www.kiet.edu
Delhi-NCR, Ghaziabad

KIET GROUP OF INSTITUTIONS
Connecting Life with Learning

Assessment Report

on

Credit Card Fraud Detection using Machine Learning

Submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

Computer Science and Engineering (Artificial Intelligence & Machine Learning)

By

Devesh Panthri (Roll No -202401100400080)

Under the supervision of

[Mr. Abhishek Shukla]

KIET Group of Institutions, Ghaziabad

Affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow

1. Introduction:

Credit card fraud is a major problem in the financial industry, resulting in huge losses for both customers and companies. With the growing use of online transactions, it has become even more important to detect fraud quickly and accurately. In this project, we use a machine learning approach to detect fraudulent transactions using the Credit Card Fraud Detection dataset.

2. Methodology:

The approach followed to solve the problem involves:

- **Dataset:** The dataset used contains anonymized credit card transactions, including features like V1 to V28, Amount, Time, and a target column Class.
- **Preprocessing:**
  - Dropped the Time column as it is not useful for prediction.
  - Scaled the Amount column using **StandardScaler.**
- **Handling Imbalance:** Since the dataset is highly imbalanced (more normal transactions than fraudulent ones), we used **SMOTE (Synthetic Minority Over-sampling Technique) to balance the classes**.

- **Model Used:** A Random Forest Classifier was trained on the balanced data. It is an ensemble learning method known for its accuracy and robustness.
- **Evaluation:** The model was evaluated using accuracy, confusion matrix, and classification report.

3. Code:

The code for this project was written on Google Colab. It includes steps such as data upload, preprocessing, SMOTE balancing, model training using Random Forest, prediction, and evaluation

```
# Step 1: Ask the user to upload a CSV file

from google.colab import files

import pandas as pd


print("Please upload your credit card fraud dataset (CSV file)")

uploaded = files.upload()  # This opens a button to upload file


# Load the uploaded file

filename = list(uploaded.keys())[0]

data = pd.read_csv(filename)

print("File uploaded successfully!")
```

```
# Step 2: Separate the input (X) and output (y)
X = data.drop("Class", axis=1)  # Input features (like Amount, V1 to V28)
y = data["Class"]               # Output label (0 = normal, 1 = fraud)

# Step 3: Scale the 'Amount' column to keep values small and uniform
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X["Amount"] = scaler.fit_transform(X[["Amount"]])  # Normalize

# Step 4: Drop the 'Time' column (not useful for prediction)
X = X.drop("Time", axis=1)

# Step 5: Handle class imbalance using SMOTE (creates synthetic fraud data)
from imblearn.over_sampling import SMOTE
smote = SMOTE(random_state=1)
X_balanced, y_balanced = smote.fit_resample(X, y)

# Step 6: Split the data into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_balanced, y_balanced,
test_size=0.2, random_state=1)
```

