

CURRENCY AUTHENTICATOR

Introduction

- The project involves building a machine learning classifier model which predicts the authenticity of the paper currency, given the features variance, skewness, curtosis and entropy.
- The model learns on the concept of supervised learning and then undergoes training and finally predicts result with the accuracy score equal to **0.99**.
- The project consists of two different frontend for user end which can be used independently. One is built using “Flask, Flasgger & Swagger API”, and another is built using “Streamlit module”.
- Finally, the complete project is deployed using Docker.

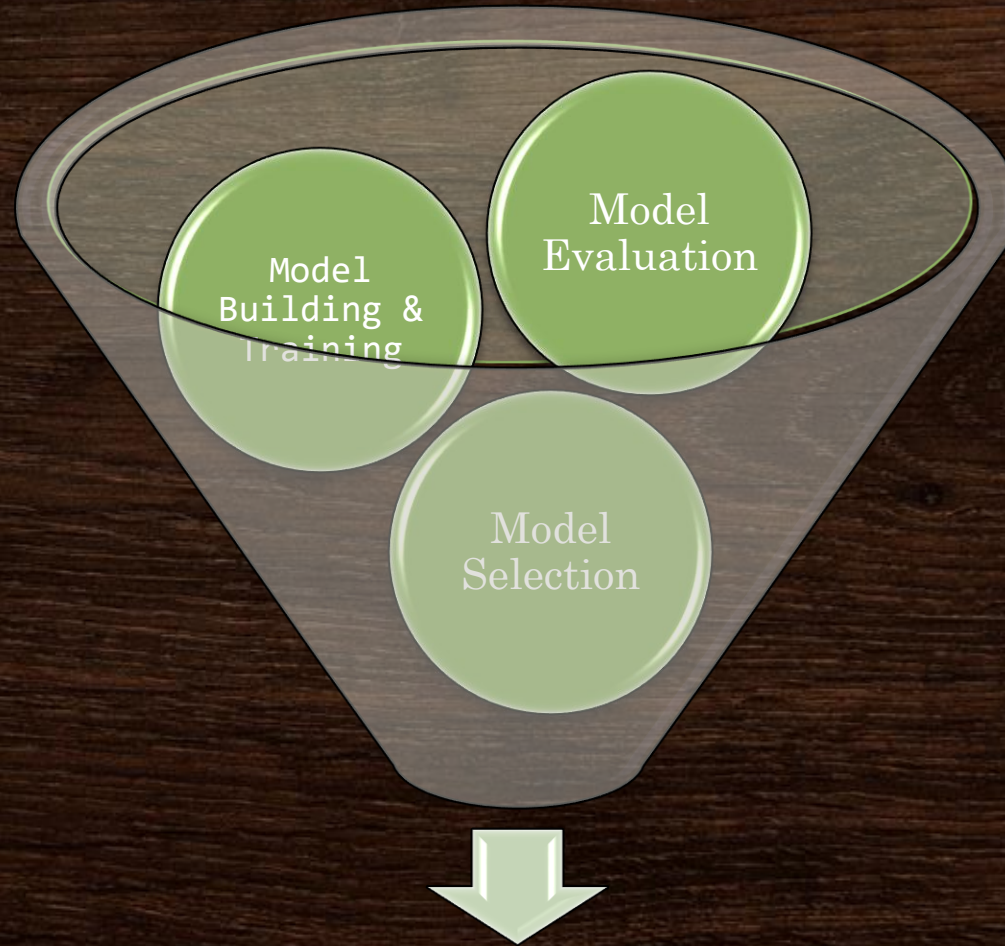
Process Diagram



Dataset Description

- The dataset “banknote_authentication.csv” is the source of information for the project. A set of images taken from genuine and forged banknote-like specimens is created.
- The final images have **400x400** pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about **660** dpi were gained. Wavelet Transform tool were used to extract features from images.
- Features such as wavelet variance, wavelet skewness, wavelet curtosis, and image entropy are extracted from the images.
- The number of instances (rows) in the dataset is **1372**, and the number of variables (columns) is **5**. In that way, the model has the following variables:
 - *variance_of_wavelet_transformed*, used as input.
 - *skewness_of_wavelet_transformed*, used as input.
 - *curtosis_of_wavelet_transformed*, used as input.
 - *entropy_of_wavelet_transformed*, used as input.
 - *class*, used as the target. It can have only two values: **1** (legal) or **0** (fraudulent).

