

TrainLab - Python-based Automatic Model Trainer and Analyzer

Sivashankar S¹ , Sai Prasad R²

Abstract

This article introduces a new Streamlit application that makes machine learning (ML) procedures easier to understand and utilize for people with varying technical skills. To tackle the accessibility issue, we have developed a web-based interface that streamlines important machine learning operations such as data transfer, exploration, and model training. The program offers an easy-to-use interface by utilizing the Streamlit framework. Users may easily submit their own datasets and obtain insightful information by using pandas_profiling's integrated data profiling feature. The program incorporates the potent pycaret library to simplify model selection and training. Users don't need to learn complicated coding techniques to compare and train different ML models with ease. This enables users, regardless of subject expertise, to fully utilize machine learning for their applications. Future Work: By adding feature engineering methods and hyperparameter tweaking tools, we want to expand the functionality of the program. Furthermore, there is a ton of potential for useful applications when the trained model is made available as a web service for real-time predictions. Through its user-friendly and approachable design, this Streamlit application helps to open up the use of machine learning (ML) in a wider range of fields. It facilitates a more open environment for machine learning research by enabling users to fully utilize ML for data analysis and predictive modeling.

Keywords: Democratization, User-Centric, Streamlit, Machine Learning, PyCaret, Data Preprocessing, Model Training, Model Comparison

1. Introduction

This paper introduces a new Streamlit application created especially to deal with this problem. We provide a customer-focused platform that makes machine learning more accessible to a wider audience by streamlining the whole workflow. Our application creates a more democratized environment for machine learning creation and exploration by enabling people with varying technical skill sets to harness the potential of machine learning for their projects. The standard machine learning workflow may be a complex process that calls for a high level of knowledge in training, assessment, model selection, and data manipulation. For people without the requisite technical experience or coding expertise, this intricacy may be a significant turnoff. Furthermore, even seasoned users may become overwhelmed by the sheer quantity of machine learning algorithms and hyperparameters accessible, making it difficult to choose the best model for a given task. With its user-friendly and interactive interface, our Streamlit application takes on these difficulties head-on. Uploading datasets is simple for users, which is an essential initial step for every machine learning project. The application's integrated data profiling features then make it easier to comprehend the data on a deeper level. Users may obtain important insights into the properties of the data, such as data types, missing values, and distribution patterns, by utilizing libraries such as pandas_profiling. Users are eventually able to create more reliable and accurate machine learning models by using this enhanced understanding to make well-informed judgments on data cleaning and preprocessing procedures. Following the preparation of the data, the program effortlessly moves to the model training stage. Here, we make use of the PyCaret library's capabilities. PyCaret offers a full suite of pre-built machine learning algorithms, which streamlines the process of choosing and training models. Depending on the type of prediction work

-
1. Sivashankar S
Rajalakshmi Engineering College, Thandalam
E-mail : sivashankars0803@gmail.com
 2. Sai Prasad R
Rajalakshmi Engineering College, Thandalam
E-mail : saiprasad7747@gmail.com

they are undertaking, users can select from a range of methods for classification, regression, or clustering. Additionally, PyCaret automates the process of training models, saving users from having to write intricate code for data splitting, model fitting, and assessment. Because of this, training and comparing many ML models takes a lot less time and effort, freeing up users to concentrate on evaluating the data and choosing the best model for their particular requirements.

Our Streamlit application has benefits beyond just making data exploration and model training easier. Additionally, the program offers functions for downloading models for later deployment and visualizing results. Users may evaluate performance data, including accuracy, precision, and recall, after training different models. This helps customers choose the model that best fits their project objectives and make well-informed decisions. The program also makes it easier for users to connect learned models for real-time prediction into other software systems or online applications. This creates opportunities for machine learning to be used practically in a variety of disciplines. All things considered, our Streamlit application marks a major advancement in the democratization of machine learning. We enable people with diverse technical backgrounds to leverage machine learning (ML) for data analysis and predictive modeling by providing an intuitive, interactive platform that optimizes the ML workflow. By streamlining data discovery, automating model training and comparison, and easing model deployment, the program makes machine learning (ML) not only more effective but also more widely available. This encourages a more welcoming atmosphere for machine learning research and development, opening the door for its broader use and practical applications.

