SKILL VERTEX

MAJOR PROJECT

FAKE CURRENCY DETECTION

-

GROUP 8

GROUP MEMBERS NAME:

1. MANSI SHERORAN
2. POOJA S
3. SATHYANARAYANAN R
4. ARUNKUMAR A

(BATCH : APR-JUL 23)

REPORT OF FAKE CURRENCY DETECTION

INTRODUCTION:

With this Machine Learning Project, we will be building a fake currency detection system for detecting fake currency and real currency. In today’s time, the production and circulation of inauthentic currency notes have been increasingly sophisticated. That’s why there needs to be a system to check whether a currency is fake or real.

So, let’s build this detection system in this project.

Fake Currency Detection System

So, what is this Fake Currency Detection System?

A fake currency detection system as the name suggests will find whether a note given to you by any person is real or fake. This system will be very useful because the circulation of inauthentic money is increasing day by day. So there needs to be a system that will prevent this from happening.

This system can be used by banks and common people. This will help the bank identify fake notes and real notes. This can also be used by common people to prevent them from being thug.

Fake Currency Detection is a real problem for both individuals and businesses. Counterfeiters are constantly finding new methods and techniques to produce counterfeit banknotes, which are essentially indistinguishable from real money. At least for the human eye. In this article, I will introduce you to Fake Currency Detection with Machine Learning.

What it does ?

Fake currency detection is a critical task in the field of finance and security, aiming to identify counterfeit banknotes from genuine ones.This project aims to develop a machine learning model to predict the authenticity of banknotes. The model uses the importanta four features such as variance, skewness, kurtosis, and entropy of wavelet-transformed images of the banknotes. The target value is 0 for real banknotes and 1 for fake banknotes.

Variance is a statistical measure that quantifies the spread of pixel intensities in an image. In the context of fake currency detection, it helps differentiate between genuine banknotes, which exhibit consistent patterns, and counterfeit banknotes, which often have irregular or blurred regions.

Skewness measures the asymmetry of the pixel intensity distribution in an image. Genuine banknotes typically exhibit low skewness values since their patterns are symmetric, whereas counterfeit banknotes may have higher skewness values due to irregular printing or manipulation.

Kurtosis is a statistical measure that characterizes the shape of the pixel intensity distribution. Genuine banknotes generally exhibit moderate kurtosis values, representing a relatively smooth distribution, while counterfeit banknotes may show higher or lower kurtosis values due to alterations or anomalies in their patterns.

Entropy is a measure of the randomness or uncertainty of pixel intensities in an image. In the context of fake currency detection, it helps distinguish between genuine banknotes with consistent patterns and counterfeit banknotes that may have random or inconsistent patterns.

How we made it?

- ✓ The dataset used in this project is the **“Banknote Authentication Dataset”** from the UCI Machine Learning Repository.
- ✓ It contains 1,372 instances of banknotes, each with five numeric input variables and one binary target variable.
- ✓ The target value is 0 for real banknotes and 1 for fake banknotes.

APPROACH:

- The main goal of the project is to create fake currency detection model that predicts whether the currency is original or fake by considering the data in the train and test data-set.
- As a first step required libraries like pandas, numpy and sklearn are imported. Next the data-set is loaded using read_csv() syntax.
- The data is loaded and ready for further steps. This data is now ready for pre-processing.

- Once the data is pre-processed, they are now taken as x_train,x_test, y_train,y_test for the prediction method using various machine learning algorithm.
- The model is builded for the fake currency detection ,next is to deploy the model to the user by collecting the data (features) from the user and predict the currency is fake or not..

PRE-PREPROCESSING:

Step 1:

- The first step of pre-processing is find out the null or empty values in the dataset.
- In this dataset there is no empty or null values.

Step 2:

- Split the data into x_train,x_test, y_train,y_test using sklearn.model_selection() in that train_test_split function.
- Then print the values of x_train,x_test, y_train,y_test.

Step 3:

- Standardize the x_train and x_test data with the use of StandardScaler().

StandardScaler:

Standard Scaler is a data preprocessing technique used to standardize or normalize the features in a dataset. It transforms the data in such a way that the mean of each feature becomes zero and the standard deviation becomes one.

It is important to note that Standard Scaler should be applied only to the training data and then the same transformation should be applied to the test data. This is to ensure that the test data is transformed in the same way as the training data, which will improve the accuracy and reliability of the model.

