ion-using-random-forest-classifier

September 5, 2023

####Name:Manohar Goud ##Roll No:21X05A6706 ##Branch:lV year cse(data science) ##College:Narsimha Reddy Engineering College

##project title:Analysis and prediction of creditcard.csv

##project statement: There are so many frauds which is going on the scoiety by credit card.By collecting the data and making effiecient analysis and classifying the fradulent transactions and valid transactions

[ ]:
```
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.metrics import accuracy_score,confusion_matrix
from sklearn.metrics import precision_score,recall_score
from sklearn.ensemble import RandomForestClassifier
```

[ ]:
```
df=pd.read_csv('creditcard.csv')
df.head()
```

[ ]:
```
   Time        V1        V2        V3        V4        V5        V6        V7  \
0     0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1     0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2     1 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3     1 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
4     2 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         V8        V9  …       V21       V22       V23       V24       V25  \
0  0.098698  0.363787  … -0.018307  0.277838 -0.110474  0.066928  0.128539
1  0.085102 -0.255425  … -0.225775 -0.638672  0.101288 -0.339846  0.167170
2  0.247676 -1.514654  …  0.247998  0.771679  0.909412 -0.689281 -0.327642
3  0.377436 -1.387024  … -0.108300  0.005274 -0.190321 -1.175575  0.647376
4 -0.270533  0.817739  … -0.009431  0.798278 -0.137458  0.141267 -0.206010

        V26       V27       V28  Amount  Class
0 -0.189115  0.133558 -0.021053  149.62    0.0
1  0.125895 -0.008983  0.014724    2.69    0.0
2 -0.139097 -0.055353 -0.059752  378.66    0.0
```

```
3 -0.221929  0.062723  0.061458  123.50    0.0
4  0.502292  0.219422  0.215153   69.99    0.0

[5 rows x 31 columns]
```

[ ]:
```
df.shape
```

[ ]:
```
(7973, 31)
```

[ ]:
```
df.describe()
```

[ ]:
```
               Time           V1           V2           V3           V4  \
count   7973.000000  7973.000000  7973.000000  7973.000000  7973.000000
mean    4257.151261    -0.299740     0.295226     0.899355     0.215736
std     3198.964299     1.498341     1.283914     1.090297     1.447057
min        0.000000   -23.066842   -25.640527   -12.389545    -4.657545
25%     1531.000000    -1.046362    -0.237359     0.372435    -0.687521
50%     3635.000000    -0.416341     0.335446     0.948695     0.223379
75%     6662.000000     1.122758     0.950582     1.597949     1.131542
max    10981.000000     1.685314     8.261750     4.101716     7.380245

                V5           V6           V7           V8           V9  …  \
count  7973.000000  7973.000000  7973.000000  7973.000000  7973.000000  …
mean     -0.025285     0.157286    -0.026445    -0.070525     0.655244  …
std       1.167218     1.325015     1.063709     1.332568     1.156618  …
min     -32.092129    -7.574798   -12.968670   -23.632502    -3.878658  …
25%      -0.630525    -0.655399    -0.517733    -0.199794    -0.085635  …
50%      -0.107337    -0.148669     0.004732     0.016128     0.613170  …
75%       0.405082     0.555200     0.527353     0.307111     1.294087  …
max      11.974269    21.393069    34.303177     3.877662    10.392889  …

               V21          V22          V23          V24          V25  \
count  7972.000000  7972.000000  7972.000000  7972.000000  7972.000000
mean     -0.053715    -0.165799    -0.035174     0.025977     0.088893
std       0.953498     0.654858     0.488322     0.601760     0.427505
min     -11.468435    -8.527145   -15.144340    -2.512377    -2.577363
25%      -0.271837    -0.581473    -0.182989    -0.340419    -0.161009
50%      -0.130344    -0.167048    -0.046107     0.089606     0.115418
75%       0.044823     0.250886     0.086806     0.421015     0.361249
max      22.588989     4.534454    13.876221     3.200201     5.525093

               V26          V27          V28       Amount        Class
count  7972.000000  7972.000000  7972.000000  7972.000000  7972.000000
mean      0.020256     0.016150     0.001161    65.413540     0.003136
std       0.517409     0.403570     0.275976   194.911169     0.055915
min      -1.338556    -7.976100    -3.054085     0.000000     0.000000
25%      -0.363180    -0.063198    -0.019081     4.617500     0.000000
```

```
50%     -0.015260     0.007101     0.018443     15.950000     0.000000
75%      0.329322     0.144700     0.080563     54.910000     0.000000
max      3.517346     4.173387     4.860769   7712.430000     1.000000

[8 rows x 31 columns]
```

