In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from pylab import rcParams
import warnings
warnings.filterwarnings('ignore')
```

In [7]:
```
data =pd.read_csv("creditcard.csv/creditcard.csv")
data
```

Out[7]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.3
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.2
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.5
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.3
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.8
5	2.0	-0.425966	0.960523	1.141109	-0.168252	0.420987	-0.029728	0.476201	0.260314	-0.5
6	4.0	1.229658	0.141004	0.045371	1.202613	0.191881	0.272708	-0.005159	0.081213	0.4
7	7.0	-0.644269	1.417964	1.074380	-0.492199	0.948934	0.428118	1.120631	-3.807864	0.6
8	7.0	-0.894286	0.286157	-0.113192	-0.271526	2.669599	3.721818	0.370145	0.851084	-0.3
9	9.0	-0.338262	1.119593	1.044367	-0.222187	0.499361	-0.246761	0.651583	0.069539	-0.7
10	10.0	1.449044	-1.176339	0.913860	-1.375667	-1.971383	-0.629152	-1.423236	0.048456	-1.7
11	10.0	0.384978	0.616109	-0.874300	-0.094019	2.924584	3.317027	0.470455	0.538247	-0.5
12	10.0	1.249999	-1.221637	0.383930	-1.234899	-1.485419	-0.753230	-0.689405	-0.227487	-2.0
13	11.0	1.069374	0.287722	0.828613	2.712520	-0.178398	0.337544	-0.096717	0.115982	-0.2
14	12.0	-2.791855	-0.327771	1.641750	1.767473	-0.136588	0.807596	-0.422911	-1.907107	0.7
15	12.0	-0.752417	0.345485	2.057323	-1.468643	-1.158394	-0.077850	-0.608581	0.003603	-0.4
16	12.0	1.103215	-0.040296	1.267332	1.289091	-0.735997	0.288069	-0.586057	0.189380	0.7
17	13.0	-0.436905	0.918966	0.924591	-0.727219	0.915679	-0.127867	0.707642	0.087962	-0.6
18	14.0	-5.401258	-5.450148	1.186305	1.736239	3.049106	-1.763406	-1.559738	0.160842	1.2
19	15.0	1.492936	-1.029346	0.454795	-1.438026	-1.555434	-0.720961	-1.080664	-0.053127	-1.9
20	16.0	0.694885	-1.361819	1.029221	0.834159	-1.191209	1.309109	-0.878586	0.445290	-0.4
21	17.0	0.962496	0.328461	-0.171479	2.109204	1.129566	1.696038	0.107712	0.521502	-1.1
22	18.0	1.166616	0.502120	-0.067300	2.261569	0.428804	0.089474	0.241147	0.138082	-0.9
23	18.0	0.247491	0.277666	1.185471	-0.092603	-1.314394	-0.150116	-0.946365	-1.617935	1.5
24	22.0	-1.946525	-0.044901	-0.405570	-1.013057	2.941968	2.955053	-0.063063	0.855546	0.0
25	22.0	-2.074295	-0.121482	1.322021	0.410008	0.295198	-0.959537	0.543985	-0.104627	0.4
26	23.0	1.173285	0.353498	0.283905	1.133563	-0.172577	-0.916054	0.369025	-0.327260	-0.2
27	23.0	1.322707	-0.174041	0.434555	0.576038	-0.836758	-0.831083	-0.264905	-0.220982	-1.0
28	23.0	-0.414289	0.905437	1.727453	1.473471	0.007443	-0.200331	0.740228	-0.029247	-0.5
29	23.0	1.059387	-0.175319	1.266130	1.186110	-0.786002	0.578435	-0.767084	0.401046	0.6
...	...	...	...	...	...	...	...	...	...	...
284777	172764.0	2.079137	-0.028723	-1.343392	0.358000	-0.045791	-1.345452	0.227476	-0.378355	0.6
284778	172764.0	-0.764523	0.588379	-0.907599	-0.418847	0.901528	-0.760802	0.758545	0.414698	-0.7
284779	172766.0	1.975178	-0.616244	-2.628295	-0.406246	2.327804	3.664740	-0.533297	0.842937	1.1
284780	172766.0	-1.727503	1.108356	2.219561	1.148583	-0.884199	0.793083	-0.527298	0.866429	0.8
284781	172766.0	-1.139015	-0.155510	1.894478	-1.138957	1.451777	0.093598	0.191353	0.092211	-0.0
284782	172767.0	-0.268061	2.540315	-1.400915	4.846661	0.639105	0.186479	-0.045911	0.936448	-2.4
284783	172768.0	-1.796092	1.929178	-2.828417	-1.689844	2.199572	3.123732	-0.270714	1.657495	0.4

	Time	V1	V2	V3	V4	V5	V6	V7	V8	