Step 3:

- Then print the values of x_train,x_test after the standardization.

By this the pre-processing ends and the same steps are applicable to test dataset.

VISUALIZATION:

Visualization is a key aspect of any project report as it helps people understand more about the topic. For visualizing the data used in fake currency detection,Python libraries (matplotlib and seaborn) is used.

Matplotlib:

- ❖ Matplotlib is a comprehensive plotting library for creating static, animated, and interactive visualizations in Python.
- ❖ It provides a wide range of plotting functions and customization options, allowing you to create various types of plots, such as line plots, scatter plots, bar plots, histograms, etc.
- ❖ Matplotlib provides a MATLAB-like interface with a pyplot module that makes it easy to create basic plots quickly.
- ❖ It offers fine-grained control over every aspect of a plot, including the figure, axes, labels, colors, markers, and annotations.
- ❖ While powerful, Matplotlib can sometimes be low-level and require more code to produce visually appealing plots.

Seaborn:

- ❖ Seaborn is a high-level data visualization library built on top of Matplotlib. It provides a simplified interface to create attractive statistical graphics.
- ❖ It is designed to work seamlessly with Pandas data structures, making it easy to visualize data stored in DataFrames.
- ❖ Seaborn offers a range of statistical visualization techniques, such as distribution plots (histograms, kernel density plots), categorical plots (bar plots, count plots), regression plots, and multi-plot grids.
- ❖ It has built-in themes and color palettes that enhance the aesthetics of plots. You can easily customize the visual aspects, such as colors, fonts, and styles.

- ❖ Seaborn integrates well with statistical analysis by providing functions to visualize relationships between variables, plot linear regression models, and display statistical summaries.
- ❖ It simplifies the creation of complex visualizations, making it suitable for exploratory data analysis and communication of results.

We can now draw a pair diagram to get an overview of the relationship between all the entities. I will also colour the observations: blue for genuine banknotes and orange for counterfeit banknotes.

PAIR PLOT:

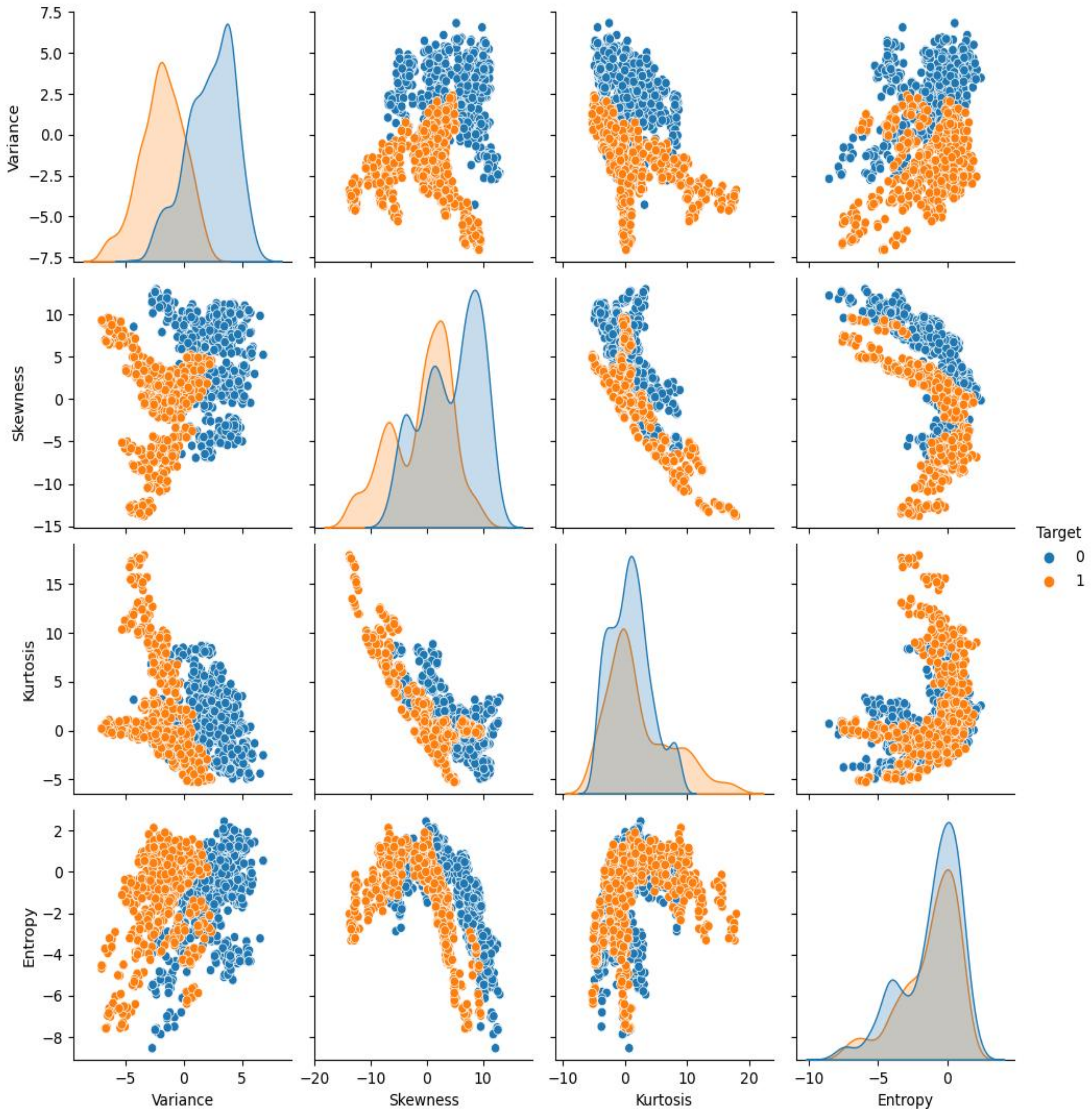
A pair plot is a type of plot that allows you to visualize the pairwise relationships between multiple variables in a dataset. It is typically created using a library such as Seaborn.