[ ]:
```python
#knowing abiut the data
```

[ ]:
```python
fraud=df[df['Class']==1]
print(fraud)
```

```
      Time        V1        V2        V3        V4        V5        V6  \
541    406 -2.312227  1.951992 -1.609851  3.997906 -0.522188 -1.426545
623    472 -3.043541 -3.157307  1.088463  2.288644  1.359805 -1.064823
4920  4462 -2.303350  1.759247 -0.359745  2.330243 -0.821628 -0.075788
6108  6986 -4.397974  1.358367 -2.592844  2.679787 -1.128131 -1.706536
6329  7519  1.234235  3.019740 -4.304597  4.732795  3.624201 -1.357746
6331  7526  0.008430  4.137837 -6.240697  6.675732  0.768307 -3.353060
6334  7535  0.026779  4.132464 -6.560600  6.348557  1.329666 -2.513479
6336  7543  0.329594  3.712889 -5.775935  6.078266  1.667359 -2.420168
6338  7551  0.316459  3.809076 -5.615159  6.047445  1.554026 -2.651353
6427  7610  0.725646  2.300894 -5.329976  4.007683 -1.730411 -1.732193
6446  7672  0.702710  2.426433 -5.234513  4.416661 -2.170806 -2.667554
6472  7740  1.023874  2.001485 -4.769752  3.819195 -1.271754 -1.734662
6529  7891 -1.585505  3.261585 -4.137422  2.357096 -1.405043 -1.879437
6609  8090 -1.783229  3.402794 -3.822742  2.625368 -1.976415 -2.731689
6641  8169  0.857321  4.093912 -7.423894  7.380245  0.973366 -2.730762
6717  8408 -1.813280  4.917851 -5.926130  5.701500  1.204393 -3.035138
6719  8415 -0.251471  4.313523 -6.891438  6.796797  0.616297 -2.966327
6734  8451  0.314597  2.660670 -5.920037  4.522500 -2.315027 -2.278352
6774  8528  0.447396  2.481954 -5.660814  4.455923 -2.443780 -2.185040
6820  8614 -2.169929  3.639654 -4.508498  2.730668 -2.122693 -2.341017
6870  8757 -1.863756  3.442644 -4.468260  2.805336 -2.118412 -2.332285
6882  8808 -4.617217  1.695694 -3.114372  4.328199 -1.873257 -0.989908
6899  8878 -2.661802  5.856393 -7.653616  6.379742 -0.060712 -3.131550
6903  8886 -2.535852  5.793644 -7.618463  6.395830 -0.065210 -3.136372
6971  9064 -3.499108  0.258555 -4.489558  4.853894 -6.974522  3.628382

            V7        V8        V9  …       V21       V22       V23  \
541  -2.537387  1.391657 -2.770089  …  0.517232 -0.035049 -0.465211
623   0.325574 -0.067794 -0.270953  …  0.661696  0.435477  1.375966
4920  0.562320 -0.399147 -0.238253  … -0.294166 -0.932391  0.172726
6108 -3.496197 -0.248778 -0.247768  …  0.573574  0.176968 -0.436207
6329  1.713445 -0.496358 -1.282858  … -0.379068 -0.704181 -0.656805
6331 -1.631735  0.154612 -2.795892  …  0.364514 -0.608057 -0.539528
6334 -1.689102  0.303253 -3.139409  …  0.370509 -0.576752 -0.669605
6336 -0.812891  0.133080 -2.214311  …  0.156617 -0.652450 -0.551572
```

```
6338 -0.746579  0.055586 -2.678679  …  0.208828 -0.511747 -0.583813
6427 -3.968593  1.063728 -0.486097  …  0.589669  0.109541  0.601045
6446 -3.878088  0.911337 -0.166199  …  0.551180 -0.009802  0.721698