2. Related Works

No	Name/Year	Title	Method Used	Advantages	Disadvantages
1	A. Géron (2017)	Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow	Scikit-learn library	- Powerful and customizable - Extensive documentation and community support	- Steeper learning curve for beginners - Requires coding knowledge
2	T. Wickham (2015)	ggplot2: Elegant Graphics for Data Analysis	ggplot2 package for R	- Creates visually appealing and informative graphs	- Limited to R programming language
3	M. Kuhn & K. Johnson (2019)	Applied Predictive Modeling	Provides a framework for building ML models	- Comprehensive guide to the ML lifecycle	- Lacks a user interface, requires coding
4	J. Brownlee (2016)	Machine Learning Mastery	Online tutorials and resources	- Wide range of topics covered - Beginner-friendly explanations	- Information scattered across various articles - Lacks interactivity
5	KNIME (2024)	KNIME Analytics Platform	Drag-and-drop interface for building workflows	- User-friendly for non-programmers - Large library of pre-built nodes	- Limited customization compared to code-based tools
6	RapidMiner (2024)	RapidMiner Studio	Visual interface for data mining and machine learning	- Streamlined data processing and model building - Extensive community support	- Free version has limitations - Can be resource-intensive

[1] A. Géron's Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Aurélien Géron's thorough book provides a useful manual for creating and implementing machine learning models with well-known libraries like Scikit-learn, Keras, and TensorFlow. It explores the foundations of deep learning ideas, model assessment metrics, data pretreatment methods, and machine learning algorithms. Although Scikit-learn offers a wide range of tools for training, choosing models, and manipulating data, putting these features into practice requires users to write code. This might be a major obstacle for individuals who are new to Python programming or novices. For individuals who are prepared to put in the effort to learn to code, Scikit-learn is a great resource because of its vast documentation and vibrant community.

[2] Wickham, T's ggplot2: Elegant Graphics for Data Analysis: The robust R tool ggplot2, developed by Hadley Wickham, makes it easier to create illuminating and eye-catching data visualizations. By defining the data aesthetics and variable connections, it provides a grammar-based method that enables users to create intricate plots. Although ggplot2 is restricted to the R programming language, it enables users to obtain important insights from their data through eye-catching visuals. Users accustomed to other programming environments, such as Python, may find this to be a hindrance.

[3] M. Kuhn & K. Johnson's *Applied Predictive Modeling* : This book, written by Max Kuhn and Kjell Johnson, offers an organized framework for creating and utilizing predictive models. It explores every step of the machine learning lifecycle, including feature engineering, data preparation, model selection, training, assessment, and deployment. Although this extensive manual offers insightful information to novices and seasoned professionals alike, it is not user-friendly. To create and train models, users must convert the principles shown into code using the programming environment of their choice. This takes some coding experience and can take a long time.

[4] J. Brownlee's *Machine Learning Mastery* : Machine Learning Mastery is an extensive online resource that was founded by Jason Brownlee. It provides a wealth of lessons, articles, and courses addressing many facets of machine learning. It serves a wide range of users by offering comprehensive talks for seasoned practitioners in addition to approachable explanations for novices. The breadth of topics covered by the site, from basic ideas to specialized algorithms and deep learning approaches, is its greatest asset. However, it is difficult to follow an organized learning route because the material is dispersed over several articles. Furthermore, there isn't an interactive environment on the site where users may try out and practice the ideas they learn.

[5] KNIME Analytics Platform : KNIME provides an approachable platform made especially for machine learning and data science applications. Through the use of a drag-and-drop interface, users may construct workflows by joining pre-built nodes that stand in for different modeling, analytical, and data manipulation tasks. Users with no coding knowledge may use KNIME because to its visual approach. KNIME also has an extensive library of pre-built nodes that includes features for data cleansing, feature engineering, training models, and visualization. On the other hand, KNIME is less flexible than code-based solutions. For skilled users, extensive customization is impeded by the restricted functions offered by the accessible nodes. Furthermore, there can be restrictions on the

number of nodes and data processing power that are accessible in the free edition of KNIME.

[6] RapidMiner Studio : Offering a visual interface for data mining and machine learning activities, RapidMiner Studio is akin to KNIME. By joining nodes that stand for various stages in data processing and modeling, users may create workflows. With features for data cleansing, transformation, model selection, training, and assessment, RapidMiner simplifies data processing and model construction. A strong community that provides more nodes and resources is also beneficial to it. Although RapidMiner's free edition is an excellent place to start, it has restrictions on the amount of data it can handle and the services it can give. Furthermore, using RapidMiner Studio may be resource-intensive, necessitating a computer with the processing capacity to handle sophisticated models and massive datasets.

