Classification Data Documentation

1. Introduction

Predict the payment method for taxi rides in Chicago based on trip data.

2. Dataset

Source: 'Chicago taxi train' CSV file with dimensions (31695 x 18):

- Relevant Columns:
  - TRIP_MILES
  - TRIP_SECONDS
  - FARE
  - TIP_RATE
  - PAYMENT_TYPE (Target)

After examining the data distribution, it was found that the PAYMENT_TYPE column had imbalanced classes.

3. Preprocessing

Steps:

- Handle missing values in TRIP_MILES and TRIP_SECONDS using the mean:

```
df['TRIP_MILES'] = df['TRIP_MILES'].fillna(df['TRIP_MILES'].mean())
df['TRIP_SECONDS'] = df['TRIP_SECONDS'].fillna(df['TRIP_SECONDS'].mean())
```

- Encode categorical columns (PAYMENT_TYPE) using one-hot encoding:

```
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
df['PAYMENT_TYPE'] = encoder.fit_transform(df['PAYMENT_TYPE'])
```

- Scale numerical features:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_features = scaler.fit_transform(df[['TRIP_MILES', 'TRIP_SECONDS', 'FARE', 'TIP_RATE']])
```

4. Machine Learning Model

Steps:

- Split data into training and testing sets:

```
from sklearn.model_selection import train_test_split
X = scaled_features
y = df['PAYMENT_TYPE']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

- Train an SVM classifier:

```
from sklearn.svm import SVC
svm_model = SVC(kernel='linear', random_state=42)
svm_model.fit(X_train, y_train)
y_pred = svm_model.predict(X_test)
```

5. Accuracy

Metrics:

- Accuracy Score:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')
```

Example Output: 86.7%

- Classification Report:

from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred))

6. SVM (Support Vector Machine)

Support Vector Machine (SVM) is a supervised machine learning algorithm.

It works well for high-dimensional spaces and uses a hyperplane to classify data points.

- Confusion Matrix:

from sklearn.metrics import confusion_matrix

print(confusion_matrix(y_test, y_pred))

Example:

[[500, 50],

[30, 420]]

- Insights: The model performed well, with minimal misclassifications in both classes.