CSCI – 555 Data Mining and Machine Learning

**The Toxicity Prediction Challenge II**

ST. FRANCIS XAVIER UNIVERSITY

Supervised By: Dr. Othman Soufan

Department Of Computer Science

**NAME:** Ganesh Sai Subramanya Varma Ruddarraju

**MAIL ID**: [x2022gvd@stfx.ca](mailto:x2022gvd@stfx.ca)

**STUDENT ID**: 202206894

**Introduction:**

In advance of the release of new chemicals, it is necessary to estimate their toxicity. Toxicology prediction has traditionally been accomplished by in-vivo techniques that involve the examination of microbes, human cells, or animals. Toxicology prediction cannot, however, be simply based on laboratory animal studies given the thousands of novel substances that are created every day. It is suggested that in-vivo approaches be replaced with in-silico techniques, which are computer-based simulations. The time, money, and animal testing associated with toxicity prediction could be cut down with the help of these strategies. The compilation of a dataset with information for over 1500 high-throughput assay endpoints and data for over 9,000 compounds. This dataset can be used in in-silico toxicity prediction investigations. The need for accurate and reliable approaches to anticipate the toxicity of newly created substances indicates the potential use of in-silico procedures as an alternative to conventional in-vivo techniques.

**Data preparation:**

**Splitting**:

Imported the rd kit packages, I have loaded the dataset to jupyer notebook. The Dataset contains a column by the name “ID”, which I have divided into 2 columns which are named Compound and AssayID. Then I extracted smiles for the data in the column “Compound”.

Used desc(Mol) function which takes the smiles as the input and returns descriptors and their values as output. Now, called the descriptor function “desc(Mol)” for both test and train files. By combining the descriptors' names and values I created a separate data frame for testing and train then I appended assayID to the newly created data frame. Created a final data frame for both test and train by concatenating the data frame created in the above step and the initial data frame. Later, exported those data frames using to\_csv function.

**Data pre-processing:**

Started with loading the descriptor file for both train and test. Then, After loading the dataset I found that 12 columns contained the null values. So, to handle the null values I have filled those null values with mean using **fillna()** method.

**Feature Selection:**

**Correlation Analysis:**

Used Spearman correlation method to remove the highly correlated features in the dataset and gave a threshold value of 0.8. Of the 209 features, there are 60 features with high correlation, because of the high correlation they are removed and the number of features left at the end is 149.

**Forward feature Selection:**

Used forward feature selection using a decision tree classifier to get the top features from the existing 149 features. This technique has given me so many sets of features but I have selected the one set which gave me the highest accuracy.

**Modelling:**

* X - Contains all the columns of training data except columns “Expected”.
* Y - Contains only the column “Expected” from the train dataset.
* Split the data using test\_train\_split into 80 and 20 percent.
* Then I used several machine learning models to predict x\_test, Some of the models used are Gradient Boost classifier, xgboost, random forest, decision tree, and litegbm.
* CatBoost model at the end gave me the best accuracy.

**CatBoost**:

Yandex created CatBoost, a framework for gradient boosting that is used for machine learning applications like classification, regression, and ranking. It employs decision trees with gradient boosting and has features like categorical feature support that can handle categorical data without the need for pre-processing or encoding. CatBoost is effective for handling complex dependencies in data, handling missing data, and handling imbalanced data by adjusting the weights of each class during training. It is a potent machine learning method that may be applied to a variety of applications, particularly those with complicated dependencies and categorical data.

**F1 Score Internal Validation:**

Text

Description automatically generated

**Predictions:**

To store the data for the predictions, I have created a new variable “**predict**” and used the pandas function **pd.DataFrame().** Used y\_predicted variable to store the predictions. I took the test data file's column **x** and concatenated it with **y\_predicted** to create the output CSV file. I exported the CSV file after concatenating it using **to\_csv**.

**Leaderboard:**

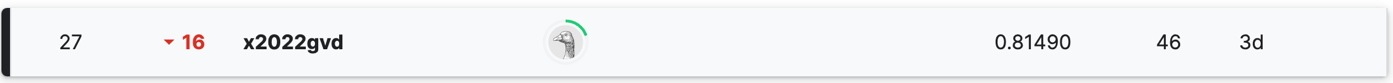
I have achieved 11th rank in the public leaderboard with a score of 81.303% and achieved 27th rank in the private leaderboard` with a score of 81.490%.

**Public**:



*Fig: Kaggle Public Leaderboard*

**Private**:

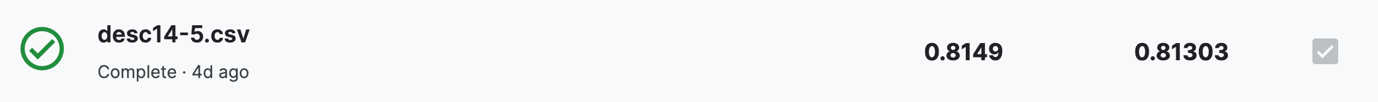


*Fig: Kaggle Private Leaderboard*

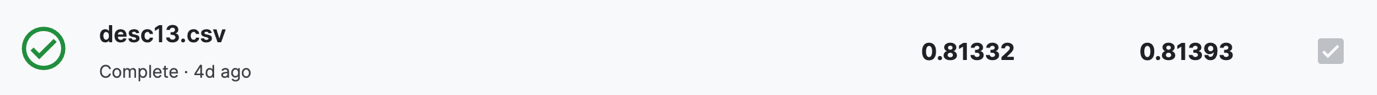
**Kaggle Score Selection:**

For the option of selecting the files in the Kaggle for the final leaderboard evaluation, I have selected two different files generated by:

1. Including all the above steps



1. All the above steps **excluding** the forward feature selection.



**Including the Forward feature selection has given me the best score in the private leaderboard.**

**Kaggle Competition Link:**

<https://www.kaggle.com/competitions/the-toxicity-prediction-challenge-ii>

**GitHub link:**

<https://github.com/GaneshRuddarraju/Toxicity_Prediction_202206894>

**Google Drive Link:**

<https://drive.google.com/drive/folders/10-nhCqogfqLQf2W6eILwzCr82Y9zZRpD?usp=share_link>