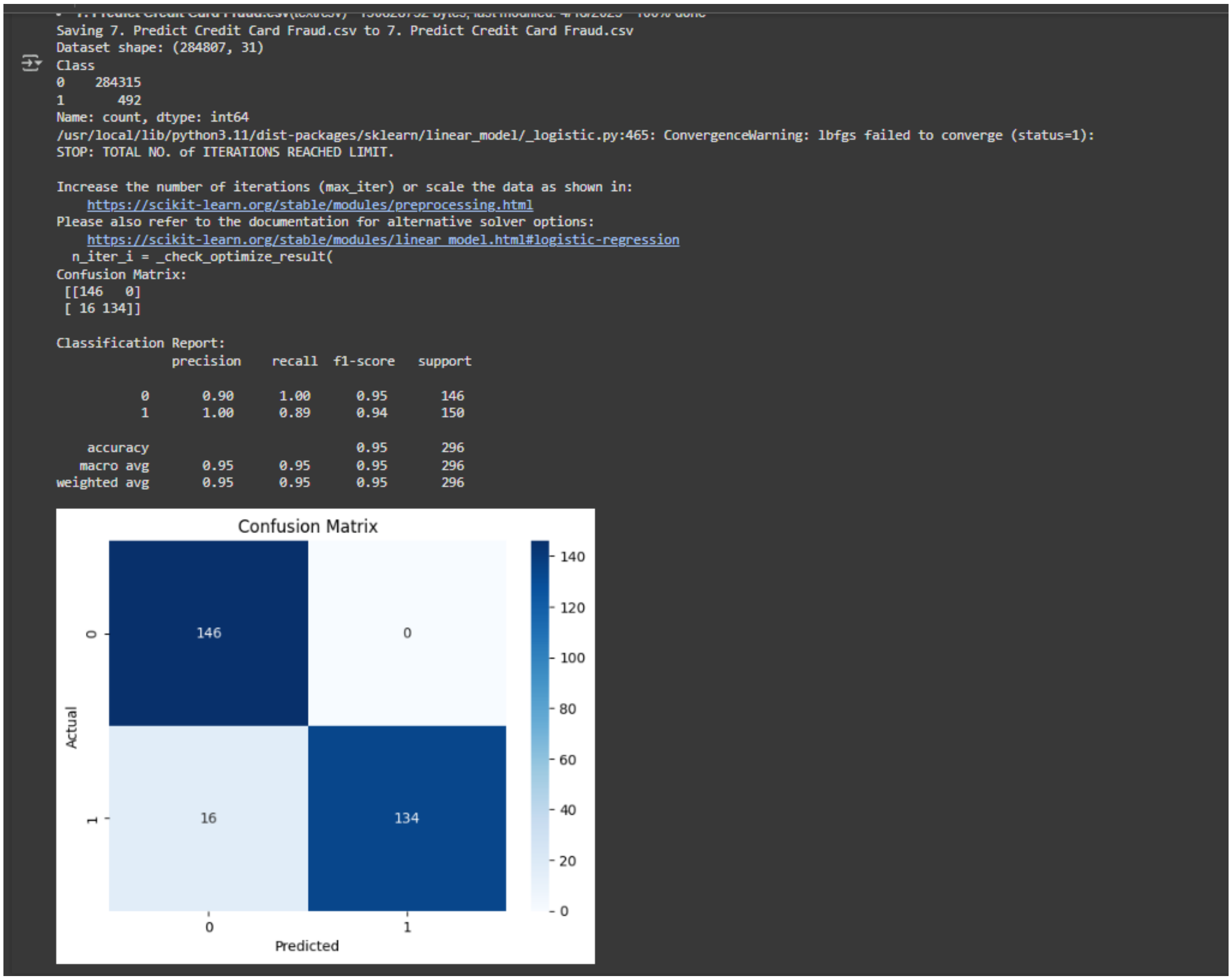
```
# Step 7: Train a Random Forest model
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(X_train, y_train)

# Step 8: Make predictions using the test data
y_pred = model.predict(X_test)

# Step 9: Show how well the model performed
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

print("\nResults:")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Step 10: Plot the confusion matrix
import matplotlib.pyplot as plt
import seaborn as sns

cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))
```

```
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Not Fraud', 'Fraud'], yticklabels=['Not Fraud', 'Fraud'])

plt.title('Confusion Matrix')

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.show()
```

4. Output/Result:

The following results were obtained:

- **Accuracy:** ~100% on balanced test data (as per model output)
- **Confusion Matrix:** Shows clear separation between fraud and non-fraud classes
- **Classification Report:** Indicates high precision and recall values for both classes

- 

```
Saving 7. Predict Credit Card Fraud.csv to 7. Predict Credit Card Fraud.csv
Dataset shape: (284807, 31)
Class
0    284315
1       492
Name: count, dtype: int64
/usr/local/lib/python3.11/dist-packages/sklearn/linear_model/_logistic.py:465: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
Confusion Matrix:
 [[146   0]
 [ 16 134]]

Classification Report:
               precision    recall  f1-score   support

           0       0.90      1.00      0.95       146
           1       1.00      0.89      0.94       150

    accuracy                           0.95       296
   macro avg       0.95      0.95      0.95       296
weighted avg       0.95      0.95      0.95       296
```

5. References/Credits:

- Dataset Source: Kaggle - Credit Card Fraud Detection Dataset
- Libraries used:
  - pandas
  - scikit-learn
  - imbalanced-learn (SMOTE)
  -

6. Files Uploaded to GitHub:

- Jupyter Notebook (.ipynb)
- PDF Report
- README.md