Summary Report

Group 5

Credit Card Fraud Detection

Abstract:

This research project aims to develop an innovative system dedicated to the detection and prevention of credit card fraud, prioritizing the security and peace of mind of cardholders. The escalating threat of credit card fraud presents a significant challenge to individuals and financial institutions, resulting in substantial financial losses and persistent concerns. Through comprehensive analysis and the application of advanced technology, this project endeavors to combat this pervasive issue, offering a robust solution to mitigate fraudulent activities.

1) Introduction

Credit card fraud has become a common problem in our increasingly digital age, costing people and businesses a lot of money. This issue is being addressed by the Credit Card Transactions Fraud Detection project, which intends to produce a reliable dataset to make it easier to develop and test machine learning models for detecting credit card fraud. Because it helps to increase the security of financial transactions and protect the interests of both consumers and businesses.

The peril of credit card fraud stands as a formidable adversary, exacting a staggering toll on both individuals and the financial landscape. In the year 2023 alone, its impact resonates ominously through statistics: an alarming 46% of global credit card fraud finds its epicenter within the United States. Moreover, projections foretell a disconcerting trajectory, with global credit card fraud poised to spiral to an unprecedented $43 billion by 2026. Within the United States, the forecast is equally foreboding, with anticipated credit card fraud losses eclipsing a daunting $12.5 billion by 2025.

Amidst this landscape, consumer sentiment crystallizes: 48% firmly assert that it falls upon the merchant to shield them from the pervasive threat of fraud. Disturbingly, more than half—55%—of fraudulent credit and debit card transactions amount to less than $100, underscoring the subtlety and frequency of these illicit activities.

The chilling reality further unfolds: every 14 seconds, an individual in the United States becomes a victim of identity theft, painting a picture of vulnerability and urgency.

Considering these disquieting statistics, our pursuit of an advanced fraud detection and prevention system assumes an unparalleled significance. Our endeavor aims not only to stem the hemorrhaging of financial losses but also to restore a sense of trust and assurance in the integrity of financial transactions, heralding a new era of security and confidence for individuals and institutions alike.

2) Data Set

The dataset used for this project, named "Credit Card Fraud Transaction Detection," encompasses a vast array of transaction-related information, comprising 1.3 million rows and 23 columns. This dataset, totaling 502 MB in size, encapsulates critical transaction details from January 1, 2019, to December 31, 2020. Its columns include essential fields such as transaction dates, credit card numbers, merchant specifics, transaction categories, and amounts, along with demographic insights like gender, names, city, state, zip codes, and occupation details. These comprehensive records provide a robust foundation for analysis, modeling, and the development of an advanced system for credit card fraud detection. The dataset's breadth across a two-year span offers an extensive temporal window, facilitating a thorough exploration of patterns and anomalies crucial for effective fraud detection and prevention strategies.

3) Exploratory Data Analysis

The dataset reveals intriguing insights into the patterns and characteristics of fraud transactions. New York emerges as the epicenter of recorded fraud transactions, closely trailed by Texas and Pennsylvania, indicating regional concentrations of fraudulent activities. Interestingly, the age group between 50 and 60 demonstrates a higher incidence of fraud transactions, hinting at potential vulnerabilities within this demographic.

Moreover, a compelling trend surfaces concerning merchant types. Entertainment and grocery merchants stand out, recording a notably larger volume of fraudulent transactions. This trend accentuates potential areas of susceptibility within these sectors, warranting closer scrutiny and fortified security measures.

Delving into regional spending habits, Missouri portrays distinctive preferences. The most significant expenditure occurs in the travel category, followed closely by grocery expenses. This disparity in spending patterns underscores the importance of understanding regional variations and preferences, potentially aiding in the refinement of fraud detection strategies tailored to specific locales and merchant categories.

The culmination of these observations not only sheds light on the geographical and demographic hotspots for fraudulent activities but also emphasizes the relevance of targeted monitoring and intervention strategies within specific merchant sectors and regional demographics.

4) Fraud Indicators

In the pursuit of identifying potential fraud indicators, this project harnesses a multifaceted approach, leveraging critical transaction data attributes such as transaction amounts, geographical coordinates (latitude and longitude) of both the individuals and merchants involved,

as well as timestamps. This comprehensive dataset empowers the development of a sophisticated model capable of discerning anomalies indicative of fraudulent activities.

