

Shape of testing data - (375, 24)

We found no missing values.

We observed that there were categorical variables, and we performed encoding on them.

['CONVB', 'SEX', 'SPORTS', 'ARTS', 'TRAVEL', 'EDUC']

The training data class distribution

0 1052

1 643

We performed basic classification models on the datasets and recorded the scores on the excel table as below.

A table with numbers and symbols

Description automatically generated

**Inference**:

The base models, DT and KNN performed the best in predicting the majority class inferred by the precision and confusion matrix, where as Linear SVC could predict all minority classes accurately as seen by it’s recall score and confusion matrix.

The MLP classifier was not great in predicting the churns in this scenario.

Performing Hyper parameter tuning on the Decision Tree model, we could identify the best parameters doing a random search:

Best Parameters: {'min\_samples\_split': 10, 'max\_leaf\_nodes': 19, 'max\_features': 22, 'max\_depth': 9, 'criterion': 'gini'}

min\_samples\_split: This hyperparameter controls the minimum number of samples required to split an internal node

max\_leaf\_nodes: This determines the maximum number of leaf nodes the tree can have.

max\_features: This represents the maximum number of features to consider when looking for the best split.

max\_depth: This parameter specifies the maximum depth of the tree. The depth of the tree is the maximum distance between the root node and any leaf node.

criterion: This is the function used to measure the quality of a split

The top 5 features we got were

A black screen with white numbers

Description automatically generated

A graph with red squares

Description automatically generated

There was no difference in the results as the base DT was pretty accurate, and doing hyper parameter tuning of few parameters we could not 100% predict the minority class. So the accuracy and other scoring methods were equal.

Using the 4 base models, we performed stacking where we combine predictions from all the models and use majority voting to get a meta model prediction.

I used a random forest as a meta model in this case- Again the result of the stacking was the same as the base DT, KNN and the hyperparameter tuned DT models.

We conclude that all these models were petty good in predicting the target values and classifying the majority class, but, we could not achieve 100% recall in predicting the minority class, as we got 2 wrong.

Overall all the DT, KNN, DT with hyper parameter tuning and stacked model performed well.