Final Report - The Toxicity Prediction Challenge

Team RR

Ravjot Singh (202005149), Raghava Akula (202003025)

Objective

The objective of this competition is to Predict whether chemicals are toxic or not by using a best machine learning model.

Data Preparation

Loading the datasets:

```
featmat=pd.read_csv("E:\\Data Mining\\the-toxicity-prediction-challenge\\feamat.csv")
```

```
train=pd.read_csv("E:\\Data Mining\\the-toxicity-prediction-challenge\\train.csv")
```

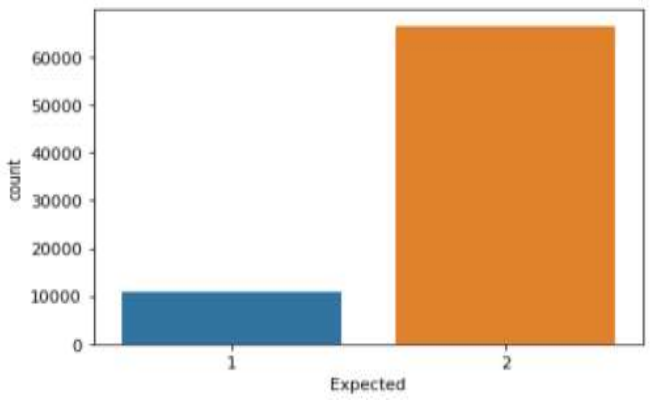
```
test=pd.read_csv("E:\\Data Mining\\the-toxicity-prediction-challenge\\test.csv")
```

We imported the train, featmat, and test from the given path, then checking the shape of the data to check number of rows and columns and further checking the datatype of the features and treating null values and infinity values.

Visualisation:

```
import seaborn as sns
sns.countplot(data=train, x="Expected")
```

```
<AxesSubplot:xlabel='Expected', ylabel='count'>
```



Cleaning the data:

First of all, we cleaned our feature matrix, by treating Infinity values and converting them to NANs and we replaced the NAN values with mean of that particular column

```
featmat.replace([np.inf, -np.inf], np.nan, inplace=True)
```

```
featmat['V15'].fillna(value=featmat['V15'].mean(), inplace=True)
```

Later we found that several columns were redundant and we removed all such columns that are containing zero's, one's.

```
featmat= featmat.loc[:, (~featmat.isin([0])).any(axis=0)]
featmat= featmat.loc[:, (~featmat.isin([1])).any(axis=0)]
```

Data Splitting:

In this step we had split our test and train datasets based on the Id column

```
train[['Id','Assay']] = train['Id'].str.split(';',expand=True)
```

```
test[['Id','Assay']] = test['x'].str.split(';',expand=True)
```

Type Conversion

Converting object type to integer

```
train['Assay'] = train.Assay.astype(int)
test['Assay'] = test.Assay.astype(int)
```

Data Merging

Merging the train and test data with featmat based on ID and making final train and test data

```
train = train.merge(featmat, left_on='Id', right_on='V1', how='left')
```

```
test = test.merge(featmat, left_on='Id', right_on='V1', how='left')
```

After removing columns that contains zero's, one's, we are left with 945 columns

```
x.shape, target.shape
```

```
((77413, 945), (11139, 945))
```

Data Modelling

Loading the independent and dependent variable

```
x= train.drop(["Id","Expected"], axis=1)
y= train['Expected']
```

splitting the data using stratified k-fold

```
from sklearn.model_selection import StratifiedKFold
folds = StratifiedKFold(n_splits = 10,random_state = None, shuffle = False)

for train_index, test_index in folds.split(x,y):
    x_train, x_test, y_train, y_test = x.iloc[train_index], x.iloc[test_index], y.iloc[train_index], y.iloc[test_index]
```

We tried to fit the model using XGBClassifier

```
import xgboost as xgb
model = xgb.XGBClassifier(booster='gbtree',max_depth=8, n_estimators=400, random_state=22)
model.fit(x_train, y_train)
model.score(x_test,y_test)
```

Then, we predicted the test data based on the splitted data

```
y_pred= model.predict(x_test)
```

In order to check the internal evaluation , we used f1 score for it.

```
from sklearn.metrics import f1_score
f1= f1_score(y_test,y_pred,average = "macro")  #None, 'micro', 'macro', 'weighted', 'samples'
print(f1)
```