3. Methodology

A rising emphasis in the rapidly developing field of machine learning is on expanding the audience's ability to utilize these potent instruments. This section explores current tools and systems designed to help individuals with different technical backgrounds understand complicated algorithms. Our solution addresses the issue of accessibility in machine learning by providing a code-free, user-friendly platform that optimizes the whole machine learning process. This section explores our application's process, outlining the steps involved in the project and its flow. Our application's core feature is its user-friendly interface, which is based on the Streamlit framework. Python code may be used directly to create web apps using Streamlit. This makes the program available through a straightforward web browser interface and removes the need for users to master sophisticated web development tools. Users are met with an application that has an easy-to-use interface with helpful text and navigational components as soon as they activate it. Any machine learning project must begin with gathering and uploading the data. Our program has an easy-to-use

file upload feature that supports CSV files, which are the most popular format for tabular data. When a user chooses a CSV file, the program opens it immediately in a pandas DataFrame, a robust Python data processing tool. Prior to starting the model training process, it is essential to comprehend the properties of your data. By integrating the `'pandas_profiling'` module, our application provides users with easy access to insightful information about their data. A thorough report that illustrates data kinds, missing values, distribution patterns, correlations, and descriptive statistics may be generated by the program upon request. Users are given the ability to recognize any problems with data quality and decide on appropriate preprocessing and data cleaning procedures with the help of this report. Model training and selection provide a number of major challenges for conventional machine learning procedures. Our application integrates the potent `'pycaret'` library to address this difficulty. Pycaret provides a collection of pre-constructed machine learning algorithms that cover tasks related to classification, regression, and clustering. The target variable that the user wants to forecast may be easily selected from the available data columns. Depending on whether the target variable is categorical or numerical, the program makes recommendations for pertinent algorithms from the Pycaret library automatically. After users have decided on their target variable and preferred model type (regression, classification, etc.), the program handles the difficult process of training and comparing models. Model training, hyperparameter adjustment for each model, and data division into training and testing sets are all automated via Pycaret. Users no longer have to write complicated code or learn the nuances of hyperparameter tuning. Following the training of any selected model, Pycaret offers a table of performance comparisons. Key performance indicators for each model are included in this table, enabling users to quickly compare results and choose the model that most closely matches their project objectives. These metrics include accuracy, precision, recall, and F1-score. Our application gives customers the ability to download models for later usage in addition to training them on the platform. This feature is

especially helpful in situations when forecasts must be made in real time. Users can incorporate the trained model into online apps or other software systems by downloading it, which will allow them to anticipate new data points outside of the application's interface. The downloaded model is stored in a common format (pickle file, for example) that can be opened with a number of Python machine learning tools.

4 Experiment

4.1 Choose ML type

The first stage is to decide what kind of machine learning technique you wish to complete (fig 1). Regression and classification are the two primary categories that our product delivers. Classification models are appropriate for predicting discrete categories (e.g., spam or not spam, customer turnover), whereas regression models are best for predicting continuous values (e.g., property price, sales figures). You may direct the program to recommend relevant methods for your particular prediction task by choosing the appropriate category.

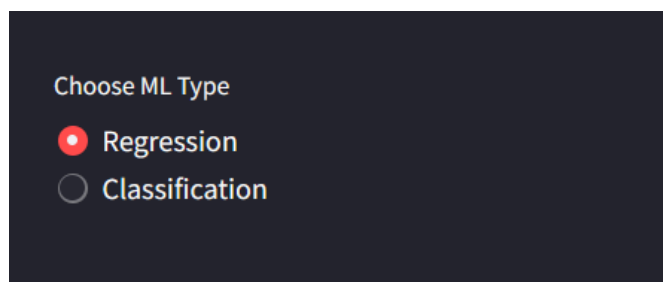


fig 1

4.2 Upload Dataset

The next step is to provide the dataset that your model will use for learning after you've selected the machine learning type. The program has an easy-to-use file upload feature (fig 2) that supports CSV (comma-separated values) files, which are a popular format for storing tabular data. All you need

to do is choose the CSV file that holds your data, and the program will take care of the rest.