284784	172768.0	-0.669662	0.923769	-1.543167	-1.560729	2.833960	3.240843	0.181576	1.282746	-0.8
284785	172768.0	0.032887	0.545338	-1.185844	-1.729828	2.932315	3.401529	0.337434	0.925377	-0.1
284786	172768.0	-2.076175	2.142238	-2.522704	-1.888063	1.982785	3.732950	-1.217430	-0.536644	0.2
284787	172769.0	-1.029719	-1.110670	-0.636179	-0.840816	2.424360	-2.956733	0.283610	-0.332656	-0.2
284788	172770.0	2.007418	-0.280235	-0.208113	0.335261	-0.715798	-0.751373	-0.458972	-0.140140	0.9
284789	172770.0	-0.446951	1.302212	-0.168583	0.981577	0.578957	-0.605641	1.253430	-1.042610	-0.4
284790	172771.0	-0.515513	0.971950	-1.014580	-0.677037	0.912430	-0.316187	0.396137	0.532364	-0.2
284791	172774.0	-0.863506	0.874701	0.420358	-0.530365	0.356561	-1.046238	0.757051	0.230473	-0.5
284792	172774.0	-0.724123	1.485216	-1.132218	-0.607190	0.709499	-0.482638	0.548393	0.343003	-0.2
284793	172775.0	1.971002	-0.699067	-1.697541	-0.617643	1.718797	3.911336	-1.259306	1.056209	1.3
284794	172777.0	-1.266580	-0.400461	0.956221	-0.723919	1.531993	-1.788600	0.314741	0.004704	0.0
284795	172778.0	-12.516732	10.187818	-8.476671	-2.510473	-4.586669	-1.394465	-3.632516	5.498583	4.8
284796	172780.0	1.884849	-0.143540	-0.999943	1.506772	-0.035300	-0.613638	0.190241	-0.249058	0.6
284797	172782.0	-0.241923	0.712247	0.399806	-0.463406	0.244531	-1.343668	0.929369	-0.206210	0.1
284798	172782.0	0.219529	0.881246	-0.635891	0.960928	-0.152971	-1.014307	0.427126	0.121340	-0.2
284799	172783.0	-1.775135	-0.004235	1.189786	0.331096	1.196063	5.519980	-1.518185	2.080825	1.1
284800	172784.0	2.039560	-0.175233	-1.196825	0.234580	-0.008713	-0.726571	0.017050	-0.118228	0.4
284801	172785.0	0.120316	0.931005	-0.546012	-0.745097	1.130314	-0.235973	0.812722	0.115093	-0.2
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	1.9
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	0.5
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	0.4
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180	0.679145	0.3
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	0.4

284807 rows × 31 columns

In [8]:
```
data.head()
```

Out[8]:

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

5 rows × 31 columns

In [9]:
```
data.isnull().sum()
```

Out[9]:
```
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 [10]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
Time      284807 non-null float64
V1        284807 non-null float64
V2        284807 non-null float64
V3        284807 non-null float64
V4        284807 non-null float64
V5        284807 non-null float64
V6        284807 non-null float64
V7        284807 non-null float64
V8        284807 non-null float64
V9        284807 non-null float64
V10       284807 non-null float64
V11       284807 non-null float64
V12       284807 non-null float64
V13       284807 non-null float64
V14       284807 non-null float64
V15       284807 non-null float64
V16       284807 non-null float64
V17       284807 non-null float64
V18       284807 non-null float64
V19       284807 non-null float64
V20       284807 non-null float64
V21       284807 non-null float64
V22       284807 non-null float64
V23       284807 non-null float64
V24       284807 non-null float64
V25       284807 non-null float64
V26       284807 non-null float64
V27       284807 non-null float64
V28       284807 non-null float64
Amount    284807 non-null float64
Class     284807 non-null int64
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

