1.INTRODUCTION

The online shopping growing day to day, credit cards are used for purchasing goods and services with the help of virtual card and physical card whereas virtual card for online transaction and physical card for offline transaction. Machine learning (ML) has emerged as a powerful tool for detecting and preventing fraud by analysing patterns in transaction data to identify potentially fraudulent activity. Credit card fraud is a widespread problem that affects millions of people and costs billions of dollars each year.

Credit card fraud is a type of financial fraud. It involves the unauthorised use of someone's credit card information to make fraudulent transactions. leading to financial losses for the cardholder. The goal is to minimize financial losses for cardholders and by quickly identifying and stopping unauthorized or suspicious transactions. Machine Learning can be applied to fraud detection, including the types of Machine Learning algorithms commonly used and the steps involved in building an Machine Learning-based fraud detection system.

Credit card fraud detection is a critical area of financial security, aimed at identifying and preventing unauthorized or illegal transactions. The goal is to protect both consumers and financial institutions from losses due to fraudulent activities.

2.SYSTEM ANALYSIS

System analysis is a project management technique and a phase of system development life cycle that divides complex projects into smaller, more easily managed segments or phases. System analysis is a critical phase in the systems development life cycle (SDLC), where the main focus is to understand and specify the requirements of a system to solve a problem or meet an opportunity. It involves a detailed examination of the components, functions, processes, and data flows within an organization to create an efficient and effective system. The purpose of the systems analysis phase is to understand the requirement and build a logical model of the new system. In the system analysis for credit card fraud detection using machine learning (ML), the primary objective is to develop a robust and efficient system capable of accurately identifying fraudulent transactions while minimizing false positives.

The system begins with the collection of a comprehensive dataset containing both legitimate and fraudulent transactions, which serves as the foundation for training and testing machine learning models. The selection of appropriate machine learning algorithms, such as logistic regression, decision trees, or random forest, is crucial for model effectiveness. These models are trained, evaluated, and optimized using metrics like accuracy, precision, recall, support and F1-score. Once trained, the models are deployed into a production environment and integrated with existing fraud detection systems. Overall, the use of machine learning in credit card fraud detection offers a proactive and adaptable approach to combating fraud, ultimately protecting both financial institutions and customers from fraudulent activities.

3.SOFTWARE REQUIREMENT SPECIFICATION

Software Requirement Specification (SRS) is a fundamental document, which forms the foundation of the software development process. SRS not only lists the requirements of a system but also has a description of its major features. The software requirements specification for the credit card fraud detection system using machine learning (ML) outlines the functional and non-functional requirements for the system.

Functionally, the system must be able to collect transaction data, preprocess it to train ML models using various algorithms, evaluate model performance, detect fraud. Non-functionally, the system must be scalable to handle a large volume of transactions, achieve high accuracy in fraud detection

A system requirement is one of the main steps involved in the development process. The system will be developed using Python for ML model development, Scikit-learn, logistic regression, random forest, decision tree for building and training ML models. The system will evaluate these models metrics like accuracy, precision, recall, support and F1-score. By leveraging ML algorithms and real-time processing, the system aims to detect and prevent fraudulent transactions, thereby minimizing financial losses for both cardholders and financial institutions. The system's ability to collect, preprocess, and analyse transaction data, coupled with its scalability, accuracy, and security features, makes it a robust solution for combating credit card fraud

3.1 Existing System:

The existing system for credit card fraud detection using machine learning (ML) incorporates various algorithms and techniques to detect fraudulent transactions. It typically involves the collection of transaction data, and this data is then preprocessed. The system will be developed using Python for machine learning model development, scikit-learn, logistic regression, random forest, decision tree for building and training ML models.

The system will evaluate these models using metrics like accuracy, precision, recall, support and F1-score.

The existing system refers to the current methods or technologies employed to detect fraudulent transactions. These existing systems often rely on statistical method, or basic machine learning algorithms to flag potentially fraudulent transactions based oncriteria or patterns.

✓ Limitations of Existing System:

- Imbalanced data
- Human expertise.
- Scalability.
- Feature Engineering.
- Data maintenance is difficult.

✓ Advantages of Existing System:

- Machine Learning Capabilities.
- High accuracy and precision.
- Scalability.
- Cost Efficiency.
- Monitoring and detection.

3.2 Proposed System:

The proposed system involves integrating supervised machine learning algorithms to enhance credit card fraud detection. This approach contrasts with traditional rule-based systems by allowing the system to learn from historical data and adapt to new fraud patterns in real-time, thereby potentially increasing detection accuracy and reducing false positives. The proposed system for credit card fraud detection using machine learning (ML) is designed to enhance the accuracy and efficiency of fraud detection while addressing the limitations of existing systems. The system will employ various ML techniques to analyse transaction data and detect fraudulent activity. Overall, the proposed system aims to provide a robust and reliable solution for credit card fraud detection using ML, ensuring the security of financial transactions.

✓ Advantages of proposed system:

- Consumes less time.
- Cost Effectiveness.
- Improved accuracy.
- Less human error.
- Reduce false positive.

3.3 Requirement Specifications

3.3.1 Hardware and Software Requirements

- **Hardware Requirements:**

Processor: Intel Core i5 and above

RAM: 4GB and higher.

Hard Disk: SSD storage for a faster data access and processing.

- **Software Requirements:**

Operating System: Windows.

Front End: Stream lit.

Language Used: Python.

Web Browser: Google Chrome.

Integrated Development Environment: Jupyter Notebook and Visual Studio code.

3.3.2 About the Technologies Used:

➢ Stream lit:

- Stream lit is a powerful open-source Python library designed for creating interactive web applications easily and quickly, primarily targeted at data scientist and machine learning practitioners. It simplifies the process of turning data scripts into shareable web apps.

- Stream lit uses a straightforward, Pythonic syntax, allowing developers to write web applications using familiar Python constructs like functions, loops, and variables.

- Stream lit offers a wide range of interactive widgets (such as sliders, checkboxes, dropdowns, buttons, etc.) that enable users to manipulate parameters and input dynamically.

- Stream lit focuses on simplicity and ease of use, allowing users to create web applications with minimal effort. Seamlessly integrate with popular python libraries such as pandas, matplotlib, plot and tensor flow to visualize data and models.

- Stream lit provides a variety of interactive widgets that you can use to gather input and control the behaviour of your application. Stream lit is the easiest way especially for the people with no front-end knowledge to put their code into a web application.

➢ Python:

- Python is a high-level, interpreted programming language known for its readability, simplicity, and versatility. It was created by Guido van Rossum and first released in 1991. Python emphasizes code readability and allows programmers to express concepts in fewer lines of code compared to other languages such as C++ or Java

- Python syntax is designed to be easy to read and write, making it an excellent choice for beginners and for rapid development.

- Python code is executed line by line at runtime by the Python interpreter. This means you can immediately see the results of your code without needing to compile it first, which speeds up development time.

- Python runs on various operating systems, including Windows, macOS, and Linux. This makes it highly portable, allowing code written in Python to run seamlessly across different systems without modification.

- Python comes with a comprehensive standard library that provides modules and packages for a wide range of tasks, from string manipulation to network programming, file I/O, and more. This reduces the need for external libraries for many common tasks.

- Python is used extensively in scientific and numerical computing, with libraries like NumPy, SciPy, and pandas providing powerful tools for data manipulation and analysis.

➢ Machine Learning:

- Machine learning is a subset of artificial intelligence (AI) that focuses on the development of algorithms and models that allow computers to learn from and make predictions or decisions based on data. The core idea behind machine learning is to enable computers to learn automatically without human intervention or explicit programming.

- Machine learning is about building and training models using data, so they can make predictions or decisions. In supervised learning, the model is trained on a labeled dataset, where each data point is paired with the correct label. The goal is to learn a mapping from input features to the correct output label, so the model can make predictions on new, unseen data.

- ML algorithms can often achieve higher accuracy than traditional methods in tasks such as image recognition, speech recognition, and natural language processing, predictions. This can lead to better outcomes in various applications.

- Machine learning has a wide range of applications, from natural language processing and computer vision to healthcare and finance. It continues to be a rapidly evolving field, with new algorithms and techniques constantly being developed to tackle increasingly complex problems.

- Machine Learning models can continuously learn from new data, allowing them to adapt to changing conditions and improve over time.

3.3.2.1 Algorithm used

➢ Decision Tree:

- Decision tree is a popular and easy-to-understand machine learning algorithm used for both classification and regression tasks. It models decisions based on a series of questions or conditions applied to the features of the data.

- At each internal node of the tree, a decision is made based on the value of a feature, and the data is split into two or more branches. This process is repeated recursively for each branch until a leaf node is reached, which represents the final decision or prediction.