6472 -3.059245  0.889805  0.415382  …  0.343283 -0.054196  0.709654
6529 -3.513687  1.515607 -1.207166  …  0.501543 -0.546869 -0.076584
6609 -3.430559  1.413204 -0.776941  …  0.454032 -0.577526  0.045967
6641 -1.496497  0.543015 -2.351190  …  0.375026  0.145400  0.240603
6717 -1.713402  0.561257 -3.796354  …  0.615642 -0.406427 -0.737018
6719 -2.436653  0.489328 -3.371639  …  0.536892 -0.546126 -0.605240
6734 -4.684054  1.202270 -0.694696  …  0.743314  0.064038  0.677842
6774 -4.716143  1.249803 -0.718326  …  0.756053  0.140168  0.665411
6820 -4.235253  1.703538 -1.305279  …  0.645103 -0.503529 -0.000523
6870 -4.261237  1.701682 -1.439396  …  0.667927 -0.516242 -0.012218
6882 -4.577265  0.472216  0.472017  …  0.481830  0.146023  0.117039
6899 -3.103570  1.778492 -3.831154  …  0.734775 -0.435901 -0.384766
6903 -3.104557  1.823233 -3.878658  …  0.716720 -0.448060 -0.402407
6971  5.431271 -1.946734 -0.775680  … -1.052368  0.204817 -2.119007

           V24       V25       V26       V27       V28   Amount  Class
541   0.320198  0.044519  0.177840  0.261145 -0.143276     0.00    1.0
623  -0.293803  0.279798 -0.145362 -0.252773  0.035764   529.00    1.0
4920 -0.087330 -0.156114 -0.542628  0.039566 -0.153029   239.93    1.0
6108 -0.053502  0.252405 -0.657488 -0.827136  0.849573    59.00    1.0
6329 -1.632653  1.488901  0.566797 -0.010016  0.146793     1.00    1.0
6331  0.128940  1.488481  0.507963  0.735822  0.513574     1.00    1.0
6334 -0.759908  1.605056  0.540675  0.737040  0.496699     1.00    1.0
6336 -0.716522  1.415717  0.555265  0.530507  0.404474     1.00    1.0
6338 -0.219845  1.474753  0.491192  0.518868  0.402528     1.00    1.0
6427 -0.364700 -1.843078  0.351909  0.594550  0.099372     1.00    1.0
6446  0.473246 -1.959304  0.319476  0.600485  0.129305     1.00    1.0
6472 -0.372216 -2.032068  0.366778  0.395171  0.020206     1.00    1.0
6529 -0.425550  0.123644  0.321985  0.264028  0.132817     1.00    1.0
6609  0.461700  0.044146  0.305704  0.530981  0.243746     1.00    1.0
6641 -0.234649 -1.004881  0.435832  0.618324  0.148469     1.00    1.0
6717 -0.279642  1.106766  0.323885  0.894767  0.569519     1.00    1.0
6719 -0.263743  1.539916  0.523574  0.891025  0.572741     1.00    1.0
6734  0.083008 -1.911034  0.322188  0.620867  0.185030     1.00    1.0
6774  0.131464 -1.908217  0.334808  0.748534  0.175414     1.00    1.0
6820  0.071696  0.092007  0.308498  0.552591  0.298954     1.00    1.0
6870  0.070614  0.058504  0.304883  0.418012  0.208858     1.00    1.0
6882 -0.217565 -0.138776 -0.424453 -1.002041  0.890780     1.10    1.0
6899 -0.286016  1.007934  0.413196  0.280284  0.303937     1.00    1.0
6903 -0.288835  1.011752  0.425965  0.413140  0.308205     1.00    1.0
6971  0.170279 -0.393844  0.296367  1.985913 -0.900452  1809.68    1.0

[25 rows x 31 columns]
```

