TRAINLAB

(Python-based Automatic Model Trainer and Analyzer)

A MINI-PROJECT REPORT

Submitted by

SIVASHANKAR S (210701252)

SAI PRASAD R (210701220)

in partial fulfillment of the award of the
degree of
BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

RAJALAKSHMI ENGINEERING COLLEGE,
THANDALAM MAY - 2024

BONAFIDE CERTIFICATE

Certified that this project **"HIGH-RES"** is the bonafide work of **"SIVASHANKAR S 210701252"** and **"SAI PRASAD R 210710220" who** carried out the project work under my supervision.

SIGNATURE

Dr. M. RAKESH KUMAR, M.E., Ph.D.,

Assistant Professor,

Computer Science & Engineering Rajalakshmi Engineering College (Autonomous)
Thandalam, Chennai-602105

This mini project report is submitted for the viva voce examination to be held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honourable chairman **MR. S. MEGANATHAN** and the chairperson **DR. M. THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honourable principal **Dr. S. N. MURUGESAN** for his able support and guidance

No words of gratitude will suffice for the unquestioning support extended to us by our Head of The Department **Dr. P. KUMAR M.E Ph.D.,** and our Academic Head **Dr. N. DURAIMURUGAN,** for being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide **Dr. M. RAKESH KUMAR, M.E Ph.D.,** for his valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of computer science engineering.

Sivashankar S (210701252)

Sai Prasad R (210701220)

TABLE OF CONTENTS

LIST OF FIGURES

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

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

This paper proposes a novel Streamlit application that bridges the gap between complex machine learning (ML) workflows and everyday users. The current ML landscape is often dominated by intricate processes requiring significant technical expertise. This poses a significant barrier for individuals who lack a strong coding background or specialized knowledge in data manipulation, model selection, training, and evaluation. Even experienced users can get overwhelmed by the vast array of algorithms and hyperparameters available, making it challenging to choose the optimal model for a specific task.

1.2 SCOPE OF THE WORK

This This Streamlit application simplifies the entire machine learning process for users. It offers Easy data exploration with data upload, profiling, and visualization to understand the data better, Effortless model training using pre-built algorithms from PyCaret, reducing coding and allowing users to focus on data analysis and model selection and Deployment-ready functionalities beyond training, including downloading models, visualizing results, and integrating models for real-world applications.

1.3 PROBLEM STATEMENT

The standard machine learning workflow is often a complex and knowledge-intensive process. This complexity discourages individuals who lack the necessary technical skills or coding experience from leveraging the power of machine learning for their projects. Even seasoned users can struggle with the sheer volume of algorithms and hyperparameters available, making it challenging to navigate the ML landscape and choose the optimal model for a specific task.

1.4 AIM AND OBJECTIVES OF THE PROJECT

Aim: This Streamlit application aims to tackle the challenges associated with the traditional machine learning workflow and make it more accessible to a wider audience. By offering an intuitive and interactive platform, and empowering users with varying technical backgrounds to leverage the power of machine learning for their projects.

Objectives: Simplifying the Machine Learning Journey. The application achieves its aim through the following key objectives:

1. Simplified Data Exploration: Streamlined data upload and integrated data profiling features empower users to gain a deeper understanding of their data.

2. Effortless Model Training: PyCaret library integration allows users to select and train models without extensive coding knowledge.

3. Enhanced Model Deployment: Users can download trained models, visualize results, and integrate models into other systems for real-world applications.

CHAPTER 2

LITERATURE SURVEY

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.

CHAPTER 3

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Processor	:	Intel i5
Memory Size	:	8GB (Minimum)
SSD	:	512 GB (Minimum)

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS 10/11
Front – End	:	Streamlit
Programming Language	:	Python

CHAPTER 4

MODULE DESCRIPTION

This is a web app built using Streamlit for easy interaction with the data and training the machine learning model. It includes several modules that employ specific components to enrich the existing functionality. They likewise permit clean loading of a dataset into use and allow straightforward viewing of data. Lewis use of PyCaret is recommended for its simplicity; it comes with preconfigured algorithms for classification and regression models that make the process of training and evaluating models easier. For model selection and evaluation, the application also has a feature that allows users to make comparisons on the performance of the models in an attempt to select the most effective ones. Lastly, models can be trained and downloaded for use in other areas of the application, adding more value to the application beyond the graphical front-end. When it comes to data sharing, the solution allows for seamless integration and establishes a clear pipeline progression from data uploading to model deployment.