In [11]:
```
data.describe().T.head()
```

Out[11]:

	count	mean	std	min	25%	50%	75%	
Time	284807.0	9.481386e+04	47488.145955	0.000000	54201.500000	84692.000000	139320.500000	172792.
V1	284807.0	3.919560e-15	1.958696	-56.407510	-0.920373	0.018109	1.315642	2.
V2	284807.0	5.688174e-16	1.651309	-72.715728	-0.598550	0.065486	0.803724	22.
V3	284807.0	-8.769071e-15	1.516255	-48.325589	-0.890365	0.179846	1.027196	9.
V4	284807.0	2.782312e-15	1.415869	-5.683171	-0.848640	-0.019847	0.743341	16.

In [12]:
```
data.shape
```

Out[12]: (284807, 31)

In [13]:
```
data.columns
```

Out[13]:
```
Index(['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',
       'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V20',
       'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'Amount',
       'Class'],
      dtype='object')
```

In [14]:
```
fraud_cases=len(data[data['Class']==1])
print(' Number of Fraud Cases:',fraud_cases)
```

```
 Number of Fraud Cases: 492
```

In [15]:
```
non_fraud_cases=len(data[data['Class']==0])
print('Number of Non Fraud Cases:',non_fraud_cases)
```

```
Number of Non Fraud Cases: 284315
```

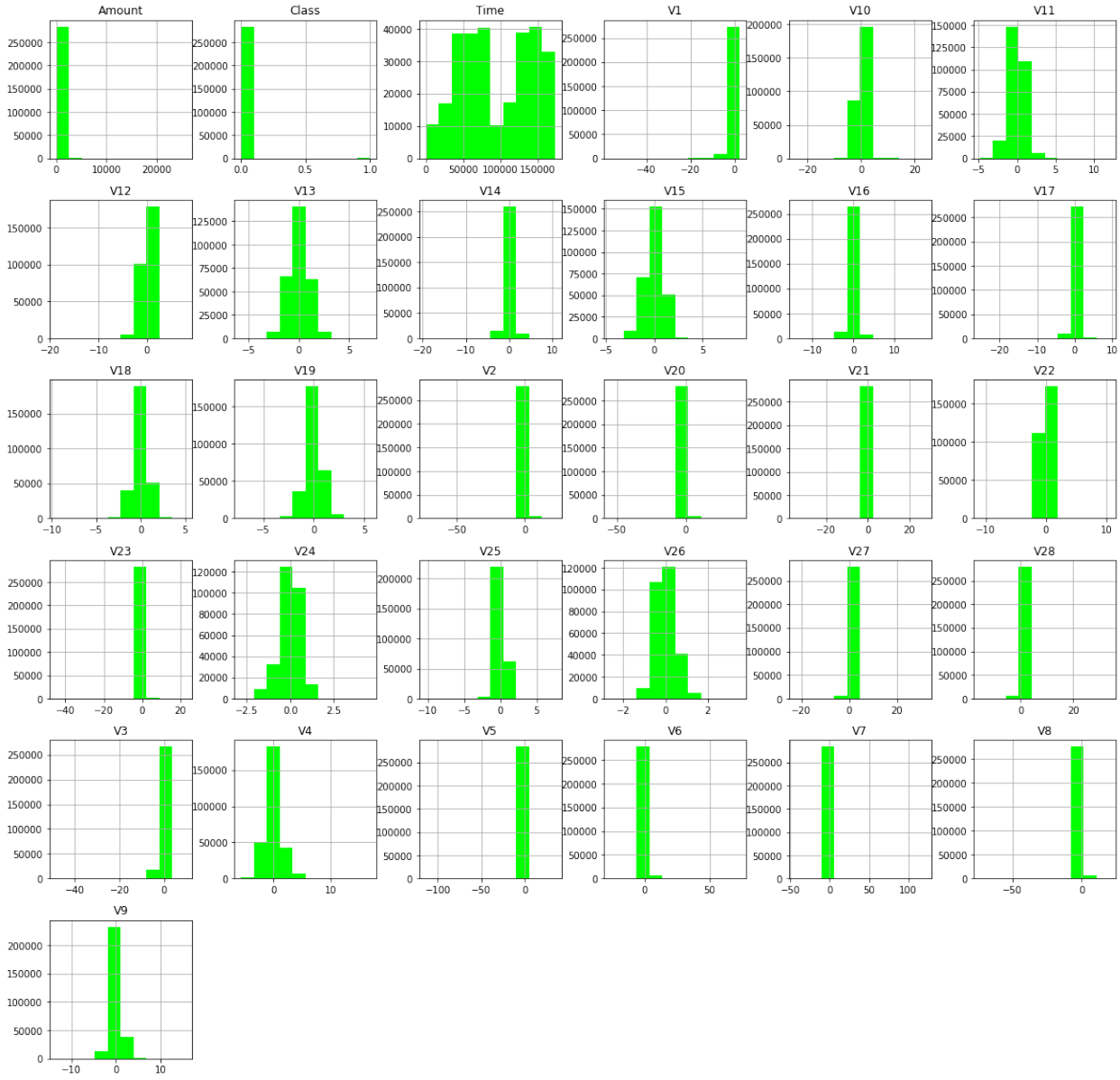
In [16]:
```
fraud=data[data['Class']==1]
genuine=data[data['Class']==0]
fraud.Amount.describe()
```

