Custom ML Module of AIDrugApp for Molecular Identification, Descriptor Calculation, and Building ML/DL QSAR Models

Divya Karade

Chemical Engineering and Process Development (CEPD) Division, CSIR-National Chemical Laboratory, Pune – 411008, India

ABSTRACT

Computer-aided drug design (CADD) techniques continue to struggle to provide a useful advance in the area of drug development due to the difficulties in an efficient exploration of the vast drug-like chemical space to uncover new chemical compounds with desired biological properties. Other challenges that users must overcome in order to fully use the potential of CADD tools and techniques include a lack of completely autonomous methods, the necessity for retraining even after deployment, and their lack of interpretability. To solve this issue, we created the 'Custom ML Tools' integrated within the framework of 'AIDrugAPP'. 'Custom ML Tools' includes four modules: 'Mol Identifier', 'DesCal', 'AutoDL', and 'Auto-Multi-ML' which give users free access to molecular identification using SMILES and compound names, similarity search, descriptor calculation, the building of ML/DL QSAR models, and their usage in predicting new data. The study demonstrates the potential of the novel tool for computational investigations in drug discovery research. The WebApp with its modules has therefore been made available for public use at: https://sars-covid-app.herokuapp.com/

Keywords: QSAR, ML/DL, WebApp, Molecular descriptors, Virtual screening, Data analysis

INTRODUCTION

Diseases are a global problem and there are thousands of diseases with no treatment that leads to approx. min. 80 % of all global deaths [1]. Drug discovery for all those diseases is a challenging process that requires billions of dollars of investments and decades of research. And above that only, less than 15% of drugs enter clinical trials resulting in an approved medicine [2, 3]. For fast and accurate drug discovery researches, novel therapeutic targets and drug-like compounds are needed. One promising approach is to use AI in drug discovery to improve screening and hit rates from large chemical datasets [4, 5]. The use of Machine Learning (ML) algorithms for computer-assisted drug discovery is expanding due to the large available public data sources[6]. The global market value of AI for drug discovery and development is expected to reach approx. $ 1.4 billion by 2024 with a growth rate (CAGR) of 40.8% [7].

AI has the potential to be useful in several aspects of drug discovery, including drug design, chemical synthesis, drug screening, polypharmacology, and drug repurposing. Deep Learning (DL) has recently emerged as a technique superior to traditional ML approaches due to its ability to perform automatic feature extractions from large amounts of data and capture nonlinear input-output relationships [8]. DL algorithms can swiftly anticipate a large number of compounds when coupled with a quantitative structure-activity relationship (QSAR)-based computational model [9, 10]. For example, the DeepChem platform incorporated with DL algorithms resulted in an easy-to-use drug development algorithm that outperformed random forests algorithms [11]. Many AI algorithms or pipelines have been created for drug discovery, such as DeepNeuralNetQSAR [12], ORGANIC [13], PotentialNet [14]. However, the existing algorithms are not that accurate and are difficult to use due to the lack of an interface. On the other hand, there are many computational chemistry libraries

available such as RDKit [15] libraries for manipulating chemical objects such as atoms, bonds and molecules, Mordred [16] libraries for calculating molecular descriptors. Developing a new model using any of these libraries or tools does require extensive coding which is a big drawback for most of the chemists, biologists, computational chemists or biologist who are not computer scientists or software engineers. Third-party software has also been developed to provide access to ML techniques for researchers who are unfamiliar with coding, however, their ability to perform ML techniques as well as other aspects of the ML pipeline is limited [17].

To address the issues discussed above, we developed an easy-to-use, freely available customized ML module called "Custom ML tools" incorporated in the framework of AIDrugApp [18]. With this, users can build their ML or DL models and can apply those models for predictions on their data. It is also useful for obtaining molecular information using Simplified molecular-input line-entry system (SMILES) and compound names and computing user-defined molecular descriptors. We believe that 'Custom ML Tools' will be valuable in the drug development efforts of both specialists and non-experts in cheminformatics and computational chemistry.

MATERIALS AND METHODS

The entire system is hosted on the Heroku cloud application platform server[19] by maintaining the program code in GitHub[20]. Python was selected as the primary programming language for developing Customized ML tools since it integrates nicely with the other tools/packages such as RDKit [21], scikit-learn[22], pandas[23], numpy[24], tensorflow[25], keras[26], etc. Streamlit [27] was used to provide a high-level Python web framework for Custom ML tools. For reading and writing molecular data files, the custom ML tool employs python modules such as pandas for handling data frames and NumPy for

numerical data processing. Before transferring to the backend program, it uses user input in the form of SMILES, chemical names, and molecular features as single string or file, depending on the type of input (single/multiple) and ML module type.

Backend Computational Engine Design

The computational engine for the Custom ML tool is divided into four modules (**Fig. 1**) and operates on the backend server for the front-end User Interface, which visualises predictions based on user input. The four modules of "Custom ML Tools" are 1. Mol_Identifier, 2. DesCal, 3. AutoDL and 4. Auto-Multi-ML.

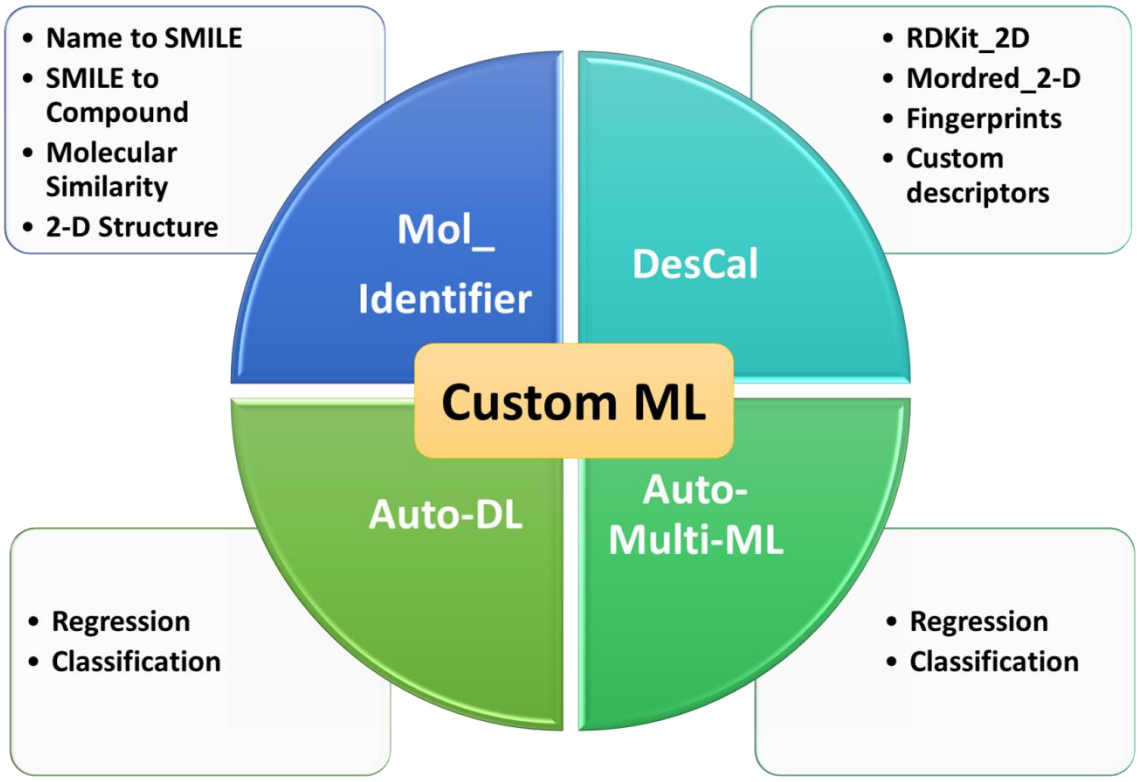


Fig. 1: Representation of the four modules integrated into Custom ML Tools

Mol_Identifier

Mol_Identifier is a Custom ML tool module that helps in converting chemical names to SMILES, molecular SMILES to compound names, 2-D structures, and identifying molecular similarities on user data. Python libraries used for developing Mol_identifier are PubChemPy[28], pandas, RDKit, mols2grid (https://github.com/cbouy/mols2grid) and matplotlib. The Mol_Identifier framework was designed in the Streamlit python library so that users can retrieve the calculations based on a single or multiple input value. To query the PubChem REST API for converting the chemical name to SMILES and SMILES to name, we utilise the PubChemPy python package. To visualize the 2-D structures of users molecules, RDKit and mols2grid python libraries were utilized. Mols2grid is an RDKit-based interactive chemical viewer for 2D structures of small molecules. RDKit libraries were also used to generate different similarity metrics such as Tanimoto, Dice, Cosine, Sokal, and McConnaughey on user data for molecular fingerprint (MACCS) based similarity calculations.

