

## ***Table of Content Page***

Chapter-1 Introduction.....	1
Chapter-2 Technology and Concepts.....	2
Chapter-3 Methodology .....	3
Chapter-4 Result and Discussion.....	9
Chapter-5 Future Enhancements.....	12
Chapter-6 Conclusion.....	13
Chapter-7 Certificate.....	14

## *List Of Figures*

3.1 Machine learning process.....	1
3.1.1.1 Importing Required Library and Dataset.....	1
3.1.1.2 No occurrence of Missing Values for Dataset.....	2
3.1.1.3 Distribution of Dataset Showing the Class imbalance.....	2
3.1.1.4 Distribution of Dataset after balancing.....	4
3.1.2 Feature Extraction.....	4
3.1.3.1 Importing Required Algorithms.....	5
3.1.3 Testing Algorithms.....	6
3.1.4 Data Visualization.....	6
3.1.6 Streamlit Hosted Website .....	7
4.1.1.1 Localhost Output 1.....	8
4.1.1.2 Localhost Output 2.....	8
4.1.2.2 Streamlit Output 1.....	10
4.1.2.2 Streamlit Output 2.....	10
7.1 Training Completion Certificate.....	14

# *Chapter 1*

## *Introduction*

The rise of digital transactions and electronic payments has brought unprecedented convenience to the modern world. However, with this convenience comes the challenge of combating fraudulent activities in the financial sector, particularly in credit card transactions.

Credit card fraud is a pervasive issue that affects both financial institutions and cardholders, leading to substantial monetary losses and undermining trust in the financial system.

In light of these challenges, I am made this Credit Card Fraud Detection project, which serves as the focus of my internship. This project aims to develop a robust and accurate fraud detection system that can identify potentially fraudulent credit card transactions in real-time.

By leveraging advanced machine learning techniques, I intend to create a solution that not only detects fraud but also minimizes false positives, ensuring a seamless experience for legitimate cardholders.

### **1.1 Dataset:**

- The dataset for this project is taken from Kaggle which is an online platform containing various datasets.
- The link for the dataset is: <https://www.kaggle.com/code/janiobachmann/credit-fraud-dealing-with-imbalanced-datasets/input>
- This dataset includes many of credit card transactions, each characterized by a range of features such as time of transaction and other information.
- Additionally, the dataset provides labels indicating whether each transaction is legitimate or fraudulent, making it suitable for supervised machine learning.

### **1.2 Problem Statement:**

- The core challenge of this project revolves around building a predictive model that can effectively distinguish between genuine and fraudulent transactions.
- The main problem this project involves evaluating multiple machine learning algorithms on credit card dataset and optimizing their hyperparameters to identify the algorithm that achieves the highest predictive accuracy.
- This exploration aims to determine the most suitable approach for solving Credit Card Fraud problem using machine learning techniques.

## ***Chapter 2***

### ***Technology and Concepts***

#### **2.1 Machine Learning:**

- Machine learning is a method of data analysis that automates analytical model building.
- It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.
- It automates tasks, enhances decision-making, and enables personalization.
- Machine Learning algorithm learns from experience E with respect to some type of task T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.
- It enables computers to learn and improve from experience without explicit programming.
- ML models make predictions or decisions based on data patterns.

It can be used in two ways:

##### ***2.1.1 Supervised Learning***

Supervised learning is a machine learning paradigm where an algorithm is trained on labeled data, learning to make predictions or classify inputs based on past examples provided in the training set.

##### ***2.1.2 Unsupervised Learning***

Unsupervised learning is a machine learning approach where algorithms analyze unlabeled data to discover hidden patterns, clusters, or structures without explicit guidance from predefined outcomes or labels.

#### **2.2 Random Forest Algorithm**

- A Random Forest is a powerful ensemble learning technique in machine learning.
- It combines multiple decision trees, each trained on a random subset of the training data, and aggregates their predictions to make more accurate and robust predictions.
- This ensemble approach helps mitigate overfitting, resulting in a more generalized model.
- They also handle missing data and outliers well, making them robust in real-world scenarios.
- Additionally, Random Forests provide a feature importance ranking, helping users identify the most influential features in their data.
- Due to their reliability and effectiveness, Random Forests are widely used in various domains, from finance to healthcare and beyond.

# Chapter 3

## Methodology

The project involved developing a Streamlit web application for credit card fraud detection, encompassing all steps of the machine learning pipeline. The application simplifies data preprocessing, model training, and deployment, ensuring user-friendly access to a robust fraud detection system for credit card transactions.

### 3.1 The process of Machine learning:

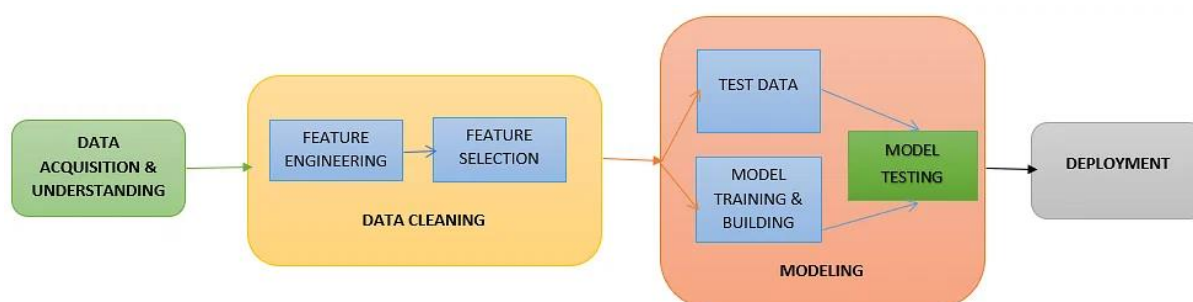


Fig 3.1: Machine learning process ([source](#))

#### 3.1.1 Data Preprocessing:

Preparing and preprocessing the dataset to ensure it is suitable for training and testing machine learning models. This involves tasks such as data cleaning, feature engineering, and handling class imbalance.