[ ]:
```
valid=df[df['Class']==0]
print(valid)
```

```
       Time        V1        V2        V3        V4        V5        V6  \
0         0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388
1         0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361
2         1 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499
3         1 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203
4         2 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921
…       …       …       …       …       …       …       …
7967  10980 -0.046786  0.030050  2.037794 -0.670130 -0.727283 -0.588537
7968  10980  1.284388 -0.013181  0.646174  0.198985 -0.568675 -0.526121
7969  10981  1.190428 -0.122329  0.954945  0.267101 -0.971026 -0.652279
7970  10981 -0.725175  0.298202  1.824761 -2.587170  0.283605 -0.016617
7971  10981  1.226153 -0.129645  0.735197  0.142752 -0.703245 -0.349641

            V7        V8        V9  …       V21       V22       V23  \
0     0.239599  0.098698  0.363787  … -0.018307  0.277838 -0.110474
1    -0.078803  0.085102 -0.255425  … -0.225775 -0.638672  0.101288
2     0.791461  0.247676 -1.514654  …  0.247998  0.771679  0.909412
3     0.237609  0.377436 -1.387024  … -0.108300  0.005274 -0.190321
4     0.592941 -0.270533  0.817739  … -0.009431  0.798278 -0.137458
…         …       …       …   …       …       …       …
7967 -0.067966 -0.370767  0.228931  …  0.264364  1.078896 -0.097768
7968 -0.448235 -0.167709  1.773223  … -0.101868 -0.030298 -0.081412
7969 -0.612992 -0.003909  1.633117  … -0.015001  0.127027  0.012079
7970  0.153659  0.045084 -0.197611  … -0.017097 -0.070535 -0.442861
7971 -0.612641  0.020507  1.648986  … -0.047936  0.040196 -0.057391

           V24       V25       V26       V27       V28  Amount  Class
0     0.066928  0.128539 -0.189115  0.133558 -0.021053  149.62    0.0
1    -0.339846  0.167170  0.125895 -0.008983  0.014724    2.69    0.0
2    -0.689281 -0.327642 -0.139097 -0.055353 -0.059752  378.66    0.0
3    -1.175575  0.647376 -0.221929  0.062723  0.061458  123.50    0.0
4     0.141267 -0.206010  0.502292  0.219422  0.215153   69.99    0.0
…         …       …       …       …       …     …     …
7967  0.375679 -0.500253 -0.159051 -0.018267 -0.061794   39.00    0.0
7968 -0.123281  0.278808  1.064001 -0.090181  0.000481   15.95    0.0
7969  0.534409  0.112179  1.004483 -0.100188 -0.004774   14.95    0.0
7970 -0.895837  0.624743 -0.510601 -0.031142  0.025564   12.95    0.0
7971 -0.012386  0.187685  1.037786 -0.100081 -0.009869   15.95    0.0

[7947 rows x 31 columns]
```

[ ]:
```
outliers=len(fraud)/len(valid)
print(outliers/100)
```

```
3.1458411979363285e-05
```

[ ]:
```
print('fraud cases are',len(fraud))
```

```
fraud cases are 25
```

[ ]:
```
print('successful valid transactions are',len(valid))
```

```
successful valid transactions are 7947
```

[ ]:
```
#amount for fraud cases
```

[ ]:
```
fraud.Amount.describe()
```

[ ]:
```
count      25.000000
mean      106.308400
std       372.676883
min         0.000000
25%         1.000000
50%         1.000000
75%         1.000000
max      1809.680000
Name: Amount, dtype: float64
```

