# **ONLINE PAYMENTS FRAUD DETECTION USING WITH MACHINE LEARNING:**

# **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)***

***(Kutagula 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*



**SRI VENKATESWARA COLLEGE OF ENGINEERING AND TECHNOLOGY**

**R.V.S. Nagar Tirupathi Road, Andhra Pradesh– 517127**

**FLASK FILE**

Creating a Flask application for online fraud detection using machine learning involves several steps

Train a machine learning model on relevant data.

Save the trained model.

Create a Flask application that loads the model and uses it to make predictions on new data.

**Here's a simple example to guide you through these steps:**

1. Train and Save the Machine Learning Model

First, train your machine learning model. For simplicity, I'll use a dummy dataset and a basic Logistic Regression model.

**python**

code

# train\_model.py

import pandas as pd

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

import joblib

# Example dataset, replace with your actual data

data = pd.read\_csv('fraud\_data.csv') # Make sure to have your dataset here

X = data.drop('label', axis=1)

y = data['label']

# Split the dataset

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Train the model

model = LogisticRegression()

model.fit(X\_train, y\_train)

# Evaluate the model

y\_pred = model.predict(X\_test)

print(f"Accuracy: {accuracy\_score(y\_test, y\_pred)}")

# Save the model

joblib.dump(model, 'fraud\_model.pkl')

Run this script to train and save your model. Make sure you have the necessary libraries installed (pandas, scikit-learn, and joblib).

**2. Create the Flask Application**

Next, create a Flask application that loads the saved model and uses it to make predictions on new data.

**Python**

code

# app.py

from flask import Flask, request, jsonify

import joblib

import numpy as np

app = Flask(\_name\_)

# Load the trained model

model = joblib.load('fraud\_model.pkl')

@app.route('/')

def home():

return "Fraud Detection API"

@app.route('/predict', methods=['POST'])

def predict():

data = request.get\_json(force=True)

# Assuming the input is a dictionary with feature names as keys

features = np.array([data[feature] for feature in sorted(data)])

# Reshape the features to match the model input

features = features.reshape(1, -1)

prediction = model.predict(features)

return jsonify({

'prediction': int(prediction[0])

})

if \_name\_ == '\_main\_':

app.run(debug=True)

**3. Running the Application**

To run your Flask application, execute the following command in your terminal:

**code**

python app.py

This will start the Flask server, and you can make POST requests to http://127.0.0.1:5000/predict with JSON data to get fraud predictions.

**Example Request**

You can use a tool like curl or Postman to send a POST request to your Flask API. Here is an example using curl:

code

curl -X POST http://127.0.0.1:5000/predict -H "Content-Type: application/json" -d '{"feature1": value1, "feature2": value2, ..., "featureN": valueN}'

Replace "feature1": value1, "feature2": value2, ..., "featureN": valueN with your actual feature names and values.

This is a basic example to get started. In a real-world application, also need to handle various aspects like input validation, error handling, logging, security, and potentially scaling your application.