

TOXICITY PREDICTION CHALLENGE II

INTRODUCTION:

The objective of the toxicity prediction challenge is to create a machine learning model that can identify whether a chemical is toxic (labeled as 1) or non-toxic (labeled as 2) by analyzing its associated features. The challenge provides 2 data files: train II data and test II data.

DATA PREPROCESSING:

- The test dataset's "x" column was renamed to "Id" using the rename() function to ensure consistency with the other dataset. The "ID" column was divided into two separate columns - "chemical id" and "Assay id" - based on the delimiters specified in the original dataset. This was achieved using the String split() method, which separates a string at a specified separator and generates a list of strings
- I used the RDkit library to generate the below Descriptors for each chemical in our train and test datasets. First we converted our chemicals to smiles and then used those smiles to generate these features. These features are selected using ChatGPT.
 1. MolWt
 2. MolLogP
 3. TPSA
 4. NumHDonors
 5. NumHAcceptors
 6. NumRotatableBonds
 7. NumAromaticRings
 8. RingCount
 9. HeavyAtomCount
 10. FractionCSP3
 11. Slogp3
 12. Slogp4
 13. Slogp5
 14. Slogp4
 15. Slogp7
 16. Smr3
 17. Smr4

18. Smr5

19. Smr6

- We also generated 512 bit of morgan fingerprints using RDkit GetMorganFingerprintAsBitVect
- Fingerprints are generated in the form of an array and we convert the array to columns.
- Before the model fitting, we split the test and train datasets as follows:
 - X_train - Contains all the columns of training data except columns "Expected" and "Chemical ID". These columns are dropped from the train dataset.
 - Y_train - Contains only the column "Expected" from the train dataset.
 - X_test - Contains all the columns from test data except column "Chemical Id". The column "Chemical Id" is dropped from the test dataset.

Modeling

I used FLAML which is an open-source Python library for automating machine learning (AutoML). It supports a variety of machine learning tasks, such as classification, regression, and clustering, and can automatically search for the best machine learning model and hyperparameters for a given task.

One of the models that Flaml supports is the LightGBM (LGBM) model, which is a gradient boosting framework that uses tree-based learning algorithms. LGBM is known for its high performance and scalability on large datasets.

In Flaml, you can use the LGBM model for a variety of tasks such as classification, regression, and ranking. Flaml can automatically tune the hyperparameters of the LGBM model using various techniques such as grid search, random search, and Bayesian optimization to achieve optimal performance on a given dataset. Overall, Flaml automates the machine learning process and makes it easier for developers and data scientists to build high-performance models without spending too much time on manual tuning of hyperparameters.






To calculate F1-score we used Classification_report() function

Submission File generation:

To make predictions, we created a new variable called "prediction" and used the pd.DataFrame() function from the pandas library to store the prediction data. We then combined it with the Id column from the test2 data set. Finally, we converted the Prediction file to a comma-separated format by using the .to_csv() function, which generated the output file.

LEADERBOARD:

I totally made 68 submissions and scored 4th in the private leaderboard.

1	▲ 6	x2022fic		0.83242	38	8d
2	▲ 27	x2022fij		0.83168	46	3d
3	▲ 13	x2021guf		0.82871	53	3d
4	—	x2022fip		0.82798	158	3d
5	▼ 4	x2020gtl		0.82771	68	4d