3.1. Data Upload Module

The data upload module enables the users to upload their datasets into the application in CSV format using the pandas library for handling the data considerations. The users can easily choose and upload their respective CSV files for ease of use, and then the data can be made easily retrievable for use in the other subsequent analysis stages. This module also entails preparing the data and passing it into the application and it is one of the most significant steps in understanding machine learning. Generally, it helps to eliminate the possibility of users' frustration while dealing with files, which are necessary for data analysis and model training, but provide more difficulties than benefits in terms of importing data.

3.2. Data Preview Module

The second module is the data preview module that offers an insight into the given database's structure and contents and employs pandas in processing the data. This module involves the display of a part of the dataset to one or more lines involving rows or columns so as to enable the user verify the data type and check for any aberrant data or error. Preview play a vital role in examining the basic content of the data to be included before further processing is done. It guarantees simplicity and usefulness for its users by providing a brief overview of the importance of data cleaning as well as possible preprocessing stages.

3.3. Data Profiling Module

It provides a comprehensive overview of the given dataset through the use of pandas_profiling library and thus is called data profiling module. This module provides data overview which includes data types, missing values, distribution patterns, correlation data, and measures of central tendencies and significant deviations. Such insights are very useful for testing hypotheses or examining substructure which might be present in the data. It is a profiling report which assists the users to decide about the right and suitable data cleaning procedures and preprocessing so that an appropriate dataset for the model training can be prepared. These two provide a detailed analysis of the data which is crucial especially when it comes to developing proper models for machine learning.

3.4. Machine Learning Models Module

The machine learning models module implemented in the PyCaret environment, which reduces the time and effort of selecting and training models. It enshrines numerous pre-existing algorithms for distinct classification and regression problems to help users in identifying the ideal model when carrying out a particular analysis on a variable. This module helps in data preprocessing in the form of data division and this module also helps in tuning the hyperparameters and training the models. It provides an end-to-end training phase that generates a performance comparison table, where you have measures such as accuracy, precision, recall, and the F1-score. This makes it easier for the users to decide on the best model to deploy by comparing the results of the run, since this will help one identify the model that fits their needs.

3.5. Performance Evaluation Modules

The performance evaluation module uses PyCaret to compare and determine the most suitable machine learning models. It appears that after training, it enlists and provides a clear and comprehensive table containing models that show various performance-related parameters like accuracy, precision, recall, and F1-score. This module allows users to understand and select the specific model as per their requirements it is specially helpful to choose accuracy or precision as per needs. Through presenting specific analyses about the utilized model’s and profited metrics, this module enables users make appropriate decision on which of the models to implement .

3.6. Model Download Module

The implement of model download module enables user to download their trained ML model for reuse in the future. As a final step in model selection, the user can save the final best model in a common file format for use in another program, possibly in pickle file format. This functionality is particularly beneficial when applying the model in real-time environments beyond the Streamlit application. As we elaborate in this module, one strength of the approach presented here is the ability of the trained models to be ported and reused on different production systems.

