**Credit Card Fraud Detection Using Machine Learning Algorithms**



Hari Krishna Jammula (harikrishnajammula@my.unt.edu)

[Kavya Vedantham](https://unt.instructure.com/groups/141850/users/343637) (kavyavedantham@my.unt.edu)

[Hepzibah Ravuri](https://unt.instructure.com/groups/141850/users/325339) (HepzibahRavuri@my.unt.edu)

[Naga Pavan Bonam](https://unt.instructure.com/groups/141850/users/343822) (NagaPavanBonam@my.unt.edu)

[Deepak Gogikar](https://unt.instructure.com/groups/141850/users/343858) (DeepakGogikar@my.unt.edu)

**Department of Information Science**

**University of North Texas, Denton, Texas**

**INFO 5502 - Principles and Techniques for Data Science**

**April 20, 2023**

**Participants and Respective Roles**

Hari Krishna Jammula - 11607185

Roles: Code Development, GitHub & Evaluation and StreamLIT

Kavya Vedantham - 11640837

Roles: Code Development, Exploratory Data Analysis and Model Prediction

Hepzibah Ravuri - 11601187

Roles: Code Development, Feature Engineering, Algorithm Selection and Model Prediction

Naga Pavan Bonam - 11638923

Roles: Project Documentation, Dataset Selection, References from Existing Projects

Deepak Gogikar - 11641199

Project Documentation, Project PPT, Dataset Selection, References from Existing Projects

**Workflow**

Our team meets during after-class hours and when everybody is available on campus to discuss the ideas, progress, and upcoming agendas that we need to work on. We also use Google Meets for online team communication to share our work and make changes in any code or documentation related to the project. We have also created a group UNT-Email id for communication and sharing purposes. Moreover, we use the GitHub platform for software project storage, tracking, and collaboration. It allows us to easily share code files and collaborate with other team members on project code building. GitHub also functions as a social networking site, where we can freely network, collaborate, and pitch our work.

**Abstract**

Overall, authentication is a critical aspect of information security, as it plays a crucial role in protecting against external threats, ensuring the integrity and accountability of information, and enabling authorized access to sensitive systems and data. Credit card fraud detection is currently the most common problem in the modern world. This is due to an increase in online transactions as well as e-commerce platforms. According to the Nilson Report, global losses from card fraud are expected to total $397.4 billion over the next ten years, with $165.1 billion of those losses occurring in the United States. Credit card fraud occurs when a card is stolen and used for unauthorized purposes, or when the fraudster uses the credit card information for his own benefit.

The main goal of our project is to create a user-friendly interface where people can detect any transactions which were not made by them, and which makes them capable of finding out if any deceptive transactions are going on with their credit cards. This will allow people to be more aware of any mis-happenings in their credit cards and be able to protect their money. By implementing this we hope to make a good effort in helping society as a lot of duplicitous activities are going on in this world.

In real time, a machine learning model can detect any deviations from regular transactions and user behaviors. ML algorithms can reduce the risk of fraud and ensure more secure transactions by detecting anomalies such as a sudden increase in transactional amount or location change. We think that by using the power of data science and machine learning, we can effectively and efficiently assist people and businesses in achieving 100% fraud-free transactions.

**Data Specification**

The dataset used was from Kaggle user Dhanush Narayanan R, a CSV dataset containing over 10 Lakh legitimate and fraud credit card transactions. The data set has 8 features, each standing for one aspect of the distance from home, distance from last transaction, ratio to median purchase price, repeat retailer, used chip, used pin number, online order and fraud. Fraud is the target variable. We have approximately 9% of fraud transactions in our data.

Using this dataset, our team has attempted to solve supervised problems and find the best algorithm for our dataset of fraud detection and predict the accuracy of the models.

Feature Explanation:

* distance\_from\_home: It is the distance from home to where the transaction happened which is in a float data type.
* distance\_from\_last\_transaction: It is a float data type which describes the distance from last transaction happened.
* ratio\_to\_median\_purchase\_price: It is the ratio which is the float data type and tells the ratio of purchased price transaction to median purchase price.
* repeat\_retailer: This feature is a float data type, and it tells if the transactions were repeated from the same retailer, where 0.0 being the transactions are not repeated at the same retailer and 1.0 being the vice versa.
* used\_chip: It tells if the transaction is through chip (credit card), through a float data type which is 1.0 for yes and 0.0 for no
* used\_pin\_number: this tells us if the transaction happened by using a PIN number, and is a float data type, 0.0 for no and 1.0 for a yes.
* online\_order: Float data type which tells if it is an online transaction, where 1.0 is yes and 0.0 is no.
* fraud: Is the transaction fraudulent or not which is denoted by float data type and 1.0 is yes, it is a fraud and 0.0 means no it is not a fraud.