Then, finally we predicted the expected variable from target data from test.csv

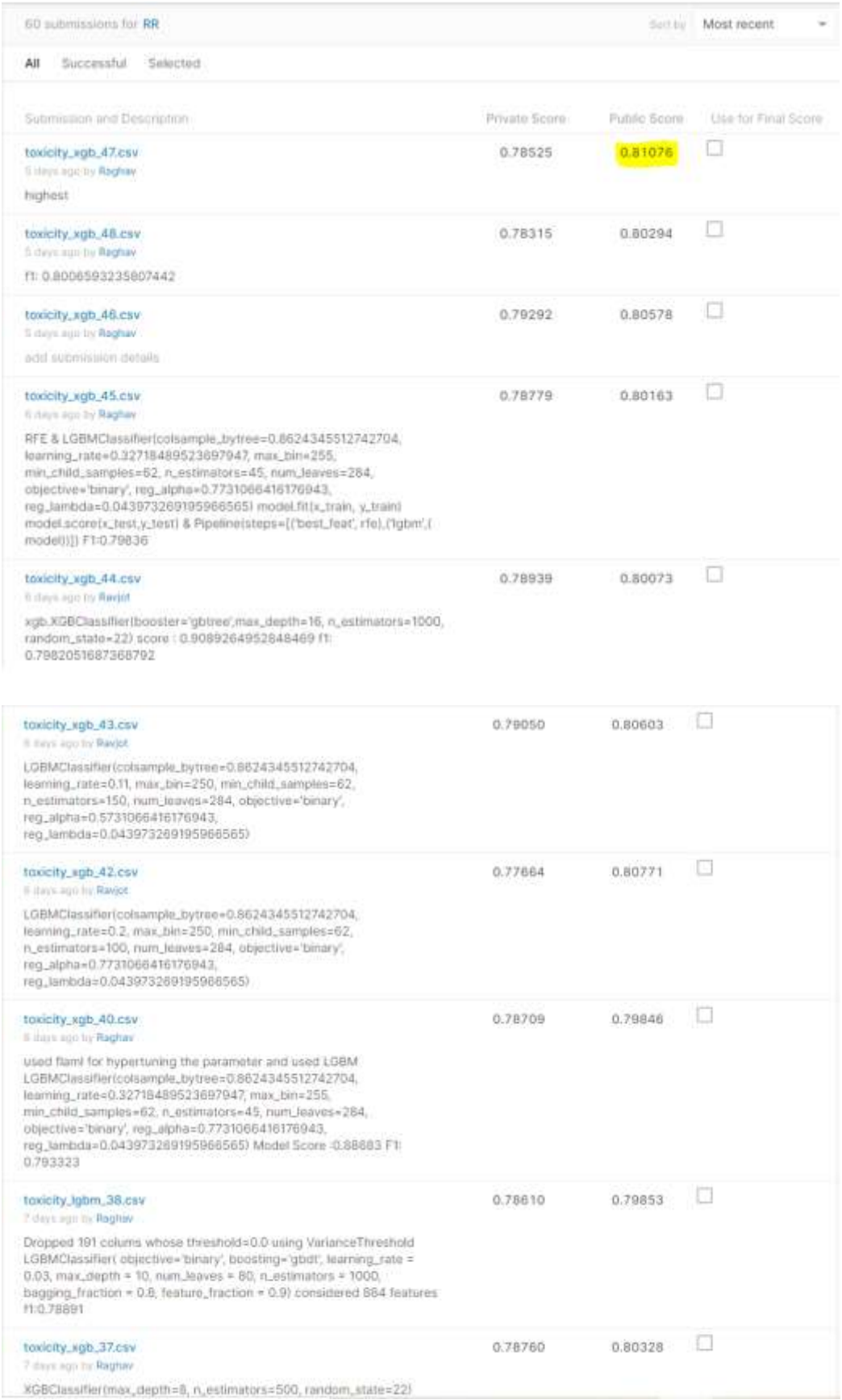
```
y_target= model.predict(target)
```

Model	F1 Score	Leaderboard Score	Features	Approach
Decision Tree	0.74572	0.75511	All features	Using train_test_split(test_size=0.3) & DecisionTreeClassfier(max_depth=4, random_state = 10)
Gradient Boosting	0.76990	0.77188	Removed all columns containing 0's and 1's or both and included Assay Id	GradientBoostingClassifier(n_estimators=100, learning_rate=1.0, max_depth=5, random_state=20)
LGBM	0.78891	0.79853	Dropped 191 columns whose threshold=0.0 using Variance Threshold Selected features: 884	LGBMClassifier(objective='binary', boosting='gbdt', learning_rate = 0.03, max_depth = 10, num_leaves = 80, n_estimators = 1000, bagging_fraction = 0.8, feature_fraction = 0.9)
LGBMClassifier	0.79836	0.80163	Consider all features and used RFE.	Used FLAML AutoML in order to find best feature for hyper tuning. LGBMClassifier (colsample_bytree=0.8624345512742704, learning_rate=0.32718489523697947, max_bin=255, min_child_samples=62, n_estimators=45, num_leaves=284, objective='binary', reg_alpha=0.7731066416176943, reg_lambda=0.043973269195966565) &Pipeline(steps=[('best_feat', rfe),('lgbm',(model))])
XGBoost	0.79954	0.80424	Selected only 201 features using SelectKbest with score_func=f_classifer whose scores_ are <10	XGBClassifier(objective="binary:logistic",max_depth=8, n_estimators=400)
XGBoost	0.7963	0.8080	Removed all columns containing 0's and 1's and V15	XGBClassifier(max_depth=8, n_estimators=400)
XGBoost	0.79919	0.80802	Removed all columns containing 0's and 1's, Selected features: 945;	XGBClassifier(booster='gbtree',max_depth=7, n_estimators=600, random_state=22)
XGBoost	0.79727	0.81076	Removed all columns containing 0's and 1's, Selected features: 945;	XGBClassifier(booster='gbtree',max_depth=8, n_estimators=400, random_state=22)

Kaggle Leader Board

Public Score: 0.81076 **Rank: 10**

Submissions Screenshots



60 submissions for RR — Sort by: Most recent

All | Successful | Selected

Submission and Description	Private Score	Public Score	Use for Final Score
toxicity_xgb_47.csv 5 days ago by Raghav highest	0.78525	0.81076	☐