DesCal

DesCal is a Custom ML tool module that helps to generate various molecular 2-D descriptors and fingerprints on user's data. It also helps to calculate customized molecular descriptors as selected by the user on their data. Python libraries used for developing DesCal are TensorFlow, Keras, scikit-learn, streamlit, pandas, NumPy, Mordred [29] and RDKit. The 'DesCal' framework was designed in the Streamlit python library so that users can retrieve the calculations based on single or multiple input values. Mordred and RDKit python libraries were used for calculating 2-D molecular descriptors and fingerprints (MACCS and Morgan for radius =1, 2, 3) on user's data. The carbon atom from the carbonyl functional group is used to calculate radii or bond depths. Radius 2 (ECFP4) and radius 3 (ECFP6) are the most

prevalent. It will essentially extract information on substructures comprising circular atom neighbourhoods, such as an atom, and its connectedness to immediate neighbours and then neighbours of those neighbours.

In the Mordred and RDKit python libraries, several molecular properties, such as LogS, density, and so on, are neither inbuilt or calculated. Therefore, we have introduced 'Custom descriptors' component in DesCal which enables users to calculate selected descriptors like AromaticProportion [30], Density and LogS (solubility of molecules) values on user's data. AromaticProportion (= Number of aromatic atoms in a molecule / Number of heavy atoms in a molecule) and Density (= Mass / Volume) were calculated using existing python libraries i.e. Mordred and RDKit. LogS values, on the other hand, were predicted using DL (DL) QSAR-based linear regression models using the Delaney [31] solubility dataset of n= 1144 molecular SMILES with observed solubility values in mol/L. The DL models for predicting LogS values were developed using the Python libraries TensorFlow and Keras. The model was trained on 2-D molecular descriptors and fingerprints generated by RDKit python libraries. Training set model validated based on the test set. The DL model has one input and two hidden layers, each with 100 neurons, with ReLU and linear activation function, Adam optimizer, and loss= Mean squared error (MSE) and Mean absolute error (MAE). During 250 epochs of training in 50 batches with 57,901 parameters, the best model was developed with a maximum Coefficient of determination (R2) score (Train: 0.99, Test: 0.93) and a minimum MSE score (Train: 0.02, Test: 0.29) and MAE score (Train: 0.09, Test: 0.38).

Auto-DL

AutoDL is a Custom ML tool module that helps to build DL models with neural networks on user's data. It also helps to predict user-specific target data based on DL models built algorithms selected by the user. Python libraries used for developing AutoDL are

TensorFlow, AutoKeras [32], scikit-learn, pandas, NumPy and matplotlib. The AutoDL framework was designed and deployed in the Streamlit so that users can build their own regression or classification based DL model and retrieve the predictions on input data (**Fig. 2**). AutoKeras is an AutoML system based on Keras. AutoDL employs both StructuredDataClassifier and StructuredDataRegressor of AutoKeras python library to construct automatic classification and regression DL models by searching the best detailed configuration for user input target data. Other python libraries such as scikit-learn to perform hyperparameter optimization by calculating MSE, MAE, Root mean squared error (RMSE), and R2 values for regression models and precision score, recall score, ROC AUC score, f1 score, confusion matrix for classification models and matplotlib was used for data visualization.

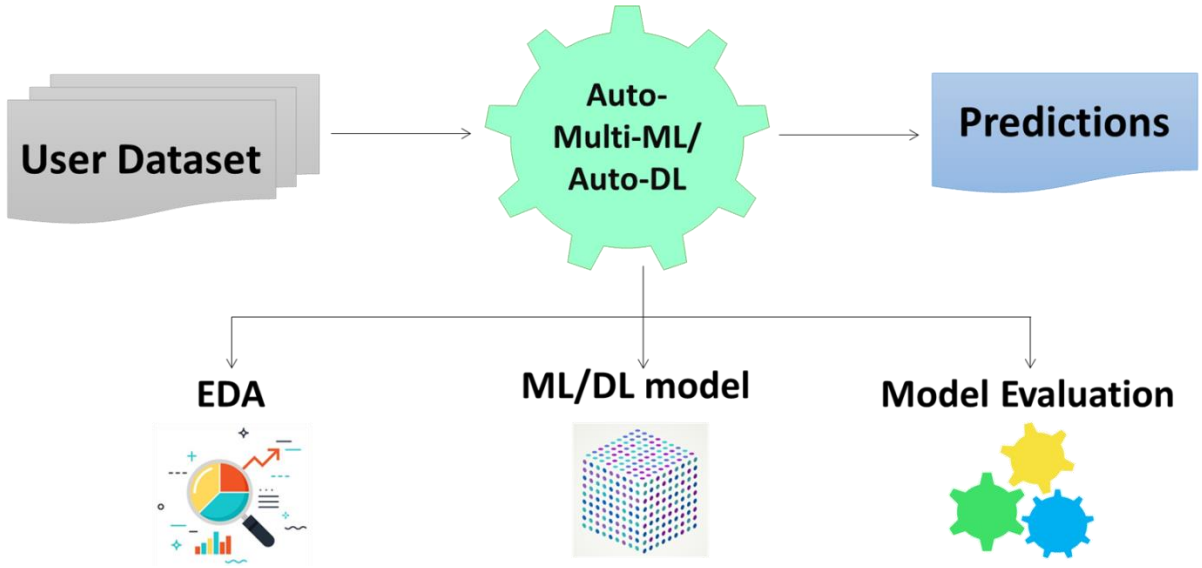


Fig. 2: An overview of the processing of automated ML/DL modules

Auto-Multi-ML

Auto-Multi-ML is a Custom ML tool module that helps to build and compare multiple ML models for interpreting best performing ML algorithms on users data. It also helps to predict

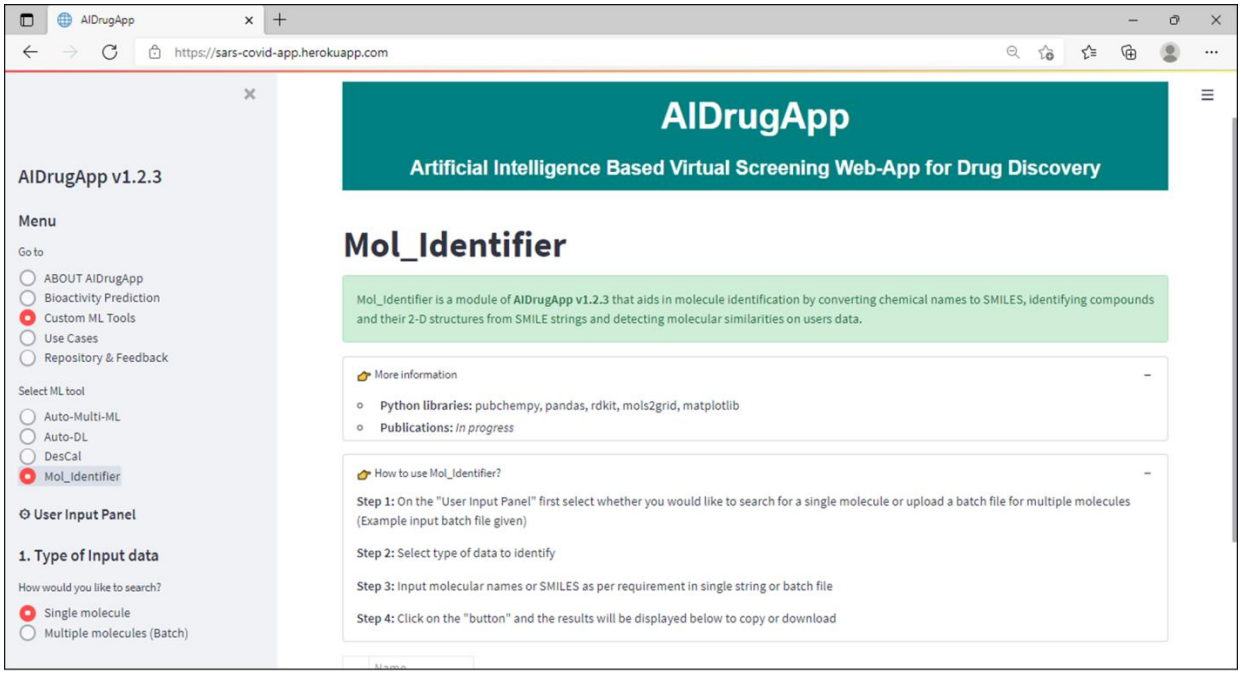
user specific target data based on ML models built and selected by the user (**Fig. 2**). Python libraries used for developing Auto-Multi-ML are scikit-learn, pandas, NumPy, Lazypredict (https://github.com/shankarpandala/lazypredict) and Sweetviz [33]. The Auto-Multi-ML framework was designed using Streamlit python library so that users can build their own regression or classification based multiple ML (ML) model and retrieve the predictions on input data. Lazypredict library was used to semi-automate ML operations by building multiple traditional ML models without much code and evaluating which models perform better without any parameter tuning. Sweetviz library was used to provide elegant, high-density images to aid with Exploratory Data Analysis (EDA)**.** Scikit-learn was used to select significant features on users target data, split data into training and test sets, and optimise hyperparameters by calculating MSE, MAE, RMSE, and R2 values for regression models and precision score, recall score, ROC AUC score, f1 score, confusion matrix for classification models.