The model operates by scrutinizing transactions against established norms, seeking outliers In various dimensions. Unusual transaction amounts outside typical spending behaviors, transactions occurring at atypical locations, and deviations from regular transaction timestamps are among the primary indicators flagged for closer scrutiny. By analyzing transactions that diverge from established patterns, this model adeptly identifies potential instances of fraud.

The incorporation of geographical coordinates enables a spatial analysis, allowing for the detection of irregularities such as transactions occurring far from a cardholder's usual locations or unexpected merchant locations, further contributing to the model's robustness in pinpointing suspicious activities.

This holistic approach, incorporating transaction amounts, geographical data, and timestamps, forms the bedrock of a sophisticated fraud detection model. By scrutinizing transactions against established patterns and discerning anomalies across multiple dimensions, the model effectively discerns potential instances of fraud, enhancing the security measures in place for credit card transactions.

5) Fraud Detection/ Machine Learning models

For fraud detection, this project employs a tailored approach utilizing distinct machine learning models designed for specific geographical regions. The segmentation into East Coast and West Coast models is informed by localized spending behaviors, transaction patterns, and historical fraud detection records unique to each region.

The East Coast model leverages machine learning algorithms trained on data specific to the spending practices and past instances of fraud prevalent in the Eastern geographical region. Similarly, the West Coast model is calibrated based on the distinct spending procedures and historical fraud patterns characteristic of the Western geographical region. This targeted approach allows for a more nuanced analysis, considering regional idiosyncrasies and enhancing the accuracy of fraud detection.

Evaluation metrics used to gauge the performance of these models encompass a range of standard measures including accuracy, precision, recall, F1-score, and Receiver Operating Characteristic (ROC) curves. These metrics enable a comprehensive assessment of the models' efficacy in correctly identifying fraudulent transactions while minimizing false positives and false negatives.

By employing region-specific models and evaluating their performance using established metrics, this project aims to optimize fraud detection capabilities, tailoring the approach to the intricacies of each geographical area's spending habits and historical fraud trends. This nuanced strategy seeks to bolster the accuracy and reliability of fraud detection systems, fortifying the defense against fraudulent activities in a targeted and effective manner.

Technical Report

1). Introduction

The Technical report provides an in-depth view of the credit card fraud detection project methodology, tools, and results.

2). Problem Statement

The primary goal is to detect fraudulent credit transactions accurately and give some analyzed reports.

3). Methodology

Sequential Model:

A Sequential Model is a linear stack of layers used in deep learning. It's particularly suitable for feedforward neural networks where you stack layers sequentially. Typically used for simple tasks and as a baseline for more complex architectures.

Convolutional Neural Network (CNN):

CNN is designed for processing structured grid data like images. It employs convolutional layers to learn spatial hierarchies of features automatically and adaptively from input data. CNNs excel in tasks like image recognition and classification.

LSTM Model:

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN). It's used for sequential data and can capture long-term dependencies. LSTMs are used in tasks like speech recognition, language modeling, and time-series analysis.

GRU Model:

Gated Recurrent Unit (GRU) is another type of RNN like LSTM. It's more computationally efficient and has a simpler architecture. GRUs are often used when a balance between performance and efficiency is required.

Auto encoder model:

An Autoencoder is an unsupervised machine learning model that aims to learn efficient representations of data. It consists of an encoder network that compresses data into a lower-dimensional representation and a decoder network that reconstructs the original data. Autoencoders are used for tasks like data denoising, dimensionality reduction, and anomaly detection.

4). Few Results

East Coast Modeling:

Implementing Sequential Model:

```
Out[54]: <keras.src.callbacks.History at 0x78be6db437f0>
```

```
In [55]:
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
6086/6086 [==============================] - 11s 2ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    193655
           1       0.77      0.48      0.59      1068

    accuracy                           1.00    194723
   macro avg       0.88      0.74      0.80    194723
weighted avg       1.00      1.00      1.00    194723

[[193497    158]
 [   552    516]]
```

```
In [ ]:
```

Implementing Convolutional Neural Network (CNN):

```
Out[56]: <keras.src.callbacks.History at 0x78be62bf7940>
```

```
In [57]:
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
6086/6086 [==============================] - 16s 3ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    193655
           1       0.66      0.60      0.63      1068

    accuracy                           1.00    194723
   macro avg       0.83      0.80      0.81    194723
weighted avg       1.00      1.00      1.00    194723

[[193330    325]
 [   429    639]]
```

```
In [ ]:
```

Implementing LSTM Model:

Out[58]: <keras.src.callbacks.History at 0x78be699931f0>

In [59]:
```
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
6086/6086 [==============================] - 34s 5ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    193655
           1       0.82      0.45      0.58      1068

    accuracy                           1.00    194723
   macro avg       0.91      0.72      0.79    194723
weighted avg       1.00      1.00      1.00    194723

[[193549    106]
 [   587    481]]
```

In [ ]:

Implementing GRU Model:

In [61]:
```
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
6086/6086 [==============================] - 49s 8ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    193655
           1       0.82      0.41      0.55      1068

    accuracy                           1.00    194723
   macro avg       0.91      0.71      0.77    194723
weighted avg       1.00      1.00      1.00    194723

[[193558     97]
 [   628    440]]
```

Implementing Auto encoder model:

Out[66]: <keras.src.callbacks.History at 0x78be6db00dc0>

In [67]:
```
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
6086/6086 [==============================] - 31s 5ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    193655
           1       0.82      0.41      0.55      1068

    accuracy                           1.00    194723
   macro avg       0.91      0.71      0.77    194723
weighted avg       1.00      1.00      1.00    194723

[[193558     97]
 [   628    440]]
```

West Coast Modeling:

Implementing Sequential Model:

Out[23]: <keras.src.callbacks.History at 0x7ed83258ae60>

In [24]:
```
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
5493/5493 [==============================] - 9s 2ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    174898
           1       0.67      0.60      0.63       859

    accuracy                           1.00    175757
   macro avg       0.83      0.80      0.82    175757
weighted avg       1.00      1.00      1.00    175757

[[174641    257]
 [   340    519]]
```

In [ ]:

Implementing Convolutional Neural Network (CNN):

Out[25]: <keras.src.callbacks.History at 0x7ed826cd6620>

In [26]:
```
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
5493/5493 [==============================] - 16s 3ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    174898
           1       0.83      0.30      0.44       859

    accuracy                           1.00    175757
   macro avg       0.91      0.65      0.72    175757
weighted avg       1.00      1.00      1.00    175757

[[174845     53]
 [   601    258]]
```

In [ ]:

Implementing LSTM Model:

Out[27]: <keras.src.callbacks.History at 0x7ed8ce83b340>

In [28]:
```
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
5493/5493 [==============================] - 30s 5ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    174898
           1       0.71      0.59      0.64       859

    accuracy                           1.00    175757
   macro avg       0.85      0.79      0.82    175757
weighted avg       1.00      1.00      1.00    175757

[[174688    210]
 [   355    504]]
```

In [ ]:

Implementing GRU Model:

Out[30]: <keras.src.callbacks.History at 0x7ed826f04f70>

In [31]:
```
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
5493/5493 [==============================] - 29s 5ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    174898
           1       0.59      0.35      0.44       859

    accuracy                           1.00    175757
   macro avg       0.79      0.68      0.72    175757
weighted avg       0.99      1.00      1.00    175757

[[174688    210]
 [   555    304]]
```

In [ ]:

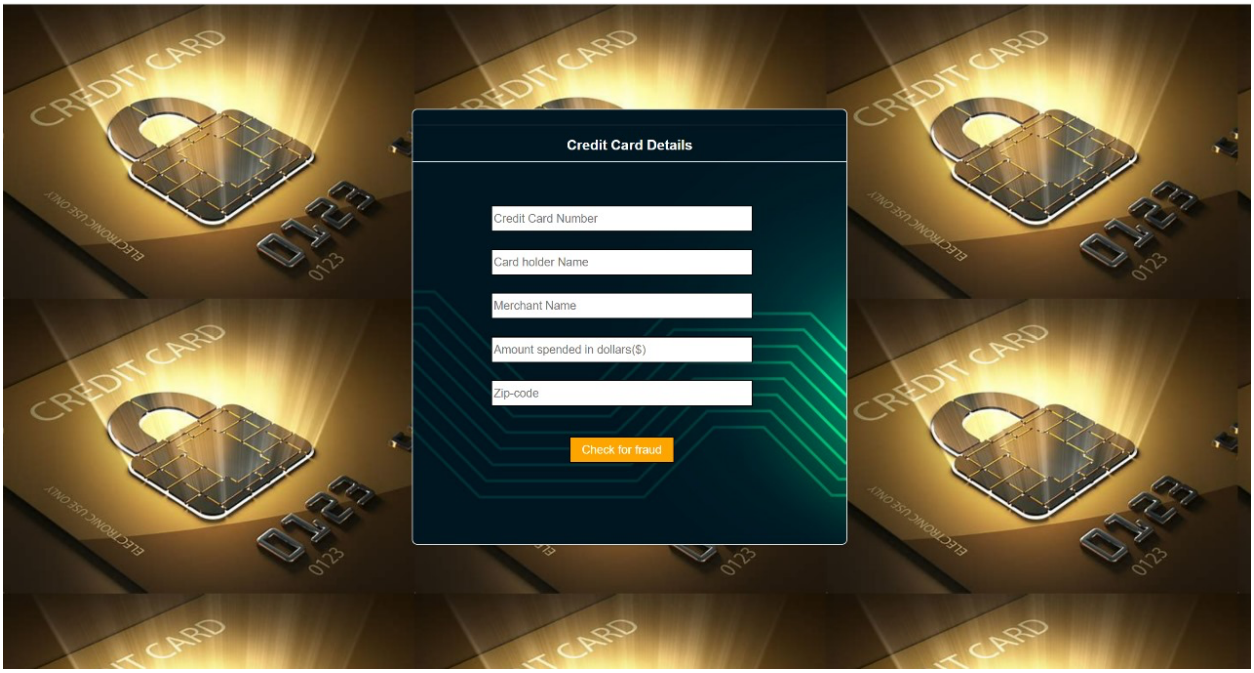
Implementing Auto encoder model:

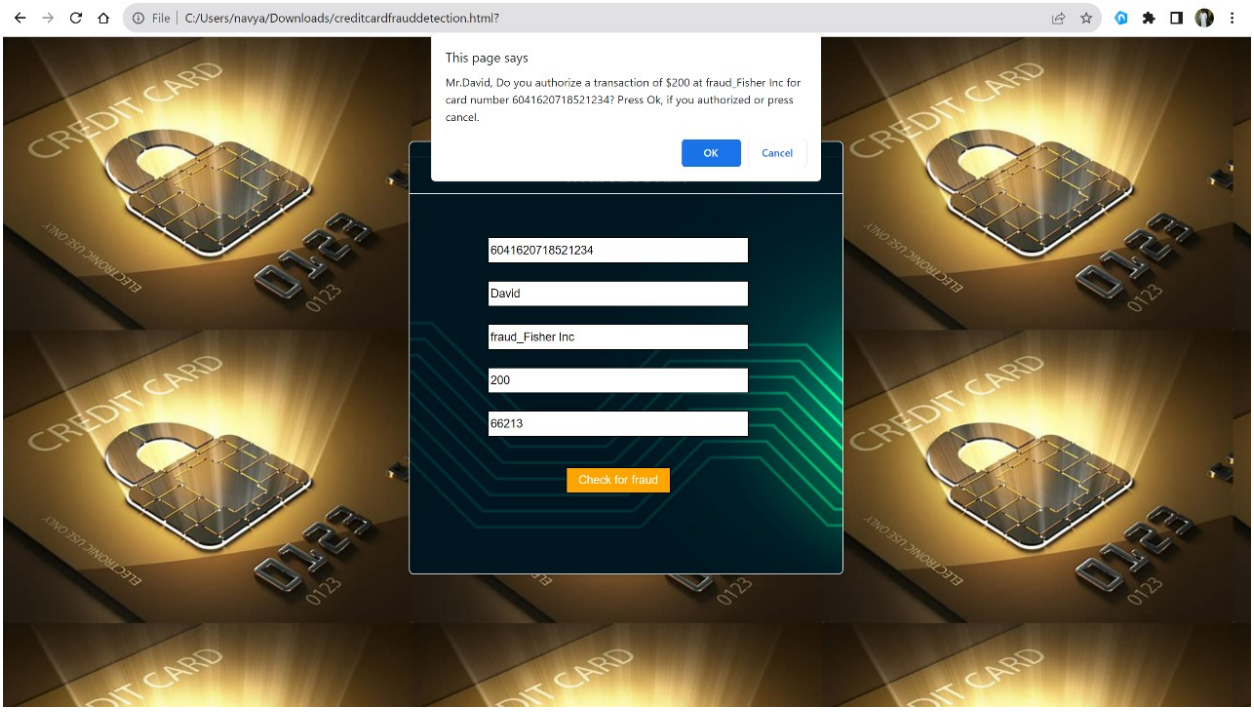
Out[32]: <keras.src.callbacks.History at 0x7ed7f671b970>