Out[16]:
```
count     492.000000
mean      122.211321
std       256.683288
min         0.000000
25%         1.000000
50%         9.250000
75%       105.890000
max      2125.870000
Name: Amount, dtype: float64
```

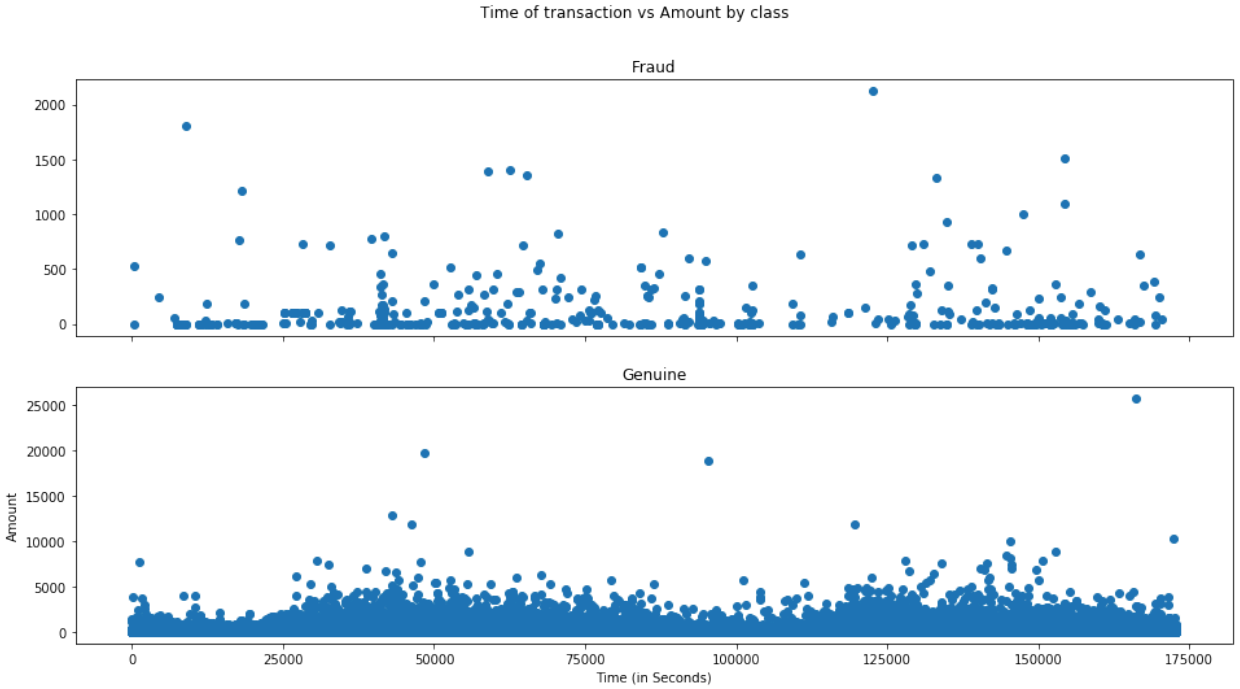
In [17]:
```
genuine.Amount.describe()
```

Out[17]:
```
count    284315.000000
mean         88.291022
std         250.105092
min           0.000000
25%           5.650000
50%          22.000000
75%          77.050000
max       25691.160000
Name: Amount, dtype: float64
```