A pair plot, also known as a scatter plot matrix, displays scatter plots for all possible combinations of the numeric variables in a dataset. It helps in understanding the relationships, patterns, and correlations between different variables. Each scatter plot in the pair plot represents the relationship between two variables, where one variable is plotted on the x-axis and the other on the y-axis.

Pair plots are useful for gaining insights into the relationships and distributions of variables in a dataset. They can help identify patterns, trends, outliers, and correlations, which are crucial in exploratory data analysis and feature selection.

To create a pair plot in Python, you can use the pairplot() function provided by the Seaborn library.

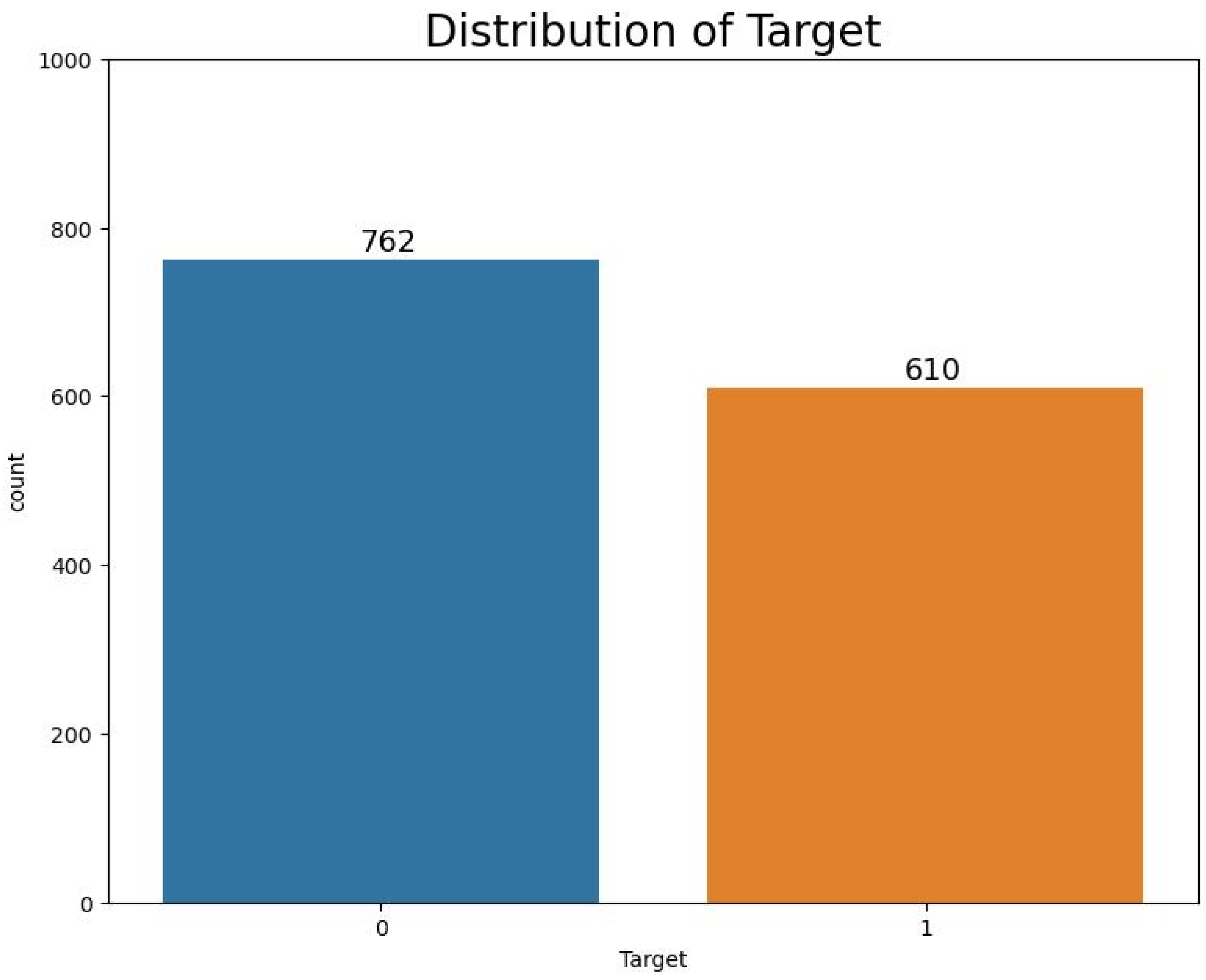
Relationship Between The Features



The above graph describes the following things,

- The distribution of both variance and skewness appears to be quite different for the two target characteristics, while kurtosis and entropy appear to be more similar.
- There are clear linear and nonlinear trends in the input features.
- Some characteristics seem to be correlated.
- Some features seem to separate genuine and fake banknotes quite well

.



The above graph describes the following things,

- It describe how the target variables are splitted into 0 and 1, which means 0 denotes 'Original Currency' records and 1 denotes 'Fake Currency' records.

ALGORITHM:

In this project we use various machine learning algorithms to detect the fake currency,by comparing accuracy score which algorithms gives more accurate value.

LOGISTIC REGRESSION:

Logistic Regression is a machine learning algorithm used for binary classification problems, where the outcome variable takes only two values (e.g., yes/no, true/false, 0/1). It is a type of regression analysis that models the probability of a certain outcome based on one or more input features.