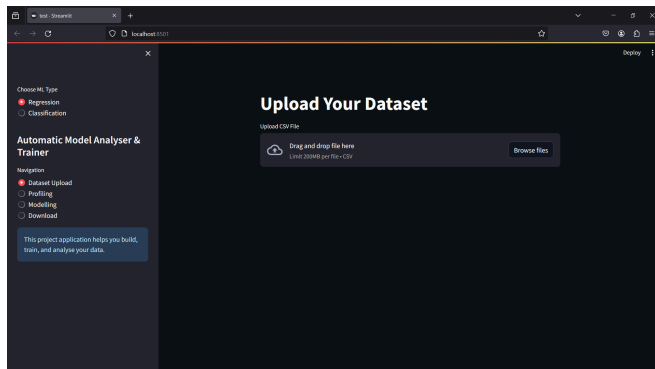


fig 2

4.3 Preview Dataset

It is essential to comprehend the composition and organization of your data prior to beginning the model training process. You can get a brief glimpse of your data using our application's additional preview function (fig 3). This can be useful for checking the quantity of rows and columns, classifying the data, or detecting any initial anomalies.

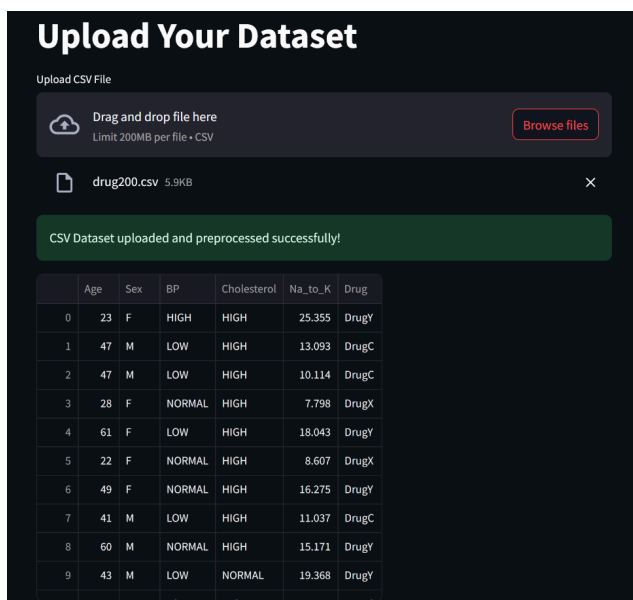


fig 3

4.4 Profiling

Utilizing the integrated data profiling capabilities will provide you a deeper insight of your data. The program uses the pandas_profiling library to do a comprehensive analysis by requesting a profile report. Data types, missing values, distribution patterns, correlations, and descriptive statistics are all shown in this report, giving you important information to help you decide what data cleaning and preprocessing procedures to do.