##### 3.1.1.1 Importing Dataset:

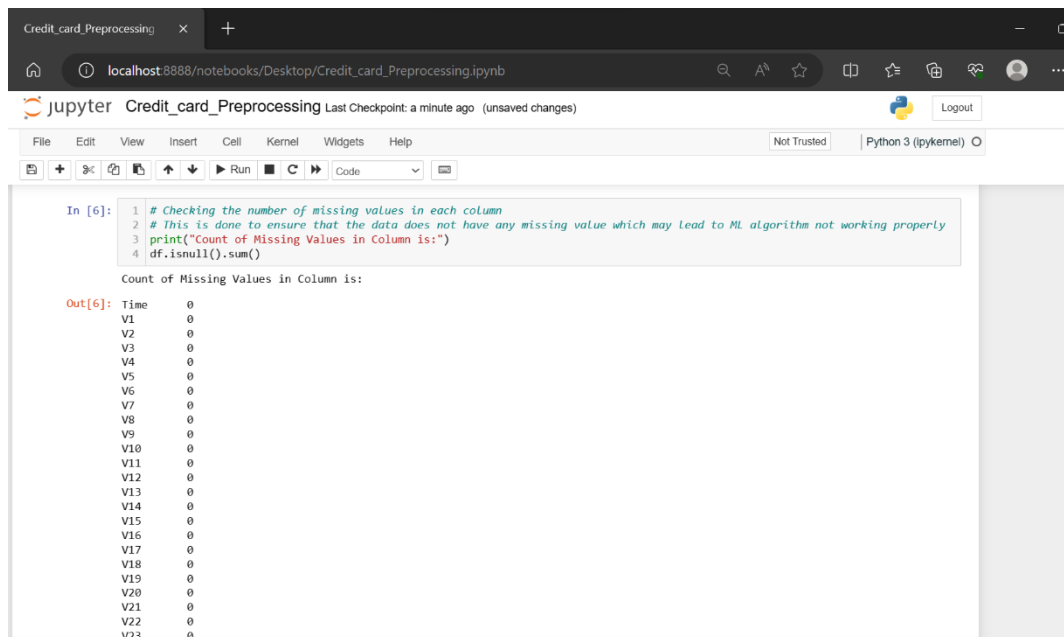
```
jupyter Credit_card_Preprocessing Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)
In [2]: 1 # Importing required libraries
        2 import pandas as pd

Data Gathering
Now we need to load the dataset.
This is done using the pd.read_csv function.
Reading csv file.
When we have a lot of data(Bigger dataset) processing takes time.
We can use nrows(Parameter for number of rows) to select required number of rows as the dataset is very big.
In [3]: 1 # Importing Dataset/CSV file
        2 df = pd.read_csv("creditcard.csv")
        3 print("Dataset is:\n",df)

Dataset is:
   Time  V1  V2  V3  V4  V5 \
0  0.0 -1.359807 -0.072781 2.536347 1.378155 -0.338321
1  0.0  1.191857  0.266151 0.166480 0.448154  0.060018
2  1.0 -1.358354 -1.340163 1.773209 0.379780 -0.503198
3  1.0 -0.966272 -0.185226 1.792993 -0.863291 -0.010309
4  2.0 -1.158233  0.877737 1.548718  0.403034 -0.407193
```

Fig 3.1.1.1: Importing Required Library and Dataset for Data Preprocessing

### 3.1.1.2 Seeing the Missing Data:



```
1 # Checking the number of missing values in each column
2 # This is done to ensure that the data does not have any missing value which may lead to ML algorithm not working properly
3 print("Count of Missing Values in Column is:")
4 df.isnull().sum()

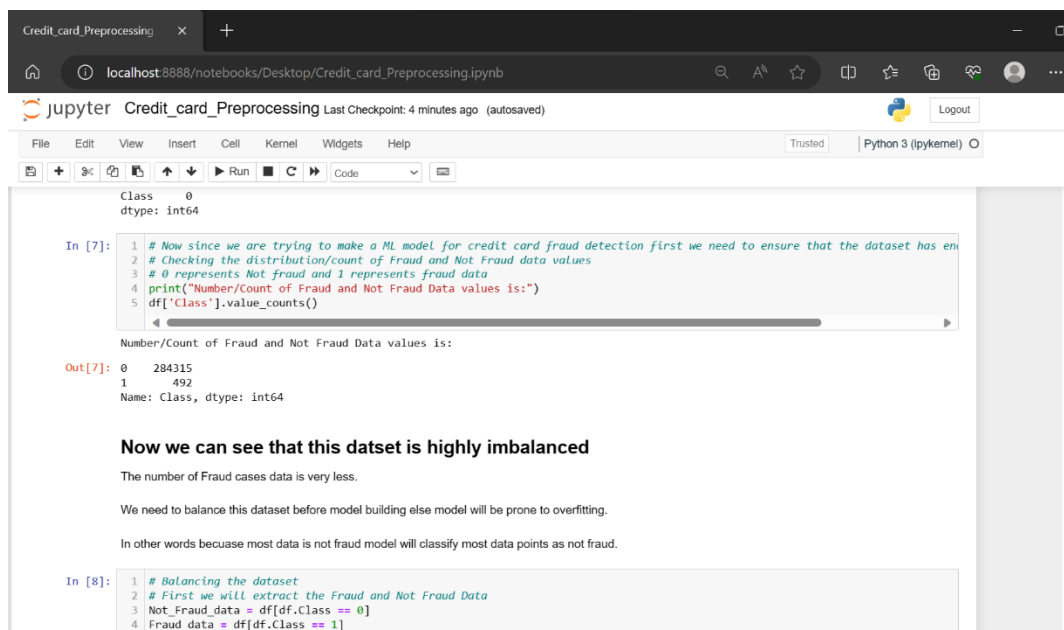
Count of Missing Values in Column is:

Out[6]: Time      0
        V1        0
        V2        0
        V3        0
        V4        0
        V5        0
        V6        0
        V7        0
        V8        0
        V9        0
        V10       0
        V11       0
        V12       0
        V13       0
        V14       0
        V15       0
        V16       0
        V17       0
        V18       0
        V19       0
        V20       0
        V21       0
        V22       0
        V23       0
```

*Fig 3.1.1.2: No occurrence of Missing Values for Dataset*

We can see that this dataset does not contain missing data.

### 3.1.1.3 Seeing the Dataset Distribution:



```
Class      0
dtype: int64

In [7]: 1 # Now since we are trying to make a ML model for credit card fraud detection first we need to ensure that the dataset has enough data
        2 # Checking the distribution/count of Fraud and Not Fraud data values
        3 # 0 represents Not fraud and 1 represents fraud data
        4 print("Number/Count of Fraud and Not Fraud Data values is:")
        5 df['Class'].value_counts()

Number/Count of Fraud and Not Fraud Data values is:

Out[7]: 0    284315
        1      492
        Name: Class, dtype: int64

Now we can see that this dataset is highly imbalanced

The number of Fraud cases data is very less.

We need to balance this dataset before model building else model will be prone to overfitting.

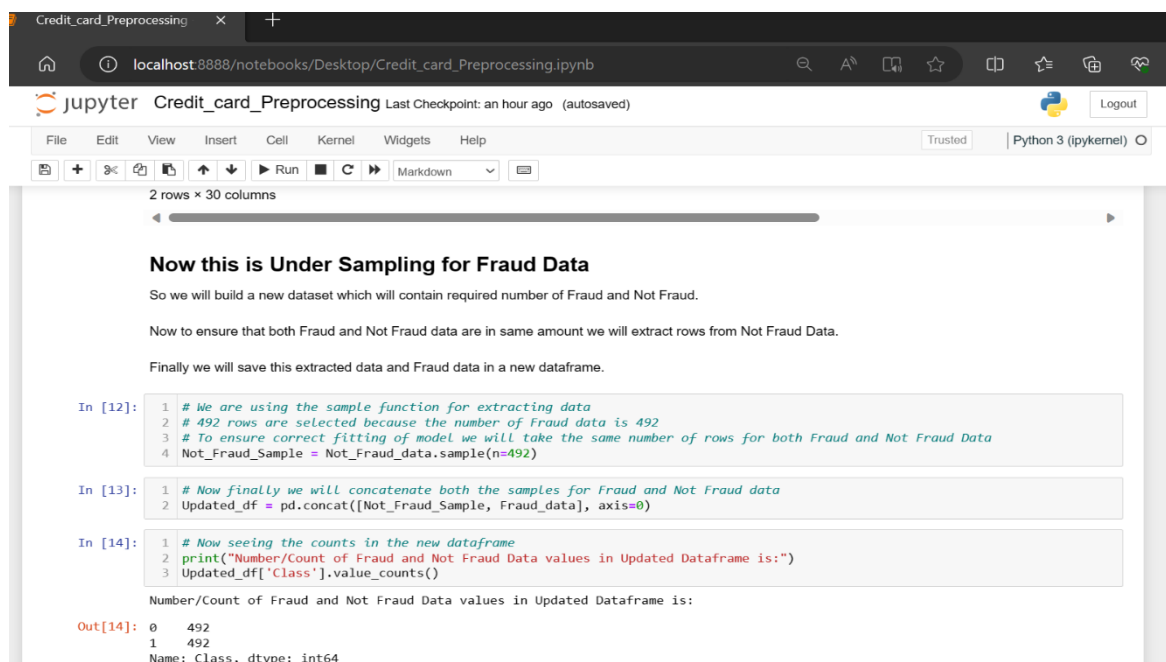
In other words because most data is not fraud model will classify most data points as not fraud.

In [8]: 1 # Balancing the dataset
        2 # First we will extract the Fraud and Not Fraud Data
        3 Not_Fraud_data = df[df.Class == 0]
        4 Fraud_data = df[df.Class == 1]
```

