<u>ONLINE PAYMENTS FRAUD DETECTION USING WITH MACHINE LEARNING</u>:

To build an application that can detect the legitimacy of the transaction in real-time and increase the security to prevent fraud.

By

(Marri yashmitha)

(Manthina raja rishika)

(Kutagulla safa)

Guided by

Prof. Ms swetha raj

A Dissertation Submitted to
SRI VENKATESWARA COLLEGE OF ENGINEERING AND TECHNOLOGY, An Autonomous Institution affiliated to 'JNTU Ananthapur' in Partial Fulfilment of the Bachelor of Technology branch of ***Computer science and Engineering***

May 2024

Model taining:

Creating a model training file for online fraud detection using machine learning involves several steps, including data preprocessing, feature engineering, model training, and evaluation. Here's a basic example using Python and popular libraries like pandas, scikit-learn, and a simple logistic regression model.

Install Required Libraries:

Ensure you have the necessary libraries installed:

Code

```
pip install pandas scikit-learn
```

Sample Data:

Assume we have a CSV file named transactions.csv with features such as transaction_id, amount, transaction_time, is_fraud, etc.

Code for Model Training:

python

code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

# Load data
data = pd.read_csv('transactions.csv')

# Basic data exploration (optional)
print(data.head())
print(data.info())
print(data.describe())

# Preprocess data
# Assume 'is_fraud' is the target variable and the rest are features
X = data.drop(['is_fraud', 'transaction_id'], axis=1)  # Dropping 'transaction_id'
as it's not useful for prediction
y = data['is_fraud']

# Handling missing values (if any)
X.fillna(X.mean(), inplace=True)

# Feature scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,
random_state=42)

# Initialize and train model
```

```
model = LogisticRegression()
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))

# Save the model for future use (optional)
import joblib
joblib.dump(model, 'fraud_detection_model.pkl')
joblib.dump(scaler, 'scaler.pkl')
```

Explanation:

Data Loading:

Load the data from a CSV file into a pandas DataFrame.

Data Preprocessing:

Drop the transaction_id column as it's not necessary for model training.
Separate features (X) and the target variable (y).
Handle missing values by filling them with the mean of each column.
Scale the features for better model performance.

Train-Test Split:

Split the data into training and testing sets (80-20 split).
Model Training:

Initialize and train a logistic regression model.
Model Evaluation:

Predict the target on the test set.
Evaluate the model using accuracy, confusion matrix, and classification report.
Save the Model:

Save the trained model and the scaler for future use using joblib.

This is a simple example to get you started. For a production-level system, you would need to perform more extensive data preprocessing, possibly use more sophisticated models like decision trees, random forests, gradient boosting machines, or neural networks, and ensure proper cross-validation and hyperparameter tuning.