CHAPTER 5

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

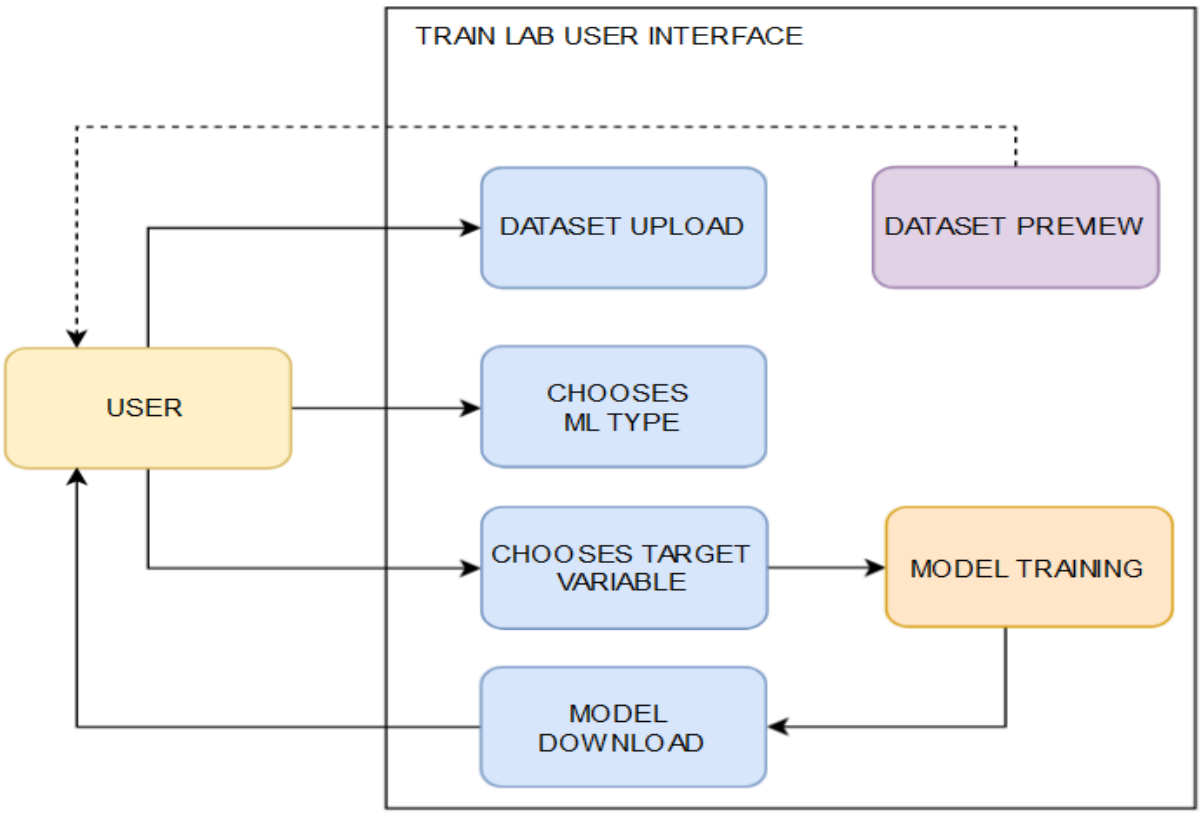


FIG.4.1 SYSTEM ARCHITECTURE DIAGRAM

User initially input their data through the “Dataset Upload” component, which lets users analyse the data set using the “Dataset Preview” tool. They then decide the type of machine learning to be used, whether regression or classification and determine the variable to be predicted/forecasted. The system employs this information for the sake of starting the “Model Training” process, as well as utilizing PyCaret for the purpose of model training and model evaluation. To organize the trained model’s deployment, the “Model Download” module also became available for the users. They have implemented an architecture that can effectively

facilitate the entire flow from uploading data, cleaning, feature engineering, feature selection, training of the model and exporting into application.