RESULTS AND DISCUSSION

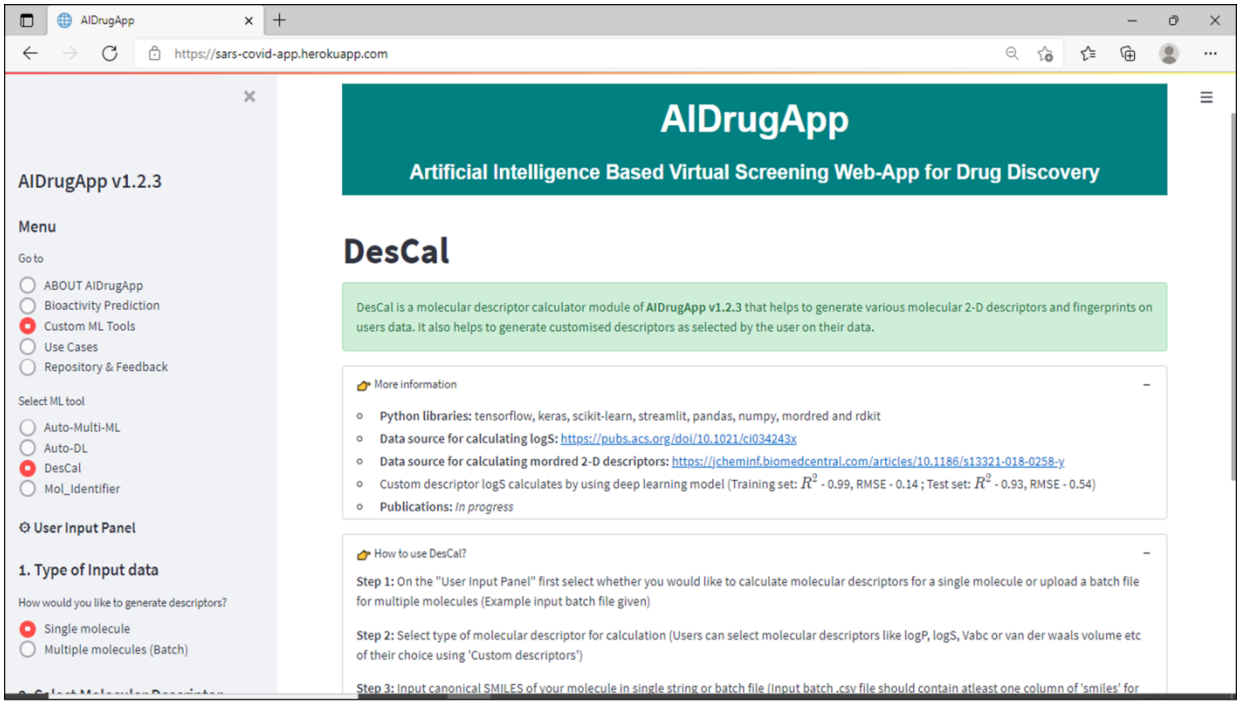
Front End User Interface

Users can use a web browser to access the publically available interface (https://sars-covid-app.herokuapp.com/). The user interface of the 'Custom ML Tools' is comprised of a User Input panel containing four key modules i.e. "Mol Identifier", "DesCal", "AutoDL", and "Auto-Multi-ML" (**Fig. 3**). The output panels, on the other hand, are loaded on the same Web page on the right side of the input panel, where the user can view the calculation results. For selecting various options, the user input panel consists of radio buttons, select boxes, checkboxes, and sliders. Users can visualise and download/ save (.csv file) the outcomes, predicted results, model data analysis, uploaded data analysis, and model evaluation results

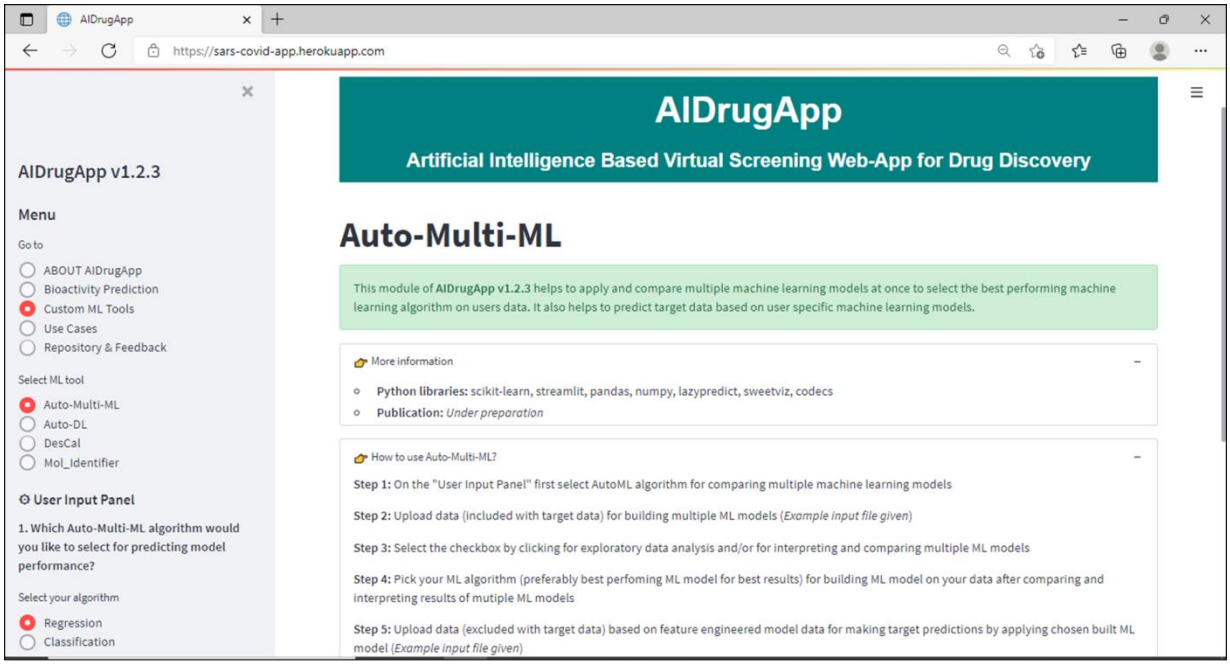
using the output panel. Every module is also enclosed with instructions for better understanding.