**Table

Description automatically generated**

*Dataset description from Streamlit UI*

*Chart, pie chart

Description automatically generated*

*Fraudulent transaction percentage in the dataset from Streamlit UI*

**Project Design**

The primary goal of our project is to concentrate on machine learning algorithms. The random forest algorithm, Decision tree algorithm and the logistic regression algorithm were used. The accuracy, precision, recall, and F1-score of the three algorithms are used to calculate the results. The Random Forest, Decision tree algorithm and Logistic regression algorithms are compared, and the algorithm with the highest accuracy, precision, recall, and F1-score are considered the best fraud detection algorithm.

**Overview of the tools and frameworks:**

* Visual Studio Code – Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.
* Python – Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation via the off side rule
* Git Hub - GitHub, Inc. is an Internet hosting service for software development and version control using Git.
* Lucid Chart - Lucidchart is a web-based diagramming application that allows users to visually collaborate on drawing, revising, and sharing charts and diagrams, and improve processes, systems, and organizational structures.
* Scikit-learn - Scikit-learn is a popular machine learning library in Python that offers a range of tools for data preprocessing, feature selection, and model selection. It provides a variety of algorithms for classification, including logistic regression, decision trees, and random forests, which can be used for fraud detection.
* Streamlit - Open-source framework for building web applications in Python. It provides a range of tools for creating interactive data visualizations and dashboards, which can be used for visualizing the results of fraud detection models.

**Flowchart of Design**

The flow of steps involved from the start which is collecting the dataset to the making of the final UI is in the below screenshot. We used Licud Chart to create this diagram for better pictorial descriptions and easy implementation.

Firstly, we collected the credit card fraud dataset and preprocessed the data which includes data cleaning and preparation of it for analysis. Exploratory Data Analysis is done after it, where we visualize and analyze the data on a deeper level. Next, we train and test the data set through various regressions and Machine Learning models. After the model training, we evaluate and find the results and build the UI.

Diagram

Description automatically generated

*flowchart of data processing and design from LucidChart*

**Data Pre-processing**

Data pre-processing, which comprises cleaning, manipulating, and preparing data for analysis, is an important stage in machine learning. It seeks to verify that the data used in machine learning models is correct, consistent, and relevant, with the goal of improving model performance.

We performed missing value and duplicate removal, as well as null value identification and removal, in order to determine the most significant features that contribute to fraud detection from the dataset. In our dataset, we have zero missing values, and zero duplicate values.

Graphical user interface, application

Description automatically generated

*Missing values in the dataset from Streamlit UI*

**Exploratory Data Analysis (EDA)**

EDA aims to find patterns, connections, and other interesting elements in the data that might not be readily obvious from a quick glance at the raw data. To evaluate and comprehend data, EDA uses both quantitative and visual methods. In order to find patterns and relationships in the data, visual approaches use charts, graphs, and other visual representations. In quantitative methods, the data are summarized and described using statistical techniques, such as computing measures of central tendency, dispersion, and correlation.

The correlation matrix analysis is an essential part of EDA that shows the correlation values, which measure the degree of linear relationship between each pair of variables. The correlation values can fall between -1 and +1. If the two variables tend to increase and decrease together, the correlation value is positive.

We are doing Exploratory Data Analysis on our credit card fraud dataset, and finding the count of the kind of transactions that took place and we are also comparing the statistics of fraud and non-fraud with other features.

Chart, bar chart

Description automatically generated

*Visualization for the count of different transaction types from Streamlit UI*

Chart, pie chart

Description automatically generated

*Distinguishing the transaction type by percentage and class from Streamlit UI*

Chart, pie chart

Description automatically generated

*Distinguishing the transaction type by percentage and class from Streamlit UI*

Chart

Description automatically generated

*Correlation matrix of the dataset from Streamlit UI*

**Modeling**

Once we have our dataset and features ready after clearing null values and unnecessary features, the next step is to select a machine learning model that can accurately predict whether a transaction is fraudulent or not. So we used classification models for fraud detection which are Logistic regression, Decision trees, Random forests.