The logistic regression model works by fitting a sigmoid function to the input data, which maps the input features to a probability between 0 and 1.

Implementation In Python:

Sklearn.linear_model has the logistic regression algorithm which is imported as LogisticRegression.

The code is given below

```
from sklearn.linear_model import LogisticRegression

LoR = LogisticRegression(random_state =  0)

Model = LoR.fit(x_train,y_train)
```

Now move for prediction using x_test data,and the result (y_pred) is compare with y_test.

```
y_pred = model.predict(x_test)
```

Using following code,Create the confusion metric and accuracy score for y_test and y_pred.

```
from sklearn.metrics import confusion_matrix,accuracy_score

cm = confusion_matrix(y_test,y_pred)

acc = accuracy_score(y_test,y_pred)
```

By using logistic regression we get **97.96 %** accuracy score.

DECISION TREE:

Decision trees are versatile and interpretable machine learning models that can be used for both classification and regression tasks Decision trees are predictive models that map observations about an item to conclusions about its target value. They represent decisions and their possible consequences in a tree-like structure.

A decision tree consists of internal nodes, branches, and leaf nodes. Internal nodes represent features or attributes, branches represent decision rules based on those attributes, and leaf nodes represent the predicted outcome or target value.The process of constructing a decision tree involves recursively partitioning the feature space based on the values of the features. At each step, the algorithm selects the best attribute to split the data, typically using metrics like information gain, Gini impurity, or entropy.It is easy to interpret and visualize, as the decision rules are explicitly represented in the form of a tree structure. They provide insights into the most important features and their impact on the prediction.