CHAPTER 6

SAMPLE CODING

test.py

```
#Importing Modules
import streamlit as st
from streamlit_pandas_profiling import st_profile_report
import pandas as pd
import os
import pandas_profiling
from pycaret.regression import setup as reg_setup, compare_models, pull, save_model
from pycaret.classification import setup as cls_setup
from sklearn.feature_extraction.text import ENGLISH_STOP_WORDS

ml_type = st.sidebar.radio("Choose ML Type", ["Regression", "Classification"])

if ml_type == "Regression":
 from pycaret.regression import setup, compare_models, pull, save_model
 setup_function = reg_setup
else:
 from pycaret.classification import setup, compare_models, pull, save_model
 setup_function = cls_setup
```

```
# Load or create an empty DataFrame
if os.path.exists('./dataset.csv'):
 df = pd.read_csv('dataset.csv', index_col=None)
 df = df.dropna()
else:
 df = pd.DataFrame()

with st.sidebar:
 st.title("Automatic Model Analyser & Trainer")
 choice = st.radio("Navigation", ["Dataset Upload", "Profiling", "Modelling", "Download"])
 st.info("This project application helps you build, train, and analyse your data.")

# Function for text data preprocessing
def preprocess_text(text):
 text = text.lower()

 # Remove stopwords
 text = ' '.join([word for word in text.split() if word not in ENGLISH_STOP_WORDS])

 return text
if choice == "Dataset Upload":
 st.title("Upload Your Dataset")
 file_csv = st.file_uploader("Upload CSV File", type=["csv"])
```

```
if file_csv:
        df = pd.read_csv(file_csv, index_col=None)

        # Data Preprocessing based on data types
        if "text" in df.select_dtypes(include=["object"]).columns:
        # Apply text data preprocessing steps
        df["text"] = df["text"].apply(lambda x: preprocess_text(x))

        # Save the preprocessed dataset
        df.to_csv('dataset.csv', index=None)
        st.success("CSV Dataset uploaded and preprocessed successfully!")
        st.dataframe(df)

if choice == "Profiling":
 st.title("Exploratory Data Analysis")
 if not df.empty:
        profile = pandas_profiling.ProfileReport(df, title="Profiling Report")
        st.subheader("Visualising various plots:")
        st_profile_report(profile)
 else:
        st.warning("Please upload a dataset before proceeding to profiling.")

if choice == "Modelling":
```

```
if not df.empty:
        st.title("Model Training")
        chosen_target = st.selectbox('Choose the Target Column', df.columns)

        if st.button('Run Modelling'):
        setup(data=df)
        setup_df = pull()
        st.subheader("Setup Information:")
        st.dataframe(setup_df)

        st.subheader("Comparing Models:")
        best_model = compare_models()
        compare_df = pull()
        st.dataframe(compare_df)

        save_model(best_model, 'best_model')
        st.success("Model training completed successfully!")
else:
        st.warning("Please upload a dataset before running the model.")

if choice == "Download":
 if os.path.exists('best_model.pkl'):
        with open('best_model.pkl', 'rb') as f:
        st.download_button('Download Model', f, file_name="best_model.pkl")
```

```
else:
        st.warning("No trained model available for download. Please run the model training
first.")
```

CHAPTER 7
SCREENSHOTS

7.1 OUTPUTS

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.

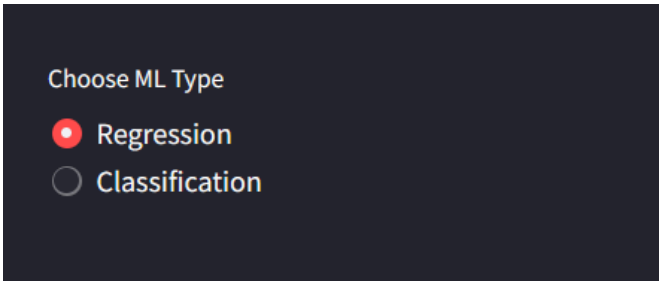


fig 1

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.