After selecting the three machine learning models, we split the dataset into training and testing dataset. Fit the  model using the training data, and use the model to make predictions on the testing data, and evaluate the performance of the model. We evaluated the metrics precision, recall, and F1 scores to know the accuracy of the ML model results.

We have compared the performance of these three models and chosen the one that gave us the best results.

**Model Evaluation**

By observing the results of the models, based on the values of precision, recall, f1-score and support, the accuracy of the logistic regression classifier on test data is 0.95800. Similarly, the accuracy of the decision tree on test data is 0.9999 and random forest classifier on the test data is 1.0000 respectively.

The results showed that while comparing other metrics like recall, f1-score and precision random forest has the better results as compared to logistic regression and decision tree models.

**Timeline

Description automatically generated**

*Logistic regression results and its accuracy from Streamlit UI*

**Table

Description automatically generated**

*Decision Tree regression results and its accuracy from Streamlit UI*

**Table, timeline

Description automatically generated**

*Random Forest results and its accuracy from Streamlit UI*

**Model Deployment**

Stremlit Integration

* Integrate the trained machine learning model into a Streamlit web application. Using Streamlit to create an interactive user interface that allows users to input credit card transactions and receive a prediction on whether the transaction is fraudulent or not.
* And we deploy the Streamlit web application to our localhost and which gives a proper User-Interface for end-users. And by monitoring the performance of the deployed model to ensure that it is working correctly and providing accurate predictions.

**Project Results**

In this paper we developed a model for credit card fraud detection using machine learning that resulted in high accuracy, precision, recall, and F1 scores using three different algorithms - decision tree, logistic regression, and random forest.

After training and testing the models on a dataset containing credit card transactions, the results showed that the random forest algorithm achieved the highest accuracy and F1 score as compared with other machine learning models (decision tree and Logistic regression model).

The conclusions drawn from this project are that machine learning algorithms, particularly random forest, can effectively detect credit card fraud with high accuracy and precision. However, it is important to note that the choice of algorithm may depend on the specific needs and resources of the organization using the model.

If given more time and/or data, further improvements to the models could be made. For example, additional features could be added to the dataset, or the hyperparameters of the algorithms could be tuned to optimize performance. Additionally, the models could be tested on new and unseen data to ensure their generalizability and robustness. And the finest possible user interface for the end user that interacts with our model via an application or a website.

**Project Milestone**

The project milestone was achieved when we successfully built and trained three machine learning algorithms, namely decision tree, logistic regression, and random forest, to detect credit card fraud. The algorithms were trained on a dataset of credit card transactions and evaluated on their accuracy, precision, recall, and f1 score.

The project milestone was reached when we identified the best-performing algorithm, random forest, and concluded the project by suggesting possible improvements to further enhance the performance of the algorithms. On top of that, we use streamlit to construct a user interface for our fraud detection machine learning model which was a unique way to represent from other referred projects.

**Repository / Archive**

Our project is available on GitHub, where it may be readily collaborated with in teams and by individuals interested in credit card fraud detection.

<https://github.com/HariKrishnaJammula/Credit-Card-Fraud-Detection-using-Machine-Learning.git>

The link above provides everyone with access to our code, dataset, and Project report, which they can use as a reference.

**Appendix – Code**

import streamlit as st

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Title of the app

st.title("Credit Card Fraud Detection Using Machine Learning Algorithms")

# Upload CSV file

uploaded\_file = st.file\_uploader("Upload your input CSV file", type=["csv"])

# Check if a file has been uploaded

if uploaded\_file is not None:

    # Read the file

    data = pd.read\_csv(uploaded\_file)

    # Show the first 5 rows of the data

    st.write("First 5 rows of the data:")

    st.write(data.head())

    # Show the shape of the data

    st.write("Shape of the data:")

    st.write(data.shape)

    # Show the data types of the columns

    st.write("Data types of the columns:")

    st.write(data.dtypes)

    # Show the summary statistics of the data

    st.write("Summary statistics of the data:")

    st.write(data.describe())

    # Show the missing values in the data

    st.write("Missing values in the data:")

    st.write(data.isnull().sum())

    # Show the correlation matrix of the data

    st.write("Correlation matrix of the data:")

    st.write(data.corr())