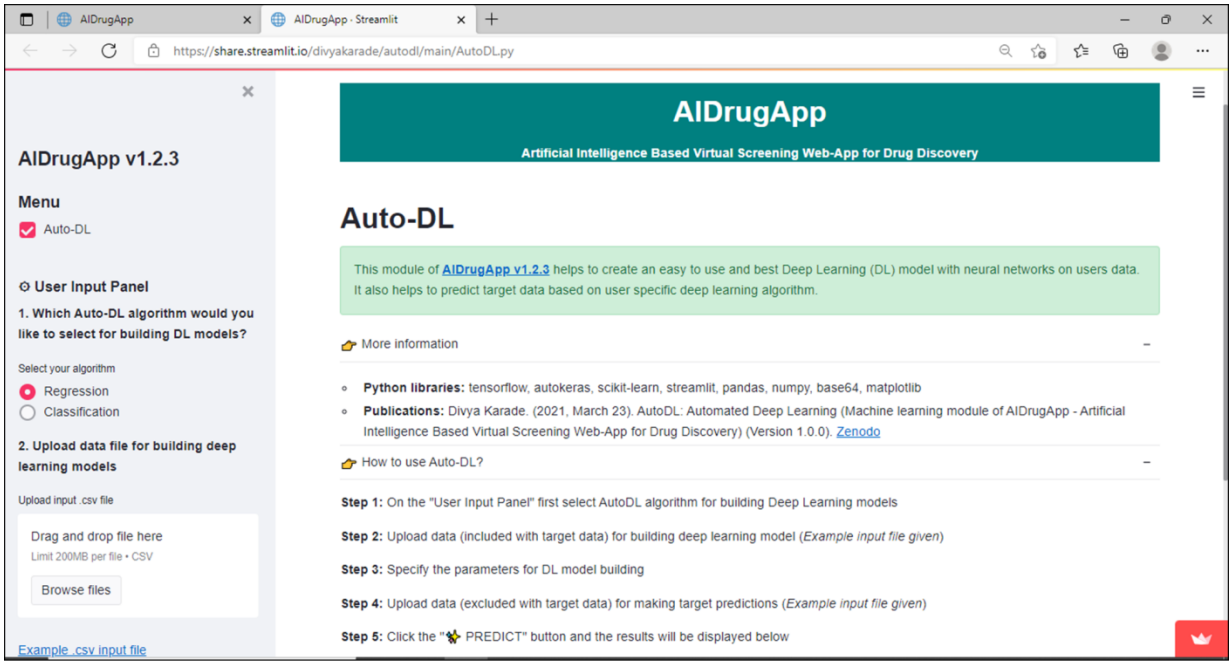
A.



B.



C.



D.

Fig. 3: Screenshots of the user interface for 'Custom ML Tools' that comprises four major modules: (**A**) 'Mol_Identifier', (**B**) 'DesCal', (**C**) 'AutoDL' and (**D**) 'Auto-Multi-ML'.

Computation of SMILES, compound names and molecular similarity scores

Name to SMILES

Users can input compound name for a ‘Single molecule’ or multiple names by uploading a .csv file in the ‘Multiple molecules (Batch)’ mode of ‘Mol_Identifier’ module. This allows users to obtain Canonical SMILES or Isomeric SMILES for their molecular names after clicking ‘GET SMILES’ button (**Fig. 4**).

SMILE to Compound

Users can input Canonical SMILES for a ‘Single molecule’ or multiple SMILES by uploading .csv file in ‘Multiple molecules (Batch)’ mode of ‘Mol_Identifier’ module. This allows users to obtain molecular names for their molecular SMILES after clicking ‘GET COMPOUND’ button (**Fig. 4**).

Molecular Similarity

Users can input a pair of Canonical SMILES for a ‘Single molecule’ or multiple SMILES by uploading a .csv file in ‘Multiple molecules (Batch)’ mode of the Mol_Identifier module. This allows users to obtain Tanimoto, Dice, Cosine, Sokal, and McConnaughey similarity scores under ‘Single molecule’ mode after clicking the ‘GET SIMILARITY’ button. Whereas, through ‘Multiple molecules (Batch)’ mode users can get similarity scores (Tanimoto, Dice, Cosine, Sokal, and McConnaughey) between multiple molecules, interpretation of similarity scores in the form of a histogram that shows the distribution of the pair-wise scores and table of pairwise similarity scores among molecules. The Tanimoto score between molecules is used to interpret similarity scores. These studies assist users in determining how similar molecules are and how ‘similar’ is comparable (**Fig. 4**).

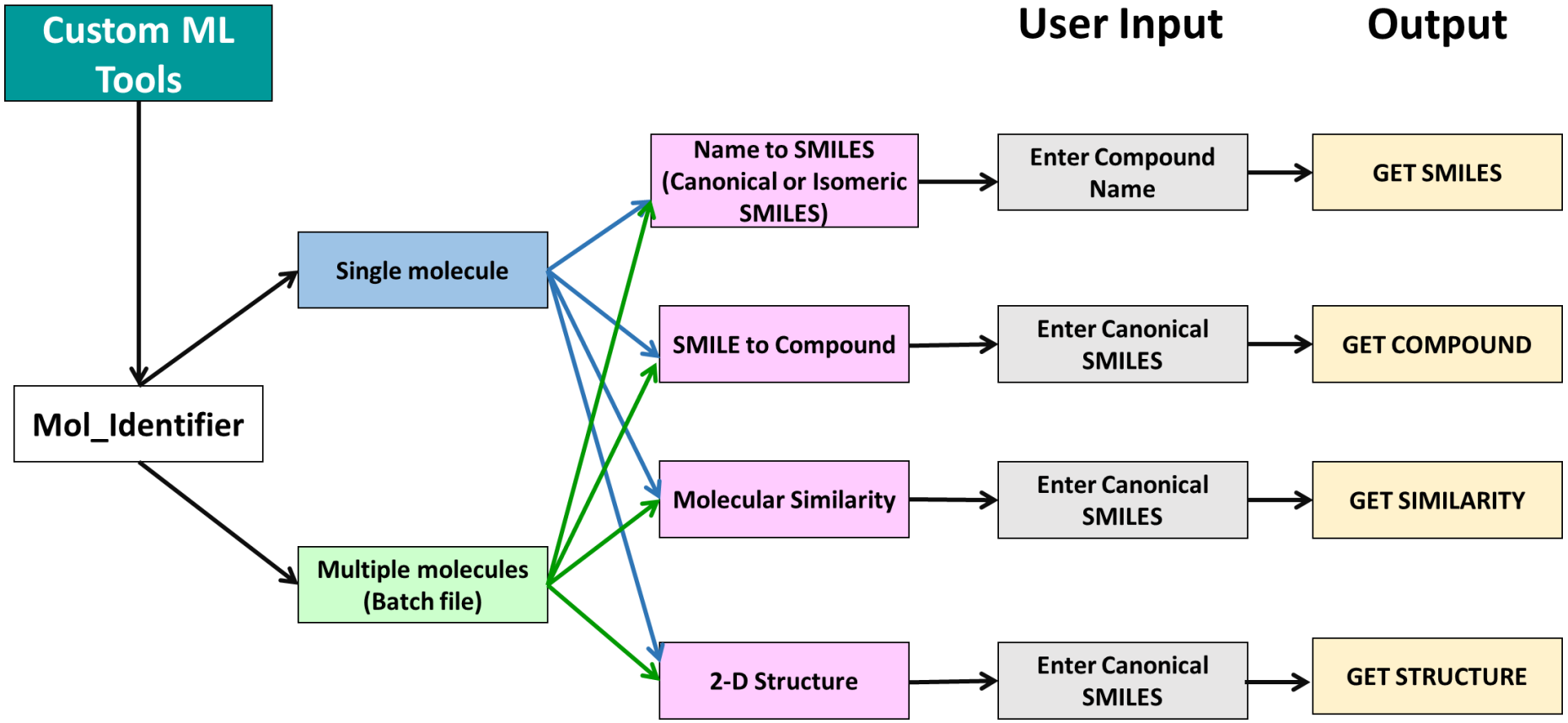


Fig. 4: Workflow highlighting the computation of SMILES, compound names, and molecular similarity scores by using 'Mol_Identifier' module of 'Custom ML Tools'.

2-D Structure

Users can input Canonical SMILES for a ‘Single molecule’ or multiple SMILES by uploading the .csv file in the ‘Multiple molecules (Batch)’ mode of the Mol_Identifier module. This allows users to obtain molecular 2-D Structure for their molecular SMILES after clicking the ‘GET STRUCTURE’ button (**Fig. 4**).

Computation of molecular descriptors and fingerprints

RDKit_2D

Users can input Canonical SMILES for a ‘Single molecule’ or multiple SMILES by uploading the .csv file in the ‘Multiple molecules (Batch)’ mode of the DesCal module. This allows users to obtain molecular 2D RDKit descriptors for their molecular SMILES after clicking the ‘CALCULATE’ button. It will output the list of 115 2D RDKit descriptors that are presently accessible (**Fig. 5**).

