MLBA Assignment 1
Documentation

Group 18: Meetakshi Setiya, Shelly Gupta, Garmit Pant.

- **Input Files:**

The program takes paths to the training data as the user input. We have used the files provided on kaggle as our training and testing data.
For training data: train_data.csv
For testing data: test_data.csv

- **Output Files:**

The program outputs a csv file Group_18_Predictions.csv containing the predicted binary class labels (0,1) for the training data.
The name of the prediction file submitted on kaggle was different though.
Prediction file submitted on kaggle : eecbrfXpredictions.csv

- **Steps To Run the Python File:**

Make sure that all dependencies and libraries used by this program are installed on your system. A list of the required libraries are:

➔ Numpy
➔ Pandas
➔ Re
➔ Sklearn
➔ Imblearn

Use the following steps to run the python file Group_18_Code.py containing the code used for training and prediction from the terminal:

1) Run the command :
   python3 Group_18_Code.py
2) The program will ask to input the train data file path. Please provide a path to the csv training dataset.
3) The program will ask to input the test data file path. Please provide a path to the csv testing dataset.

4) After the program is completed running, it will create an output file *Group_18_Predictions_Output.csv* containing the predictions corresponding to the inputted testing dataset csv file. The program will also display the cross validation scores, classification report of the validation dataset and confusion matrix for the validation dataset.

- **Steps used for creation of model, testing and prediction:**

Step 1 - Feature Generation:

First, we tried using the composition for feature generation. Then we tried ONE HOT ENCODING technique and it gave better results as compared to composition.
We chose one hot encoding because this technique considers both the position of amino acid as well as their composition in the sequences.

We used one hot encoding without dropping X. The training data contains sequences with X

ONE HOT ENCODING:
We stored all the 20 amino acids in the VARIABLE 'codes'. The method one_hot_encode takes the sequence as input and returns the one-hot encoded sequence. We also flattened the no. of features of sequence in the method

X_train : one hot encoded train sequences
X_test: one hot encoded test sequences
y_train: train labels

Step 2 - Splitting The Training Data

We split the whole training data into train data and validation data. Validation data is 20% of training data to be used for validation and testing later.
trainX : splitted train data
validationX: splitted validation data

Step 3 - Model Selection And Training

We used models SVMClassifier, xgboost, random forest and we achieved the best model using EasyEnsembleClassifier with base estimator Random Forest Classifier.

Model Information

best model: EasyEnsembleClassifier (imblearn)
base_estimator: RandomForestClassifier
n_estimators: 10
Sampling_strategy: auto

Rest of the parameters for both Random Forest and Easy Ensemble Classifier were kept default.

We selected model EasyEnsembleClassifier with base estimator Random Forest Classifier because it improves the generalization performance for imbalanced datasets. The Easy Ensemble handles the imbalance by forming balanced samples of the training dataset by selecting all examples from the minority class and a subset from the majority class. The downside of undersampling, i.e. valuable information is not used in the training process, is overcome by creating multiple subsamples. We did not use the default base estimator since using the RandomForestClassifier as the base estimator gave the best result, thus resulting in classifier stacking.

STEP 4 - Scoring

cross validation: to check the model was not overfitted.
criteria:: Roc_auc
model efficiency: we measured the efficiency of the model using classification_report, confusion matrix.
classification report : gives precision, f1-score

The repeated stratified k fold cross validation method was used to cross validate the model. The value of k was set as 10 and the value of repeats was set as 5.
Since the dataset was very imbalanced, ill-fitted models gave a high accuracy score. Thus, we used ROC_AUC scores, checking for precision and recall to judge the model.

STEP 5 - Fit the model to whole training data set

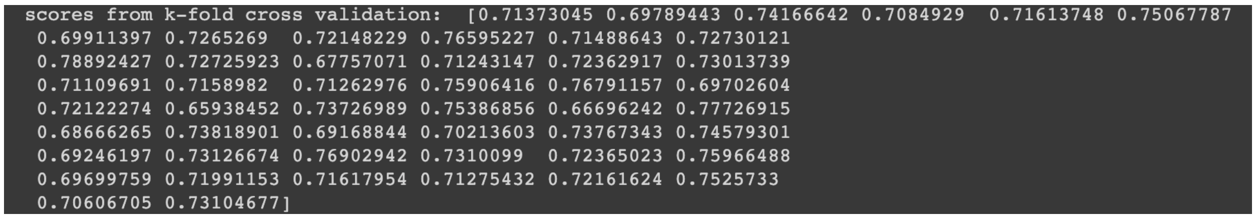
The selected model was then refitted to the entire training dataset (including the validation dataset) to make the final predictions on the testing dataset.