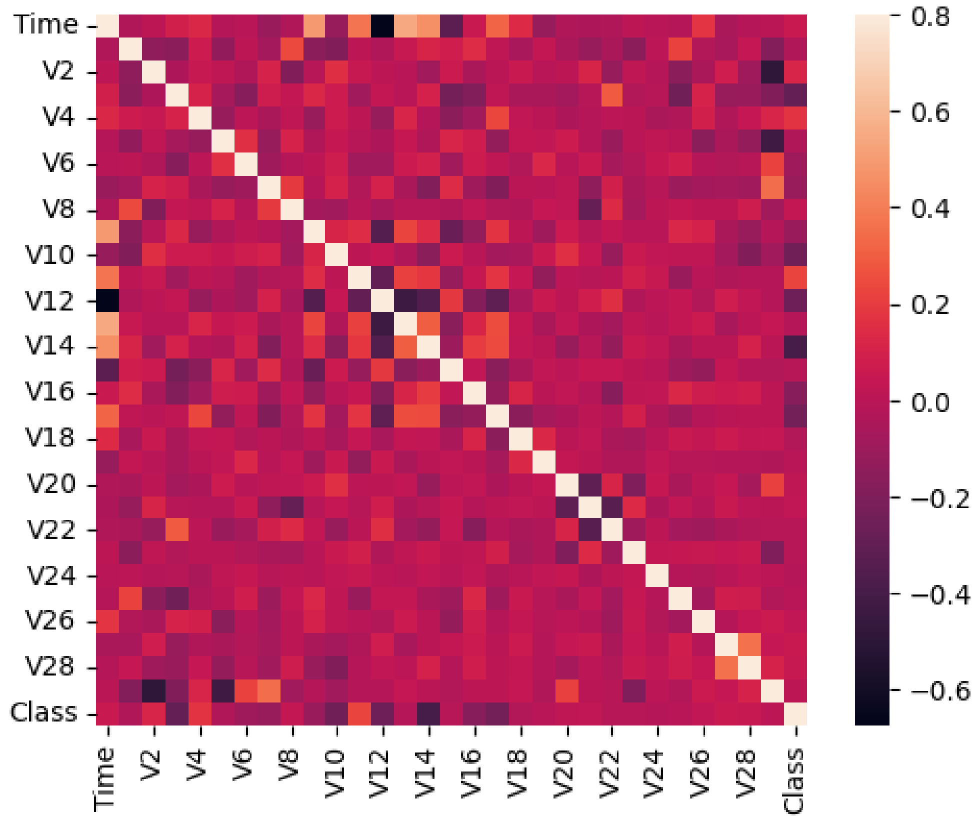
[ ]:
```
#amount fo valid transaction cases
```

[ ]:
```
valid.Amount.describe()
```

[ ]:
```
count    7947.000000
mean       65.284891
std       194.126547
min         0.000000
25%         4.795000
50%        15.950000
75%        54.990000
max      7712.430000
Name: Amount, dtype: float64
```

[ ]:
```
#knowing the correlation of the  features in the dataset
```

[ ]:
```
corr=df.corr()
sns.heatmap(corr,vmax=.8,square=True)
plt.show()
```



[25]:
```
#checking whether the columns consists of null values are not
```

[27]:
```
df.isnull().sum()
```

[27]:
```
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      1
```

```
V16       1
V17       1
V18       1
V19       1
V20       1
V21       1
V22       1
V23       1
V24       1
V25       1
V26       1
V27       1
V28       1
Amount    1
Class     1
dtype: int64
```

null values are present in the dataset hence it effects the accuracy score of the model and model may performs very poor hence cleaning of dataset is very mandatory for the dataset.removing the null values from the dataset to to train the model well and apply the suitable model for it

[28]:
```
df.dropna(inplace=True)
```

[30]:
```
df.isnull().sum()# no more null values are present in the dataset
```

[30]:
```
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
```

[31]:
```
#separating the x and y values to train and test the dataset and to apply the␣
↪randomforestclassifier
```

