CS-GY 9963-Advanced Research Project Report

No-Free Lunch Web Application

By N.V.M.Krishna Chaitanya Kotcherlakota

Guided By Prof. Raman Kannan

Introduction:

The No-Free Lunch theorem in Machine Learning states that " No Particular Classifier can outperform all the other classifiers for every dataset", This Project helps in aiding the process of making the lunch free for anyone with a dataset. This project aims to reduce the development time and enhance the productivity of the user, also this project facilitates No-Code Machine Learning i.e allowing the end user to model their data with no need for a single line of Code.

Project Requirements:

- Python 3.6 or 3.7
- Scikit-Learn
- Seaborn
- Matplotlib
- Flask
- HTML & CSS
- Java Script
- BootStrap

Installation:

- pip install -r requirements.txt
- python run app.py
- Open http://localhost:9000

Steps Of Execution:

1. Splitting the data
2. Selecting the dependent and independent variables
3. Training and Testing set
4. Evaluate or Review the EDA performed -- heatmap, correlation graphs of the whole dataset
5. Specify the Learning Model
6. Training
7. Review the performance metrics of each classifier selected

Implementation:

Variance and Bias:

Variance means the variety of predicted values made by a machine learning model (target function). Bias means the distance of the predictions from the actual (true) target values. A high-biased model means its prediction values (average) are far from the actual values. Also, high-variance prediction means the prediction values are highly varied.

Training:

Occam's Razor as an Inductive Bias :
Inductive bias is the assumption that is being made to the learning algorithm to create a hypothesis beyond the training data in order to be able to classify the unseen data. Occam's razor involves a preference for a simpler hypothesis that best fits the data. Following this hypotheses, the models are created with a default classifier parameters,

performance on the training set :

The models are build and are trained on the training data, once the models are the model are trained we are calculating the proportion of the variance in the dependent variable that is predictable from the independent variable(s) this is often referred as r2 value, this servers as a value that we could be using to determine if the model is overfitting on the training dataset.

Assessing learnability:
It is important to asses the learnability of the model before running it on the test dataset, we have used a heart dataset for evaluating the learnability through the application, in this we see decision tree is shown as overfitting as we see the training accuracy is 1 which means the model is failing to generalize the behavior of data, there are several reasons for overfitting,

- ★ The training data contains large amounts of irrelevant information, called noisy data.
- ★ The training data size is too small and does not contain enough data samples to accurately represent all possible input data values.
- ★ The model trains for too long on a single sample set of data.
- ★ The model complexity is high, so it learns the noise within the training data

Testing:

performance on the testing set :

once the models are the model are trained we are calculating the proportion of the variance in the dependent variable that is predictable from the independent variable(s) this is often referred as r2 value, across the testing dataset, this servers as a value that we could be using to determine if the model is underfitting on the testing dataset.

Assessing Generalizability:

generalization is a definition to demonstrate how well a trained model can classify or forecast unseen data. Training a generalized machine learning model means, in general, it works for all subsets of unseen data. An example is when we train a model to classify between dogs and cats. If the model is provided with a dog image dataset with only two breeds, it may obtain a good performance. But, it possibly gets a low classification score when it is tested by other breeds of dogs as well. This issue can result in classifying an actual dog image as a cat from the unseen dataset. Therefore, data diversity is a very important factor in order to make a good prediction.

Variance-bias trade-off

The prediction results of a machine learning model stand somewhere between a) low-bias, low-variance, b) low-bias, high-variance c) high-bias, low-variance, and d) high-bias, high-variance. A low-biased, high-variance model is called overfit and a high-biased, low-variance model is called underfit. By generalization, we find the best trade-off between underfitting and overfitting so that a trained model obtains the best performance. An overfit model obtains a high prediction score on seen data and low one from unseen datasets. An underfit model has low performance in both seen and unseen datasets.