*Fig 3.1.1.3: Distribution of Dataset Showing the Class imbalance*

We can see that there is a huge class imbalance in this dataset so for this we extracted data for both the classes and then we are going to create a new dataframe which will contain equal distribution and then this will serve as the data for our model.

### 3.1.1.4 Balancing the Dataset Class Distribution:



```
Credit_card_Preprocessing x +
localhost:8888/notebooks/Desktop/Credit_card_Preprocessing.ipynb
jupyter Credit_card_Preprocessing Last Checkpoint: an hour ago (autosaved)
Python 3 (ipykernel)
File Edit View Insert Cell Kernel Widgets Help Trusted
2 rows x 30 columns

Now this is Under Sampling for Fraud Data
So we will build a new dataset which will contain required number of Fraud and Not Fraud.
Now to ensure that both Fraud and Not Fraud data are in same amount we will extract rows from Not Fraud Data.
Finally we will save this extracted data and Fraud data in a new dataframe.

In [12]: 1 # We are using the sample function for extracting data
2 # 492 rows are selected because the number of Fraud data is 492
3 # To ensure correct fitting of model we will take the same number of rows for both Fraud and Not Fraud Data
4 Not_Fraud_Sample = Not_Fraud_data.sample(n=492)

In [13]: 1 # Now finally we will concatenate both the samples for Fraud and Not Fraud data
2 Updated_df = pd.concat([Not_Fraud_Sample, Fraud_data], axis=0)

In [14]: 1 # Now seeing the counts in the new dataframe
2 print("Number/Count of Fraud and Not Fraud Data values in Updated Dataframe is:")
3 Updated_df['Class'].value_counts()

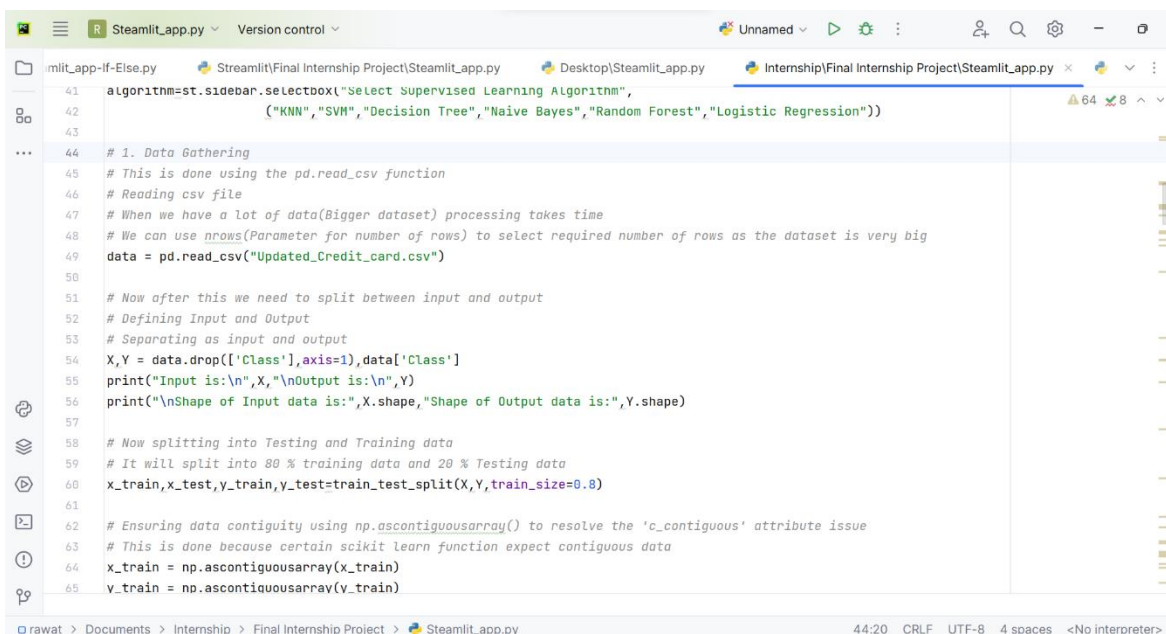
Number/Count of Fraud and Not Fraud Data values in Updated Dataframe is:
Out[14]: 0    492
1    492
Name: Class, dtype: int64
```

Fig 3.1.1.4: Distribution of Dataset after balancing

Now in the updated dataframe we have equal number of both Fraud and Not Fraud data and this will help avoid overfitting. This dataframe is saved as csv file which is the dataset for our model.

### 3.1.2 Feature Extraction:

Now our text target is to find the Features for Input and Output in the model.



```
Steamlit_app.py Version control
Steamlit(Final Internship Project)Steamlit_app.py Desktop\Steamlit_app.py Internship\Final Internship Project\Steamlit_app.py
41 algorithm=st.sidebar.selectbox("Select Supervised Learning Algorithm",
42                               ("KNN", "SVM", "Decision Tree", "Naive Bayes", "Random Forest", "Logistic Regression"))
43
44 # 1. Data Gathering
45 # This is done using the pd.read_csv function
46 # Reading csv file
47 # When we have a lot of data(Bigger dataset) processing takes time
48 # We can use nrows(Parameter for number of rows) to select required number of rows as the dataset is very big
49 data = pd.read_csv("Updated_Credit_card.csv")
50
51 # Now after this we need to split between input and output
52 # Defining Input and Output
53 # Separating as input and output
54 X,Y = data.drop(['Class'],axis=1,data['Class'])
55 print("Input is:\n",X,"\nOutput is:\n",Y)
56 print("\nShape of Input data is:",X.shape,"Shape of Output data is:",Y.shape)
57
58 # Now splitting into Testing and Training data
59 # It will split into 80 % training data and 20 % Testing data
60 x_train,x_test,y_train,y_test=train_test_split(X,Y,train_size=0.8)
61
62 # Ensuring data contiguity using np.ascontiguousarray() to resolve the 'c_contiguous' attribute issue
63 # This is done because certain scikit learn function expect contiguous data
64 x_train = np.ascontiguousarray(x_train)
65 y_train = np.ascontiguousarray(y_train)
```