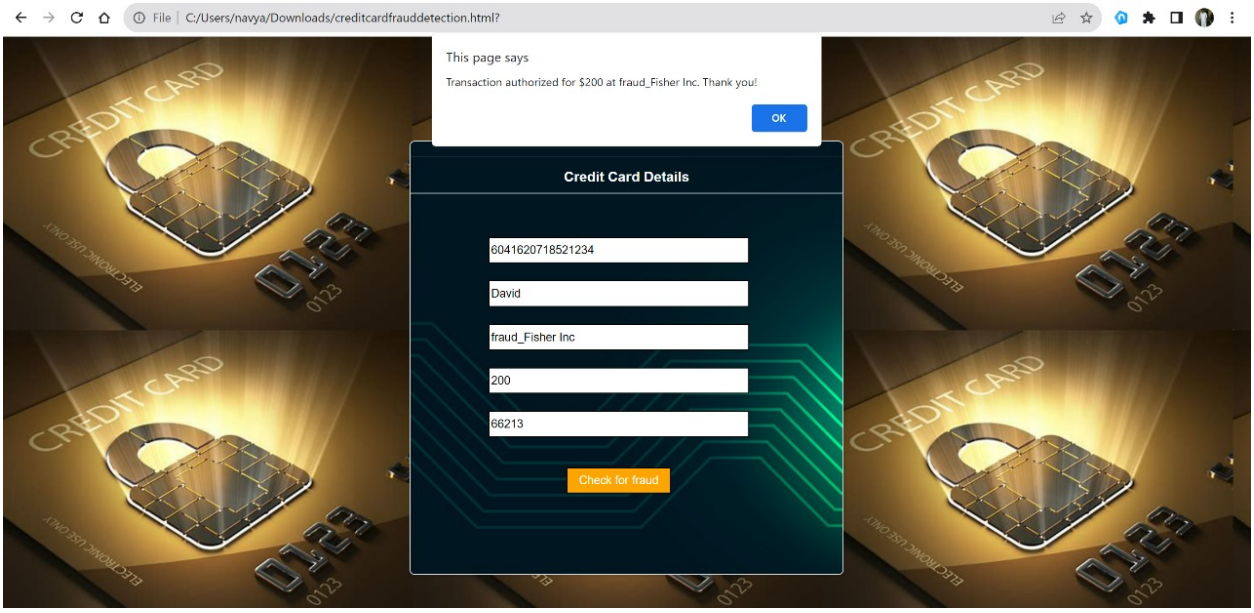
In [33]:
```
# Evaluate the model on the test set
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5)  # Convert probabilities to binary predictions

# Print classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

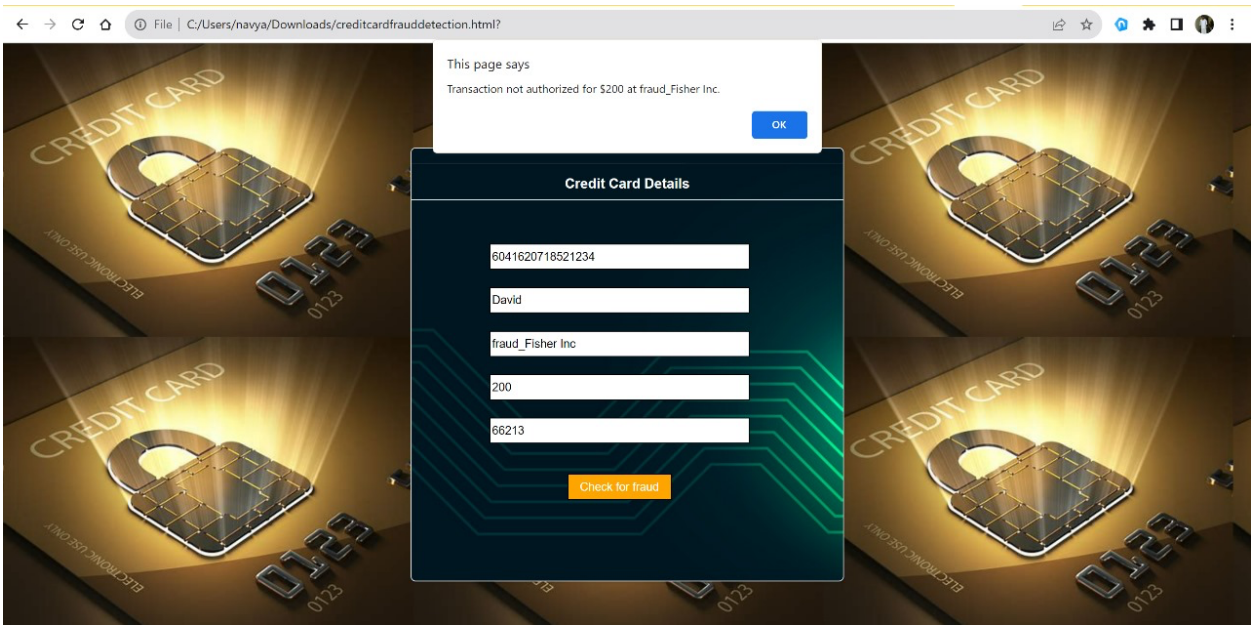
```
5493/5493 [==============================] - 30s 5ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    174898
           1       0.59      0.35      0.44       859