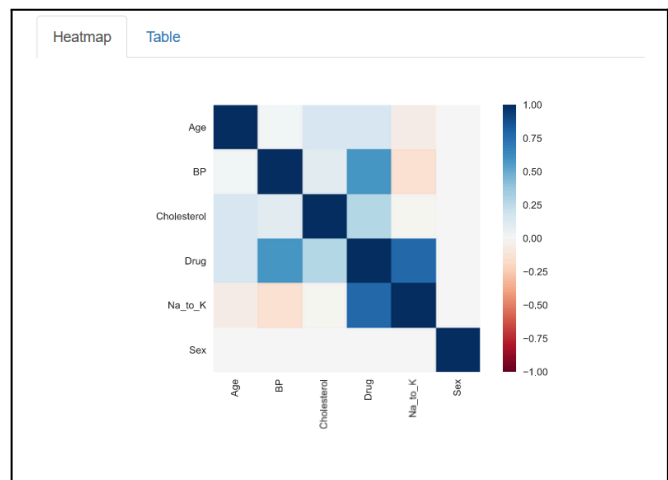


fig 4.1

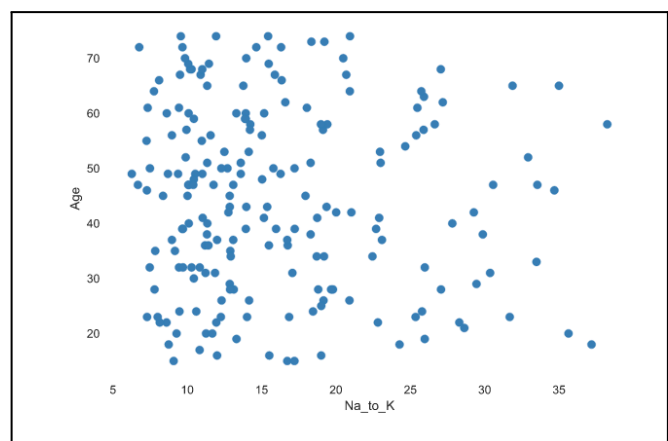


fig 4.2

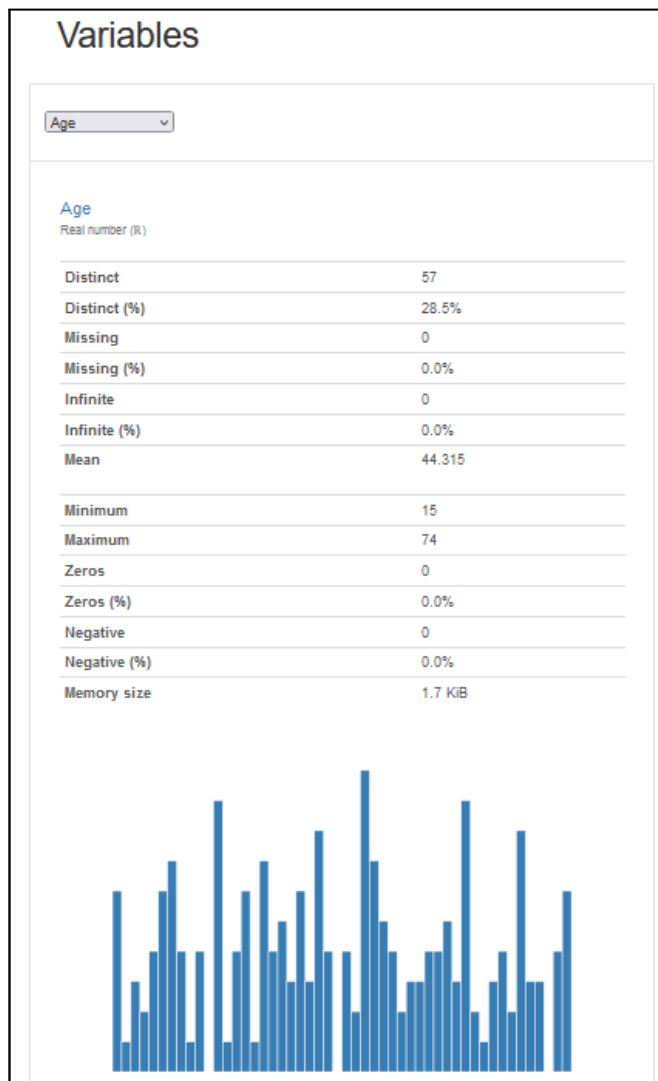


fig 4.3

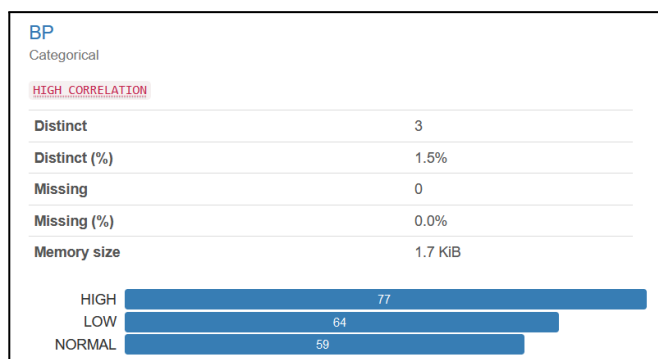


fig 4.4

4.5 Model Training

This Streamlit application uses the Pycaret library's power for its model training procedure. Pycaret provides an array of pre-constructed algorithms appropriate for the chosen task, based on the user's chosen target variable (fig 5). The application does all of the training for you, saving you from having to write complicated code. This entails dividing the data into testing and training sets, adjusting the hyperparameters for every model, and then training the models. Following training, Pycaret creates a performance comparison table that presents each trained model's accuracy, precision, recall, and F1-score. Through the analysis of this table, users may determine which model performs best in relation to the particular criteria of their project (e.g., optimizing accuracy or emphasizing precision). This "best fitting algorithm" can then be used for making predictions on new data points.