Model Development



Final Model

Model Selection

- The dataset has 4 feature columns and 1 target column. The target column has 2 discrete values: **1** (legal) or **0** (fraudulent). It clearly shows we can approach with Binary classification algorithm for our model selection.
- *Random Forest Classification* is selected under model selection process.
- It is an ensemble tree-based learning algorithm. The Random Forest Classifier is a set of decision trees from randomly selected subset of training set. It aggregates the votes from different decision trees to decide the final class of the test object.
- Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. The reason for this wonderful effect is that the trees protect each other from their individual errors.

Model Building & Training

- Model is built by splitting the dataset into training set and testing set in the ratio of **70%** and **30%** respectively.
- Random Tree Classifier is imported by using the library Scikit-learn, commonly known as sklearn and used as “*sklearn.ensemble*”

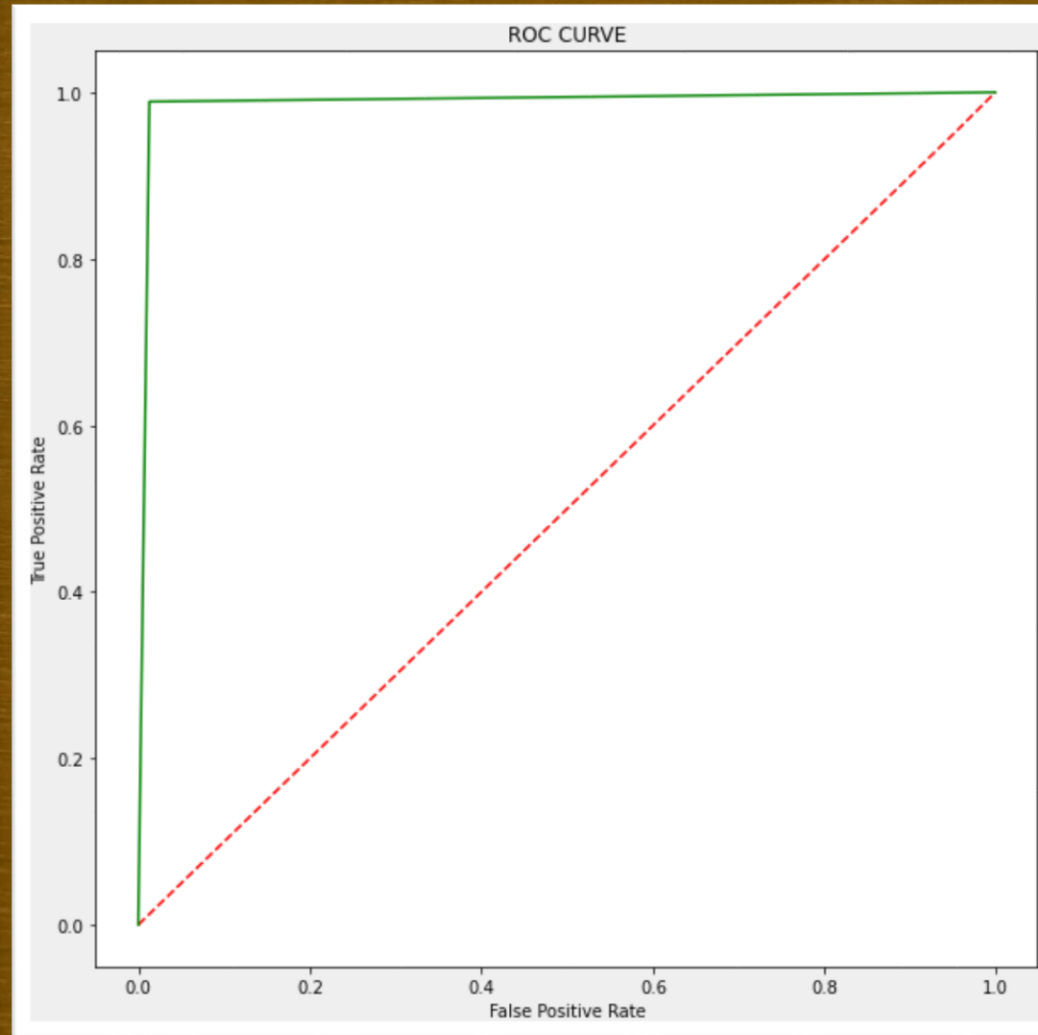
Model Evaluation

- Model is evaluated using various evaluating metrics such as Accuracy Score, F1 Score and a complete Classification Report is generated too.
- The details of the evaluation obtained is mentioned below:-
 - Accuracy Score: **0.99**
 - F1 Score: **0.99**