Performance Metrics Used In This Project

1) Accuracy- The percentage of labels that our model successfully predicted is represented by accuracy, the accuracy function from Sklearn library accepts values for the classification model's predicted labels and true labels of the sample as its arguments and computes the accuracy score. Here, we iterate through each pair of true and predicted labels in parallel to record the number of correct predictions. We then divide that number by the total number of labels to compute the accuracy score.
2) TN,TP,FP,FN -“TN” stands for True Negative which shows the number of negative examples classified accurately. Similarly, “TP” stands for True Positive which indicates the number of positive examples classified accurately. The term “FP” shows False Positive value, i.e., the number of actual negative examples classified as positive; and “FN” means a False Negative value which is the number of actual positive examples classified as negative. One of the most commonly used metrics while performing classification is accuracy.

3) An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters
    a) True Positive Rate
    b) False Positive Rate

Comparative Analysis of models

Tabulating the metrics:

The well-trained classifiers are subjected to the various above mentioned performance metrics, and we have calculated the accuracy scores of the classifier's on both the testing and training data and tabulated these values in the below table which is currently being generated at the end of the web application, and this table dynamically updates the selection of the dependent variable and independent variables.

Results

Results:

	Models	Test Accuracy	Train Accuracy	AUC	Model Behaviour
0	logistic	0.710526	0.612335	0.704530	Acceptable
1	decision tree	0.539474	0.969163	0.539721	Over fitting
2	naive bayes	0.631579	0.603524	0.633449	Acceptable
3	random forest	0.684211	0.687225	0.671777	Acceptable

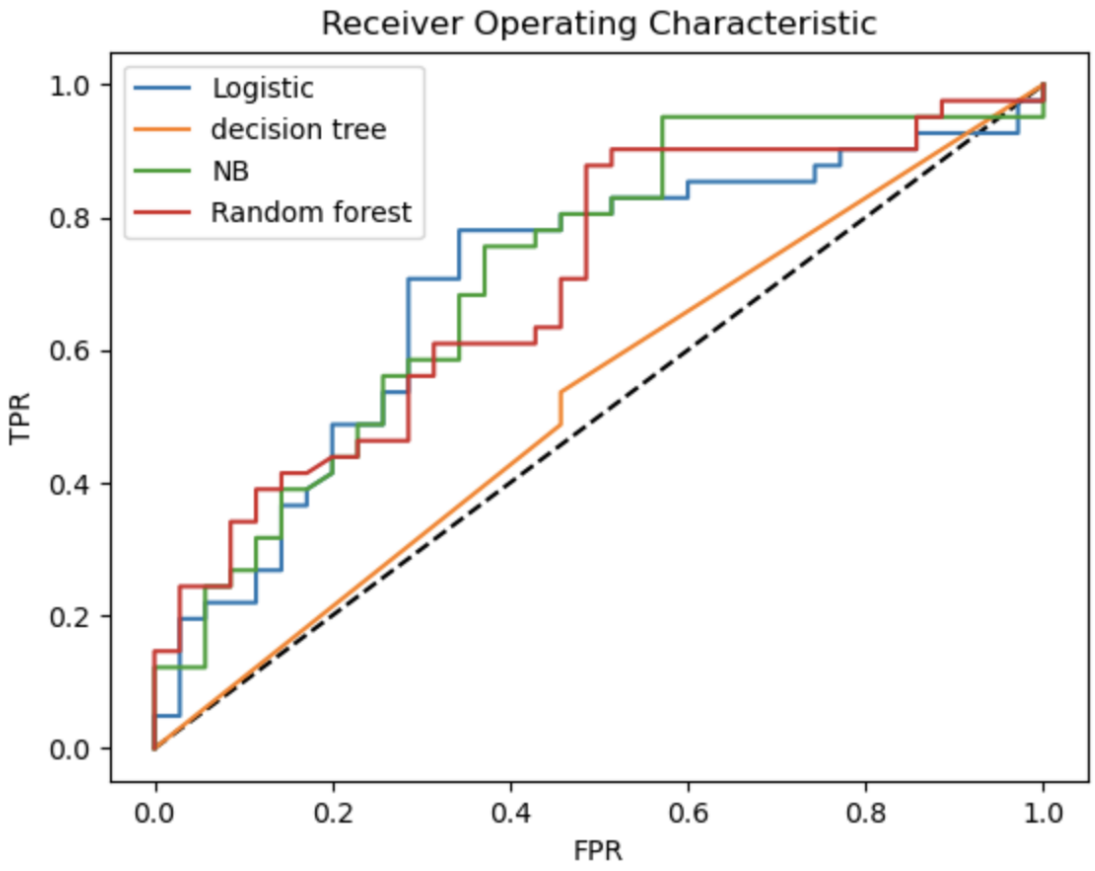
The best performing model is : logistic

The Model with best AUC score is : logistic

RoC Curves:

The ROC curve with individual classifiers performance is built along with the accuracy score's. This table is updated dynamically with the change in the selection of the dependent and independent variables, the below graph shows the table generated for a sample dataset.

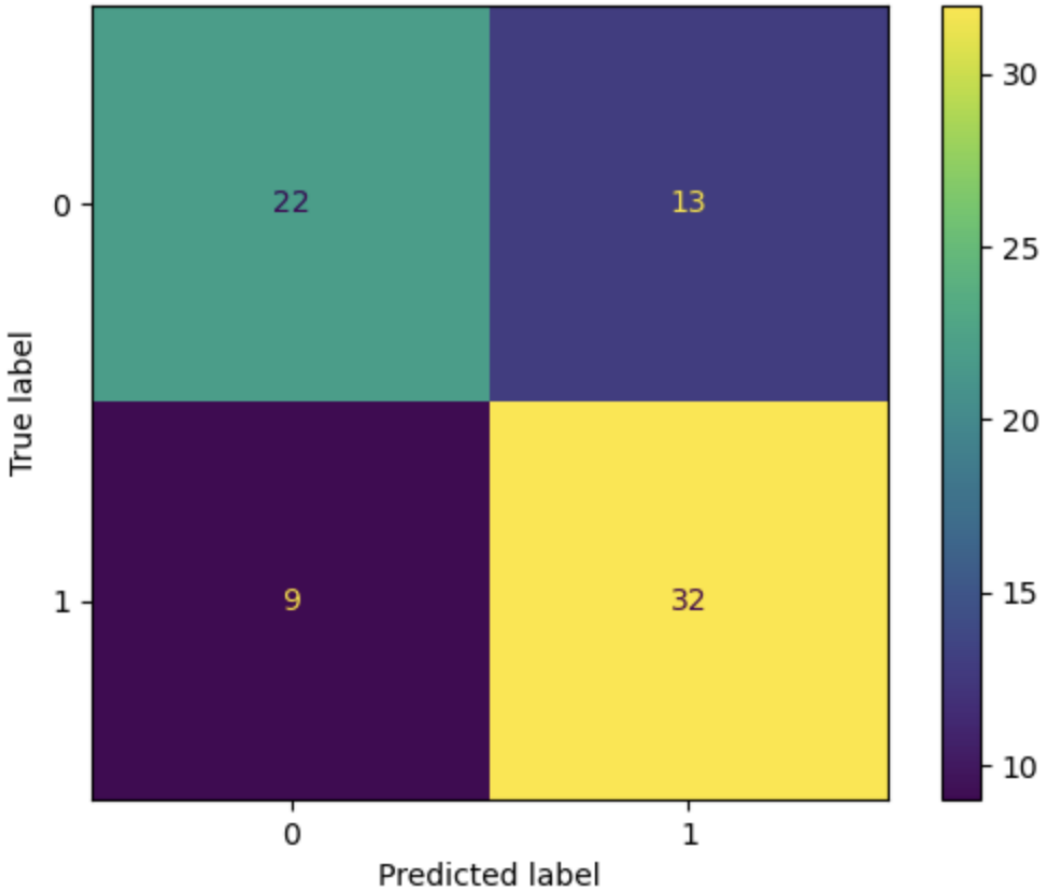
ROC