- Decision trees are often used as the building blocks for more complex ensemble learning algorithms, such as Random Forests and Gradient Boosting Machines, which combine multiple decision trees to improve predictive performance.

- Overall, decision trees are a versatile and intuitive machine learning algorithm that is widely used in practice for a variety of tasks. decision trees are prone to overfitting, especially when they are deep and complex.

- The construction of a decision tree is a process that recursively splits the data into subsets based on the most significant attribute at each step. The goal is to create homogenous subsets that contain similar values for the target variable. This process continues until the subset at a node has all the same values of the target variable, or when splitting no longer adds value to the predictions.

➢ Logistic Regression:

- Logistic regression is a statistical method used for binary classification tasks, where the goal is to predict the probability that an instance belongs to a particular class.

- Logistic regression outputs the probability of the instance belonging to the default class (class 0) or the other class (class 1). It models the relationship between the dependent variable (target) and one or more independent variables (features) by estimating probabilities using a logistic function.

- Logistic regression is used when the dependent variable (or target variable) is categorical and has only two possible outcomes, often coded as 0 and 1.

- Logistic regression is a valuable tool for detection tasks, providing interpretable results and the ability to quantify the impact of individual features on the detection outcome.

- Logistic regression provides a straightforward way to interpret the relationship between the independent variables and the probability of the outcome. Logistic regression is computationally efficient and relatively simple to implement, especially compared to more complex machine learning algorithms. This interpretability makes logistic regression particularly useful when the goal is to understand the underlying factors driving a particular outcome.

➢ Random Forest:

- Random Forest is a popular machine learning algorithm known for its versatility and effectiveness in a variety of tasks, especially in classification and regression problems. It belongs to the ensemble learning method, where multiple individual models are combined to improve the overall performance.

- Random Forest has several advantages, including its ability to handle large datasets with high dimensionality, its resistance to overfitting, and its effectiveness in dealing with missing values. Overall, Random Forest is a powerful and versatile algorithm that is widely used in practice for its robustness and ease of use.

- A simple model that splits the data into subsets based on feature values. Each node represents a feature, each branch represents a decision rule, and each leaf represents an outcome.

- Random Forest is a collection of decision trees, where each tree is trained on a random subset of the training data and a random subset of the features. This randomness helps to decorrelate the trees, reducing the risk of overfitting and improving the generalization of the model.

- Random Forest is an ensemble learning method, meaning it combines the predictions of multiple individual models to improve accuracy. In the case of Random Forest, the individual models are decision trees.

4.SYSTEM DESIGN

4.1 Architecture:

The architecture of a credit card fraud detection system involves various components and processes designed to identify and prevent fraudulent transactions in real-time. This architecture typically integrates data collection, preprocessing, machine learning models, and decision-making algorithms.

- ✓ Interfaces with the user (eg:web) showing transaction status
- ✓ Manages data storage, retrieval, and preprocessing.
- ✓ Contains the fraud detection models and algorithms.
- ✓ Processes for collecting, cleaning, and preprocessing data, followed by training and validating models.
- ✓ Deployment of models to production environments, enabling them to make predictions on new transactions.

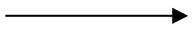
4.2 Data Flow Diagram:

A Data Flow Diagram (DFD) is a graphical representation that depicts the flow of data through a system. For a credit card fraud detection system, a DFD can illustrate how transaction data is processed and analysed to identify potential fraud. DFD graphically representing the functions, or processes, which capture, manipulate, store and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. A structure of DFD allows starting from a broad overview and expands it to a hierarchy of detailed diagrams.

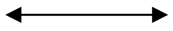
DFD has often been used due to the following reasons:

- Logical information flow of the system.
- Visualization of system process.
- Supporting the design and optimization of fraud detection algorithms and workflow.
- Documenting the system for future reference and for training a new member.
- Clarifying interaction between different components like transaction system, fraud detection engines.

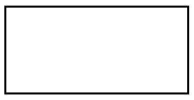
The following symbols are used in the Data Flow Diagram:



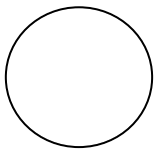
This arrow diagram represents the flow of data in one direction.



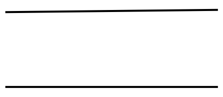
This arrow diagram represents the flow of data in bi-direction.



This rectangular symbol represents the activity of the application.



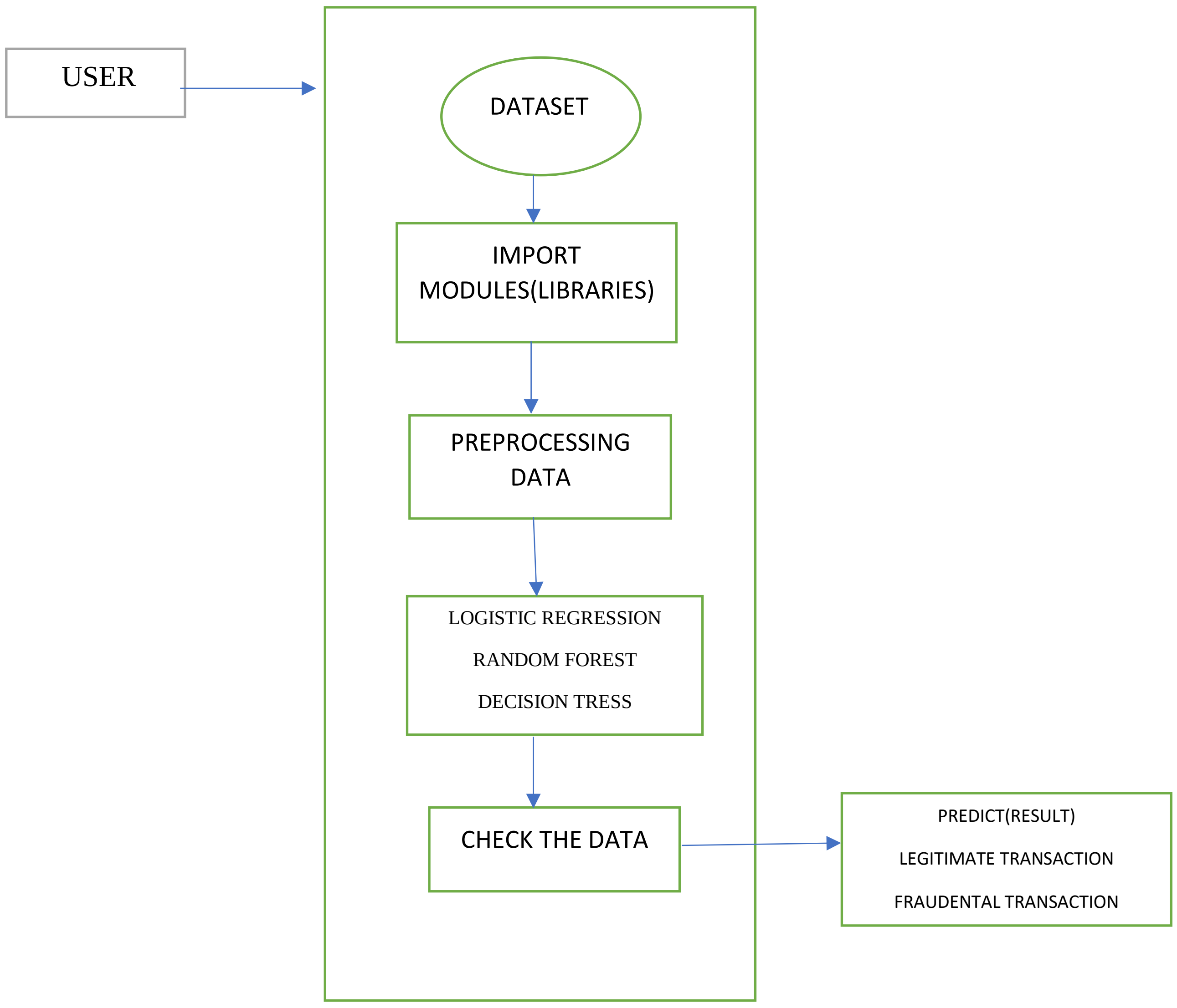
This circular symbol denotes the users / source of inputs to the activity of the application.



This symbol denotes the Database and Database Tables of the application.

➢ DATA FLOW DIAGRAM

CREDIT CARD FRAUD DETECTION USING PREDICTIVE MODLE



➢ Working of the project:

- Credit card fraud detection using machine learning involves training a model to distinguish between legitimate and fraudulent transactions.

