The Credit Card Fraud Detection falls under Classification Machine Learning Problem.

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
df = pd.read_csv('/kaggle/input/creditcardfraud/creditcard.csv')
df
```

Out [2]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431
...	...	...	...	...	...	...	...	...	...	...	...	...
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	1.914428	...	0.213454
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	0.584800	...	0.214205
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	0.432454	...	0.232045
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180	0.679145	0.392087	...	0.265245
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	0.486180	...	0.261057

284807 rows × 31 columns

In [3]:
```
df.shape
```

Out [3]:
```
(284807, 31)
```

In [4]:
```
df.head()
```

Out [4]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278

5 rows × 31 columns

In [5]:
```
df.isnull().sum()
```

Out [5]:
```
Time      0
V1        0
V2        0
V3        0
V4        0
V5        0
V6        0
V7        0
V8        0
V9        0
V10       0
V11       0
V12       0
V13       0
V14       0
V15       0
V16       0
V17       0
V18       0
V19       0
V20       0
V21       0
V22       0
V23       0
V24       0
V25       0
V26       0
V27       0
```

```
V28       0
Amount    0
Class     0
dtype: int64
```

In [6]:
```
X = df.drop('Class', axis=1)
Y = df['Class']
```

In [7]:
```
X
```

Out [7]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V20
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	0.251412
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.069083
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.524980
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.208038
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	0.408542
...	...	...	...	...	...	...	...	...	...	...	...	...
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	1.914428	...	1.475829
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	0.584800	...	0.059616
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	0.432454	...	0.001396
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180	0.679145	0.392087	...	0.127434
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	0.486180	...	0.382948

284807 rows × 30 columns

In [8]:
```
Y
```

Out [8]:
```
0         0
1         0
2         0
3         0
4         0
         ..
284802    0
284803    0
284804    0
284805    0
284806    0
Name: Class, Length: 284807, dtype: int64
```

In [9]:
```
print(X.shape, Y.shape)
```

```
(284807, 30) (284807,)
```

In [10]:
```
df
```

Out [10]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431
...	...	...	...	...	...	...	...	...	...	...	...	...
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	1.914428	...	0.213454
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	0.584800	...	0.214205
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	0.432454	...	0.232045
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180	0.679145	0.392087	...	0.265245
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	0.486180	...	0.261057

284807 rows × 31 columns

In [11]:
```
df.groupby('Class').mean()
```

Out [11]:

Class	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V2
0	94838.202258	0.008258	-0.006271	0.012171	-0.007860	0.005453	0.002419	0.009637	-0.000987	0.004467	...	-0.00064
1	80746.806911	-4.771948	3.623778	-7.033281	4.542029	-3.151225	-1.397737	-5.568731	0.570636	-2.581123	...	0.372319

2 rows × 30 columns

In [12]:
```
#Train and Test Data Split
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, random_state=42
```

In [13]:
```
print(Y.shape, Y_train.shape, Y_test.shape)
```

```
(284807,) (227845,) (56962,)
```

In [14]:
```
print(X_train.mean(), X_test.mean(), X.mean())
```

```
Time      94885.093704
V1            0.000795
V2           -0.000481
V3           -0.000383
V4           -0.000020
V5            0.000175
V6           -0.001171
V7            0.000077
V8           -0.000434
V9            0.000702
V10          -0.000389
V11          -0.000787
V12           0.002718
V13          -0.000493
V14          -0.000059
V15           0.000744
V16          -0.001070
V17           0.000390
V18           0.000056
V19           0.000664
V20          -0.000999
V21           0.000370
V22           0.000303
V23           0.000050
V24          -0.000277
V25          -0.000655
V26           0.000126
V27          -0.000070
V28           0.000154
Amount       88.176298
dtype: float64 Time      94528.926811
V1           -0.003181
V2            0.001924
V3            0.001532
V4            0.000079
V5           -0.000698
V6            0.004685
V7           -0.000307
V8            0.001736
V9           -0.002807
V10           0.001558
V11           0.003149
V12          -0.010872
V13           0.001972
V14           0.000234
V15          -0.002976
V16           0.004280
V17          -0.001561
V18          -0.000225
V19          -0.002654
V20           0.003996
V21          -0.001479
V22          -0.001211
V23          -0.000202
V24           0.001109
V25           0.002622
V26          -0.000504
V27           0.000282
V28          -0.000615
Amount       89.042896
dtype: float64 Time      9.481386e+04
V1        1.168375e-15
V2        3.416908e-16
V3       -1.379537e-15
V4        2.074095e-15
V5        9.604066e-16
V6        1.487313e-15
V7       -5.556467e-16
V8        1.213481e-16
V9       -2.406331e-15
V10       2.239053e-15
V11       1.673327e-15
V12      -1.247012e-15
V13       8.190001e-16
V14       1.207294e-15
V15       4.887456e-15
V16       1.437716e-15
V17      -3.772171e-16
V18       9.564149e-16
V19       1.039917e-15
V20       6.406204e-16
V21       1.654067e-16
V22      -3.568593e-16
V23       2.578648e-16
V24       4.473266e-15
V25       5.340915e-16
V26       1.683437e-15
V27      -3.660091e-16
V28      -1.227390e-16
Amount    8.834962e+01
dtype: float64
```

In [15]:
```
#Decision Tree
```

```
from sklearn.tree import DecisionTreeClassifier
```

In [16]:
```
# Initialize the model
model = DecisionTreeClassifier(random_state=42)
```

In [17]:
```
# Train the model
model.fit(X_train, Y_train)
```

Out [17]:
```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

In [18]:
```
# Predictions
prediction = model.predict(X_test)
```

In [19]:
```
# Accuracy
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(Y_test, prediction)
print(f'Accuracy: {accuracy}')
```

```
Accuracy: 0.9991397773954567
```

Decision Tree Classification has 99.91 percentage accuracy.