Mordred_2-D

Users can input Canonical SMILES for a ‘Single molecule’ or multiple SMILES by uploading the .csv file in the ‘Multiple molecules (Batch)’ mode of the DesCal module. This allows users to obtain molecular Mordred_2-D descriptors for their molecular SMILES after clicking the ‘CALCULATE’ button. It will output the list of 1613 Mordred_2-D descriptors that are presently accessible (**Fig. 5**).

Fingerprints

Users can input Canonical SMILES for a ‘Single molecule’ or multiple SMILES by uploading the .csv file in the ‘Multiple molecules (Batch)’ mode of the DesCal module. This allows users to obtain fingerprints like MACCS and Morgan fingerprints also known as Extended Connectivity Circular Fingerprints (ECFP) for radius = 1, 2 and 3 for their

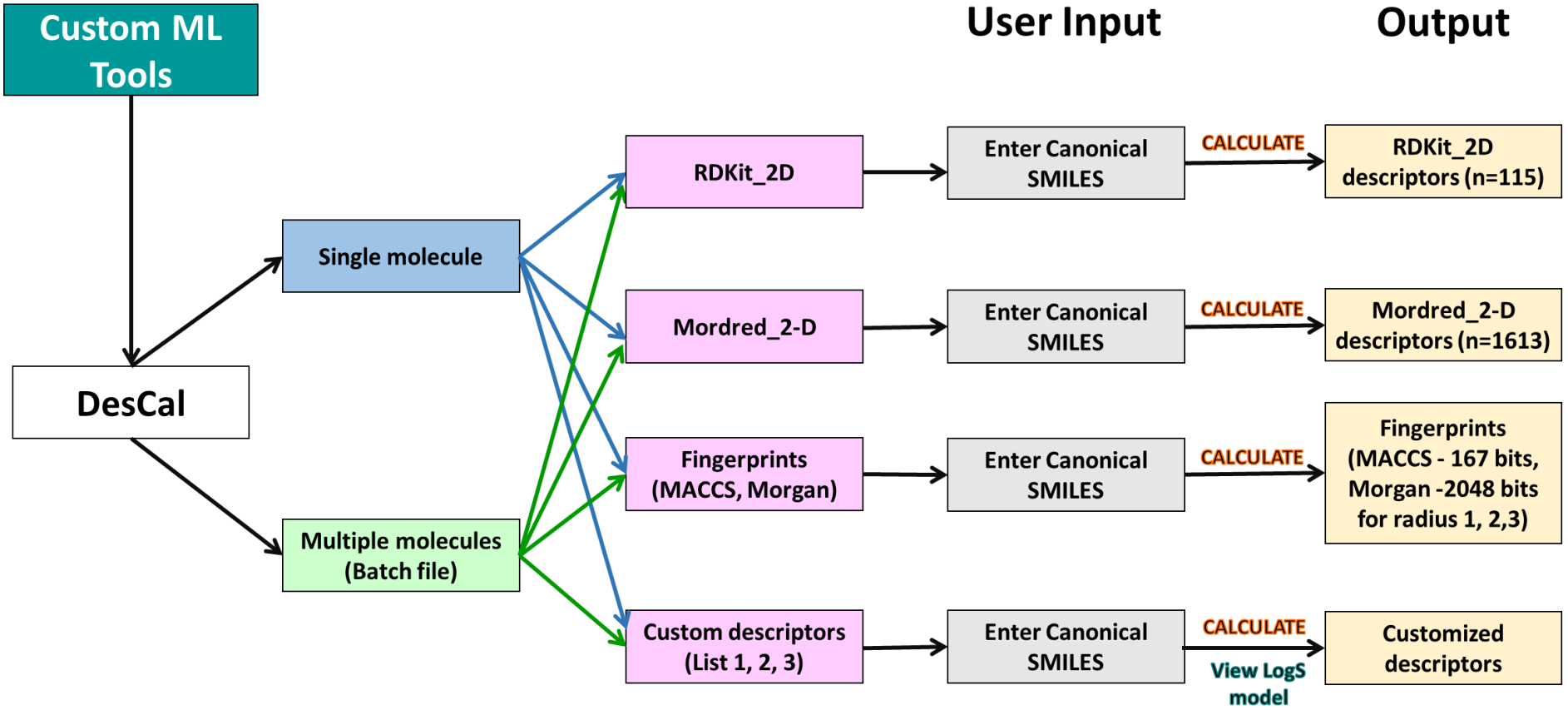


Fig. 5: Workflow highlighting the computation of molecular 2-D descriptors and fingerprints by using 'DesCal' module of 'Custom ML Tools'.

molecular SMILES after clicking the 'CALCULATE' button. In both fingerprints, each bit position reflects the presence or absence of certain substructures. Users can extract smaller fragments with a smaller radius, and bigger fragments with a bigger radius. Users can extract MACCS fingerprints with 167 bits and Morgan fingerprints with 2048 bits that are presently accessible (**Fig. 5**).

Custom descriptors

Users can input Canonical SMILES for a 'Single molecule' or multiple SMILES by uploading the .csv file in the 'Multiple molecules (Batch)' mode of the DesCal module. This allows users to obtain Custom descriptors of their choice from the given lists for their molecular SMILES after clicking the 'CALCULATE' button (**Fig. 5**). Users can extract molecular descriptors of their choice as available in three lists, which are as follows:

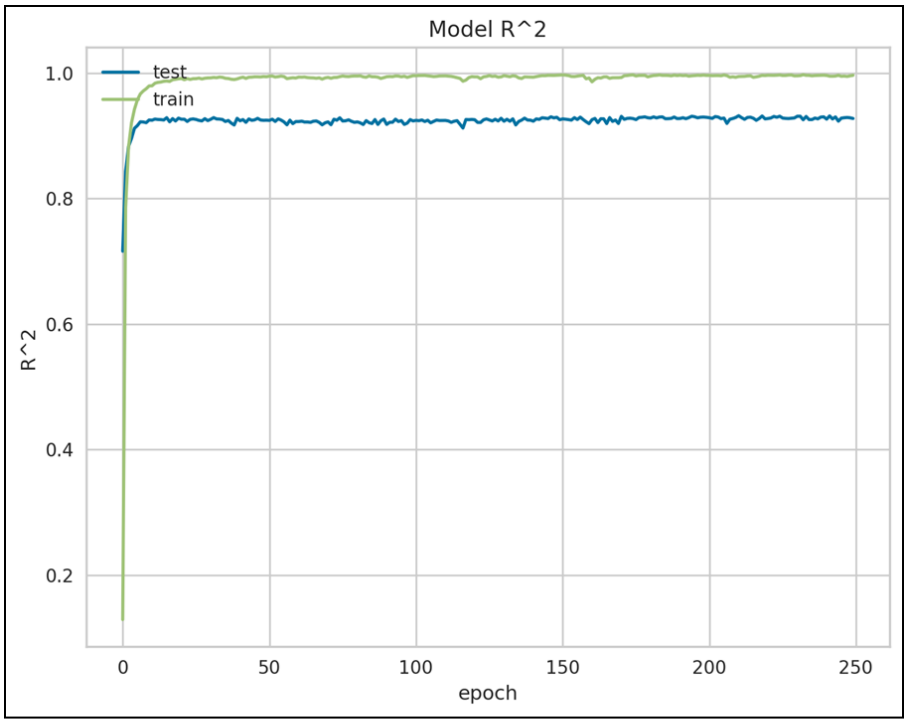
List 1: 'MolLogP', 'MolWt', 'NumHAcceptors', 'NumHDonors', 'NumRotatableBonds', 'RingCount', 'TPSA', 'HeavyAtomCount'.

