# Credit Card Fraud Detection — Full Documentation + Complete Code

This PDF contains full project documentation and ALL notebook code extracted exactly as-is (no modifications).

1. Overview

-----------

This project performs credit card fraud detection using Machine Learning.

The notebook includes dataset loading, EDA, class imbalance inspection, model training, and evaluation.

2. Libraries Used

-----------------

- pandas
- numpy
- seaborn / matplotlib
- sklearn.model_selection
- sklearn.preprocessing
- sklearn.linear_model / sklearn.ensemble
- sklearn.metrics

3. Workflow Steps

-----------------

1. Load dataset

2. Check imbalance (fraud vs non-fraud)

3. Prepare features and labels

4. Train/Test split

5. Train classifier

6. Evaluate using precision, recall, F1

4. Important Variables

----------------------

- data / df

- X, Y

- model

- predictions

5. Notes

--------

- Accuracy alone is misleading

- Recall is more important for fraud detection

6. Complete Notebook Code

-------------------------

Below are all code cells exactly as they appear.

### *Code Cell 1*

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

### *Code Cell 2*

```
#Load the csv file
credit_card_data = pd.read_csv('/content/drive/MyDrive/Data Science/Projects/10 Credit Card Fraud De
credit_card_data.head()
```

### *Code Cell 3*

```
credit_card_data.shape
```

### *Code Cell 4*

```
credit_card_data.info()
```

### *Code Cell 5*

```
credit_card_data.isnull().sum()
```

### *Code Cell 6*

```
#distribution of legit transaction and fraudulant transaction
credit_card_data['Class'].value_counts()
```

### *Code Cell 7*

```
#seperate the data for analysis
legit = credit_card_data[credit_card_data.Class == 0]
fraud = credit_card_data[credit_card_data.Class == 1]
print(legit.shape)
print(fraud.shape)
```

### *Code Cell 8*

```
#statistical measures of the data
legit.Amount.describe()
```

### *Code Cell 9*

```
fraud.Amount.describe()
```

### *Code Cell 10*

```
#compare the values for both transactions
credit_card_data.groupby('Class').mean()
```

### *Code Cell 11*

```
legit_sample = legit.sample(n=492) #random sampling
```

### *Code Cell 12*

```
legit_sample.shape
```

### *Code Cell 13*

```
new_dataset = pd.concat([legit_sample, fraud], axis=0)
new_dataset.shape
```

### *Code Cell 14*

```
new_dataset.head()
```

### *Code Cell 15*

```
new_dataset.tail()
```

### *Code Cell 16*

```
new_dataset['Class'].value_counts()
```

### *Code Cell 17*

```
new_dataset.groupby('Class').mean()
```

### *Code Cell 18*

```
x = new_dataset.drop(['Class'], axis=1)
y = new_dataset['Class']
```

### *Code Cell 19*

```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2, stratify=y,random_state=2)
x.shape, x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

### *Code Cell 20*

```
model = LogisticRegression()
#training the Logistic Regression
model.fit(x_train, y_train)
```

### *Code Cell 21*

```
x_train_predicition = model.predict(x_train)
training_data_accuracy = accuracy_score(x_train_predicition, y_train)
print('Accuracy on training data: ', training_data_accuracy*100)
```

### *Code Cell 22*

```
x_test_predicition = model.predict(x_test)
test_data_accuracy = accuracy_score(x_test_predicition, y_test)
print('Accuracy on test data: ', test_data_accuracy*100)
```

### *Code Cell 23*

```
input_data = x_test.iloc[1].values.reshape(1,-1)
prediction = model.predict(input_data)
print(prediction)

if prediction == 0:
  print('The transaction is legit')
else:
  print('The transaction is fraud')
```

### *Code Cell 24*

```
x_test.iloc[1]
```

### *Code Cell 25*

```
# (Empty)
```