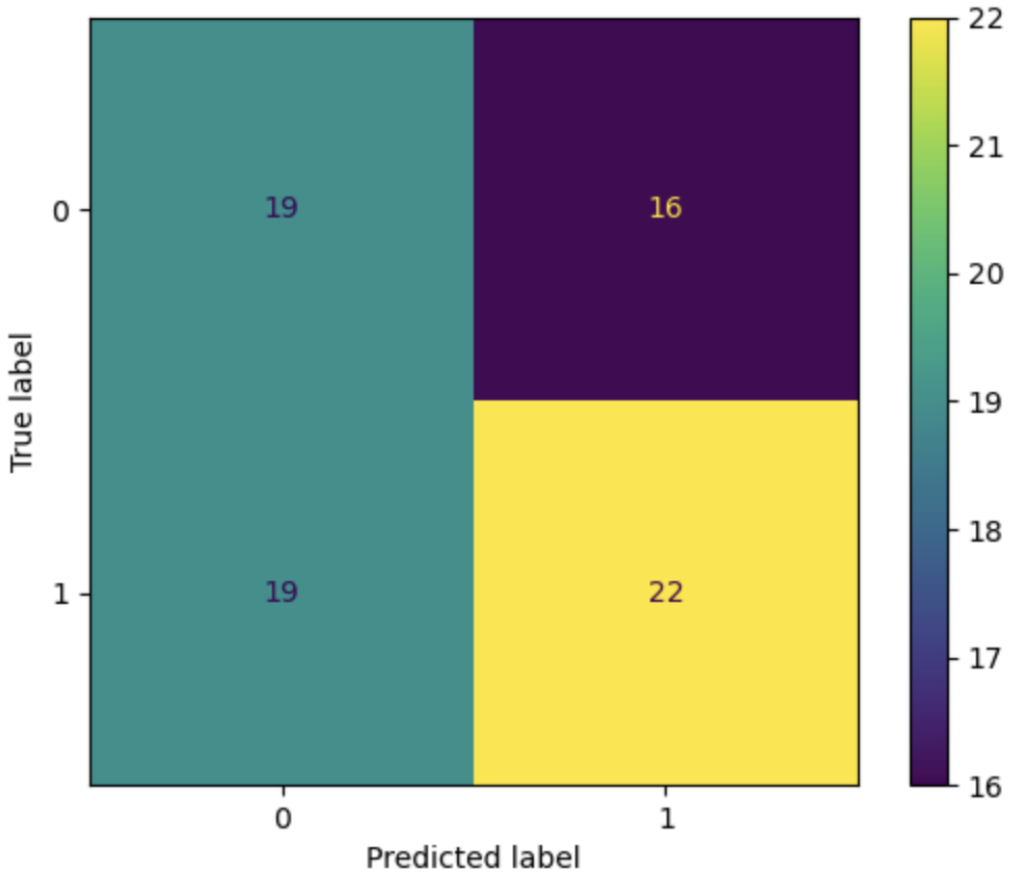
Confusion matrix:

Confusion matrix is a very popular measure used while solving classification problems. It can be applied to binary classification as well as for multiclass classification problems, the below image represents the confusion matrix developed using the tool on the Tests performed over a sample dataset.