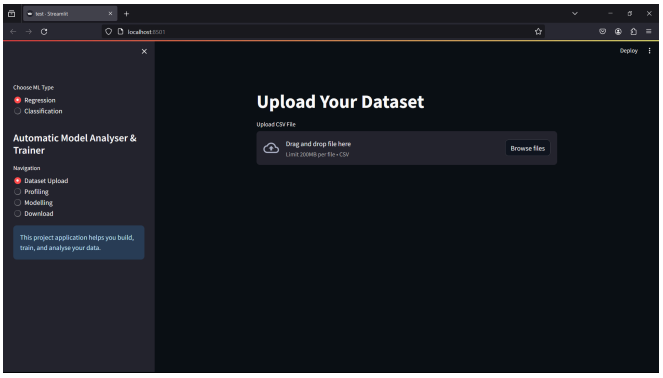
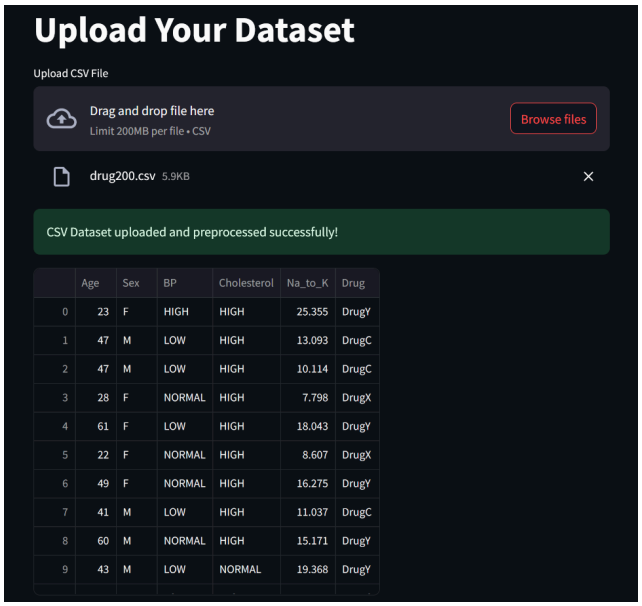


fig 2

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.



	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	DrugY
1	47	M	LOW	HIGH	13.093	DrugC
2	47	M	LOW	HIGH	10.114	DrugC
3	28	F	NORMAL	HIGH	7.798	DrugX
4	61	F	LOW	HIGH	18.043	DrugY
5	22	F	NORMAL	HIGH	8.607	DrugX
6	49	F	NORMAL	HIGH	16.275	DrugY
7	41	M	LOW	HIGH	11.037	DrugC
8	60	M	NORMAL	HIGH	15.171	DrugY
9	43	M	LOW	NORMAL	19.368	DrugY

fig 3

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.