- First step is used to import the necessary libraries and modules to perform common task. Libraries in programming serves as essential tools for developers for providing a collection of pre-written code. The libraries used are:

    - NumPy: NumPy, short for Numerical Python, is a fundamental library in Python used for numerical and scientific computing. It provides support for arrays, matrices, and many mathematical functions to operate on these data structures. NumPy is widely used in data science, machine learning, and scientific research due to its powerful features and high performance.

    - Pandas: Pandas is a powerful and flexible open-source data analysis and manipulation library for Python. It provides data structures and functions needed to manipulate structured data seamlessly. Pandas is especially well-suited for working with tabular data (data in a table format) similar to SQL databases or Excel spreadsheets.

    - Matplotlib: Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It is the most widely used plotting library in the Python ecosystem and serves as the foundation for many other visualization libraries.

    - Seaborn: Seaborn is a Python data visualization library based on Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphics. It is particularly well-suited for visualizing complex datasets, enabling users to generate a wide variety of plots with minimal code.

- Next, process starts with reading the dataset which contains transaction details, including many features.

- Then, next step is understanding the data in detail using different methods:

  - head () which is used to find top 5 rows (by default).
  - info () which is used to give the summary of the data frame, including datatypes of each column and row and number of non-null values
  - describe () which is used to provide statistics such as count, min, standard deviation etc. for each numerical column in the data frame.
  - value counts () which is used to count the occurrences of unique values in a series or a data frame column.
  - shape () it is an attribute in pandas and numpy which is used to determine the dimension of the data structure.
  - group by () The group by method in pandas is a powerful tool for grouping data based on one or more columns and performing aggregate operations on each group.

- And the next step in our project, is assigning the logistic regression to the model and then fitting the model and then training and testing the model in order to predict the model with an accuracy.

- Next step is separating the imbalanced legitimate and fraudulent transactions and balancing legitimate.

- Next step is splitting the dataset into training and testing the data. And creating a dictionary to store the logistic regression, decision tree, random forest model for the comparision purpose.

- Various machine learning algorithms are used in this project to predict, such as logistic regression, random forests or decision tree can be used for training.

- And then training and evaluating the model.

- After training, the model's performance is evaluated using the test set, typically using metrics like precision, recall, F1-score, precision, support.

- Next step is using streamlit for the displaying credit card fraud detection model. Streamlit application displays the accuracy of the models.

- After then by entering user transaction in an input column it predicts whether it is a legitimate or fraudulent transaction.

- Along with that, it displays model accuracy comparison using bar plot, line graph, confusion matrix and classification report for each model.

4.3 E-R Diagram (Entity-Relationship Diagram):

Entity Relationship Diagram depicts the various relationships among entities, considering each objective as entity. Entity relationships are described by their dependence on each other, as well as the extent of the relationship between the data stores. It depicts the relationship between data Objects. While ER models are mostly developed for designing relational databases in terms of concept visualization and in terms of physical database design, there are still other situations when ER diagrams can help. The ER diagram is a notation that is used to conduct the data modelling activity.

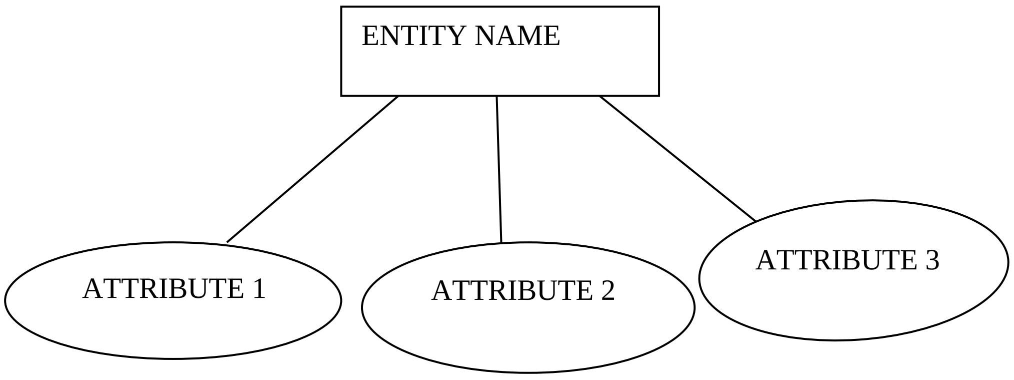
i. Entity:

An ERD entity is a definable thing or concept within a system. In ER models, an entity is shown as a rectangle, with its name on top and its attributes listed in the body of the entity shape.

ENTITY NAME

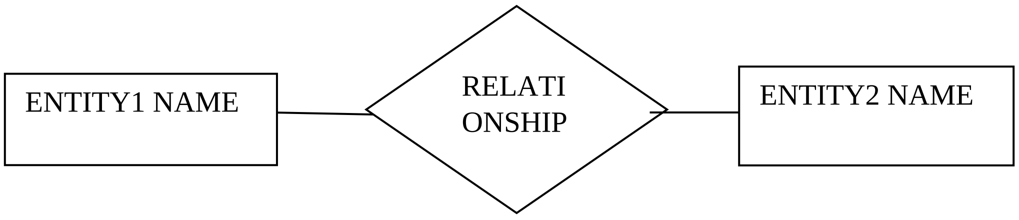
ii. Attributes:

Also known as a column, an attribute is a property or characteristic of the entity that holds it. An attribute has a name that describes the property and a type that describes the kind of attribute it is, such as varchar for a string, and int for integer. It is represented by an oval shape in the ER Diagram. i.e.

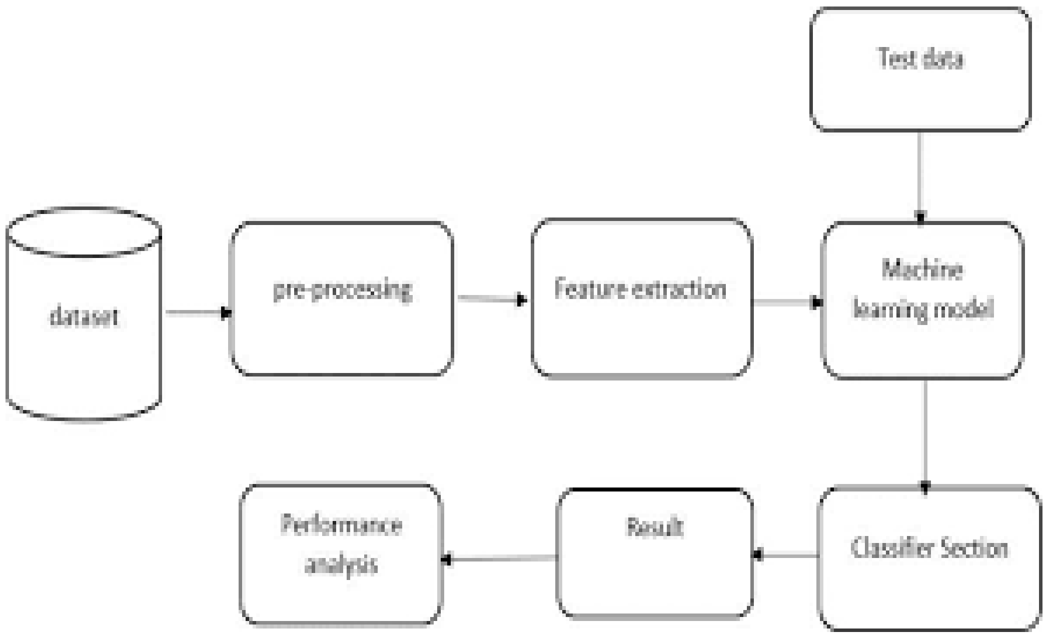


iii. Relationship:

A relationship between two entities signifies that the two entities are associated with each other. It is represented by a Rhombus shape in ER Diagram.



ENTITY RELATION DIAGRAM (FUNCTIONALITY):



5.IMPLEMENTATION

Implementation is the process of converting a or a revised system design into an operational one. The Objective is to put the new or revised system that has been tested into operation while holding costs, risks, and personal irritation to the minimum. A critical aspect or the implementation process is to ensure that there will be no disrupting the functioning or the organization. The best method for gaining control while implanting any new system would be to use well planned test for testing all new programs.