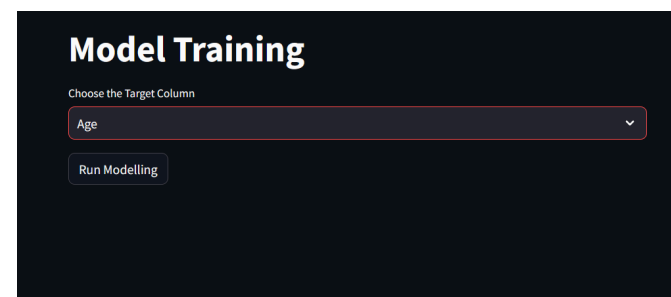
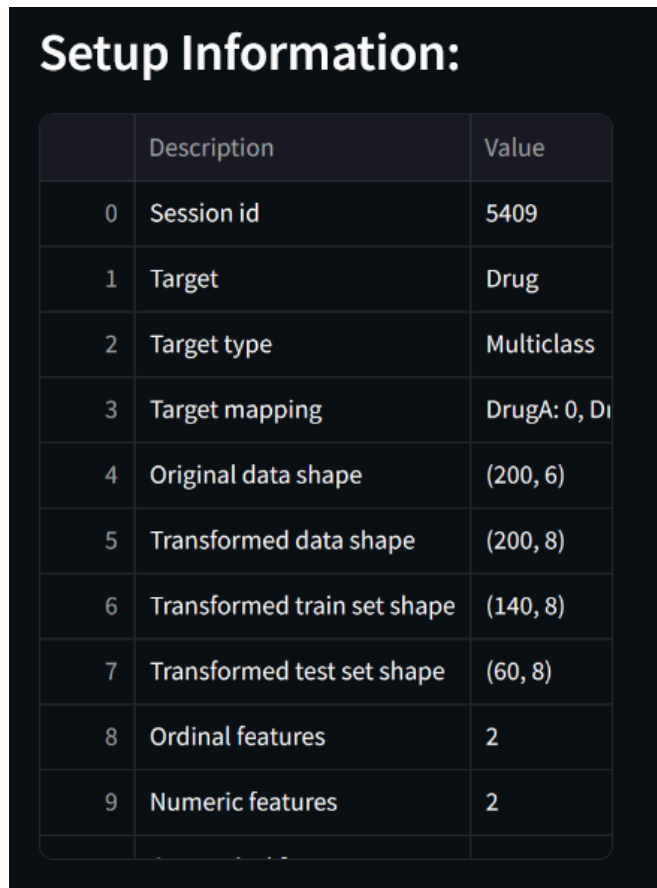


fig 5.1

Our Streamlit application's "Setup Information" tab serves as its command center and provides an overview of the data and settings for the active machine learning session. It shows information such as a distinct session ID, the goal variable (classification or regression) that you want to forecast, and the original and modified forms (number of rows and columns) of your data. Any preparation or data cleaning that was done within the application is reflected in this transformation. The screen also provides a breakdown of the characteristics, classifying them as numeric or ordinal (categorical with order). The program needs

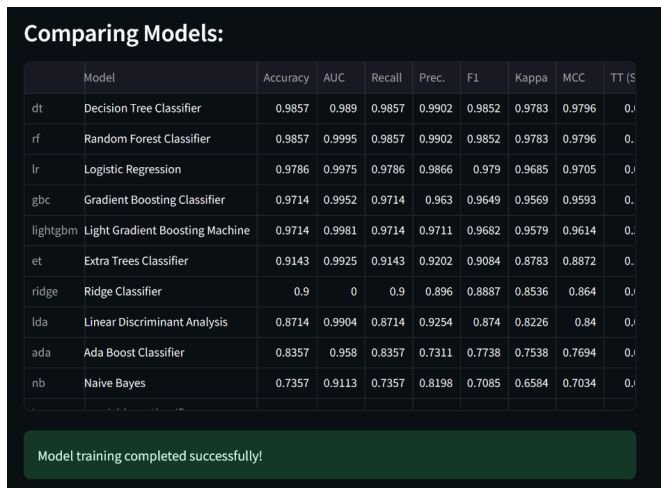
this data type information in order to choose the right machine learning algorithms for training. The information on this page, serves as essentially a snapshot of the data and configuration used to train the model (fig 5.2), offers important insights into the phase of model training that is started by clicking the "Run Modeling" button.