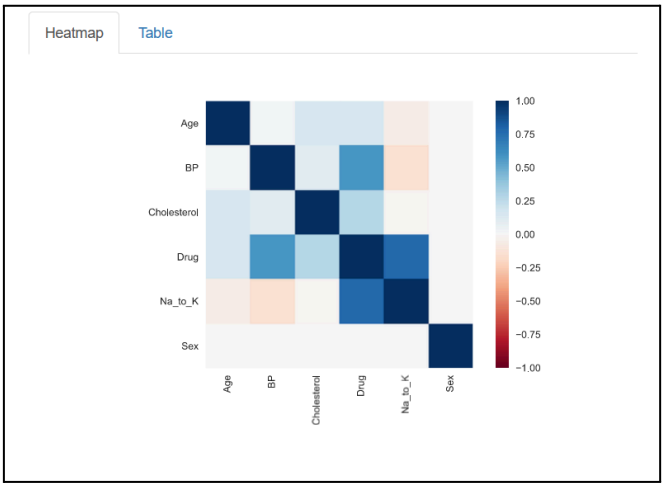


fig 4.1

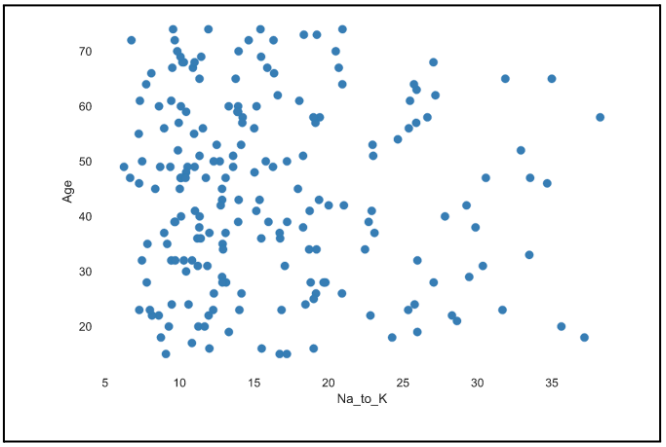

70
60
50
40
30
20
Age
5
10
15
20
25
30
35
Na_to_K

fig 4.2

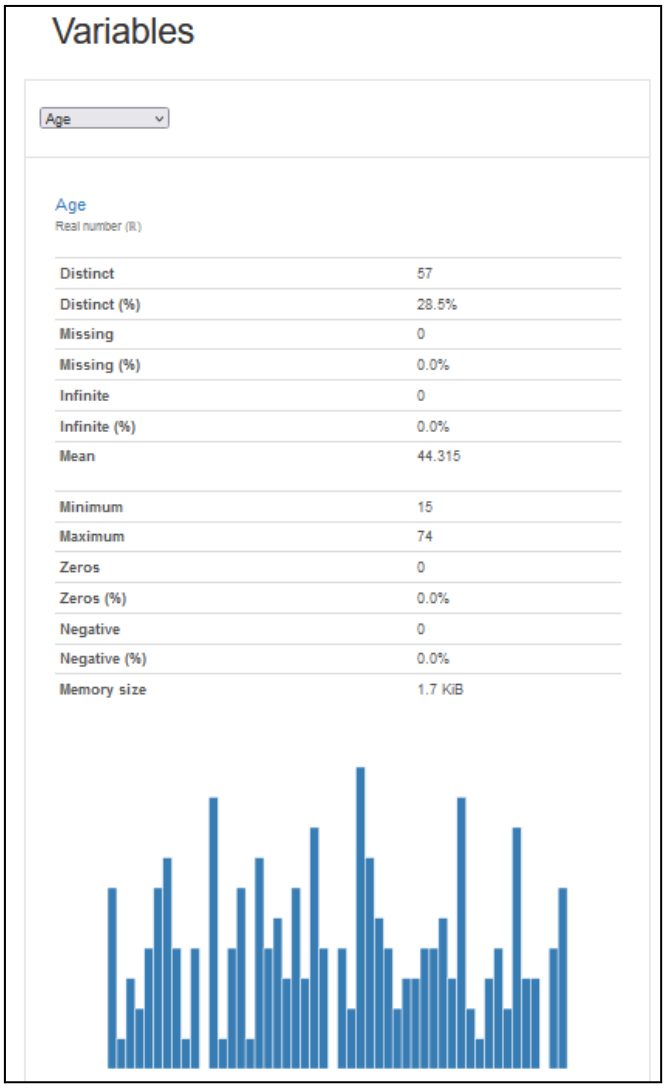

Variables
Age
Age
Real number (ℝ)
Distinct 57
Distinct (%) 28.5%
Missing 0
Missing (%) 0.0%
Infinite 0
Infinite (%) 0.0%
Mean 44.315
Minimum 15
Maximum 74
Zeros 0
Zeros (%) 0.0%
Negative 0
Negative (%) 0.0%
Memory size 1.7 KiB

fig 4.3

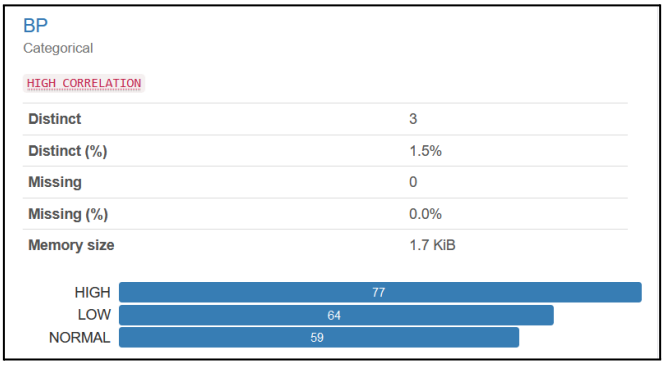


fig 4.4

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.