List 2: 'nAromAtom', 'Wpath', 'Vabc', 'nAcid', 'nBase'

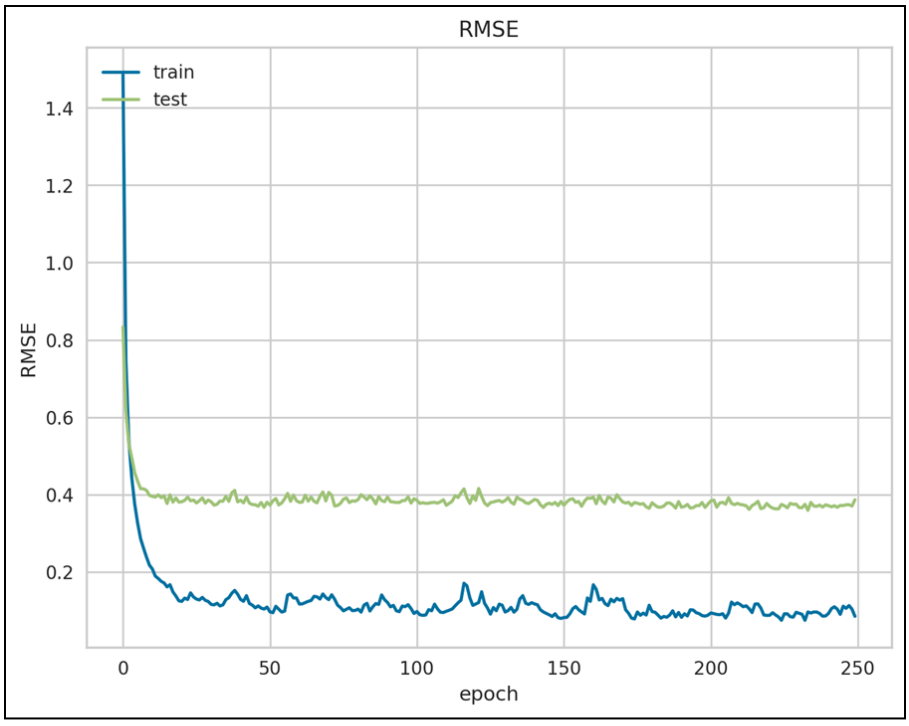
List 3: 'logS', 'AromaticProportion', 'Density'

While predicting/ calculating 'logS' values, users can also view the DL based QSAR model data for LogS and model evaluation by clicking the check box. The model is trained on RDKit 2-D descriptors and MACCS fingerprints generated for Delaney [31] solubility dataset of n= 1144 molecular SMILES with observed solubility values in mol/L. The data has been divided into 75:25 train test splits. Users can also view the model summary and metrics used for model prediction evaluation each for training and test set. Training curves for R^2, RMSE and Linear regression plots for training and test sets are shown in **Fig. 6**. The best model was developed with the highest Coefficient of determination (R2) score (Train: 0.99, Test: 0.93)

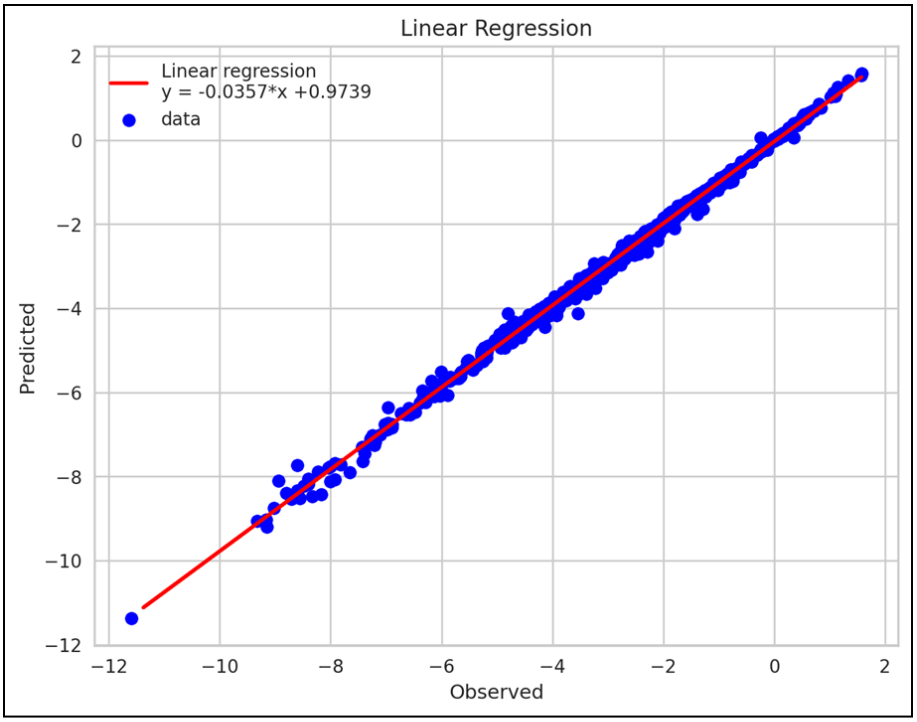
and the lowest MSE score (Train: 0.02, Test: 0.29) and MAE score (Train: 0.02, Test: 0.29). (Train: 0.09, Test: 0.38).



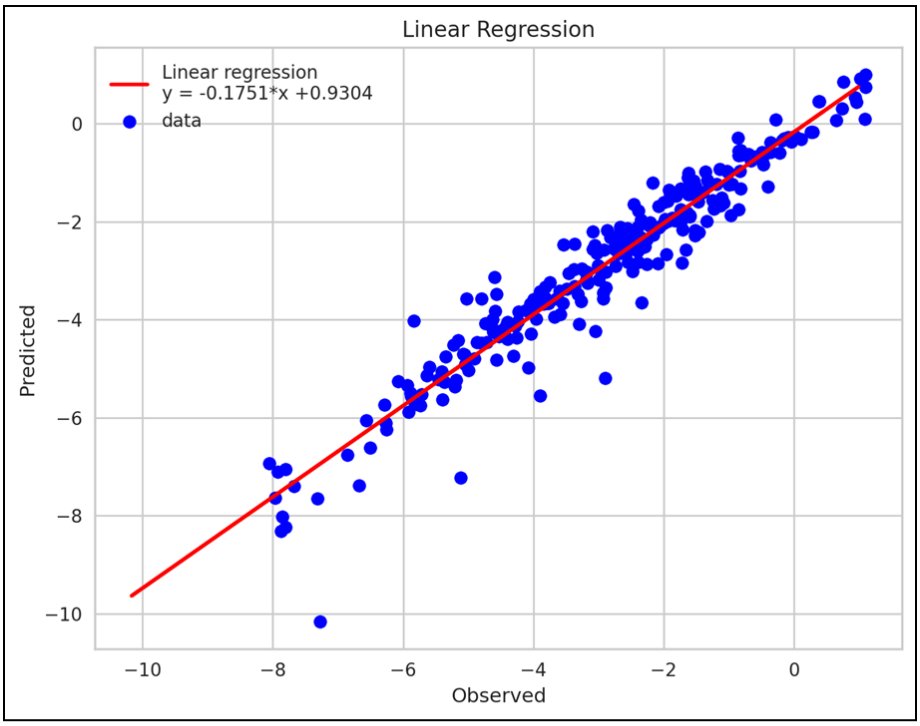
A.



B.



C.



D.

Fig. 6: Training curves for (**A.**) R2, (**B.**) RMSE and Linear regression plots for (**C.**) training and (**D.**) test sets for logS Model

Building DL model and predicting target data using Auto-DL

Users can input molecular descriptors in a .csv file with target data for a 'Classification' or 'Regression' mode of the Auto-DL module. This allows users to obtain the best DL model for their input data. To predict using the same DL model, users must enter their data before clicking the 'PREDICT' button. It will generate the train-test split of the target data, as well as its model summary, model evaluation scores, and predicted results for visualization and download using the same DL model that was developed.

The entire process takes only four steps (**Fig. 7**) to complete by pressing a button, as follows:

Step 1: In the User input-side panel, select the type of algorithm ('Classification' or 'Regression') for building the DL model.

Step 2: Upload descriptor data (included with target data) for building DL model (Example input file provided)

Step 3: For developing the model, specify parameters such as 'Train-Test split percent', 'random seed number', 'maximum trial number', and 'epochs number'.

Step 4: Upload descriptor data (excluded with target data) for making target predictions (Example input file provided) and click on the 'PREDICT' button to display the results.