	Description	Value
0	Session id	5409
1	Target	Drug
2	Target type	Multiclass
3	Target mapping	DrugA: 0, DrugB: 1, DrugC: 2
4	Original data shape	(200, 6)
5	Transformed data shape	(200, 8)
6	Transformed train set shape	(140, 8)
7	Transformed test set shape	(60, 8)
8	Ordinal features	2
9	Numeric features	2

fig 5.2

By analyzing the performance comparison table (fig 5.3) presented within the application, users can select the model with the most suitable performance for their specific requirements. This "best-performing model" is then considered the "best algorithm" in the context of the application and the user's project. This trained model can then be used for making predictions on new data points, extending the application's functionality.



	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (\$
dt	Decision Tree Classifier	0.9857	0.989	0.9857	0.9902	0.9852	0.9783	0.9796	0.
rf	Random Forest Classifier	0.9857	0.9995	0.9857	0.9902	0.9852	0.9783	0.9796	0.
lr	Logistic Regression	0.9786	0.9975	0.9786	0.9866	0.979	0.9685	0.9705	0.
gbc	Gradient Boosting Classifier	0.9714	0.9952	0.9714	0.963	0.9649	0.9569	0.9593	0.
lightgbm	Light Gradient Boosting Machine	0.9714	0.9981	0.9714	0.9711	0.9682	0.9579	0.9614	0.
et	Extra Trees Classifier	0.9143	0.9925	0.9143	0.9202	0.9084	0.8783	0.8872	0.
ridge	Ridge Classifier	0.9	0	0.9	0.896	0.8887	0.8536	0.864	0.
lda	Linear Discriminant Analysis	0.8714	0.9904	0.8714	0.9254	0.874	0.8226	0.84	0.
ada	Ada Boost Classifier	0.8357	0.958	0.8357	0.7311	0.7738	0.7538	0.7694	0.
nb	Naive Bayes	0.7357	0.9113	0.7357	0.8198	0.7085	0.6584	0.7034	0.

Model training completed successfully!

fig 5.3

4.6 Download Model

You can download the trained model for later use once the selected model or models have been trained (fig 6). When you wish to generate predictions in real time based on fresh data points outside of the application's UI, this feature is very helpful. The trained model can be extended in practicality by integrating it into web applications or other software systems by downloading it (saved in a standard format).

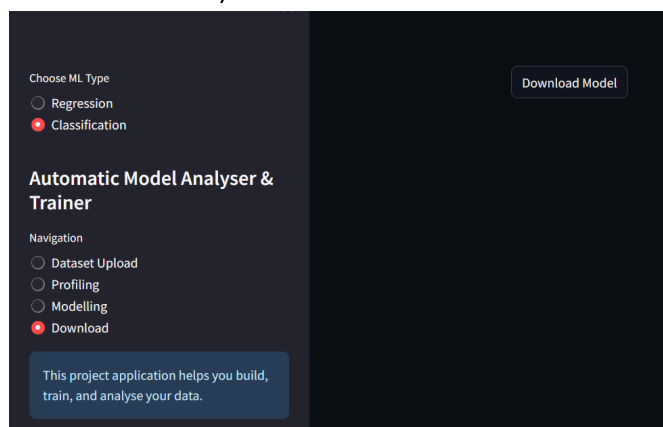


fig 6

5. Results

The Result could be viewed by loading the model in a jupyter notebook using 'load_model' function in pycaret module. It results in the pipeline on how the model has been trained and in addition to this the logs also give us the clear overview on how the model is trained (figs 7.1,7.2 and 7.3).