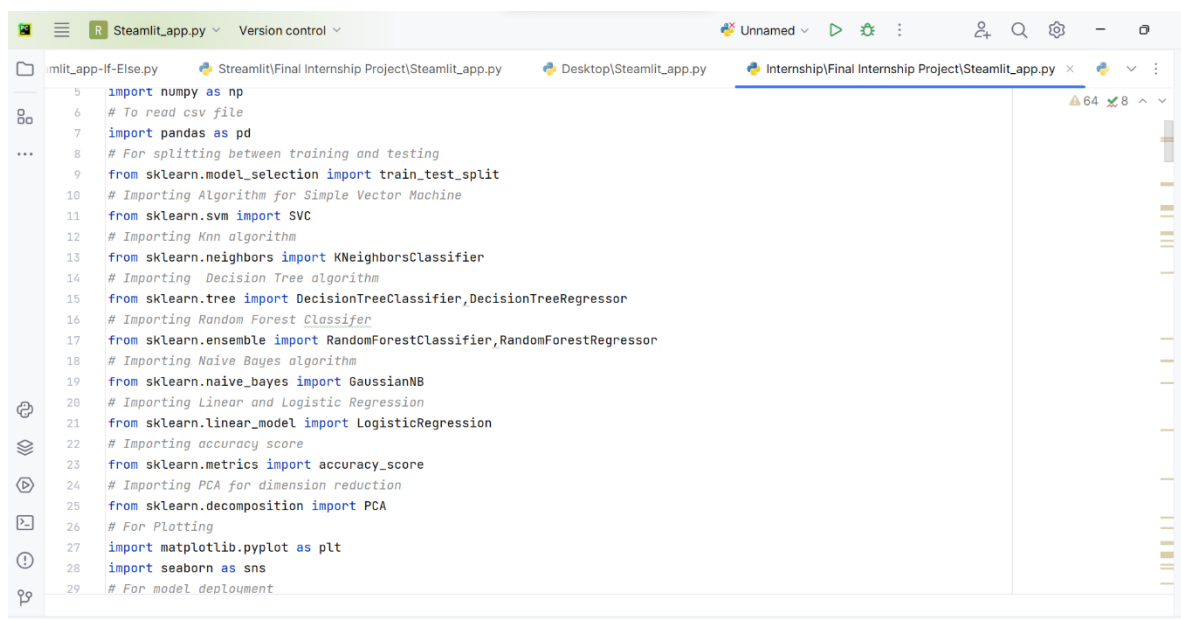
Fig 3.1.2: Feature Extraction

Now we have extracted features for input and output and also splitted testing and training data. After Splitting into training and testing data, to ensure compatibility we also convert to numpy contiguous array to ensure stability and compatibility with scikit-learn functions. The next task is model development or choosing algorithm.

### 3.1.3 Model/Algorithm Development:

#### 3.1.3.1 Exploratory Data Analysis:

- After feature extraction several supervised learning algorithm are applied in this dataset and the highest accuracy is given by random forest algorithm.
- Thus Random Forest algorithm works best for this dataset.



*Fig 3.1.3.1: Importing Required Algorithms*

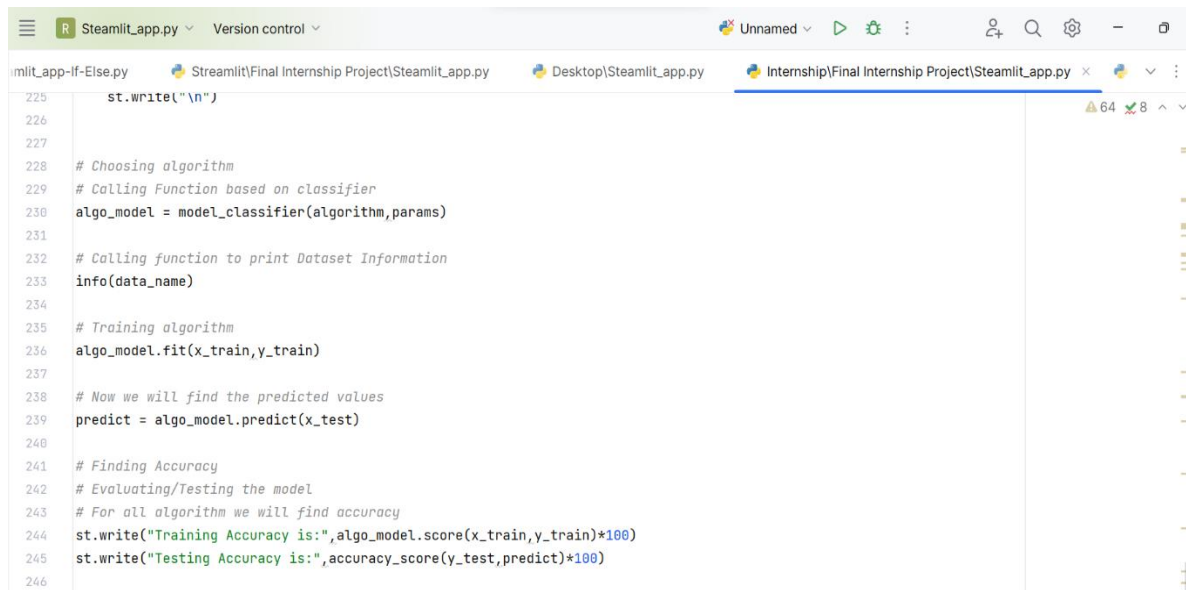
#### 3.1.3.2 Hyperparameter Tuning:

- Hyperparameter tuning is the process of optimizing the settings or configuration of a machine learning model to enhance its performance.
- This involves systematically adjusting hyperparameters, such as learning rates, regularization strengths, and tree depths, through techniques like grid search or random search to find the best combination that yields the highest accuracy or minimizes the loss function.
- Hyperparameter tuning is essential for fine-tuning model behavior and achieving optimal results in various machine learning tasks.
- In this project I have created a streamlit app in which we can see the performance of supervised learning algorithms in this dataset and also tune parameters and see the effect on overall accuracy.



### 3.1.3 Model/Algorithm Testing:

Finally we check the training and testing accuracy for our model for all datasets.