- **Coding Part:**

```
import numpy as np

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score


# Load the credit card dataset

credit_card_dataset = pd.read_csv("creditcard.csv.csv")

# Display the first few rows of the dataset to understand its structure

credit_card_dataset.head()

# Display the shape of the dataset (number of rows and columns)

credit_card_dataset.shape
```

```
# Access the 'Class' column which indicates whether a transaction is legitimate (0) or fraudulent (1)

credit_card_dataset['Class']

# Count the number of legitimate and fraudulent transactions

credit_card_dataset['Class'].value_counts()

# Display statistical summary of the dataset

credit_card_dataset.describe()

# Separate the dataset into legitimate and fraudulent transactions

legit = credit_card_dataset[credit_card_dataset.Class==0]

fraud = credit_card_dataset[credit_card_dataset['Class']==1]

# Count the number of each unique combination of columns in legitimate transactions

legit.value_counts()

# Count the number of each unique combination of columns in fraudulent transactions

fraud.value_counts()

# Display all legitimate transactions

Legit

# Display all fraudulent transactions

Fraud

# Count the number of each unique combination of columns in fraudulent transactions (redundant)

fraud.value_counts()
```

```
# Display the shape of the legitimate transactions (number of rows and columns)

legit.shape

# Display the shape of the fraudulent transactions (number of rows and columns)

fraud.shape

# Randomly sample a subset of legitimate transactions to balance the dataset

legit_sample = legit.sample(n=284315)

# Concatenate the sampled legitimate transactions with all the fraudulent transactions

credit_card_dataset = pd.concat([legit_sample,fraud],axis=0)

# Display the shape of the balanced dataset

credit_card_dataset.shape

# Count the number of legitimate and fraudulent transactions in the balanced dataset

credit_card_dataset['Class'].value_counts()

# Calculate and display the mean of each feature for legitimate and fraudulent transactions

credit_card_dataset.groupby('Class').mean()

# Define the feature set (X) by dropping the 'Class' column

X = credit_card_dataset.drop('Class', axis=1)

# Define the target variable (Y) as the 'Class' column

Y = credit_card_dataset['Class']

# Display the shape of the feature set

X.shape

# Display the shape of the target variable

Y.shape
```

```
# Create a Logistic Regression model

model = LogisticRegression()

# Split the dataset into training and testing sets

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y,
random_state=42)

# Train the Logistic Regression model on the training data

model.fit(X_train, Y_train)

# Make predictions on the test data

ypred = model.predict(X_test)

# Calculate and display the accuracy of the model

accuracy_score(ypred, Y_test)


%%writefile app.py

import streamlit as st

import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score

from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score, classification_report,confusion_matrix

import matplotlib.pyplot as plt

import seaborn as sns

# Load data

data = pd.read_csv('credit_card.csv.csv')

# Separate legitimate and fraudulent transactions

legit = data[data.Class == 0]

fraud = data[data.Class == 1]

# Undersample legitimate transactions to balance the classes

legit_sample = legit.sample(n=len(fraud), random_state=2)

data = pd.concat([legit_sample, fraud], axis=0)

# Split data into training and testing sets

X = data.drop(columns="Class", axis=1)

y = data["Class"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y, random_state=42)

# Create a dictionary to store model names and their corresponding models

models = {

    "Logistic Regression": LogisticRegression(),

    "Random Forest": RandomForestClassifier(),

    "Decision Tree": DecisionTreeClassifier()

}
```

```
# Create a dictionary to store accuracy scores

accuracy_scores = {}

# Train and evaluate models

for model_name, model in models.items():

    model.fit(X_train, y_train)

    y_pred = model.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)

    accuracy_scores[model_name] = accuracy

color_map = {

    "Logistic Regression": "green",

    "Random Forest": "blue",

    "Decision Tree": "red"

}


# Streamlit app

st.title("Credit Card Fraud Detection Different Algorithms")

st.subheader("Model Comparison")

# Display accuracy scores

st.write("Model Accuracy Scores:")

st.write(accuracy_scores)
```

```
# Train logistic regression model

model = RandomForestClassifier()

model.fit(X_train, y_train)

# Evaluate model performance

train_acc = accuracy_score(model.predict(X_train), y_train)

test_acc = accuracy_score(model.predict(X_test), y_test)

st.title("Credit Card Fraud Detection Using RandomForestClassifier Model")

st.write("Enter the following features to check if the transaction is legitimate or fraudulent:")

# Create input fields for user to enter feature values

input_df = st.text_input('Input All features')

input_df_lst = input_df.split(',')

# Create a button to submit input and get prediction

submit = st.button("Submit")

if submit:

    # get input feature values

    features = np.array(input_df_lst, dtype=np.float64)

    # make prediction

    prediction = model.predict(features.reshape(1,-1))

# display result

    if prediction[0] == 0:

        st.title("Legitimate transaction")

    else:

        st.title("Fraudulent transaction")
```

```
# Plot bar chart for accuracy comparison
fig, ax = plt.subplots(figsize=(6, 2))
bars = ax.bar(accuracy_scores.keys(), accuracy_scores.values(), color=[color_map[model] for model in accuracy_scores.keys()])
ax.set_ylabel('Accuracy')
ax.set_title('Model Accuracy Comparison')
plt.xticks()
st.pyplot(fig)
# Plot line graph for accuracy comparison
fig, ax = plt.subplots(figsize=(8, 4))  # Adjust figsize as needed
for model_name, color in color_map.items():
    ax.plot(list(accuracy_scores.keys()), list(accuracy_scores.values()), marker='o', linestyle='-', color=color, label=model_name)
ax.set_ylabel('Accuracy')
ax.set_title('Model Accuracy Comparison')
ax.legend()
plt.xticks()  # Rotate x-axis labels for better readability
# Instead of st.pyplot(), pass the figure directly
st.pyplot(fig)
# Reset the Matplotlib figure after using it with Streamlit
plt.close(fig)
```

```
# Confusion Matrix Heatmaps

for model_name, model in models.items():

    st.write(f"Confusion Matrix for {model_name}:")

    y_pred = model.predict(X_test)

    cm = confusion_matrix(y_test, y_pred)

    cm_df = pd.DataFrame(cm, index=['Actual Legit', 'Actual Fraud'], columns=['Predicted Legit',
'Predicted Fraud'])

    st.write(cm_df)

# Plot confusion matrix as heatmap

    fig, ax = plt.subplots(figsize=(5, 3))

    sns.heatmap(cm_df, annot=True, cmap='Blues', fmt='d', cbar=False, ax=ax)

    ax.set_title(f'Confusion Matrix for {model_name}')

    ax.set_xlabel('Predicted')

   ax.set_ylabel('Actual')

    st.pyplot(fig)

# Histogram Graph

  st.title("Histogram for Transaction Amount")

  fig, ax = plt.subplots()

  sns.histplot(data['Amount'], bins=30, kde=True, ax=ax)

  ax.set_xlabel('Amount')

  ax.set_ylabel('Frequency')
```

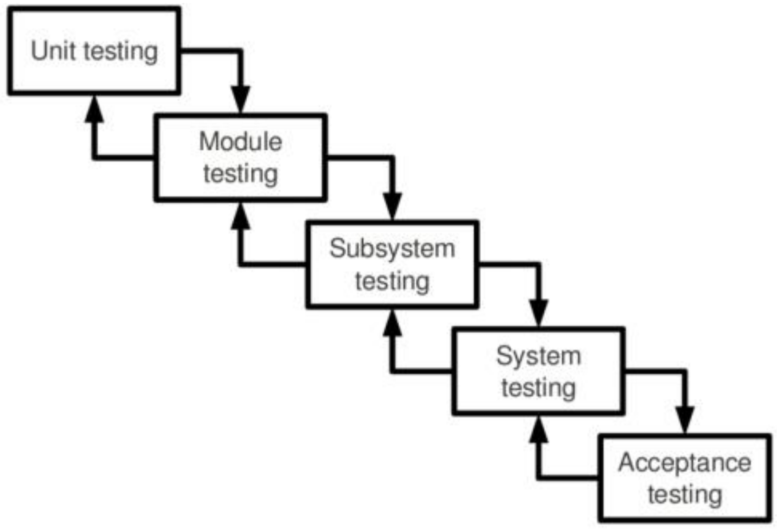
```
st.pyplot(fig)

st.write("Classification Report:")

for model_name, model in models.items():

 y_pred = model.predict(X_test)

 report = classification_report(y_test, y_pred, output_dict=True)

 report_df = pd.DataFrame(report).transpose()

 st.write(f"Classification Report for {model_name}:")

 st.dataframe(report_df.style.background_gradient(cmap='coolwarm'))


!streamlit run app.py & npx localtunnel --port 8501
```

6.TESTING

6.1 About Testing:

- Testing is the major process involved in software quality assurance (QA) is iterative Here test data is prepared and is used to test the modules individually. System testing ensures that all components or the system function - as a unit by actually forcing the system to fail.
- The test causes should be planned before testing begins. Then as the testing progresses, testing shifts focus in an attempt to find errors in integrated clusters of modules. The philosophy behind testing is to find errors. Actually, testing is the state of implementation that is aimed at ensuring that the system works actually and efficiently before implementation.
- Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. Thus, the system testing is a confirmation that all is correct and an opportunity to show the user that the system works. The final step involves Validation testing, which determines whether the software function as the user expected. The end-user rather than the system developer conduct this test most software developers as a process called "Alpha and Beta test" to uncover that only the end user seems able to find. This is the final Step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an online fashion. Here system maintenance is done every month or year based on company policies, and is checked for errors like runtime errors, long run errors and other maintenances like table verification and during the requirement analysis and design, the output is a document that is usually textual and non-executable. After the coding phase, computer programs are available that can be executed for testing purpose. This implies that testing not only has to uncover errors introduced during coding, but also errors introduced during the previous phases.