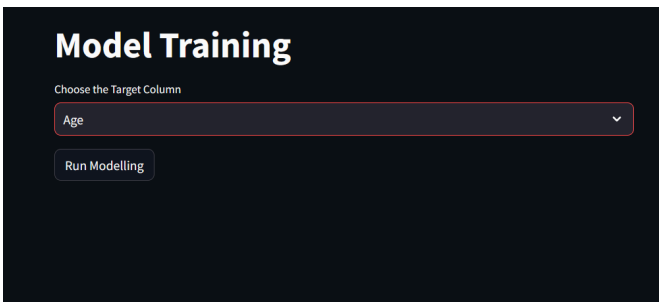
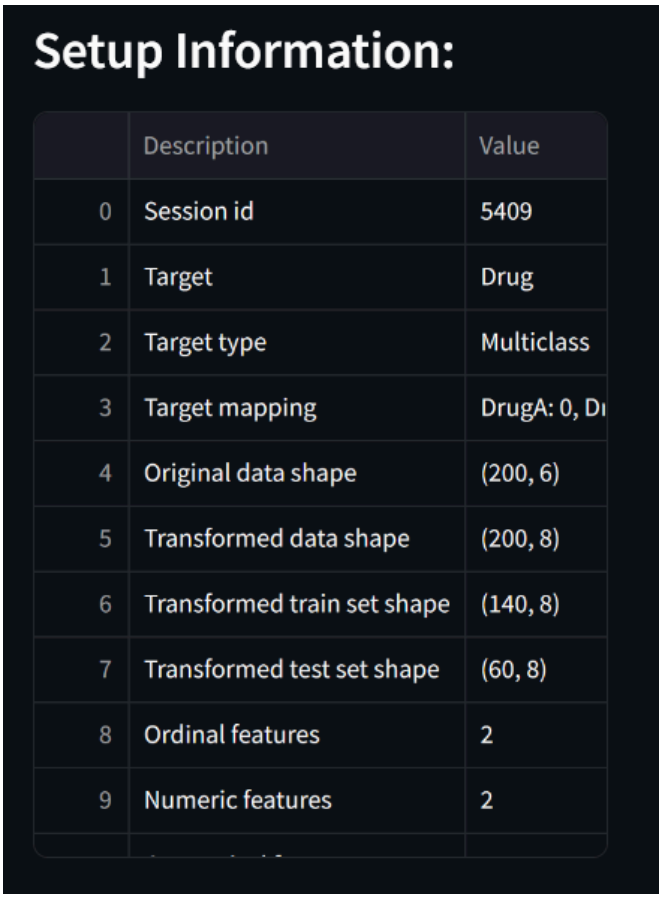


fig 5.1

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.

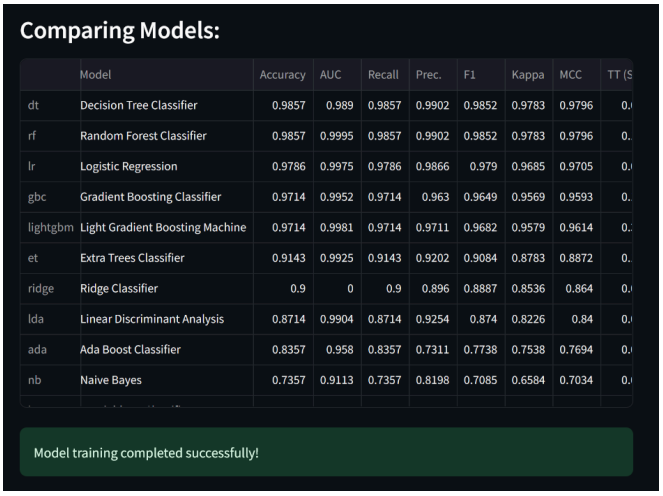


Setup Information:

	Description	Value
0	Session id	5409
1	Target	Drug
2	Target type	Multiclass
3	Target mapping	DrugA: 0, Dı
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.



Comparing Models:

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (S
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

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).