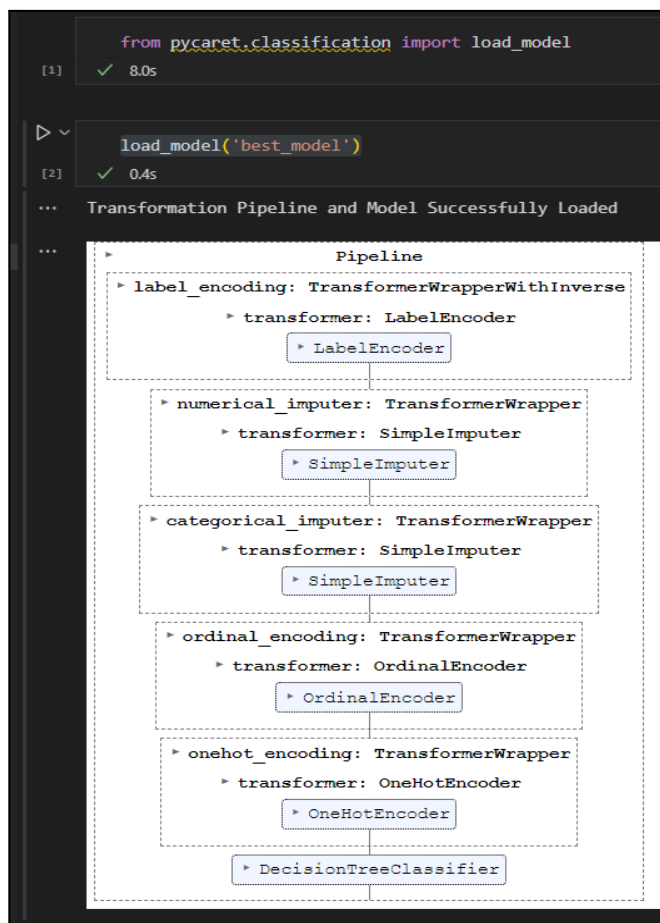


fig 7.1

fig 7.2

fig 7.3

6 Conclusion and Future Works

This paper presented a novel Streamlit application designed to democratize machine learning workflows. The program enables people with varying technical backgrounds to leverage machine learning for predictive modeling and data analysis through an intuitive, code-free interface. The program simplifies the whole machine learning process, including data upload, optional exploration, model training and comparison, and optional model download. By removing the technological obstacles that are frequently connected to conventional machine learning pipelines, our all-encompassing strategy promotes an atmosphere that is more welcoming to ML research and innovation. In the future, we hope to add support for image-based datasets, incorporate feature engineering features, provide choices for manual hyperparameter tweaking, and allow the real-time deployment of trained models as web services. By making machine learning more accessible and useful for a wider range of activities and skill levels, these developments hope to provide a more effective and adaptable tool.

References

- [1] [Aurélien Géron, "Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems," O'Reilly Media, Inc., 2017.](#)
- [2] [Trevor Hastie, Robert Tibshirani, and Jerome Friedman, "The Elements of Statistical Learning," Springer Series in Statistics New York, NY, USA, 2009.](#)
- [3] [Ian Goodfellow, Yoshua Bengio, and Aaron Courville, "Deep Learning," MIT Press, 2016.](#)
- [4] [F. Chollet, "Deep Learning with Python," Manning Publications Co., 2017. \(Deep Learning for image recognition\)](#)
- [5] [J. Brownlee, "Mastering Machine Learning with Python," Packt Publishing Ltd, 2017. \(Machine Learning for various tasks\)](#)
- [6] [R. J. Hyndman and G. A. Athanasopoulos, "Forecasting: principles and practice," OTexts, 2018. \(Machine Learning for time series forecasting\)](#)
- [7] [J. Friedman, T. Hastie, and R. Tibshirani, "The Elements of Statistical Learning," Springer Series in Statistics New York, NY, USA, 2013. \(Machine Learning for regression analysis\)](#)
- [8] [KNIME \(2024\) \[Online\].](#)
- [9] [RapidMiner \(2024\) \[Online\].](#)
- [10] [Google Colab \(2024\) \[Online\].](#)
- [11] [M. Kuhn and K. Johnson, "Applied Predictive Modeling," Springer International Publishing, 2013.](#)
- [12] [J. Brownlee, "Machine Learning Mastery" \[Online\].](#)
- [13] [Kaggle Learn \(2024\) \[Online\]](#)
- [14] [Hadley Wickham, "ggplot2: Elegant Graphics for Data Analysis," Springer-Verlag New York, 2016. \(Data Visualization\)](#)
- [15] [Lars Madsen, Thomas P. Jensen, and Lars Kai Hansen, "Multivariate Analysis: A User's Guide," Chapman and Hall/CRC, 2014. \(Statistical Analysis\)](#)