6.2 Types of Testing



The various types of testing done are:

- Unit Testing
- Integration Testing
- Validation Testing
- System Testing
- Acceptance Testing

1) Unit Testing:

Unit testing verification efforts on the smallest unit of software design, module. This is known as “Module Testing". The modules are tested separately. This testing is carried out during stage itself. In these testing Steps, each module is found to be working satisfactorily as regard to the expected output from the module.

2) Integration Testing:

Integration testing is a systematic technique for constructing tests to uncover error within the interface. In the project, all the modules are combined and then the entire is tested as a whole. In the integration-testing Step, all the error uncovered is corrected for the next testing steps.

3) Validation Testing:

To uncover functional errors, that is, to check whether functional characteristics confirm to specification or not specified is done at this phase to check the correctness.

4)System Testing:

Once individual module testing completed, modules are assembled to perform as a system. Then the top down testing, which begins from upper level to lower level module testing, to done to check whether the entire system is performing satisfactorily. After unit and integration testing are over then the system as whole is tested. There are two general strategies for system testing. They are:

- ✓ Code Testing
- ✓ Specification Testing

• **Code Testing:** This strategy examines the logic of the program. A path is a specific combination of conditions handled by the program. Using this strategy, every path through the program is tested.

• **Specification Testing:** This strategy examines the specifications stating what the program should do and how it should perform under various conditions. The test cases are developed for each condition of developed System and processed. It is found that the system developed perform according to its specified requirements. The system is used experimentally to ensure that the software will run according to tits specification and in the way user expects. Specification Testing is done successfully by entering various types of end data. It is checked for both valid and invalid data and found System is working properly as per requirement.

5)Acceptance Testing:

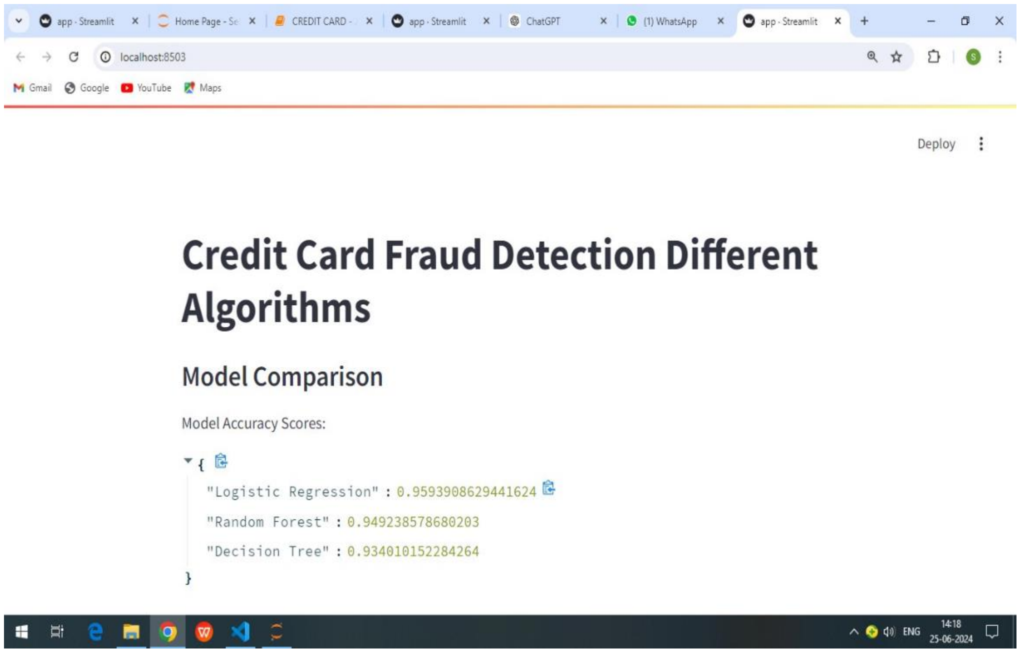
When the system has no measure problem with its accuracy, the system passes through a final acceptance test. This test confirms that the system needs the original goal, Objective and requirements established during analysis. If the system fulfils all the requirements, it is finally acceptable and ready for operation.

6)Evaluation:

- Accuracy: Accuracy is a performance metric used to evaluate the overall effectiveness of a classification model. It is defined as the ratio of correctly predicted instances (both positive and negative) to the total number of instances in the dataset. Accuracy measures the proportion of true results (both true positives and true negatives) among the total number of cases examined.
- Precision: Precision is a performance metric used to evaluate the accuracy of a classification model, particularly in the context of binary classification. It is defined as the ratio of correctly predicted positive observations to the total predicted positive observations. Precision is a measure of the exactness of a model
- Recall: Recall, also known as sensitivity or true positive rate, is a performance metric used to evaluate the accuracy of a classification model, particularly in identifying all relevant instances within a dataset. Recall measures the model's ability to correctly identify positive instances.
- F1 Score: The F1 score is a performance metric used to evaluate the accuracy of a classification model, particularly when dealing with imbalanced datasets. It is the harmonic mean of precision and recall, providing a single metric that balances the trade-off between the two.
- Support: The "support" metric in classification evaluation refers to the number of true instances for each class in the dataset. It provides the context for the calculation of precision, recall, and F1 score, showing how many actual samples of each class are present. The support metric is often presented in classification reports alongside precision, recall, and F1 score.

7. SCREENSHOT

- Home Page:



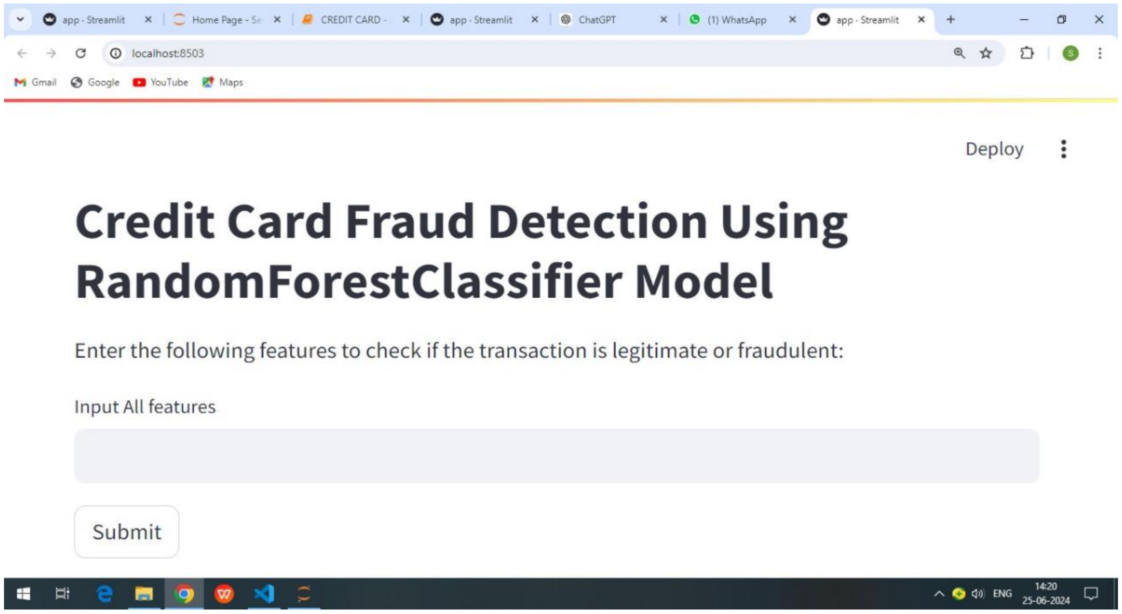
Credit Card Fraud Detection Different Algorithms

Model Comparison

Model Accuracy Scores:

```
{
  "Logistic Regression" : 0.9593908629441624
  "Random Forest" : 0.949238578680203
  "Decision Tree" : 0.934010152284264
}
```

- Entering input from the user:

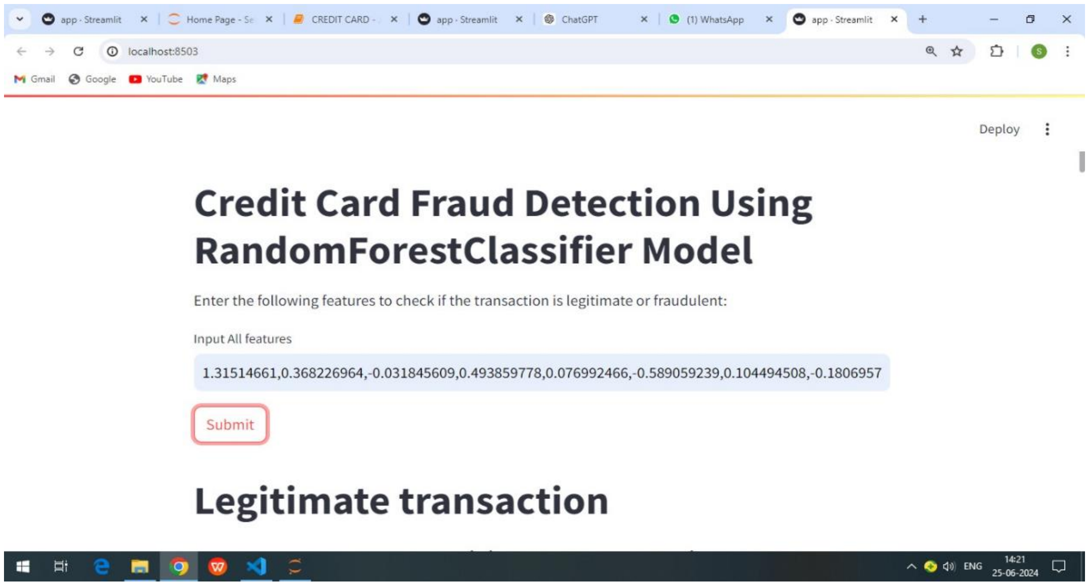


Credit Card Fraud Detection Using RandomForestClassifier Model

Enter the following features to check if the transaction is legitimate or fraudulent:

Input All features

- Checking output by giving input values:


Credit Card Fraud Detection Using RandomForestClassifier Model

Enter the following features to check if the transaction is legitimate or fraudulent:

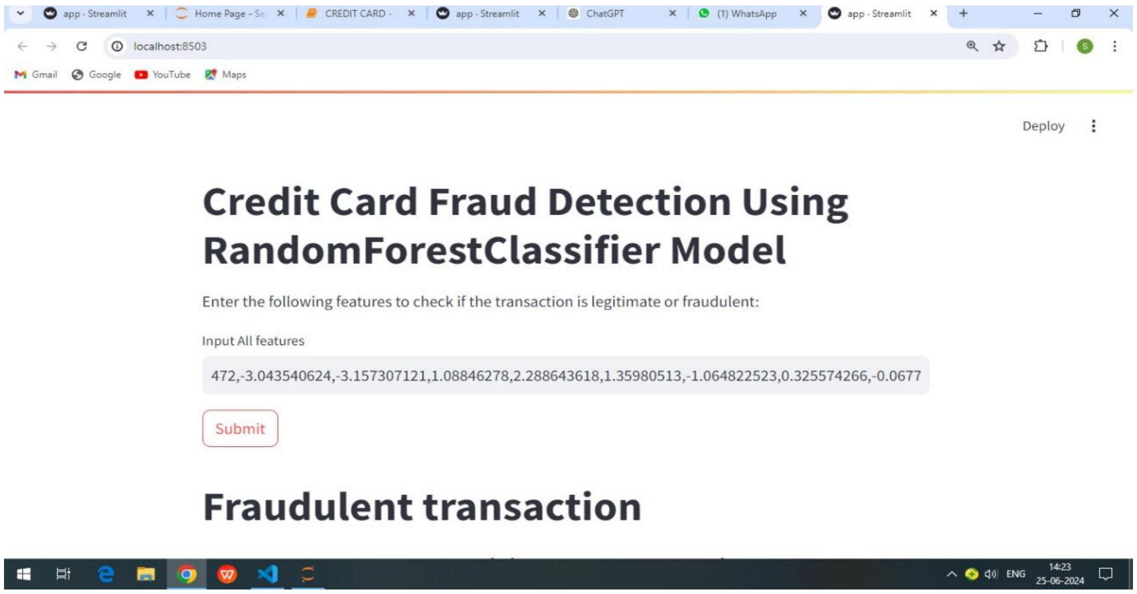
Input All features

1.31514661,0.368226964,-0.031845609,0.493859778,0.076992466,-0.589059239,0.104494508,-0.1806957

Submit

Legitimate transaction

- Checking output by giving different input:


Credit Card Fraud Detection Using RandomForestClassifier Model

Enter the following features to check if the transaction is legitimate or fraudulent:

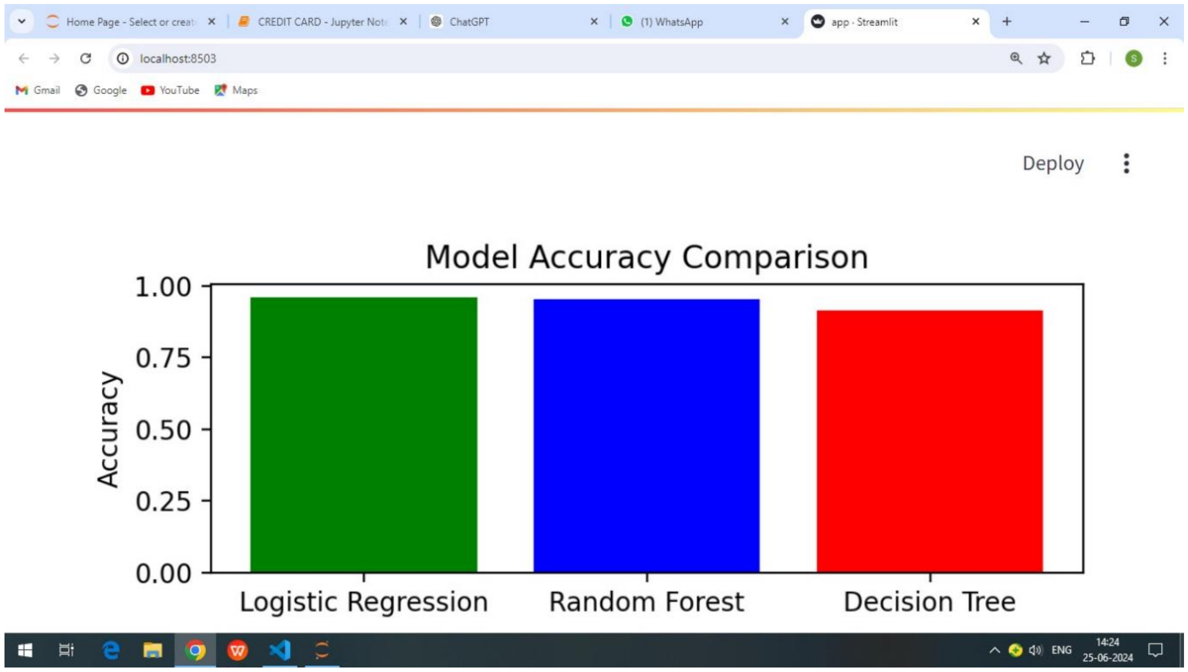
Input All features

472,-3.043540624,-3.157307121,1.08846278,2.288643618,1.35980513,-1.064822523,0.325574266,-0.0677