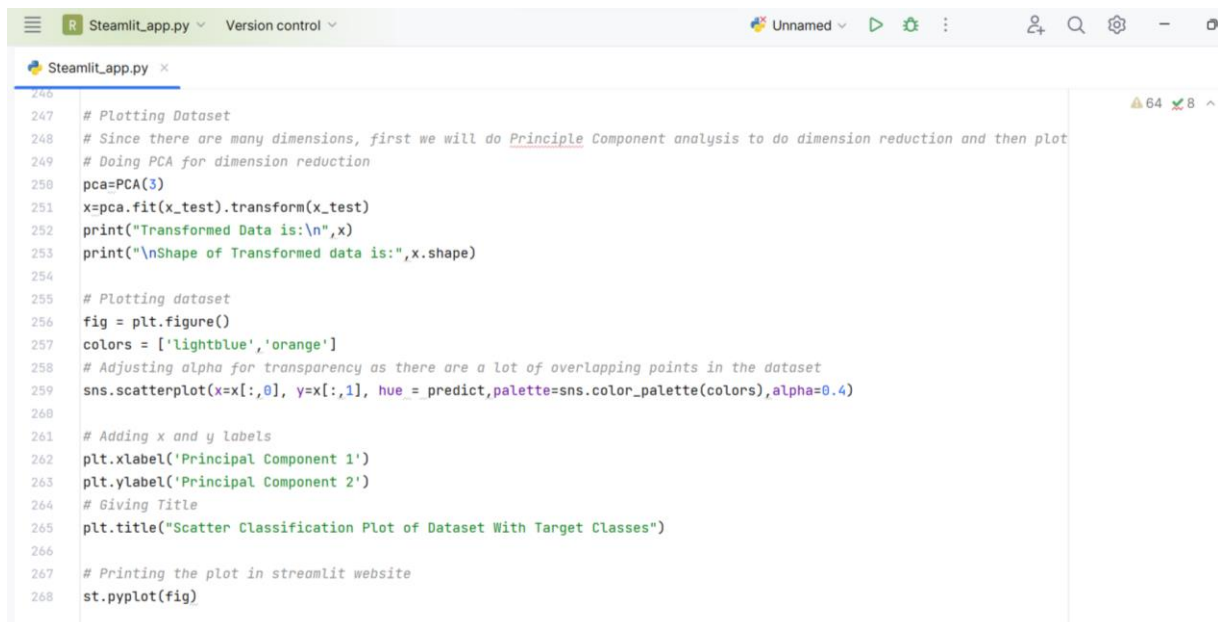
A screenshot of a code editor window with multiple tabs. The active tab is 'Steamlit\_app.py'. The code is in Python and includes comments for each step: choosing an algorithm, calling a classifier function, printing dataset info, training the model, finding predicted values, and calculating accuracy. The code is as follows:

```
225 st.write("\n")
226
227
228 # Choosing algorithm
229 # Calling Function based on classifier
230 algo_model = model_classifier(algorithm,params)
231
232 # Calling function to print Dataset Information
233 info(data_name)
234
235 # Training algorithm
236 algo_model.fit(x_train,y_train)
237
238 # Now we will find the predicted values
239 predict = algo_model.predict(x_test)
240
241 # Finding Accuracy
242 # Evaluating/Testing the model
243 # For all algorithm we will find accuracy
244 st.write("Training Accuracy is:",algo_model.score(x_train,y_train)*100)
245 st.write("Testing Accuracy is:",accuracy_score(y_test,predict)*100)
246
```

*Fig 3.1.3: Testing Algorithms*

The highest accuracy is given by Random Forest.

### 3.1.4 Data Visualization:

A screenshot of a code editor window with a single tab 'Steamlit\_app.py'. The code is in Python and includes comments for each step: plotting the dataset, performing PCA for dimension reduction, and creating a scatter plot. The code is as follows:

```
247 # Plotting Dataset
248 # Since there are many dimensions, first we will do Principle Component analysis to do dimension reduction and then plot
249 # Doing PCA for dimension reduction
250 pca=PCA(3)
251 x=pca.fit(x_test).transform(x_test)
252 print("Transformed Data is:\n",x)
253 print("\nShape of Transformed data is:",x.shape)
254
255 # Plotting dataset
256 fig = plt.figure()
257 colors = ['lightblue','orange']
258 # Adjusting alpha for transparency as there are a lot of overlapping points in the dataset
259 sns.scatterplot(x=x[:,0], y=x[:,1], hue = _predict,palette=sns.color_palette(colors),alpha=0.4)
260
261 # Adding x and y labels
262 plt.xlabel('Principal Component 1')
263 plt.ylabel('Principal Component 2')
264 # Giving Title
265 plt.title("Scatter Classification Plot of Dataset With Target Classes")
266
267 # Printing the plot in streamlit website
268 st.pyplot(fig)

```

*Fig 3.1.4: Data Visualization*

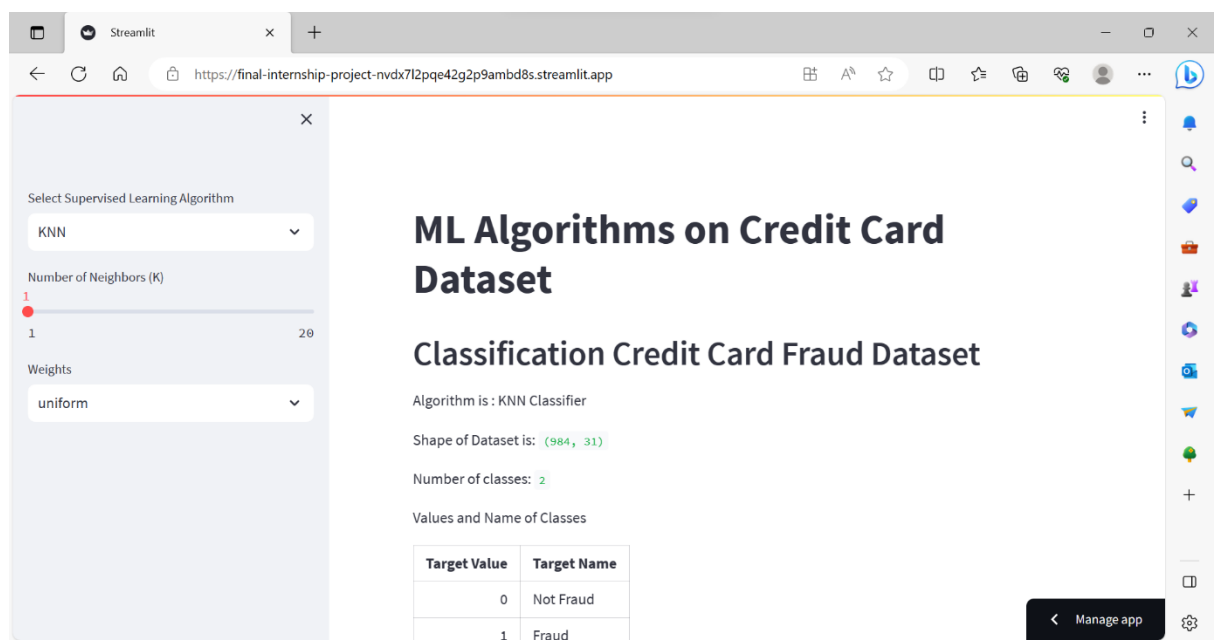
First Dimensionality reduction is done on dataset and then final plotting is done for all algorithms and varying parameters.

### 3.1.5 Deployment:

- The final project deployment is done using streamlit.
- In the streamlit app we can see all algorithms and their accuracy on the dataset along with the data visualization according to the parameters and algorithm.
- We can also see the working of various algorithms on this dataset by plotting scatter plot for the algorithm and dataset.

### 3.1.6 Hosting with GitHub and Streamlit:

- Streamlit is an open-source Python library that simplifies the creation of web applications for Datascience and ML.
- I have also hosted this app using GitHub and streamlit and it can be seen at: [Streamlit \(final-internship-project-nvdx7l2pqe42g2p9ambd8s.streamlit.app\)](https://final-internship-project-nvdx7l2pqe42g2p9ambd8s.streamlit.app)
- Since the repository is private code cannot be seen.
- But anyone with a public URL can access this app.



*Fig 3.1.6: Streamlit Hosted Website*

- For successful deployment we first ensure that it runs successfully in our local machine
- After running it in our local machine we can deploy it using GitHub and Streamlit.
- All the code files are added in the GitHub Repository.
- One requirement file is made with the extension .txt which contains the name of all required libraries or dependencies.
- This file is also uploaded in repository.
- Finally after using all this files, we can host our app in streamlit.