[32]:
```
x=df.drop(['Class'],axis=1)
x
```

[32]:
```
       Time        V1        V2        V3        V4        V5        V6  \
0         0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388
1         0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361
2         1 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499
3         1 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203
4         2 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921
…       …         …         …         …         …         …         …
7967  10980 -0.046786  0.030050  2.037794 -0.670130 -0.727283 -0.588537
7968  10980  1.284388 -0.013181  0.646174  0.198985 -0.568675 -0.526121
7969  10981  1.190428 -0.122329  0.954945  0.267101 -0.971026 -0.652279
7970  10981 -0.725175  0.298202  1.824761 -2.587170  0.283605 -0.016617
7971  10981  1.226153 -0.129645  0.735197  0.142752 -0.703245 -0.349641

            V7        V8        V9  …       V20       V21       V22  \
0     0.239599  0.098698  0.363787  …  0.251412 -0.018307  0.277838
1    -0.078803  0.085102 -0.255425  … -0.069083 -0.225775 -0.638672
2     0.791461  0.247676 -1.514654  …  0.524980  0.247998  0.771679
3     0.237609  0.377436 -1.387024  … -0.208038 -0.108300  0.005274
4     0.592941 -0.270533  0.817739  …  0.408542 -0.009431  0.798278
…          …         …       …  …   …         …         …
7967 -0.067966 -0.370767  0.228931  …  0.322583  0.264364  1.078896
7968 -0.448235 -0.167709  1.773223  … -0.063281 -0.101868 -0.030298
7969 -0.612992 -0.003909  1.633117  … -0.150267 -0.015001  0.127027
7970  0.153659  0.045084 -0.197611  … -0.001388 -0.017097 -0.070535
7971 -0.612641  0.020507  1.648986  … -0.122552 -0.047936  0.040196

           V23       V24       V25       V26       V27       V28  Amount
0    -0.110474  0.066928  0.128539 -0.189115  0.133558 -0.021053  149.62
1     0.101288 -0.339846  0.167170  0.125895 -0.008983  0.014724    2.69
2     0.909412 -0.689281 -0.327642 -0.139097 -0.055353 -0.059752  378.66
```

```
3    -0.190321 -1.175575  0.647376 -0.221929  0.062723  0.061458  123.50
4    -0.137458  0.141267 -0.206010  0.502292  0.219422  0.215153   69.99
…         …         …         …         …         …         …      …
7967 -0.097768  0.375679 -0.500253 -0.159051 -0.018267 -0.061794   39.00
7968 -0.081412 -0.123281  0.278808  1.064001 -0.090181  0.000481   15.95
7969  0.012079  0.534409  0.112179  1.004483 -0.100188 -0.004774   14.95
7970 -0.442861 -0.895837  0.624743 -0.510601 -0.031142  0.025564   12.95
7971 -0.057391 -0.012386  0.187685  1.037786 -0.100081 -0.009869   15.95

[7972 rows x 30 columns]
```

[33]:
```
x.shape
```

[33]:
```
(7972, 30)
```

[35]:
```
y=df['Class']
y
```

[35]:
```
0       0.0
1       0.0
2       0.0
3       0.0
4       0.0
       …
7967    0.0
7968    0.0
7969    0.0
7970    0.0
7971    0.0
Name: Class, Length: 7972, dtype: float64
```

[36]:
```
y.shape
```

[36]:
```
(7972,)
```

it does not take any columns for p training and testing the data so we are only taking values without taking columns

[37]:
```
x1=x.values
x1
```

[37]:
```
array([[ 0.00000000e+00, -1.35980713e+00, -7.27811733e-02, …,
         1.33558377e-01, -2.10530535e-02,  1.49620000e+02],
       [ 0.00000000e+00,  1.19185711e+00,  2.66150712e-01, …,
        -8.98309914e-03,  1.47241692e-02,  2.69000000e+00],
       [ 1.00000000e+00, -1.35835406e+00, -1.34016307e+00, …,
        -5.53527940e-02, -5.97518406e-02,  3.78660000e+02],
```

```
       …,
       [ 1.09810000e+04,  1.19042824e+00, -1.22329144e-01, …,
        -1.00188315e-01, -4.77439733e-03,  1.49500000e+01],
       [ 1.09810000e+04, -7.25174766e-01,  2.98202350e-01, …,
        -3.11419393e-02,  2.55638666e-02,  1.29500000e+01],
       [ 1.09810000e+04,  1.22615304e+00, -1.29645121e-01, …,
        -1.00081361e-01, -9.86920840e-03,  1.59500000e+01]])
```