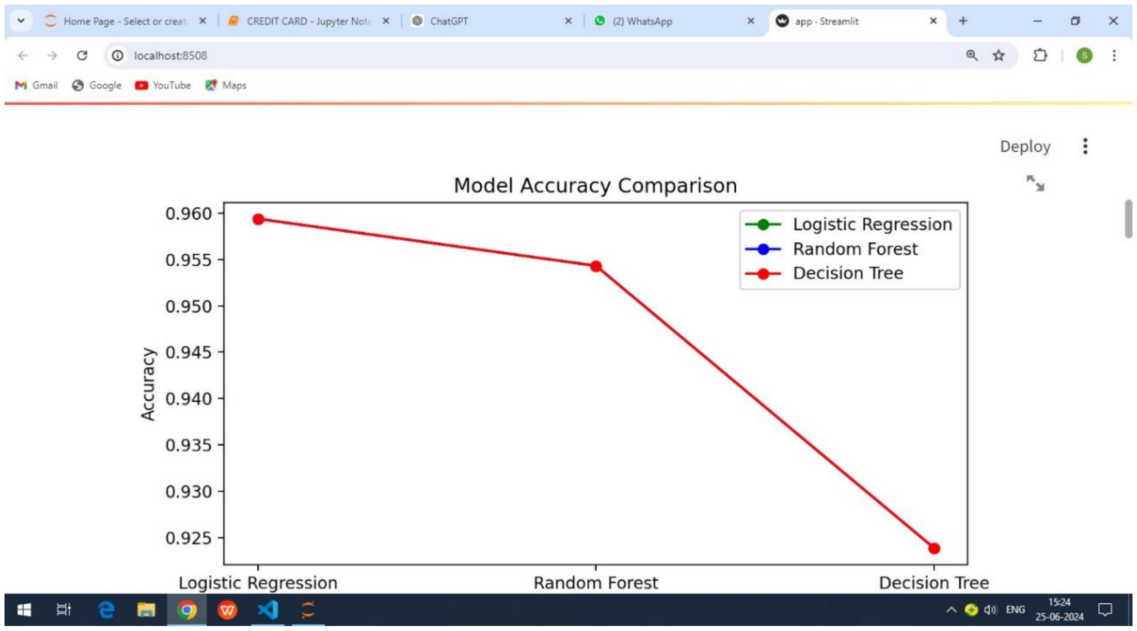
Submit

Fraudulent transaction

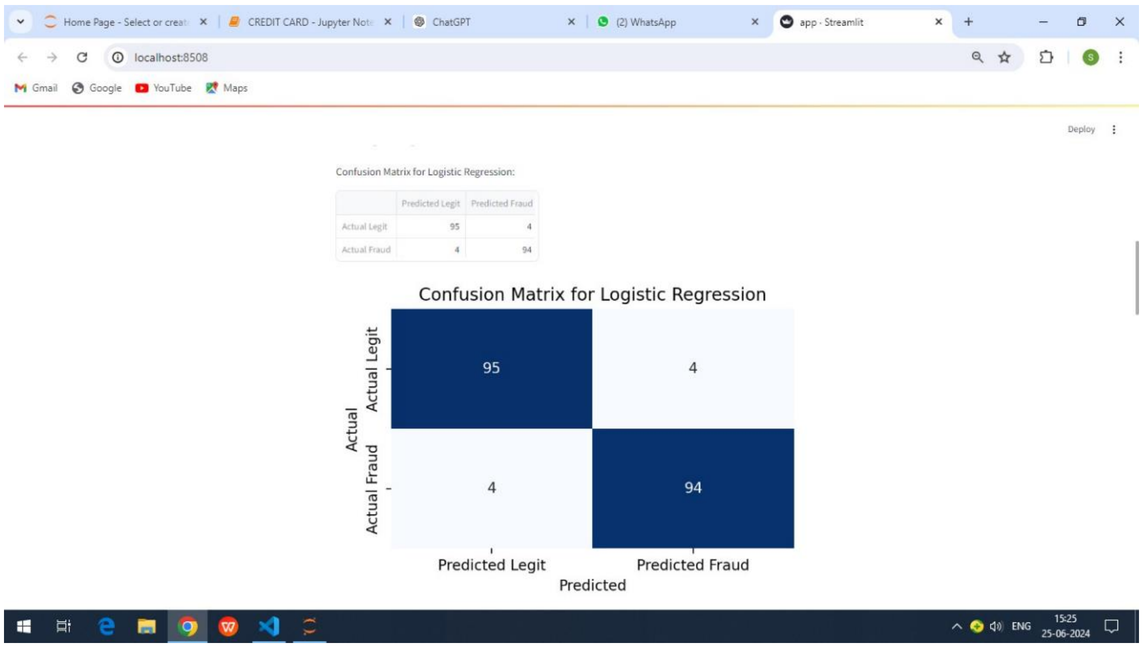
- Model accuracy comparision:



- Line Graph:



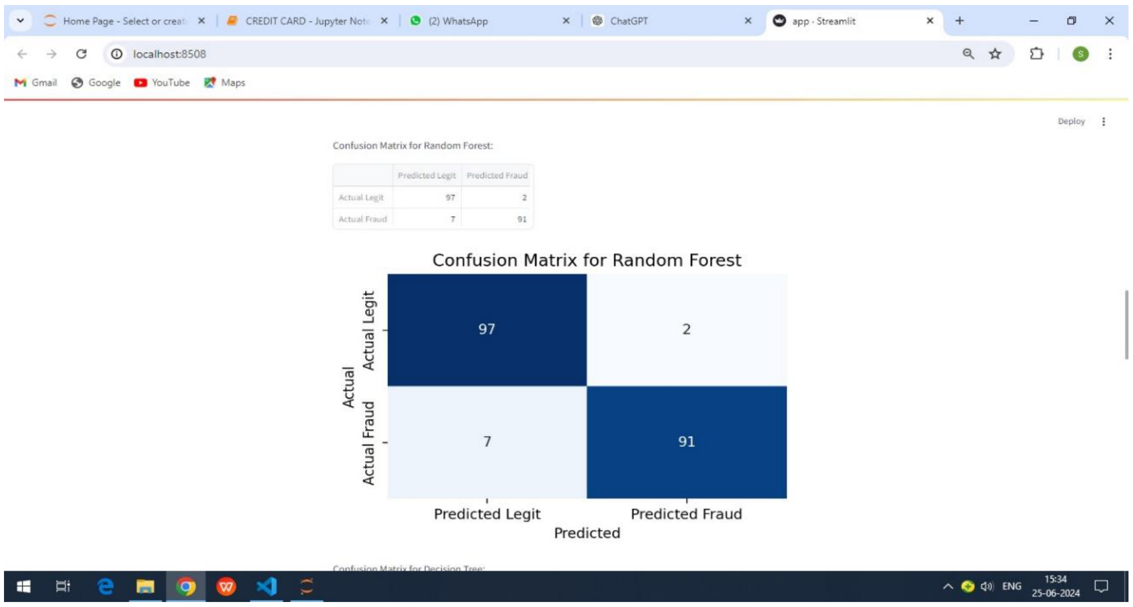
- Confusion matrix for logistic regression:



Confusion Matrix for Logistic Regression:

	Predicted Legit	Predicted Fraud
Actual Legit	95	4
Actual Fraud	4	94

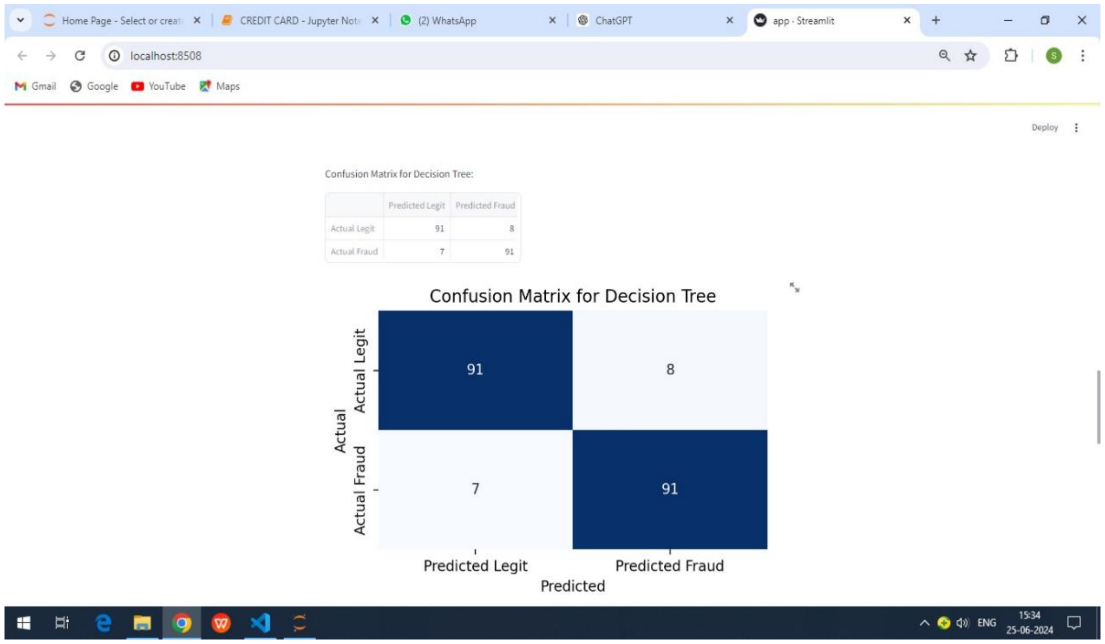
- Confusion matrix for random forest:



Confusion Matrix for Random Forest:

	Predicted Legit	Predicted Fraud
Actual Legit	97	2
Actual Fraud	7	91

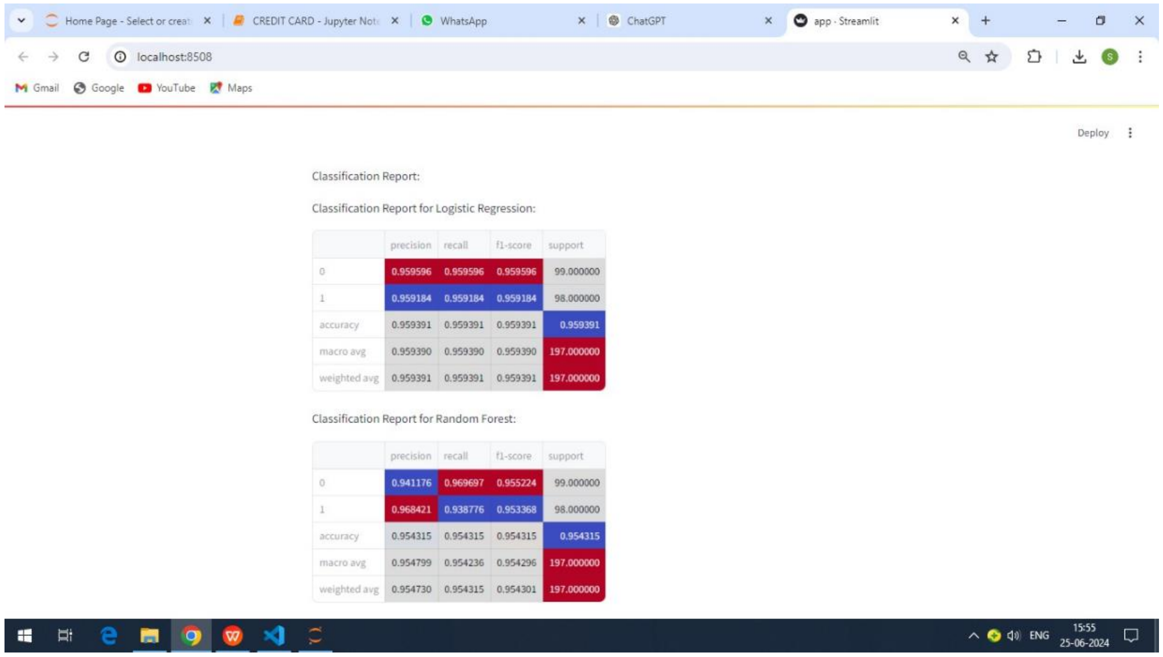
- Confusion for decision tree:



Confusion Matrix for Decision Tree:

	Predicted Legit	Predicted Fraud
Actual Legit	91	8
Actual Fraud	7	91

- Classification report:



Classification Report:

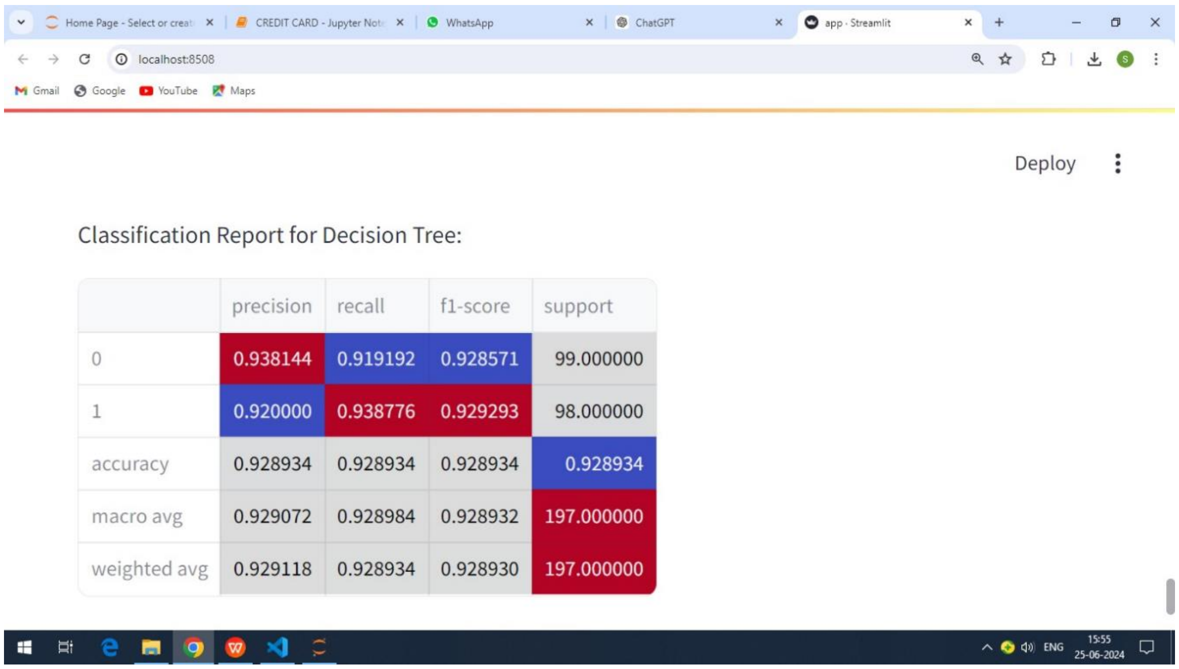
Classification Report for Logistic Regression:

	precision	recall	f1-score	support
0	0.959596	0.959596	0.959596	99.000000
1	0.959184	0.959184	0.959184	98.000000
accuracy	0.959391	0.959391	0.959391	0.959391
macro avg	0.959390	0.959390	0.959390	197.000000
weighted avg	0.959391	0.959391	0.959391	197.000000

Classification Report for Random Forest:

	precision	recall	f1-score	support
0	0.941176	0.969697	0.955224	99.000000
1	0.968421	0.938776	0.953368	98.000000
accuracy	0.954315	0.954315	0.954315	0.954315
macro avg	0.954799	0.954236	0.954296	197.000000
weighted avg	0.954730	0.954315	0.954301	197.000000

- Classification report:



Classification Report for Decision Tree:

	precision	recall	f1-score	support
0	0.938144	0.919192	0.928571	99.000000
1	0.920000	0.938776	0.929293	98.000000
accuracy	0.928934	0.928934	0.928934	0.928934
macro avg	0.929072	0.928984	0.928932	197.000000
weighted avg	0.929118	0.928934	0.928930	197.000000

8. CONCLUSION:

- Machine learning (ML) has significantly improved credit card fraud detection by enabling the development of sophisticated models that can identify fraudulent transactions with high accuracy. ML models, such as logistic regression, decision trees, random forests, can effectively learn pattern in transaction data, making them valuable tools for fraud detection.
- Machine Learning models can achieve high levels of accuracy in detecting fraudulent transactions, helping financial institutions minimize losses due to fraud.
- By reducing false positives (legitimate transactions incorrectly flagged as fraudulent), Machine Learning models can enhance the overall customer experience by minimizing disruptions to legitimate transactions.
- Credit card fraud detection is a crucial aspect of financial security, leveraging advanced technologies such as machine learning to protect consumers and financial institutions. Effective fraud detection relies high-quality data, and sophisticated anomaly detection techniques to identify suspicious activities swiftly.
- By combining multiple detection methods and understanding user behaviour, these systems can adapt to emerging fraud tactics while minimizing false positives. Collaboration between financial entities and adherence to privacy regulations are essential for robust fraud prevention.

9.FUTURE ENHANCEMENTS

- Enhancing credit card fraud detection using machine learning can significantly improve the accuracy, efficiency, and robustness of fraud detection systems. One key area of enhancement is advanced feature engineering, where new features are created from transaction data to capture user behaviour, transaction frequency, and aggregated patterns over time.
- Continuously improving feature engineering by exploring new features derived from existing ones or from external data sources can enhance model performance.
- Building ensemble models (e.g., combining predictions from multiple models like Random Forests etc.) can improve overall performance and robustness.
- Implementing advanced techniques to handle imbalanced data, such as oversampling, under sampling, or using algorithms that are inherently robust to class imbalance, can improve model performance.
- The future of credit card fraud detection will be significantly enhanced by the integration of advanced machine learning models, big data analytics, and artificial intelligence, enabling more precise and adaptable detection of fraudulent activities. Incorporating behavioural biometrics and blockchain technology will add additional layers of security, making it harder for fraudsters to alter transaction records and mimic user behaviours.

10.BIBILOGRAPHY

Reference Books:

- **Machine Learning:** Srikanth S, Skyward Publishers
- **Python:** Srikanth S, Skyward Publishers

Websites:

- www.stackoverflow.com
- www.youtube.com
- www.kaggle.com
- www.w3school.com
- www.bootstrap.com
- www.googlechrome.com