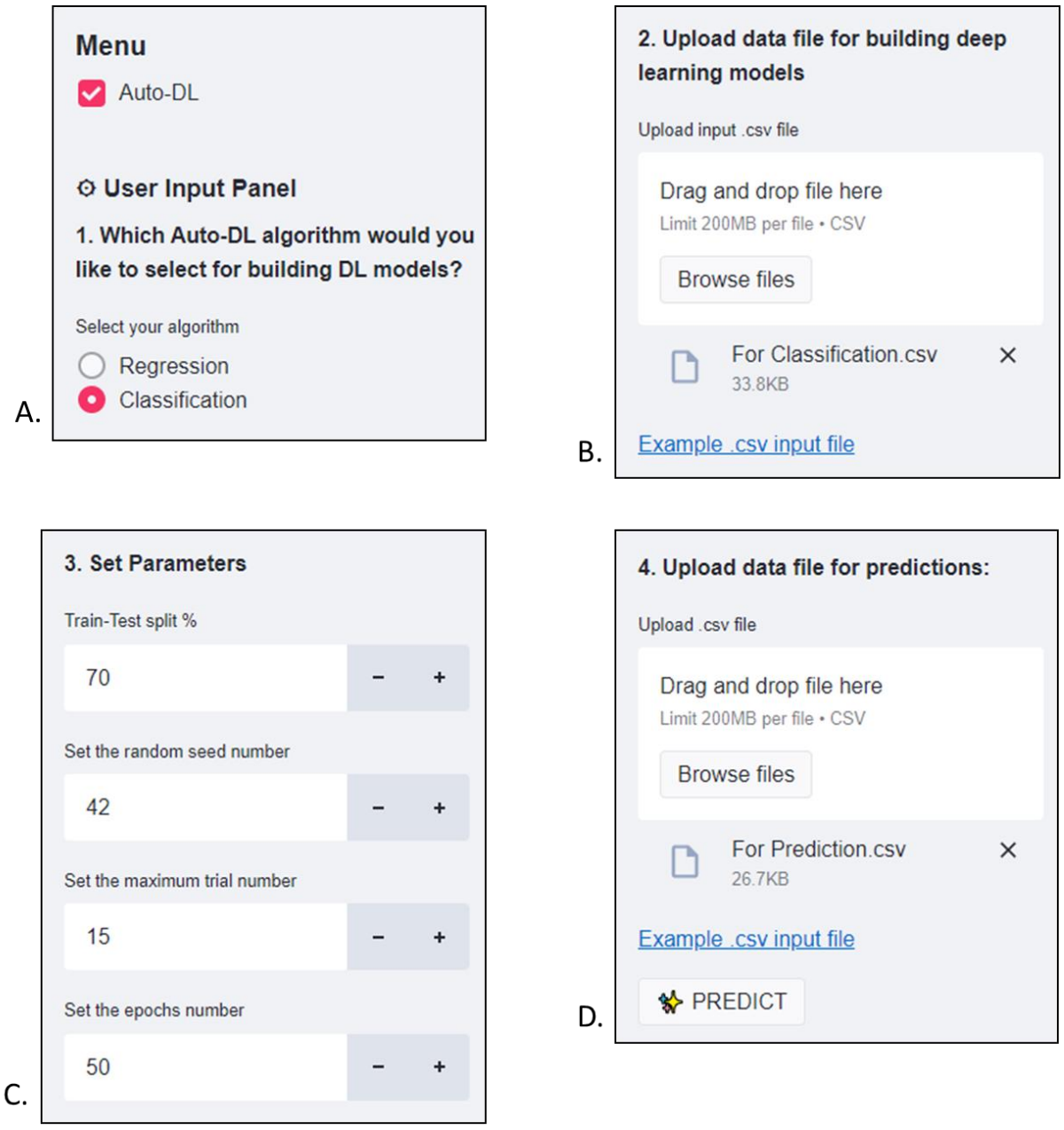


Fig. 7: Steps (**A - D**) for developing an automated DL model and using it to predict data

Building multiple ML models and predicting target data using Auto-Multi-ML

Users can input molecular descriptors in a .csv file with target data for a 'Classification' or 'Regression' mode of the Auto-Multi-ML module. This allows users to obtain multiple ML models (n=40) and their interpretation (RMSE, R^2 and time taken for building model) for their input data. Users are also allowed to select the ML model from the provided list for prediction on new data based on the interpretation of multiple models. To predict using the best ML model, users must enter their data before clicking the 'PREDICT' button. It will generate an Exploratory Data Analysis (EDA) report for uploaded data in the form of SWEETVIZ_report [33], feature engineered data with significant features, a train-test split of the feature engineered data, as well as an EDA report for feature engineered data in the form of SWEETVIZ_report, interpretation of multiple models for training and test set, and predicted results for visualization and download using the best ML model that was developed.

The entire process takes only five steps (**Fig. 8**) to complete by pressing a button, as follows:

Step 1: In the User input-side panel, select the type of algorithm ('Classification' or 'Regression') for building the multiple ML models.

Step 2: Upload descriptor data (included with target data) for building multiple ML models (Example input file provided)

Step 3: For exploratory data analysis and/or interpreting and comparing multiple ML models, tick the checkboxes for 'EDA of uploaded data', 'EDA of feature engineered data', and 'Interpret multiple ML models'.

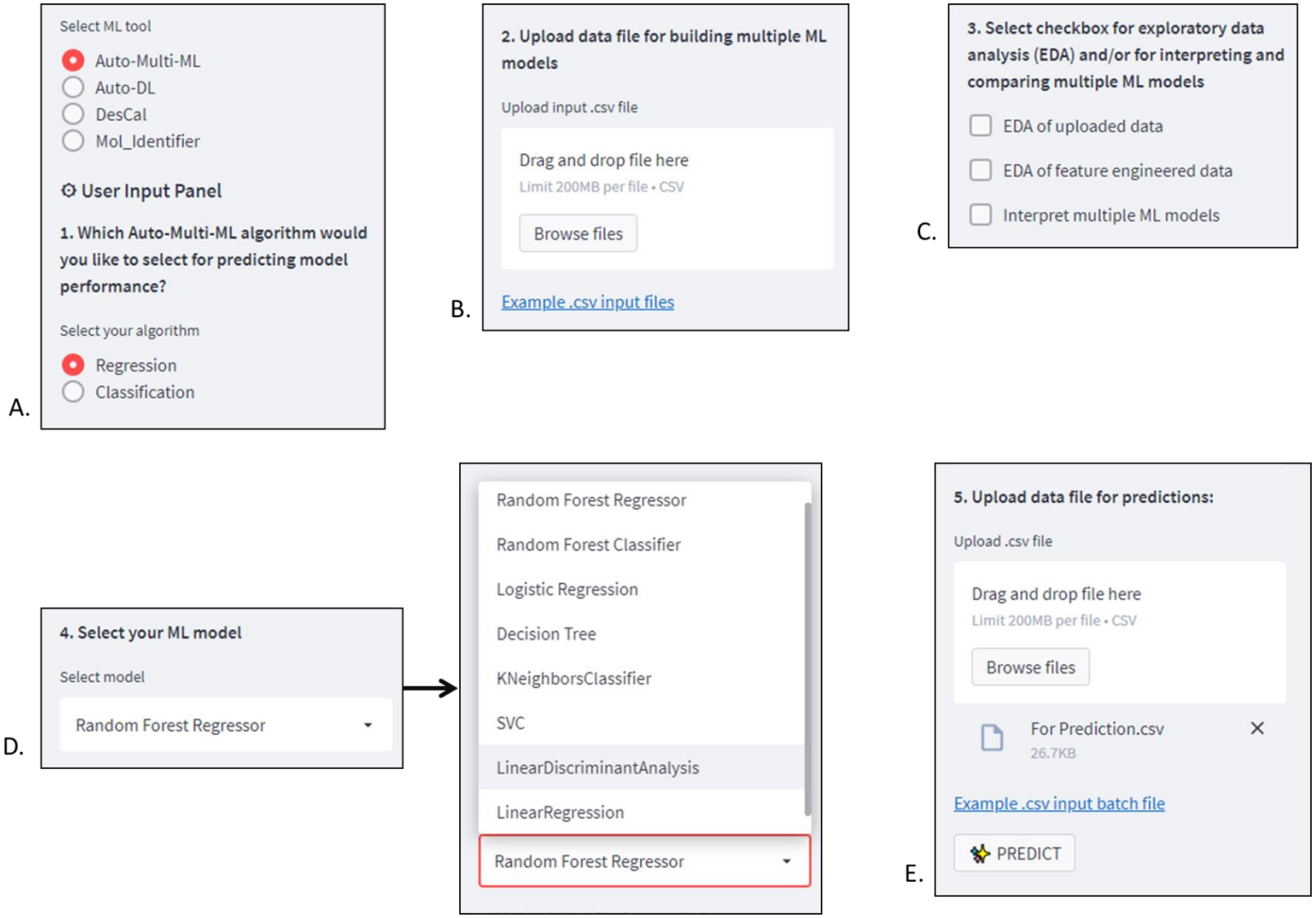


Fig. 8: Steps (A - E) for developing automated multiple ML models and using them to predict data

Step 4: After evaluating and understanding the outcomes of different ML models, select ML algorithm (ideally the top-performing ML model for the best results) for building an ML model on new data and predicting target data.