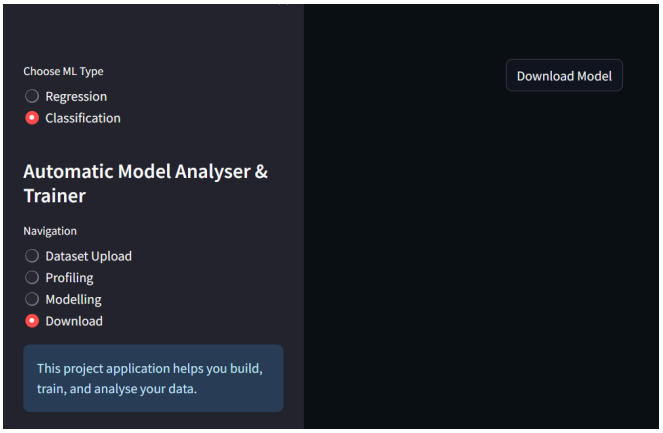


fig 6

7.2 RESULT

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).

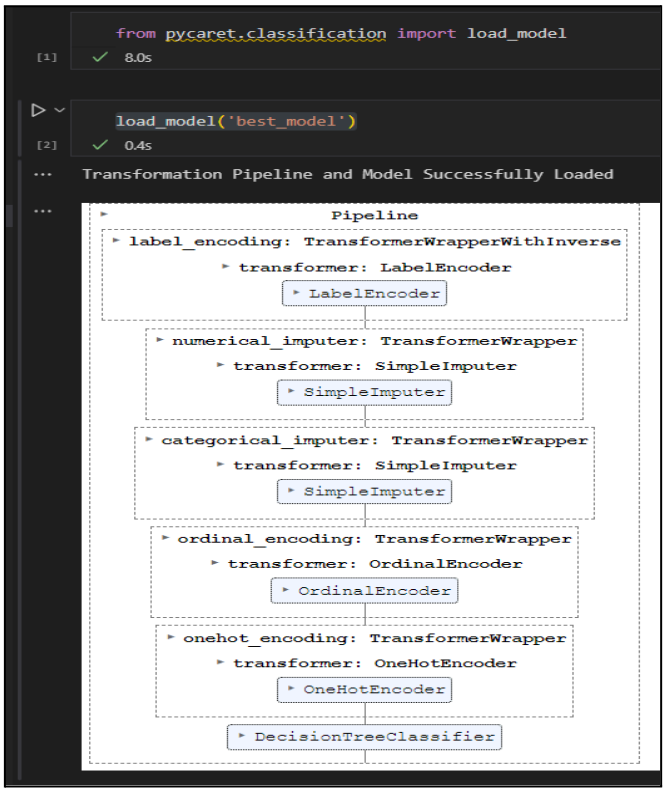


fig 7.1

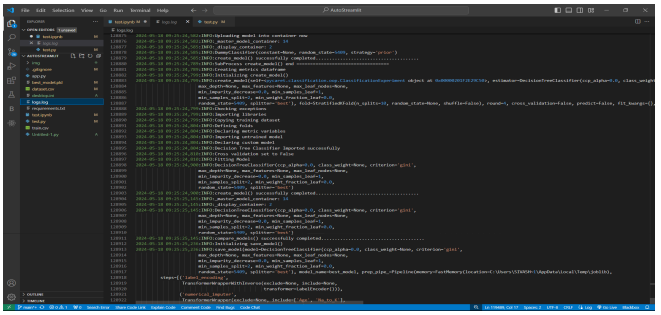


fig 7.2

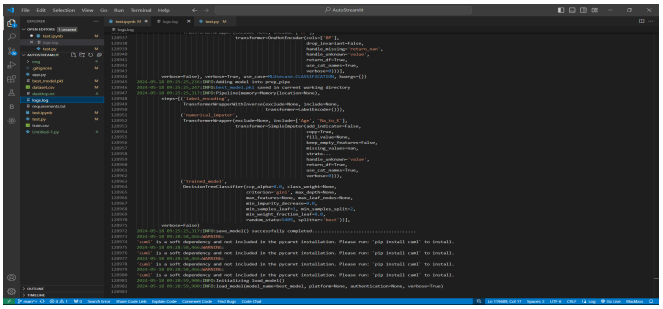


fig 7.3

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION

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.

FUTURE ENHANCEMENTS

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)