STEP 6 - Prediction on final test dataset

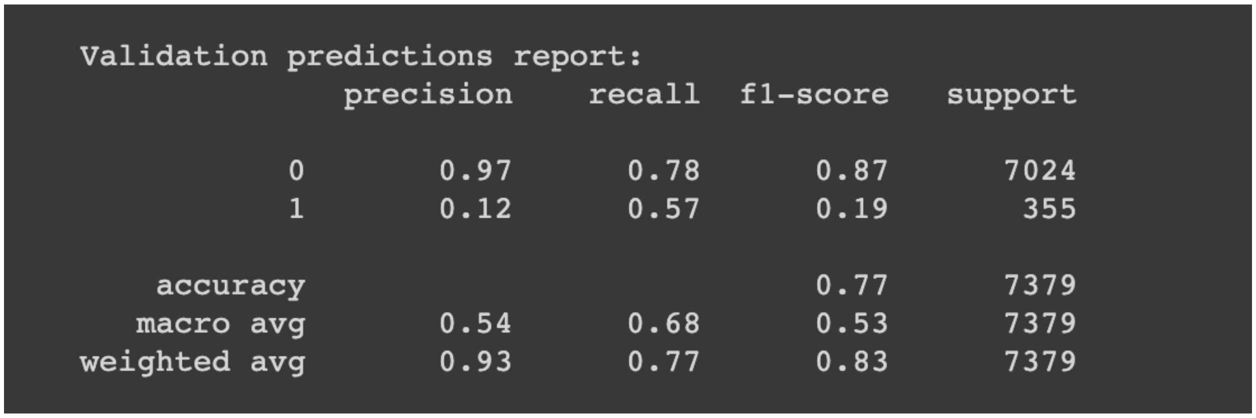
Train the model on entire training dataset and make prediction on test data. The labels and corresponding indices were made into a dataframe and exported as a CSV. The CSV will be present in the same directory the program is in.

STEP 7 - FINAL RESULTS AND SCREENSHOTS

The scores from cross validation are ROC_AUC scores



```
scores from k-fold cross validation:  [0.71373045 0.69789443 0.74166642 0.7084929  0.71613748 0.75067787
 0.69911397 0.7265269  0.72148229 0.76595227 0.71488643 0.72730121
 0.78892427 0.72725923 0.67757071 0.71243147 0.72362917 0.73013739
 0.71109691 0.7158982  0.71262976 0.75906416 0.76791157 0.69702604
 0.72122274 0.65938452 0.73726989 0.75386856 0.66696242 0.77726915
 0.68666265 0.73818901 0.69168844 0.70213603 0.73767343 0.74579301
 0.69246197 0.73126674 0.76902942 0.7310099  0.72365023 0.75966488
 0.69699759 0.71991153 0.71617954 0.71275432 0.72161624 0.7525733
 0.70606705 0.73104677]
```

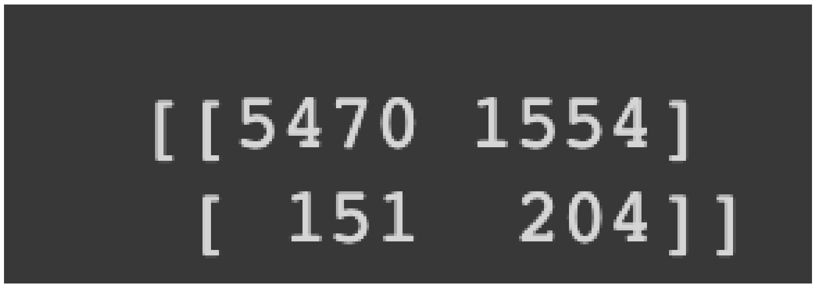


```
Validation predictions report:
              precision    recall  f1-score   support

           0       0.97      0.78      0.87      7024
           1       0.12      0.57      0.19       355

    accuracy                           0.77      7379
   macro avg       0.54      0.68      0.53      7379
weighted avg       0.93      0.77      0.83      7379
```

Confusion matrix for validation data set



```
[[5470 1554]
 [ 151  204]]
```

For the final test data set predictions:

Number of 0s: 7324, Number of 1s: 2258

Public leaderboard score: 0.61862

Private leaderboard score: 0.60419

Other model submission results (public leaderboard):
one class svm ~ 57% score (best) approx,
xgboost ~ 56% score (best) approx
Svm ~ 50% score