Step 5: Upload descriptor data (excluded with target data) based on feature-engineered model data for making target predictions by applying selected built ML model (Example input file provided) and click on the 'PREDICT' button to display and download the results.

CONCLUSIONS

The "Custom ML Tools" is an integral part of the "AIDrugApp" that provides users a convenient and online way to identify compounds using SMILES, compound names, and similarity scores. It also enables users to compute molecular descriptors, build ML/DL models, and predict data using built models. It can be used for various scientific applications such as QSAR, similarity search, ADMET prediction, virtual screening and data analysis. It is designed with an elegant and user-friendly graphical interface for a cost-free and login-free WebApp platform. Users may utilise interactive features to input simple data and analyse it quickly, and they don't need any prior knowledge of coding or computer-aided design. Finally, the study reveals that the new tool has the potential to help the whole community (biologists, chemists, and nonexperts) with their drug discovery efforts.

AUTHOR INFORMATION

CORRESPONDING AUTHOR

Phone: +91-8830379882, E-mail: divya.karade@gmail.com

CONFLICT OF INTEREST

The authors confirm that this article's content has no conflict of interest.

ACKNOWLEDGEMENTS

The authors would like to thank the Government of India, Drug Discovery Hackathon 2020 (https://innovateindia.mygov.in/ddh2020/) and India International Science Festival 2020 organizers team for providing financial support as part of the DDH2020 and IISF2020 program.

ABBREVIATIONS

CADD: Computer-Aided Drug Design; DL: Deep Learning; ML: Machine Learning; QSAR: Quantitative Structure-Activity Relationship; SMILES: Simplified Molecular Input Line Entry System; MSE: Mean Squared Error; RMSE: Root Mean Squared Error; MAE: Mean Absolute Error; R^2: Coefficient of determination

REFERENCES

[1] X. Lin, Y. Xu, X. Pan, J. Xu, Y. Ding, X. Sun, X. Song, Y. Ren, P.-F. Shan, Global, regional, and national burden and trend of diabetes in 195 countries and territories: an analysis from 1990 to 2025, J Scientific reports, 10 (2020) 1-11.

[2] K.I. Kaitin, C.P. Milne, A dearth of new meds, J Scientific American, 305 (2011) 16-16.

[3] K.I. Kaitin, The quest to develop new medicines to treat Alzheimer's disease: present trends and future prospects, J Clinical therapeutics, 37 (2015) 1618-1621.

[4] E. Smalley, AI-powered drug discovery captures pharma interest, J Nature biotechnology, 35 (2017) 604-606.

[5] J. Jiménez-Luna, F. Grisoni, N. Weskamp, G. Schneider, Artificial intelligence in drug discovery: Recent advances and future perspectives, J Expert Opinion on Drug Discovery, (2021) 1-11.

[6] L. Zhao, H.L. Ciallella, L.M. Aleksunes, H. Zhu, Advancing computer-aided drug discovery (CADD) by big data and data-driven machine learning modeling, Drug discovery today, (2020).

[7] M. Eisenstein, AI in Drug Discovery Starts to Live Up to the Hype: Machine learning–based techniques are finding a place in early-stage target discovery and in drug development workflows, J Genetic Engineering and Biotechnology News, 41 (2021) 38-40.

[8] R. Gupta, D. Srivastava, M. Sahu, S. Tiwari, R.K. Ambasta, P. Kumar, Artificial intelligence to deep learning: Machine intelligence approach for drug discovery, J Molecular Diversity, (2021) 1-46.

[9] M. Bule, N. Jalalimanesh, Z. Bayrami, M. Baeeri, M. Abdollahi, The rise of deep learning and transformations in bioactivity prediction power of molecular modeling tools, Chemical Biology Drug Design, (2021).

[10] D. Wang, J. Yu, L. Chen, X. Li, H. Jiang, K. Chen, M. Zheng, X. Luo, A hybrid framework for improving uncertainty quantification in deep learning-based QSAR regression modeling, Journal of cheminformatics, 13 (2021) 1-17.

[11] B. Ramsundar, B. Liu, Z. Wu, A. Verras, M. Tudor, R.P. Sheridan, V. Pande, Is multitask deep learning practical for pharma?, Journal of chemical information modeling, 57 (2017) 2068-2076.

[12] H.S. Chan, H. Shan, T. Dahoun, H. Vogel, S. Yuan, Advancing drug discovery via artificial intelligence, Trends in pharmacological sciences, 40 (2019) 592-604.

[13] N. Brown, In silico medicinal chemistry: computational methods to support drug design, Royal Society of Chemistry2015.

[14] J.C. Pereira, E.R. Caffarena, C.N. Dos Santos, Boosting docking-based virtual screening with deep learning, Journal of chemical information modeling, 56 (2016) 2495-2506.

[15] L. G, RDKit: open-source cheminformatics. http://www.rdkit.org.

[16] H. Moriwaki, Tian, YS., Kawashita, N., Takagi, T., Mordred: a molecular descriptor calculator, J Cheminform, 10 (2018).

[17] Ö. Sahin, Introduction to Apple ML Tools, Develop Intelligent iOS Apps with Swift, Springer2021, pp. 17-39.

[18] K. Divya, K. Vikas, AIDrugApp: Artificial Intelligence-based Web-App for Virtual Screening of Inhibitors against SARS-COV-2, Journal of Experimental & Theoretical Artificial Intelligence (Accepted/forthcoming) (ChemRxiv Preprint: https://doi.org/10.26434/chemrxiv.13312661.v1), (2020).

[19] B.-H. Lee, E.K. Dewi, M.F. Wajdi, Data security in cloud computing using AES under HEROKU cloud, 2018 27th Wireless and Optical Communication Conference (WOCC), IEEE, 2018, pp. 1-5.

[20] L. Dabbish, C. Stuart, J. Tsay, J. Herbsleb, Social coding in GitHub: transparency and collaboration in an open software repository, Proceedings of the ACM 2012 conference on computer supported cooperative work, 2012, pp. 1277-1286.

[21] G. Landrum, RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling, Academic Press, 2013.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, Scikit-learn: Machine learning in Python, Journal of machine Learning research, 12 (2011) 2825-2830.

[23] W. McKinney, pandas: a foundational Python library for data analysis and statistics, J Python for high performance scientific computing, 14 (2011) 1-9.

[24] C.R. Harris, K.J. Millman, S.J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. Smith, Array programming with NumPy, Nature, 585 (2020) 357-362.

[25] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, Tensorflow: A system for large-scale machine learning, 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16), 2016, pp. 265-283.

[26] N.K. Manaswi, Understanding and working with Keras, Deep Learning with Applications Using Python, Springer2018, pp. 31-43.

[27] P. Singh, Deploy machine learning models to production: with flask, streamlit, docker, and kubernetes on google cloud platform, Apress2021.

[28] M. Swain, PubChemPy: A way to interact with PubChem in Python, 2014.

[29] H. Moriwaki, Y.-S. Tian, N. Kawashita, T. Takagi, Mordred: a molecular descriptor calculator, Journal of cheminformatics, 10 (2018) 1-14.

[30] B. Ramsundar, P. Eastman, P. Walters, V. Pande, Deep learning for the life sciences: applying deep learning to genomics, microscopy, drug discovery, and more, " O'Reilly Media, Inc."2019.

[31] J.S. Delaney, ESOL: estimating aqueous solubility directly from molecular structure, Journal of chemical information computer sciences, 44 (2004) 1000-1005.

[32] H. Jin, Q. Song, X. Hu, Auto-keras: An efficient neural architecture search system, Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 1946-1956.

[33] J. Peng, W. Wu, B. Lockhart, S. Bian, J.N. Yan, L. Xu, Z. Chi, J.M. Rzeszotarski, J. Wang, DataPrep. EDA: Task-Centric Exploratory Data Analysis for Statistical Modeling in Python, Proceedings of the 2021 International Conference on Management of Data, 2021, pp. 2271-2280.