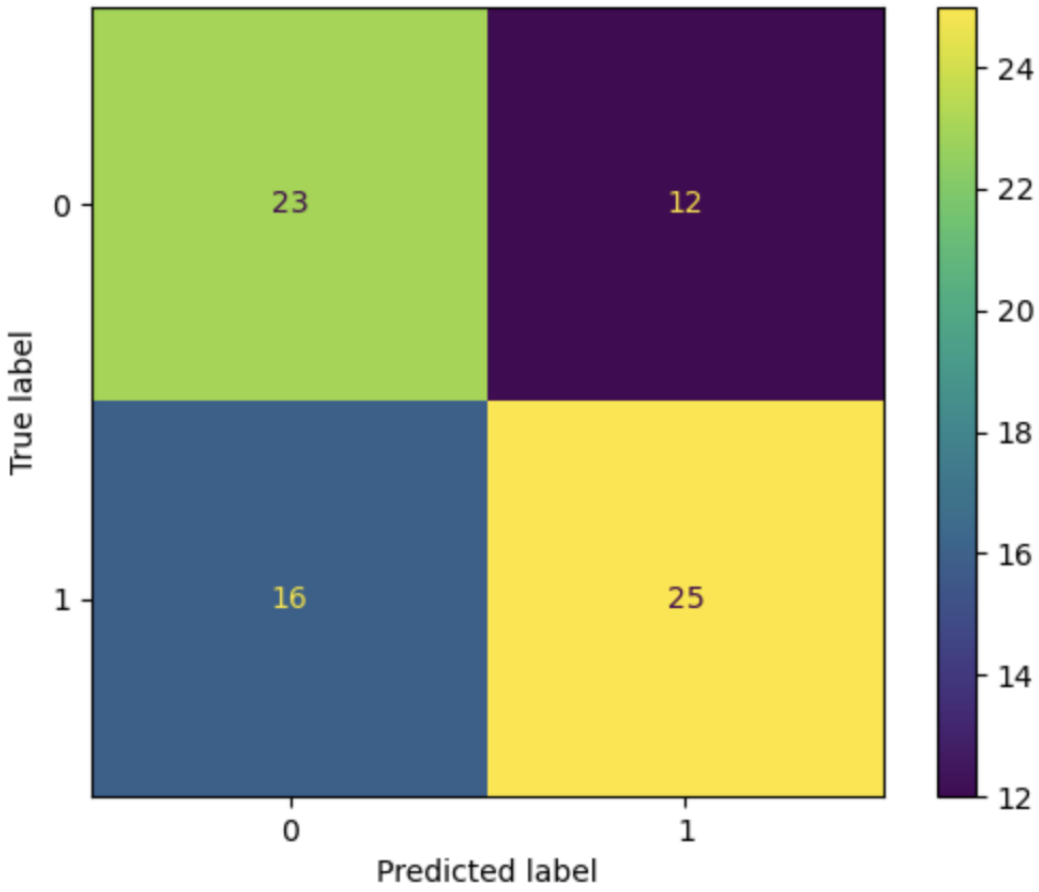
Logistic Regresion



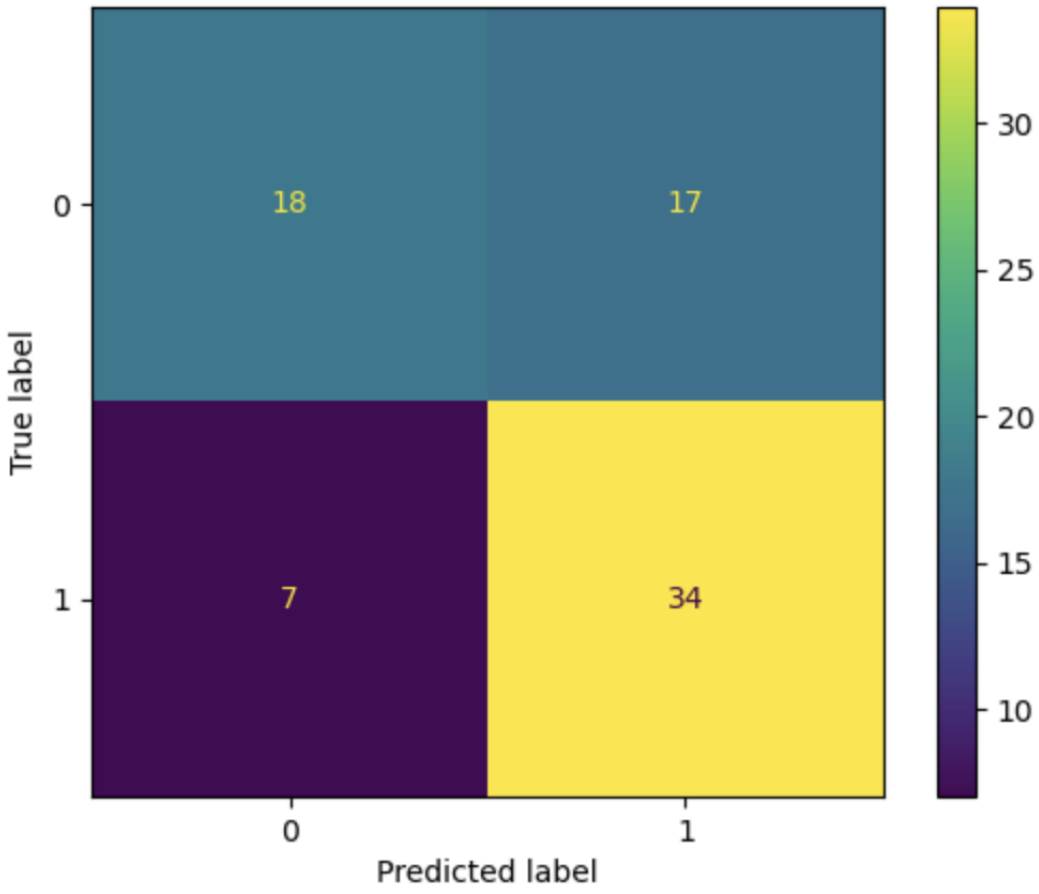
Decision Tree



Naive Bayes



Random Forest



Experimenting In Real Time

Promise Of No-Code ML:

This project aims to reduce the development time and enhance the productivity of the user, also this project facilitates No-Code Machine Learning i.e allowing the end user to model their data with no need for a single line of Code. The Web application is powered by Flask, a micro-web Framework, that runs the server at port 9000, facilitating the interaction between the web page and the back-end code built on Python. The entire application is built on a single page, with an idea of not redirecting the user to different pages. Initially the user uploads the dataset and the javascript handles the input file and stores it in the UPLOADS folder. After the dataset is uploaded the program redirects to app.route(‘/’) in the flask code,

where the data is scanned for null and 0 values across all the columns in the dataset. And the top 10 rows will be displayed on the left side of the website.

The next segment allows the user to select the columns and target variable across all the dataset columns and the next step would be displaying the pairplot and the heatmap using seaborn library. Based on the correlation values from the heatmap, we can reselect the columns and target variables and reselect the model parameters and run across the classifiers, once the classifiers generate the output, it is sent to the web page and we see a ROC plot along with a table of results showing the individual classifier performance. The present implementation has the following classifies built in the system,

- Naive Bayes Classifier
- Logistic Regression
- Decision Tree Classifier
- Random Forest Classifier

Time Compression achieved with the tool:

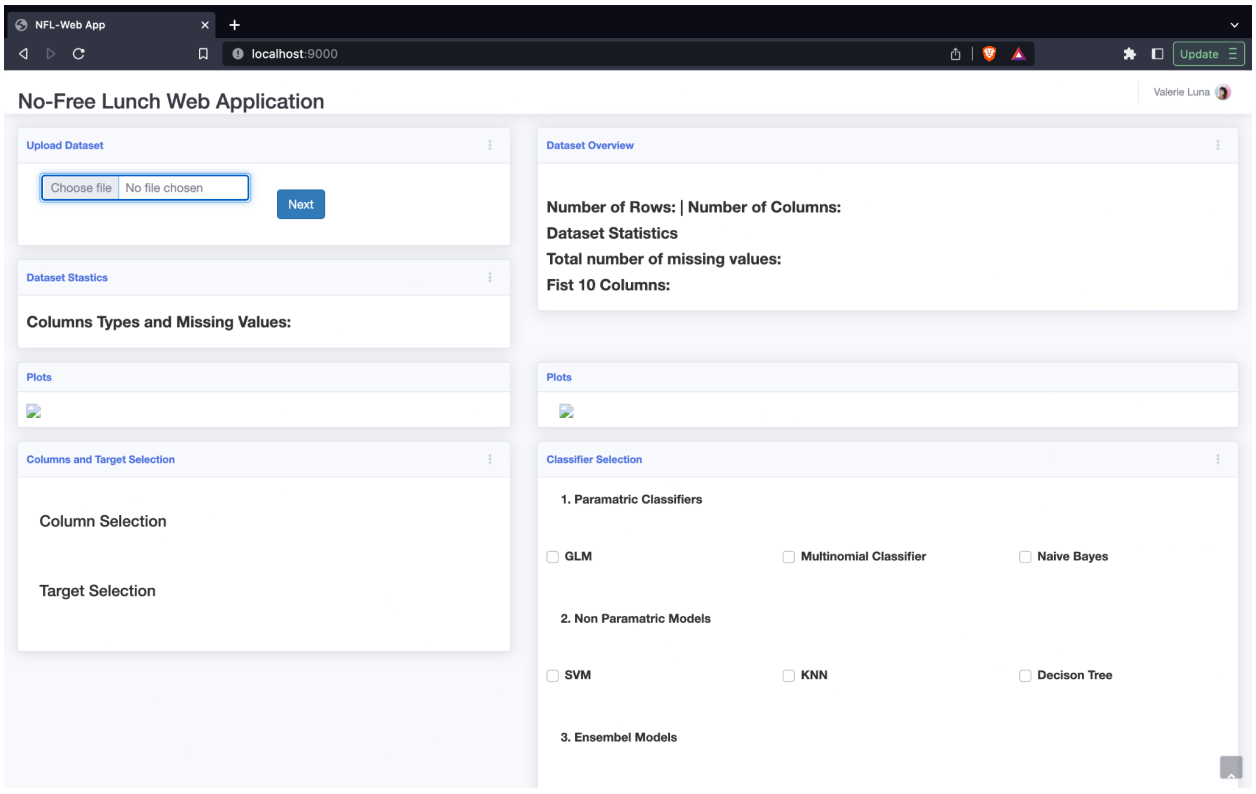
We have performed and timed two different experiments on the sample dataset chosen, The below table shows the time taken to perform each set of tasks, the time displayed is calculated based on the change in the target variables or changing the dependent variable assuming there is a code that's already built and changing the parameter and re-running the classifier's took around 1 min 30 sec and 1 min 15 sec and using this tool it took 10 sec and 15 sec respectively excluding the execution time in both the cases, the below table explains the time taken using the tool and without using the tool.

Experiment-No.	Without Using The Tool	Using This Tool
1.	1 min 10 sec	10 sec
2.	1 min 35 sec	11 sec

Conclusion:

The tool has reduced the development time and increased the robustness of the model development with an absolute use of No-Code, This is scalable to a larger version of No-Code ML platform's with significant options in customization made to the user's requirements. The present version of the tool work's on single button click and is built in the simplest way allowing the user to focus more on the output and thus reducing the issues of bug's or errors developed during manual code development process. On further development this tool would be made available online deployed on a cloud and making it available to the end-user through a web page, thus reducing the infrastructure needed by the machine learning practitioners..

Output:



NFL-Web App

localhost:9000

Update

Choose file | No file chosen

File heart.csv successfully uploaded.

Next

Number of Rows: 303 | Number of Columns: 14

Dataset Statistics

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exan
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000

Total number of missing values: 0

Fist 10 Columns:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
7	44	1	1	120	263	0	1	173	0	0.0	2	0	3	1

Dataset Stastics

Columns Types and Missing Values:

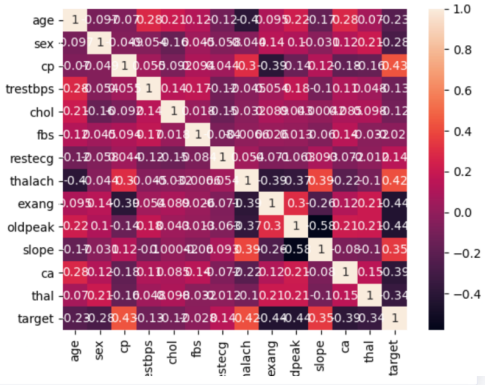
	Columns	Missing Values	Data Type
0	age	0	int64
1	sex	0	int64
2	cp	0	int64
3	trestbps	0	int64
4	chol	0	int64
5	fbs	0	int64
6	restecg	0	int64
7	thalach	0	int64
8	exang	0	int64
9	oldpeak	0	float64
10	slope	0	int64
11	ca	0	int64
12	thal	0	int64
13	target	0	int64