# Chapter-4

## Result and Discussion

### 4.1 Results:

#### 4.1.1 Localhost:

After running the app using Streamlit we can see the output as follows on the localhost:

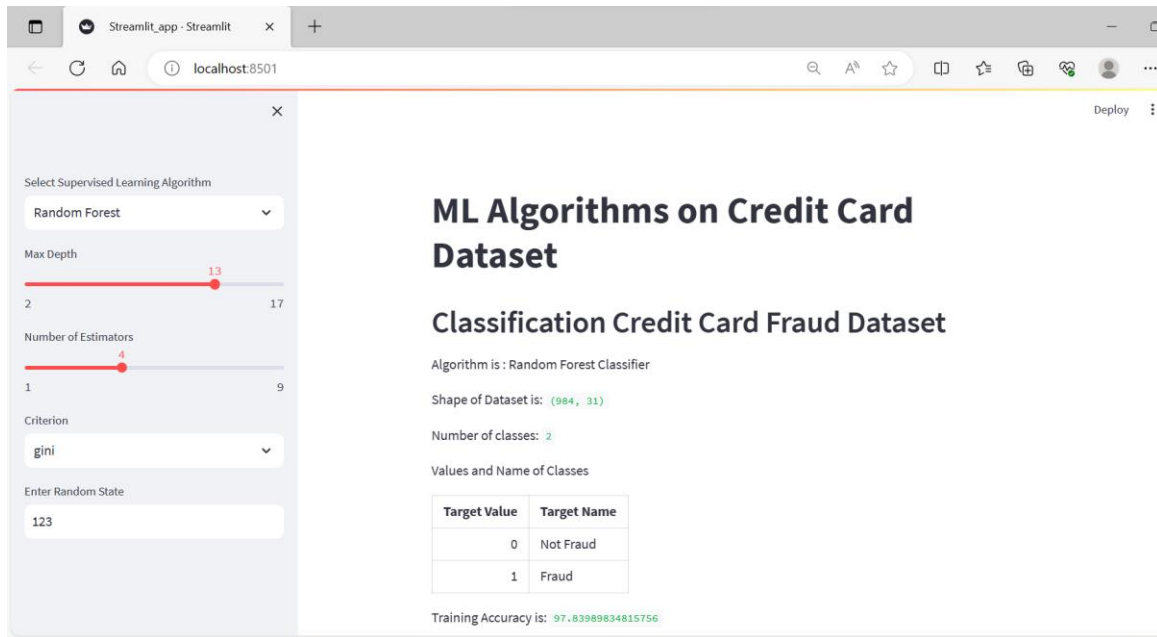


Fig 4.1.1.1: Localhost Output 1

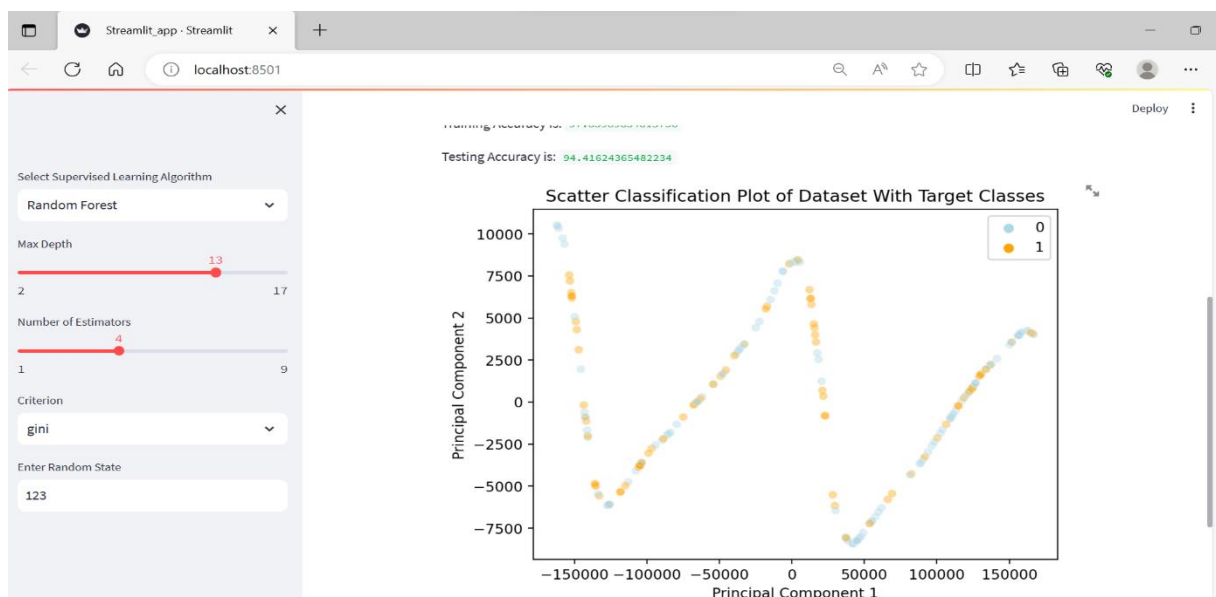


Fig 4.1.1.2: Localhost Output 2

## 4.1.2 Streamlit Server:

After visiting the link we can see the output as follows on the browser:

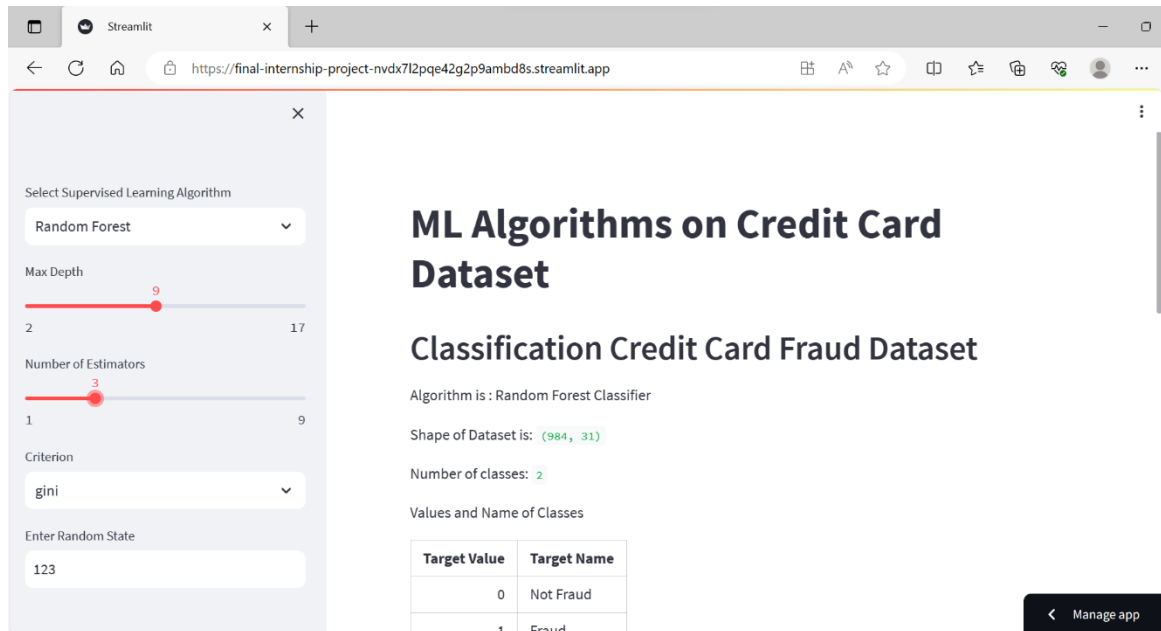


Fig 4.1.2.1: Streamlit Output 1

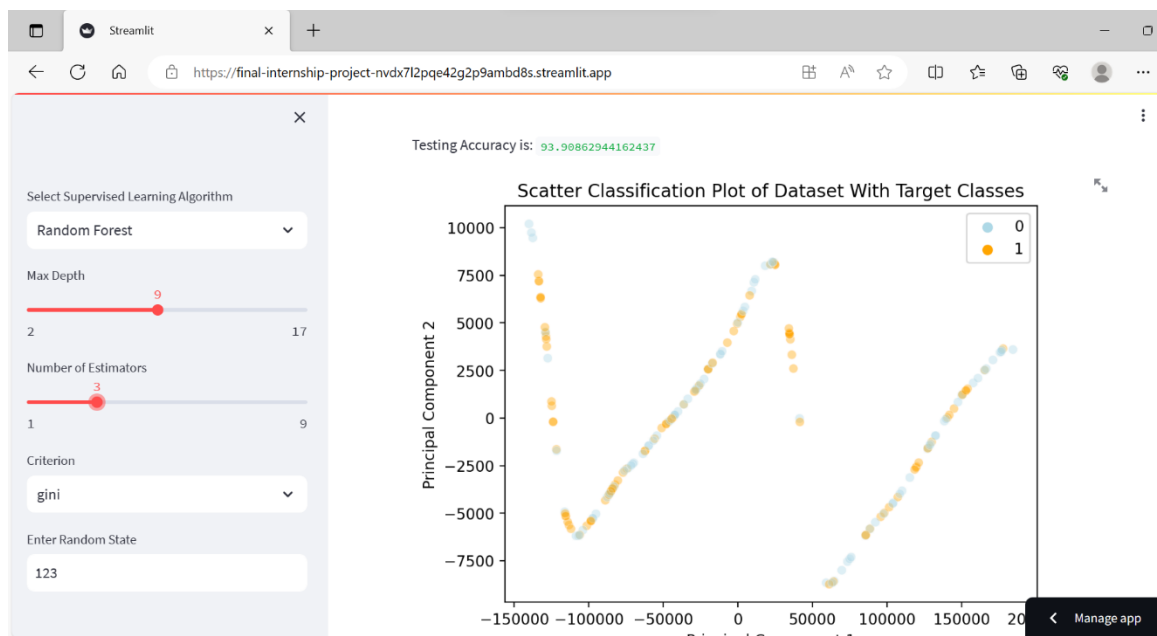


Fig 4.1.2.2: Streamlit Output 2

## 4.2 Discussion:

The overarching aim of this project is to create a fraud detection system that excels in accurately identifying potentially fraudulent credit card transactions while minimizing false positives. Achieving this delicate balance is essential for preserving customer trust while shielding financial institutions from substantial monetary losses.

Credit card fraud is a pervasive problem with far-reaching consequences. It impacts not only financial institutions but also cardholders, resulting in substantial monetary losses and eroding trust in the financial system.

Fraudulent transactions can inflict severe financial hardships on individuals, tarnish the reputation of financial institutions, and contribute to elevated fees and interest rates for consumers.

Consequently, the development of effective countermeasures against credit card fraud is crucial in today's digital economy.

The selection of an appropriate dataset is paramount for training and evaluating the fraud detection system.

In this project, the dataset from Kaggle furnishes a diverse array of credit card transactions. Each transaction is characterized by a multitude of features, including transaction time, transaction amount, and several anonymized attributes.

Critically, this dataset includes labels that indicate whether each transaction is legitimate or fraudulent, thus facilitating a supervised machine learning approach. The real-world nature of this dataset ensures that the system's performance is tested against the intricacies and nuances of actual transactions.

The Credit Card Fraud Detection project is focused on creating an effective real-time fraud detection system using advanced machine learning techniques. Utilizing a dataset sourced from Kaggle, which includes labeled credit card transactions, the project's primary goal is to accurately differentiate between genuine and fraudulent activities.

Future enhancements may involve the incorporation of deep learning models to further improve detection accuracy and explore anomaly detection methods for added sophistication.

Additionally, ongoing monitoring and adaptation will be essential to address the ever-evolving strategies employed by fraudsters, thus ensuring the ongoing security and trustworthiness of the financial system.

## *Chapter – 5*

### *Future Enhancements*

As this project unfolds, several avenues for future enhancements emerge, promising to further elevate the effectiveness of the fraud detection system:

1. **Deep Learning Integration:** The incorporation of deep learning models, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), holds the potential to enhance the model's capacity to detect subtle patterns in transaction data, potentially yielding even higher accuracy.
2. **Anomaly Detection:** The exploration of anomaly detection techniques, including unsupervised learning approaches like Isolation Forests and One-Class SVMs, could prove invaluable in identifying unusual or suspicious transactions that may not adhere to typical patterns.
3. **Continuous Monitoring:** The tactics employed by credit card fraudsters are in a perpetual state of evolution. To effectively combat new fraud strategies, it will be imperative to implement a system for continuous monitoring and model retraining.
4. **Ethical Considerations:** With the increasing utilization of AI in finance, ethical concerns such as privacy and fairness need to be thoughtfully addressed. Future enhancements should incorporate robust measures to ensure compliance with privacy regulations and uphold model fairness.

The Credit Card Fraud Detection project constitutes a proactive response to a pressing issue in the financial sector. The ongoing proliferation of digital transactions necessitates robust and adaptable solutions for combatting fraudulent activities effectively.

By harnessing advanced machine learning techniques and a meticulously curated dataset, this project is poised to make a substantial contribution to bolstering the security and trustworthiness of the financial system.

As it evolves, the integration of deep learning, anomaly detection, continuous monitoring, and ethical considerations will further enhance its capabilities and impact.

## *Chapter – 6*

### *Conclusion*

My vocational training experience with Diginique Techlabs has been both enriching and rewarding. Throughout my time at this dynamic project, I have had the opportunity to gain valuable insights into the world of Machine Learning and Datascience.

This experience has allowed me to apply the theoretical knowledge gained during our academic journey and witness its practical implications in a real-world setting. Working on this Credit Card Fraud Detection project has exposed me to a wide array of responsibilities, working and designing of real world machine learning applications.

Additionally, this vocational training has given me a deeper appreciation for the complexities and challenges involved in making of Machine Learning models and deployment of this model. I had the privilege of learning from experienced mentors and colleagues who have provided invaluable guidance and support throughout this internship.

As I submit this vocational training report, I want to express my gratitude to the Diginique Techlabs for their unwavering encouragement and trust in my abilities.

This project marks my contribution in the realm of data science and machine learning. By leveraging the power of Streamlit, I successfully developed an interactive and user-friendly web application for data analysis and visualization.

The application's ease of use and accessibility make it a valuable tool for both experts and non-technical users, facilitating data-driven decision-making.

The meticulous data preprocessing, feature engineering, and model selection processes laid the foundation for robust machine learning models.

Hyperparameter tuning improved model performance, and careful evaluation provided critical insights into model behavior.

Deploying the application ensures its practical utility, and ongoing monitoring and maintenance will ensure its continued relevance and accuracy.

