Model Development Phase Report

Date	June 2025
Team ID	SWTID1749841176
Project Title	Online Payments Fraud Detection Using Machine Learning
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code:

Note: For ease of working with the data and for model training we have used a label encoder to encode the string values

i) **"is Fraud" as 1**
ii) **"is not Fraud" as 0**

```
#data manipulation
import numpy as np
import pandas as pd

#data visualization
import matplotlib.pyplot as plt
import seaborn as sns

#scientific computing
from scipy import stats

#machine learning models and tools
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier, ExtraTreesClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, f1_score, classification_report, confusion_matrix

#gradient boosting
import xgboost as xgb

#utilities
import pickle
import warnings
warnings.filterwarnings('ignore')
```

Random Forest Classifier:

```
rfc = RandomForestClassifier()
rfc.fit(x_train, y_train)

y_test_predict1 = rfc.predict(x_test)
test_accuracy = accuracy_score(y_test, y_test_predict1)
test_accuracy
```

```
y_train_predict1 = rfc.predict(x_train)
train_accuracy = accuracy_score(y_train, y_train_predict1)
train_accuracy
```

Decision Tree Classifier:

```
dtc = DecisionTreeClassifier()
dtc.fit(x_train, y_train)

y_test_predict2 = dtc.predict(x_test)
test_accuracy = accuracy_score(y_test, y_test_predict2)
test_accuracy
```

```
y_train_predict2 = dtc.predict(x_train)
train_accuracy = accuracy_score(y_train, y_train_predict2)
train_accuracy
```

Extra Trees Classifier:

```
etc = ExtraTreesClassifier()
etc.fit(x_train,y_train)
y_test_predict3 = etc.predict(x_test)
test_accuracy = accuracy_score(y_test, y_test_predict3)
test_accuracy
```

```
y_train_predict3 = etc.predict(x_train)
train_accuracy = accuracy_score(y_train,y_train_predict3)
train_accuracy
```

SupportVectorMachine Classifier:

```
svc = SVC()
svc.fit(X_train, y_train)

y_test_pred4 = svc.predict(X_test)
accuracy = accuracy_score(y_test, y_test_pred4)
print("Accuracy:", accuracy)

y_train_predict4 = svc.predict(X_train)
print(classification_report(y_train,y_train_predict4))

pd.crosstab(y_test,y_test_pred4)
```

XgBoost Classifier:

```
xgb1 = xgb.XGBClassifier()
xgb1.fit(x_train,y_train)

y_test_predict5 = xgb1.predict(x_test)
test_accuracy = accuracy_score(y_test, y_test_predict5)
test_accuracy
```

```
y_train_predict5 = xgb1.predict(x_train)
train_accuracy = accuracy_score(y_train,y_train_predict5)
train_accuracy
```

Model Validation and Evaluation Report:

Model	Classification Report	F1 Score	Confusion Matrix

Random Forest Classifier	`print(classification_report(y_test, y_test_predict1))` ✓ 0.5s precision recall f1-score support 0 1.00 1.00 1.00 1270883 1 0.98 0.79 0.88 1641 accuracy 1.00 1272524 macro avg 0.99 0.90 0.94 1272524 weighted avg 1.00 1.00 1.00 1272524	94%	`confusion_matrix(y_test, y_test_predict1)` ✓ 0.0s array([[1270856, 27], [343, 1298]])
Decision Tree Classifier	`print(classification_report(y_test, y_test_predict2))` ✓ 0.5s precision recall f1-score support 0 1.00 1.00 1.00 1270883 1 0.89 0.88 0.88 1641 accuracy 1.00 1272524 macro avg 0.95 0.94 0.94 1272524 weighted avg 1.00 1.00 1.00 1272524	94%	`confusion_matrix(y_test, y_test_predict2)` ✓ 0.1s array([[1270708, 175], [200, 1441]])
Extra Trees Classifier	`print(classification_report(y_test,y_test_predict3))` ✓ 0.5s precision recall f1-score support 0 1.00 1.00 1.00 1270883 1 0.99 0.78 0.87 1641 accuracy 1.00 1272524 macro avg 0.99 0.89 0.93 1272524 weighted avg 1.00 1.00 1.00 1272524	93%	`confusion_matrix(y_test, y_test_predict3)` ✓ 0.0s array([[1270869, 14], [369, 1272]])

Support Vector Machine Classifier	`print(classification_report(y_train,y_train_predict4))`
	`pd.crosstab(y_test,y_test_pred4)`
	Accuracy: 0.8890016233766234
	precision recall f1-score support
	0 0.85 0.99 0.92 13123
	1 0.98 0.66 0.79 6588
	accuracy 0.88 19711
	macro avg 0.92 0.83 0.85 19711
	weighted avg 0.89 0.88 0.87 19711
		85%	`confusion_matrix(y_test, y_test_predict4)`
			array([[1270856, 27],
			[343, 1298]])

XgBoost Classifier	`print(classification_report(y_test, y_test_predict5))` ✓ 0.5s precision recall f1-score support 0 1.00 1.00 1.00 1270883 1 0.91 0.76 0.82 1641 accuracy 1.00 1272524 macro avg 0.95 0.88 0.91 1272524 weighted avg 1.00 1.00 1.00 1272524	91%	`confusion_matrix(y_test, y_test_predict5)` ✓ 0.1s array([[1270754, 129], [402, 1239]])