[39]:
```
y1=y.values
y1
```

[39]:
```
array([0., 0., 0., …, 0., 0., 0.])
```

[40]:
```
# training and testing the dataset
```

[41]:
```
x_train,x_test,y_train,y_test=train_test_split(x1,y1,test_size=0.
 ↪2,random_state=42)
```

[42]:
```
print(x_train)
```

```
[[ 1.17900000e+03  6.57389339e-01 -6.43789396e-01 …  1.14239002e-02
   7.49778628e-02  2.51350000e+02]
 [ 1.88000000e+02  1.16843339e+00  3.19977589e-01 …  2.18927266e-02
   1.93366400e-02  8.09000000e+00]
 [ 0.00000000e+00 -1.35980713e+00 -7.27811733e-02 …  1.33558377e-01
  -2.10530535e-02  1.49620000e+02]
 …
 [ 6.54000000e+02 -8.33568321e-01  6.06174188e-01 …  1.64383985e-01
   2.74361005e-01  9.90000000e+00]
 [ 1.05180000e+04 -2.26083429e+00 -7.58476478e-01 …  2.49838190e-01
  -5.77953345e-03  9.50000000e-01]
 [ 9.67300000e+03 -1.61547335e+00  1.50325911e+00 … -1.11060384e+00
   1.15793236e-01  2.99900000e+01]]
```

[43]:
```
print(x_test)
```

```
[[ 5.75300000e+03 -1.12863936e+00  1.24763953e+00 …  1.82763452e-01
   1.07998112e-01  5.90000000e+00]
 [ 4.69000000e+03 -1.30060458e+00  5.98826086e-01 … -8.28637702e-01
  -9.99499308e-02  1.56900000e+01]
 [ 2.94200000e+03 -4.55381586e-01  4.65230036e-01 …  4.40550611e-03
  -5.23812598e-02  9.48000000e+01]
 …
 [ 7.58000000e+03  1.13427464e+00  2.42404189e-01 … -3.27342627e-03
   1.71692354e-02  3.60000000e+01]
 [ 9.82800000e+03 -9.96630171e-01  1.16049279e+00 …  1.09208486e-01
   1.03158112e-01  8.76000000e+00]
```

```
[ 4.43700000e+03  1.28067328e+00  1.21095747e-01 …  4.90805040e-06
  4.75607379e-03  1.00000000e+00]]
```

[44]:
```
print(y_train)
```

```
[0. 0. 0. … 0. 0. 0.]
```

[45]:
```
print(y_test)
```

```
[0. 0. 0. … 0. 0. 0.]
```

##MODEL BUILDING

[46]:
```
rfc=RandomForestClassifier()
```

[47]:
```
rfc.fit(x_train,y_train)
```

[47]:
```
RandomForestClassifier()
```

##PREDICTING

[48]:
```
y_pred=rfc.predict(x_test)
print(y_pred)
```

```
[0. 0. 0. … 0. 0. 0.]
```

##ACCURACY SCORE , PREICISION,RECALL SCORE

[51]:
```
accuracy=accuracy_score(y_test,y_pred)
print('ACCURACY SCORE IS {}'.format(accuracy))
```

```
ACCURACY SCORE IS 1.0
```

[55]:
```
precision=precision_score(y_test,y_pred)
print('the precision score is {}'.format(precision))
```

```
the precision score is 1.0
```

[56]:
```
recall=recall_score(y_test,y_pred)
print('the recall score is {}'.format(recall))
```

```
the recall score is 1.0
```