Classification Report

	precision	recall	f1-score	support
0	0.99	0.99	0.99	232
1	0.98	0.99	0.99	180
accuracy			0.99	412
macro avg	0.99	0.99	0.99	412
weighted avg	0.99	0.99	0.99	412

ROC Curve




Frontend Development

- The project works by taking 4 input features from the user. Frontend is required so that the user can input these features easily.
- To integrate the classifier model with the frontend, we have created a pickle file. The model is dumped into the pickle file so that it can be easily integrated and used in accordance with the frontend.
- We have created two different frontend using two different libraries.
 - Flask and Flasgger (Swagger API)
 - Streamlit
- Both of the frontend works absolutely fine and are mutually independent.
- As we run the project, we get an IP address, which can be accessed through web browser and we can make our model predict the result.

Frontend with Flask & Flasgger

- Flask is a "micro-framework" based on Werkzeug's WSGI toolkit and Jinja 2's templating engine.
- Flasgger is a Flask extension to help the creation of Flask APIs with documentation and live playground powered by SwaggerUI.
- A simple flask app is created by using Flask library which is then encapsulated inside the Swagger API which can be used through Flasgger library.
- It provides an IP Address which can be accessed via a web browser.
- The interface is designed to take either a single set of input features or multiple sets of input features in the form of a csv (comma separated values) file.
- As the execute button is clicked, the result is displayed in the response body column, which would be either **1** or **0**.
- Here, representation of class will be:
 - **1** denotes "Authentic"
 - **0** denotes "Not Authentic".

Diagram of frontend:

 **Swagger**
supported by SMARTBEAR

/apispec_1.json

Explore

A swagger API 0.0.1

/apispec_1.json

powered by Flasgger

[Terms of service](#)

default

GET

/predict

Let's Authenticate the Paper Currency

Built by ABHISHEK GAURAV & VICKY KUMAR

Parameters

Cancel

Name	Description
variance * required number (query)	<input type="text" value="variance"/>
skewness * required number (query)	<input type="text" value="skewness"/>
curtosis * required number (query)	<input type="text" value="curtosis"/>
entropy * required number (query)	<input type="text" value="entropy"/>

Execute

Frontend with Streamlit

- Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science.
- Streamlit library interacts with the pickle file, which is actually the classifier model and then outputs the result.
- As the predict button is clicked, the result is displayed in the response body column, which would be either **1** or **0**.
- Here, representation of class will be:
 - **1** denotes “Authentic”
 - **0** denotes “Not Authentic”.

Diagram of frontend:

PAPER CURRENCY AUTHENTICATOR

Currency Authenticator ML App

Variance

Skewness

Curtosis

Entropy

Predict

The output is

About

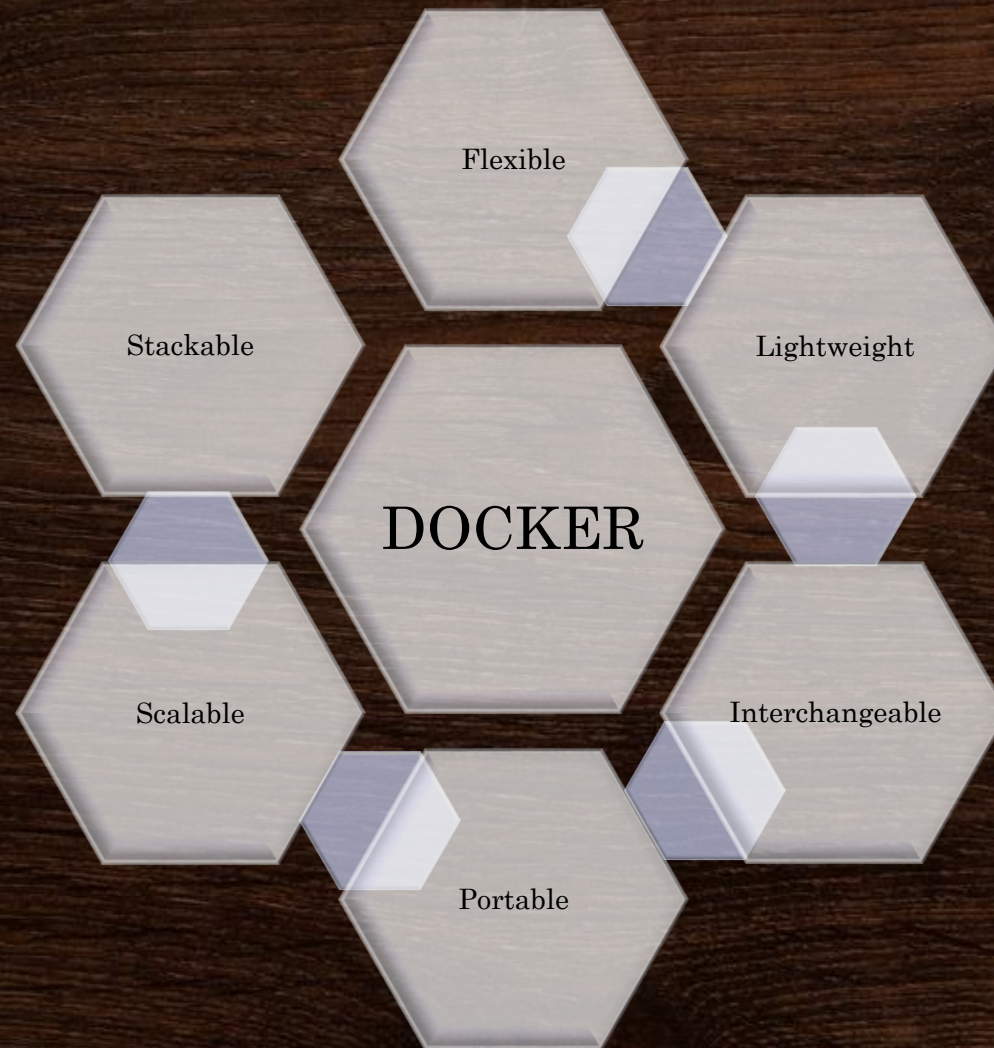
Built with Streamlit

Built by Abhishek Gaurav & Vicky Kumar

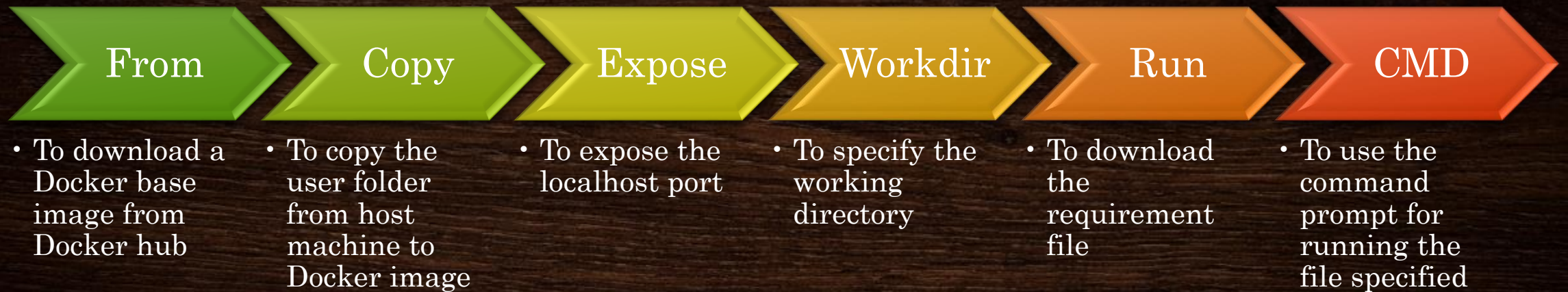
Deployment on Docker

- Deployment is all of the activities that make a software system available for use. The general deployment process consists of several interrelated activities with possible transitions between them.
- Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.
- Docker provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security allow you to run many containers simultaneously on a given host.
- When working with Docker, a 'Dockerfile' defines what goes on in the environment inside your container. Access to resources like networking interfaces and disk drives are virtualized inside this environment, which is isolated from the rest of your system.

Characteristics of Docker



Docker Commands Used



Conclusion

- Regardless of whichever frontend is being used, the obtained result is same.
- This depicts the mutual independency relationship of result with respect to the frontend.
- The accuracy of the predicted result is proved by performing the accuracy metrics.
- The results generated by the evaluation metrics of the project are the following:
 - Accuracy Score: 0.99
 - F1 Score: 0.99
 - Precision and Recall Score:
 - For 0 : 0.99 and 0.99
 - For 1 : 0.98 and 0.99