Decision trees can handle both categorical and numerical features. For categorical features, the tree branches represent different categories, while for numerical features, the branches represent different ranges or thresholds. Decision trees have a wide range of applications, including but not limited to, fraud detection, credit scoring, medical diagnosis, customer segmentation, and recommendation systems.

Implementation In Python:

```
from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import confusion_matrix,accuracy_score

dtc = DecisionTreeClassifier()

model1 = dtc.fit(x_train, y_train)

y_pred1 = model1.predict(x_test)

cm1 = confusion_matrix(y_test,y_pred1)

acc1 = accuracy_score(y_test,y_pred1)
```

By using Decision Tree we get **98.25 %** accuracy score.

XGBOOST:

XGBoost (eXtreme Gradient Boosting) is a powerful and popular machine learning algorithm known for its high performance and scalability.It is an optimized implementation of the gradient boosting framework, which combines multiple weak prediction models (typically decision trees) to create a strong predictive model. It is designed to be highly efficient and provides significant performance improvements over traditional gradient boosting implementations.

XGBoost applies a gradient boosting algorithm that sequentially adds decision trees to minimize a specified loss function. Each subsequent tree is built to correct the errors made by the previous trees.It uses gradient descent optimization techniques to minimize the loss function during the tree construction process. This helps in finding the best splitting points and improving the predictive accuracy.It supports both regression and classification tasks, making it suitable for a wide range of predictive modeling problems. It offers several advanced features, including handling missing values internally, regularization techniques (such as L1 and L2 regularization), custom loss functions, and early stopping to prevent overfitting.

XGBoost provides a rich set of hyperparameters that allow fine-tuning of the model for optimal performance. These hyperparameters control various aspects of the boosting process, tree construction, and regularization.

Implementation In Python:

```
from xgboost import XGBClassifier
from sklearn.metrics import confusion_matrix,accuracy_score

xgb = XGBClassifier()
model2 = xgb.fit(x_train, y_train)
y_pred2 = model2.predict(x_test)
cm2 = confusion_matrix(y_test,y_pred2)

acc1 = accuracy_score(y_test,y_pred2)
```

By using XGBOOST we get **100.00%** accuracy score.

USAGE:

- To run the project, follow these steps:
- Clone this repository to your local machine.
- Install the dependencies listed above.
- Open a terminal and navigate to the project directory.
- Run the fake currency detection .ipynb jupyter notebook to train and evaluate the machine learning model.

What is the purpose of currency detector?

A currency detector or currency validator is a device that determines whether note or coins are genuine or counterfeit.

What is the scope of fake currency detection project?

This system focuses on the improvement and implementation of the fake currency detection application. The scope of the project is to provide techniques and methods that appear suitable while

CONCLUSION:

The main motivation behind the development of this model is to provide a better way for people to detect fraudian currency notes using an easily available device. Features of currency note like serial number, security thread, identification mark, Mahatma Gandhi portrait were extracted. The process starts from image acquisition to calculation of intensity of each extracted features.

In conclusion, this report presents a machine learning-based approach for detecting fake currency using features such as variance, skewness, kurtosis, and entropy. The objective of the study was to develop an effective and reliable method to distinguish genuine currency notes from counterfeit ones.

The analysis began by collecting a dataset consisting of various currency notes, both genuine and counterfeit, and extracting relevant features from the images of the notes. The features extracted included variance, skewness, kurtosis, and entropy, which provide valuable information about the distribution and complexity of pixel intensities.

Using these features, a machine learning model was trained to classify the currency notes as genuine or fake. Several popular algorithms, such as decision trees, logistic regression, and XGBoost, were utilized and their performance compared. The model was trained on a labeled dataset, and appropriate evaluation metrics such as accuracy score and confusion metices were used to assess its effectiveness.The results obtained demonstrated that the selected features, namely variance, skewness, kurtosis, and entropy, were effective in distinguishing between genuine and fake currency notes. The machine learning models achieved high accuracy and performed well in detecting counterfeit currency.

The findings of this study indicate the potential of using variance, skewness, kurtosis, and entropy as informative features for fake currency detection. The utilization of machine learning algorithms provided an automated and reliable approach for identifying counterfeit notes, which can aid financial institutions, businesses, and individuals in minimizing financial losses and maintaining the integrity of currency transactions.

------------------------------ THANK YOU ----------------------------