##CONFUSION MATRIX

[60]:
```
cm=confusion_matrix(y_test,y_pred)
cm
```

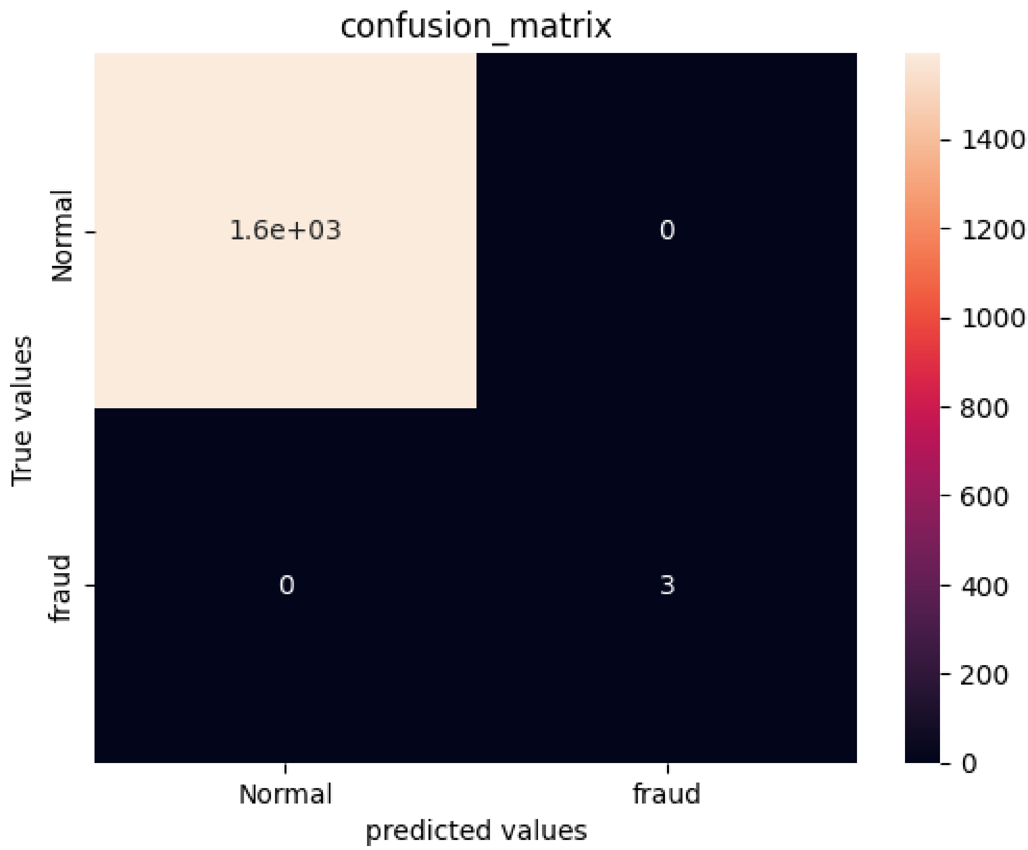
[60]:
```
array([[1592,    0],
       [   0,    3]])
```

[61]:

```
#visualizing the confusion matrix
```

[63]:

```
labels=['Normal','fraud']
sns.heatmap(cm,xticklabels=labels,yticklabels=labels,annot=True)
plt.title('confusion_matrix')
plt.ylabel('True values')
plt.xlabel('predicted values')
plt.show()
```



[ ]:

```

```

##conclusion:Credit card fraud is a serious issue that can lead to financial loss and identity theft. It is a type of fraud committed using a payment card, such as a credit card or debit card. The purpose may be to obtain goods or services or to make payment to another account, which is controlled by a criminals. Credit card fraud can occur when unauthorized users gain access to an individual's credit card information in order to make purchases, other transactions, or open new accounts 1. There are various techniques used for credit card frauds such as paper-based fraud, application fraud, financial fraud, skimming to commit fraud, etc2. To detect credit card fraud, refer to the sources mentioned in the results

```
[ ]:
```