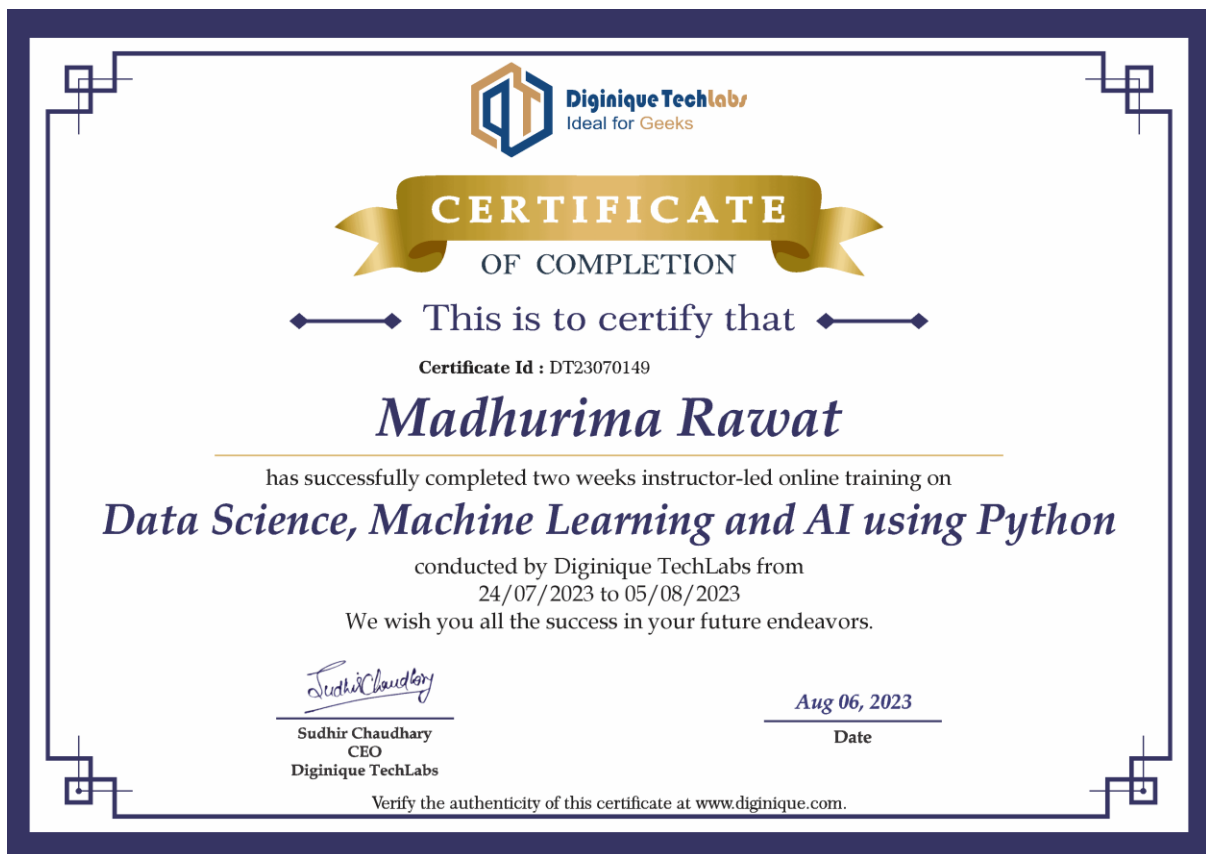
Overall, this project demonstrates the synergy between machine learning and user-friendly interfaces, enabling effective data exploration and knowledge dissemination.

In conclusion, this Vocational Training has been a significant step forward in my professional development, and I look forward to applying the lessons learned here as I embark on our future career. Thank you for this incredible opportunity.

## Chapter – 7

### Certificate

Enclosed is my training certificate, demonstrating my dedication to continuous learning and expertise in **Data Science, Machine Learning and AI using Python**, reinforcing the quality and reliability of this report.



*Fig 7.1: Training Completion Certificate*



## *Appendix*

### **Objectives:**

The objectives of the Credit Card Fraud Detection model span various phases, from initial design to deployment:

- **Model Design:** Careful selection of machine learning algorithms and the creation of a robust fraud detection model.
- **Hyperparameter Tuning:** Extensive fine-tuning of model hyperparameters to maximize its effectiveness.
- **Real-Time Deployment:** Successful deployment of the model to ensure practical use.

### **Technology Stack:**

The technology stack used in this project ensured a seamless user experience and efficient deployment:

- **Streamlit Framework:** Streamlit was chosen for the front-end user interface, guaranteeing a responsive and dynamic user experience.
- **Python:** Python served as the core programming language for model development, preprocessing, and deployment.

### **Features:**

The “Credit Card Fraud Detection” model incorporates an array of features to enhance its functionality and accuracy:

- **Algorithm Diversity:** Implementation of multiple machine learning algorithms, including logistic regression, random forests, and gradient boosting, to assess their performance.
- **Hyperparameter Tuning:** Rigorous tuning of hyperparameters to optimize each algorithm's fraud detection capabilities.
- **Parameter Sensitivity Analysis:** Exploration of how varying algorithm parameters impacts fraud detection accuracy and overall performance.

### **Preprocessing:**

Preprocessing of the dataset was a pivotal step to ensure its compatibility with various machine learning models:

- **Data Cleaning:** Thorough data cleaning processes were executed to address missing values and outliers.

- **Feature Engineering:** The dataset underwent feature engineering (mainly Feature Extraction) to enhance its predictive power.
- **Scaling and Transformation:** Data scaling and transformation techniques were applied to normalize features and improve model performance.

## Testing:

Testing played a crucial role in validating the model's reliability and accuracy:

- **Model Validation:** Comprehensive validation to assess the model's ability to detect both legitimate and fraudulent transactions.
- **Performance Metrics:** Utilization of performance metrics such as accuracy score for thorough evaluation.

## Future Enhancements:

For future iterations of the Credit Card Fraud Detection model, several enhancements and features could be considered:

- **Advanced Algorithms:** Exploration of advanced machine learning and deep learning algorithms to further enhance fraud detection accuracy.
- **Ensemble Methods:** Implementation of ensemble learning techniques to combine the strengths of multiple algorithms.
- **Anomaly Detection:** The exploration of anomaly detection techniques, including unsupervised learning approaches like Isolation Forests and One-Class SVMs, could prove invaluable in identifying unusual or suspicious transactions that may not adhere to typical patterns.
- **Continuous Monitoring:** The tactics employed by credit card fraudsters are in a perpetual state of evolution. To effectively combat new fraud strategies, it will be imperative to implement a system for continuous monitoring and model retraining.
- **Ethical Considerations:** With the increasing utilization of AI in finance, ethical concerns such as privacy and fairness need to be thoughtfully addressed.

## Certificate:

In this section, my training certificate is attached, confirming my commitment to continuous learning and proficiency in Data Science, Machine Learning, and AI using Python, which enhances the credibility of this report.

The comprehensive appendix offers detailed insights into the Credit Card Fraud Detection project, covering objectives, technology stack, features, preprocessing, testing, and future enhancements. Overall, this project involves precise algorithm selection, hyperparameter tuning, Streamlit-based user interface, data preprocessing, rigorous testing, and ongoing improvements.

## ***Project Resources***

**Machine learning Process:** <https://www.datacamp.com/tutorial/tutorial-machine-learning-pipelines-mlops-deployment>

**Dataset:** <https://www.kaggle.com/code/janiobachmann/credit-fraud-dealing-with-imbalanced-datasets/input>

### **Deployment:**

Credit Card Fraud Detection App: You can access the deployed Credit Card Fraud Detection application by clicking:

<https://final-internship-project-nvdx7l2pqe42g2p9ambd8s.streamlit.app/>

### **GitHub Repository:**

Codes related to this project can be seen at my GitHub Repository:

GitHub Repository: <https://github.com/madhurimarawat/Final-Internship-Project>