 # Plot scatterplot matrix for all numeric columns

 #   st.write("Scatterplot matrix for all numeric columns:")

 #   sns.pairplot(data.select\_dtypes(include=['int64', 'float64']))

 #   st.pyplot()

    fraud\_counts = data['fraud'].value\_counts()

# Create a pie chart with the fraud counts

    fig, ax = plt.subplots()

    ax.pie(fraud\_counts, labels=fraud\_counts.index, autopct='%1.1f%%')

    ax.set\_title('Fraudulent Transactions')

    ax.axis('equal')

    st.pyplot(fig)

    # Count the number of fraud and non-fraud records

    fraud\_count = len(data[data['fraud'] == 1])

    non\_fraud\_count = len(data[data['fraud'] == 0])

# Calculate the percentages

    fraud\_percentage = fraud\_count / len(data) \* 100

    non\_fraud\_percentage = non\_fraud\_count / len(data) \* 100

# Create the bar graph

    labels = ['Fraud', 'Non-Fraud']

    percentages = [fraud\_percentage, non\_fraud\_percentage]

# Display the chart using Streamlit

    st.title('Percentage of Fraud and Non-Fraud Records')

    st.bar\_chart({'Labels': labels, 'Percentages': percentages})

# Create your heatmap

    st.title("Correlation Matrix (Heat Map)")

    fig, ax = plt.subplots()

    sns.heatmap(data.corr().round(3), annot=True, vmin=-1, vmax=1, cmap="coolwarm", ax=ax)

    sns.set(rc={"figure.figsize":(10,10)})

# Display the plot in Streamlit app

    st.pyplot(fig)

    st.title('Fraud Data EDA')

# Set the figure size

    plt.figure(figsize=(10, 10))

# Create a subplot for each variable of interest

    plt.subplot(2, 2, 1)

    data['repeat\_retailer'].value\_counts().plot(kind='bar', color='purple')

    plt.title('Repeat Retailer')

    plt.subplot(2, 2, 2)

    data['used\_chip'].value\_counts().plot(kind='bar', color='green')

    plt.title('Used Chip')

    plt.subplot(2, 2, 3)

    data['used\_pin\_number'].value\_counts().plot(kind='bar', color='blue')

    plt.title('Used Pin Number')

    plt.subplot(2, 2, 4)

    data['online\_order'].value\_counts().plot(kind='bar', color='red')

    plt.title('Online Order')

# Set the overall title of the figure

    plt.suptitle('Fraud Data EDA')

# Display the figure using Streamlit

    st.pyplot(plt)

    import streamlit as st

    from sklearn.model\_selection import train\_test\_split

    st.title("Fraud Detection App")

    # Define the feature columns and target variable

    feature\_columns = ["distance\_from\_home", "distance\_from\_last\_transaction",

                       "ratio\_to\_median\_purchase\_price", "repeat\_retailer",

                       "used\_chip", "used\_pin\_number", "online\_order"]

    target\_variable = "fraud"

    # Split the data into training and testing sets

    X = data[feature\_columns]

    y = data[target\_variable]

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=39)

    # Display the dataset

    st.subheader("Dataset")

    st.write(data)

    # Display the feature columns and target variable

    st.subheader("Feature Columns")

    st.write(feature\_columns)

    st.subheader("Target Variable")

    st.write(target\_variable)

    # Display the training and testing sets

    st.subheader("Training and Testing Sets")

    st.write("X\_train:", X\_train.shape)

    st.write("y\_train:", y\_train.shape)

    st.write("X\_test:", X\_test.shape)

    st.write("y\_test:", y\_test.shape)

    import streamlit as st

    from sklearn.linear\_model import LogisticRegression

    from sklearn import metrics

    # Create a logistic regression model and fit it to the training data

    logreg = LogisticRegression(max\_iter=200)

    logreg.fit(X\_train, y\_train)

    # Make predictions on the test data

    y\_pred = logreg.predict(X\_test)

    # Calculate the accuracy of the model

    accuracy = metrics.accuracy\_score(y\_test, y\_pred)

    # Display the accuracy score

    st.subheader("Accuracy of logistic regression classifier on test set:")

    st.write("{:.5f}".format(accuracy))