In [18]:
```
data.hist(figsize=(20,20),color='lime')
plt.show()
```

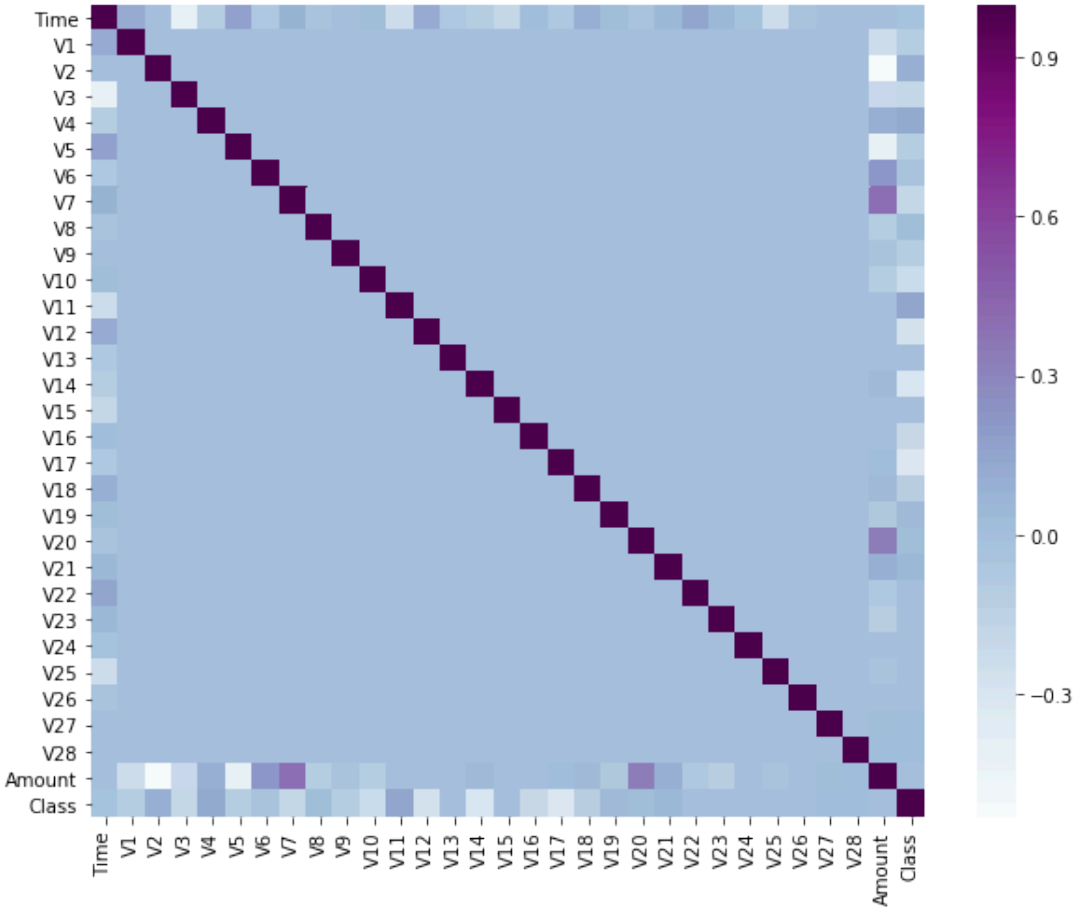


In [19]:
```
rcParams['figure.figsize'] = 16, 8
f,(ax1, ax2) = plt.subplots(2, 1, sharex=True)
f.suptitle('Time of transaction vs Amount by class')
ax1.scatter(fraud.Time, fraud.Amount)
ax1.set_title('Fraud')
ax2.scatter(genuine.Time, genuine.Amount)
ax2.set_title('Genuine')
plt.xlabel('Time (in Seconds)')
plt.ylabel('Amount')
plt.show()
```



In [20]:
```
plt.figure(figsize=(10,8))
corr=data.corr()
sns.heatmap(corr,cmap='BuPu')
```

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x266bc582dd8>



In [21]:
```
from sklearn.model_selection import train_test_split
```

In [22]:
```
X=data.drop(['Class'],axis=1)
y=data['Class']
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.30,random_state=123)
```

In [23]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [24]:
```
rfc=RandomForestClassifier()
model=rfc.fit(X_train,y_train)
prediction=model.predict(X_test)
```

In [25]:
```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test, prediction)
print(f"The accuracy using Random Forest Classifier is {accuracy:2f}")
```

```
The accuracy using Random Forest Classifier is 0.999508
```

In [26]:
```
from sklearn.linear_model import LogisticRegression
```

In [27]:
```
X1=data.drop(['Class'],axis=1)
y1=data['Class']
X1_train,X1_test,y1_train,y1_test=train_test_split(X1,y1,test_size=0.30,random_state=123
```

In [28]:
```
lr=LogisticRegression()
model2=lr.fit(X1_train,y1_train)
prediction2=model2.predict(X1_test)
```

In [29]:
```
accuracy_score(y1_test,prediction2)
print(f"The accuracy using Logistic Regression is {accuracy}")
```

```
The accuracy using Logistic Regression is 0.9995084442259752
```

In [30]:
```
from sklearn.tree import DecisionTreeRegressor
```

In [31]:
```
X2=data.drop(['Class'],axis=1)
y2=data['Class']
dt=DecisionTreeRegressor()
X2_train,X2_test,y2_train,y2_test=train_test_split(X2,y2,test_size=0.3,random_state=123)
```

In [32]:
```
model3=dt.fit(X2_train,y2_train)
prediction3=model3.predict(X2_test)
```

In [33]:
```
accuracy_score(y2_test,prediction3)
print(f"The accuracy using Decision Tree  is {accuracy}")
```

```
The accuracy using Decision Tree  is 0.9995084442259752
```

In [ ]:
```

```