    accuracy                           1.00    175757
   macro avg       0.79      0.68      0.72    175757
weighted avg       0.99      1.00      1.00    175757

[[174688    210]
 [   555    304]]
```

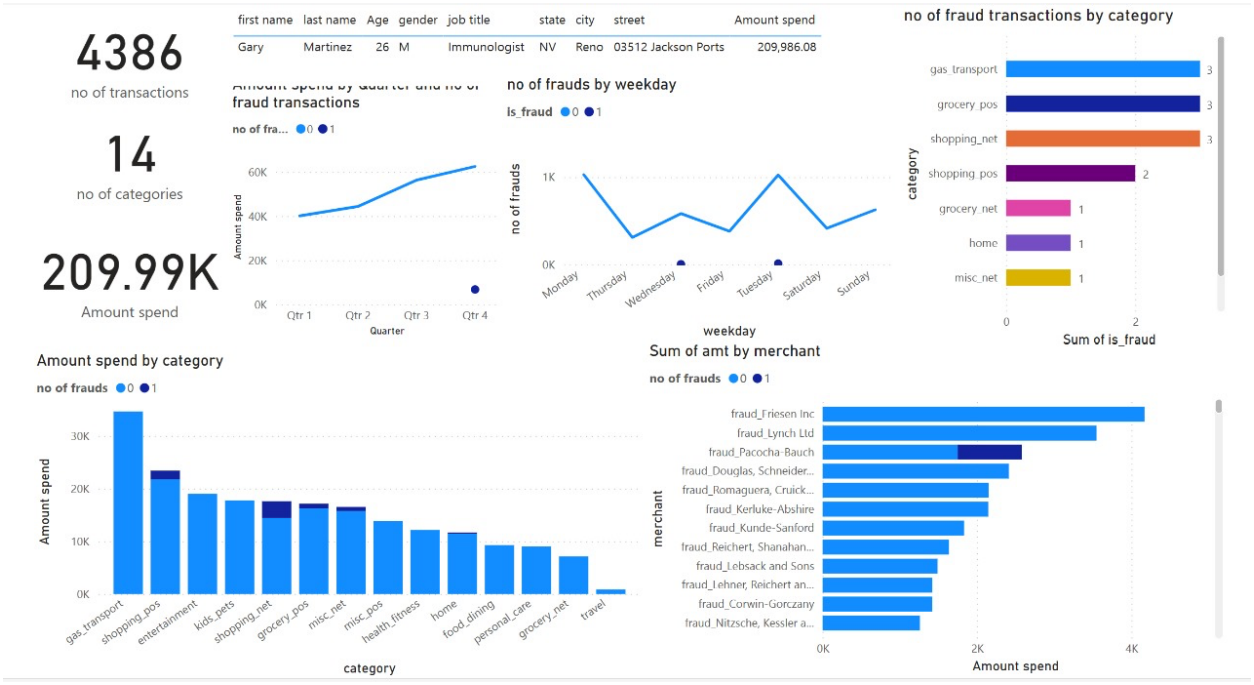
Website Front-end:




File | C:/Users/navya/Downloads/creditcardfrauddetection.html?
This page says
Mr.David, Do you authorize a transaction of $200 at fraud_Fisher Inc for card number 6041620718521234? Press Ok, if you authorized or press cancel.
OK
Cancel
6041620718521234
David
fraud_Fisher Inc
200
66213
Check for fraud


File | C:/Users/navya/Downloads/creditcardfrauddetection.html?
This page says
Transaction authorized for $200 at fraud_Fisher Inc. Thank you!
OK
Credit Card Details
6041620718521234
David
fraud_Fisher Inc
200
66213
Check for fraud





GitHub Link: https://github.com/BinduParvati7/Capstone5588/