NFL-Web App

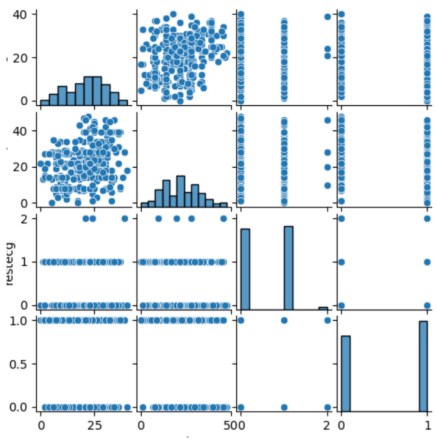
localhost:9000/preprocess

Update

Plots



Plots



Columns and Target Selection

Column Selection

- ☐ age
- ☐ sex
- ☐ cp
- ☐ trestbps
- ☐ chol
- ☐ fbs
- ☐ restecg
- ☐ thalach
- ☐ exang
- ☐ oldpeak
- ☐ slope
- ☐ ca
- ☐ thal
- ☐ target

Classifier Selection

1. Paramatric Classifiers

☐ GLM ☐ Multinomial Classifier ☐ Naive Bayes

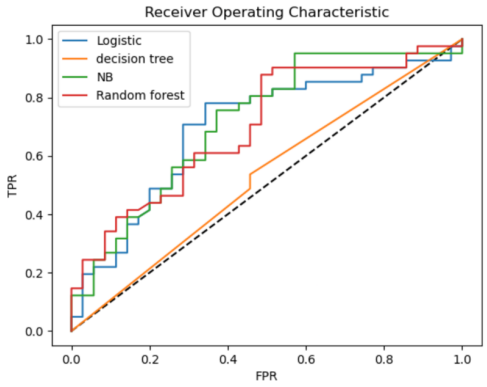
2. Non Paramatric Models

NFL-Web App

localhost:9000/preprocess

Update

ROC



Results

Results:

	Models	Test Accuracy	Train Accuracy	AUC	Model Behaviour
0	logistic	0.710526	0.612335	0.704530	Acceptable
1	decision tree	0.539474	0.969163	0.539721	Over fitting
2	naive bayes	0.631579	0.603524	0.633449	Acceptable
3	random forest	0.684211	0.687225	0.671777	Acceptable

The best performing model is : logistic

The Model with best AUC score is : logistic

Confusion Matrix

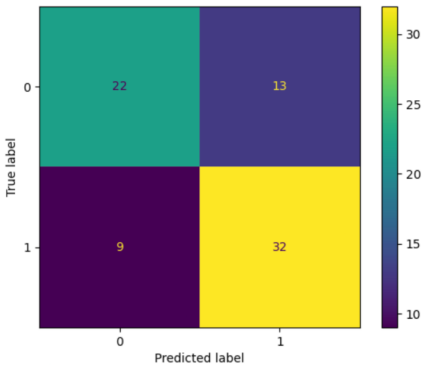
Logistic Regresion

NFL-Web App

localhost:9000/preprocess

Update

Logistic Regresion



Decision Tree

REFERENCES

1. https://aws.amazon.com/what-is/overfitting/#:~:text=Overfitting%20occurs%20when%20the%20model,to%20several%20reasons%2C%20such%20as%3A&text=The%20training%20data%20size%20is,all%20possible%20input%20data%20values.
2. https://docs.aws.amazon.com/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html
3. https://towardsdatascience.com/underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6fe4a8a49dbf
4. https://deepai.space/what-is-generalization-in-machine-learning/#:~:text=In%20machine%20learning%2C%20generalization%20is,classify%20between%20dogs%20and%20cats
5. https://www.javatpoint.com/accuracy_score-in-sklearn
6. https://www.sciencedirect.com/topics/engineering/confusion-matrix
7. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
8. https://www.bmc.com/blogs/mean-squared-error-r2-and-variance-in-regression-analysis/#:~:text=What%20is%20r2%20score%3F&text=%E2%80%9D%20%E2%80%A6the%20proportion%20of%20the%20variance,with%20no%20variance%20at%20all.