toxicity_xgb_48.csv 5 days ago by Raghav f1: 0.8006593235807442	0.78315	0.80294	☐
toxicity_xgb_46.csv 5 days ago by Raghav add submission details	0.79292	0.80578	☐
toxicity_xgb_45.csv 6 days ago by Raghav RFE & LGBMClassifier(colsample_bytree=0.8624345512742704, learning_rate=0.32718489523697947, max_bin=255, min_child_samples=62, n_estimators=45, num_leaves=284, objective='binary', reg_alpha=0.7731066416176943, reg_lambda=0.043973269195966565) model.fit(x_train, y_train) model.score(x_test,y_test) & Pipeline(steps=[('best_feat', rfe),('lgbm',(model))]) F1:0.79836	0.78779	0.80163	☐
toxicity_xgb_44.csv 6 days ago by Ravjot xgb.XGBClassifier(booster='gbtree',max_depth=16, n_estimators=1000, random_state=22) score : 0.9089264952848469 f1: 0.7982051687368792	0.78939	0.80073	☐
toxicity_xgb_43.csv 6 days ago by Ravjot LGBMClassifier(colsample_bytree=0.8624345512742704, learning_rate=0.11, max_bin=250, min_child_samples=62, n_estimators=150, num_leaves=284, objective='binary', reg_alpha=0.5731066416176943, reg_lambda=0.043973269195966565)	0.79050	0.80603	☐
toxicity_xgb_42.csv 6 days ago by Ravjot LGBMClassifier(colsample_bytree=0.8624345512742704, learning_rate=0.2, max_bin=250, min_child_samples=62, n_estimators=100, num_leaves=284, objective='binary', reg_alpha=0.7731066416176943, reg_lambda=0.043973269195966565)	0.77664	0.80771	☐
toxicity_xgb_40.csv 6 days ago by Raghav used flaml for hypertuning the parameter and used LGBM LGBMClassifier(colsample_bytree=0.8624345512742704, learning_rate=0.32718489523697947, max_bin=255, min_child_samples=62, n_estimators=45, num_leaves=284, objective='binary', reg_alpha=0.7731066416176943, reg_lambda=0.043973269195966565) Model Score :0.88683 F1: 0.793323	0.78709	0.79846	☐
toxicity_lgbm_38.csv 7 days ago by Raghav Dropped 191 colums whose threshold=0.0 using VarianceThreshold LGBMClassifier(objective='binary', boosting='gbdt', learning_rate = 0.03, max_depth = 10, num_leaves = 60, n_estimators = 1000, bagging_fraction = 0.8, feature_fraction = 0.9) considered 864 features f1:0.78891	0.78610	0.79853	☐
toxicity_xgb_37.csv 7 days ago by Raghav XGBClassifier(max_depth=8, n_estimators=500, random_state=22)	0.78760	0.80328	☐