    # Display the classification report

    st.subheader("Classification Report")

    report = metrics.classification\_report(y\_test, y\_pred, digits=6)

    st.code(report, language="text")

    import streamlit as st

    from sklearn.tree import DecisionTreeClassifier

    from sklearn import metrics

    # Create a decision tree model and fit it to the training data

    decision\_tree = DecisionTreeClassifier()

    decision\_tree.fit(X\_train, y\_train)

    # Make predictions on the test data

    y\_pred = decision\_tree.predict(X\_test)

    # Calculate the accuracy of the model

    accuracy = metrics.accuracy\_score(y\_test, y\_pred)

    # Display the accuracy score

    st.subheader("Accuracy of decision tree classifier on test set:")

    st.write("{:.5f}".format(accuracy))

    # Display the classification report

    st.subheader("Classification Report")

    report = metrics.classification\_report(y\_test, y\_pred, digits=6)

    st.code(report, language="text")

    import streamlit as st

    from sklearn.ensemble import RandomForestClassifier

    from sklearn import metrics

    from sklearn.metrics import classification\_report

    import streamlit as st

    from sklearn.ensemble import RandomForestClassifier

    from sklearn import metrics

# Create a Random Forest Classifier and fit it to the training data

    rfc = RandomForestClassifier(n\_estimators=100)

    rfc.fit(X\_train, y\_train)

# Make predictions on the test data

    y\_pred = rfc.predict(X\_test)

# Calculate the accuracy of the model

    accuracy = metrics.accuracy\_score(y\_test, y\_pred)

# Display the accuracy score

    st.subheader("Accuracy of Random Forest Classifier on test set:")

    st.write("{:.5f}".format(accuracy))

# Display the classification report

    st.subheader("Classification Report")

    report = metrics.classification\_report(y\_test, y\_pred, digits=6)

    st.code(report, language="text")

**Related Projects**

The editor of the blog Credit card fraud detection: how machine learning can protect your business from scams describes a detailed review of credit card fraud detection of an organization or a single person. The blog also discussed who becomes a victim of credit card scams and how machine learning aids in fraud detection. When and how should machine learning be used in fraud detection processes? It covers all the fundamentals of credit card fraud detection.

The website analyticsvidhya.com has a brief description of machine learning algorithms and illustrates how the random forest algorithm and other required algorithms for our project can be used.

The research credit card fraud detection based on attention mechanism and LSTM deep model was published in springer, the goal of this paper is to create a novel system for credit card fraud detection based on sequential data modeling and LSTM deep recurrent neural networks which are very advance related to our project.

**References**

Ibtissam Benchaji, Samira Douzi, Bouabid El Ouahidi, and Jaafar Jaafari. Enhanced credit card fraud detection based on attention mechanism and deep model. Journal of Big Data, 8(1), 2021.

Editor. Credit card fraud detection: How machine learning can protect your business from scams, Aug 2020.

Sruthi E R. Understand random forest algorithms with examples (updated 2023), Feb 2023.

GeeksforGeeks. (2023, February 6). A beginners guide to Streamlit. GeeksforGeeks. Retrieved April 23, 2023, from https://www.geeksforgeeks.org/a-beginners-guide-to-streamlit/

Dhanush Narayanan R. Credit Card Fraud Detection Dataset – “<https://www.kaggle.com/datasets/dhanushnarayananr/credit-card-fraud> “

Ray\_Johnson. (2022, February 18). Streamlit server 1.5.1 from command line, hard to stop with CTRL C. Streamlit. Retrieved April 23, 2023, from https://discuss.streamlit.io/t/streamlit-server-1-5-1-from-command-line-hard-to-stop-with-ctrl-c/22277

K, G. M. (2020, July 18). *Machine learning basics: Decision tree regression*. Medium. Retrieved April 23, 2023, from https://towardsdatascience.com/machine-learning-basics-decision-tree-regression-1d73ea003fda

GeeksforGeeks. (2022, December 30). *Random Forest hyperparameter tuning in python*. GeeksforGeeks. Retrieved April 23, 2023, from https://www.geeksforgeeks.org/random-forest-hyperparameter-tuning-in-python/

Benedekrozemberczki. (n.d.). *Benedekrozemberczki/awesome-fraud-detection-papers: A curated list of data mining papers about fraud detection.* GitHub. Retrieved April 23, 2023,from https://github.com/benedekrozemberczki/awesome-fraud-detection-papers