Project Name = Credit Card Fraud Detection

Import Library

In [82]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [83]:
```
df=pd.read_csv("C:\\Users\\DISHA__COMPUTERS\\Desktop\\Internship\\5) credi
```

In [84]:
```
df
```

Out[84]:

	Time	V1	V2	V3	V4	V5	V6	V
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.23959
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.07880
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.79146
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.23760
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.59294
...	...	...	...	...	...	...	...	
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.91821
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.02433
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.29682
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.68618
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.57700

284807 rows × 31 columns

EDA

In [85]:
```
df.head()
```

Out[85]:

	Time	V1	V2	V3	V4	V5	V6	V7	V
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.09869
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.0851(
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.24767
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.37743
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.27053

5 rows × 31 columns

In [86]:
```
df.tail()
```

Out[86]:

	Time	V1	V2	V3	V4	V5	V6	V
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.91821
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.02433
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.29682
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.68618
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.57700

5 rows × 31 columns

In [87]:
```
df.describe()
```

Out[87]:

	Time	V1	V2	V3	V4	
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.84807(
mean	94813.859575	3.918649e-15	5.682686e-16	-8.761736e-15	2.811118e-15	-1.55210
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.38024
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.13743
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.91597
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.43358
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.11926
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.48016

8 rows × 31 columns

In [88]:
```
df.shape
```

Out[88]:
```
(284807, 31)
```

In [89]:
```
df.columns
```

Out[89]:
```
Index(['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V1
0',
       'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V
20',
       'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'Amount',
       'Class'],
      dtype='object')
```

In [90]:
```
type(df)
```

Out[90]:
```
pandas.core.frame.DataFrame
```

In [95]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64  
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

In [92]:
```
df.Class.value_counts()
```

Out[92]:
```
0    284315
1       492
Name: Class, dtype: int64
```

Data Preprocessing and Data Cleaning

In [93]:
```
df.isnull().sum()
```

Out[93]:
```
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 [94]:
```
df.dropna(axis=0,inplace=True)
```

In [49]:
```
df
```

Out[49]:

	Time	V1	V2	V3	V4	V5	V6	V
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.2395
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.0788
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.7914
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.2376
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.5929
...	...	...	...	...	...	...	...	...
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.9182
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.0243
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.2968
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.6861
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.5770

284807 rows × 31 columns

In [96]:
```
df.duplicated()
```

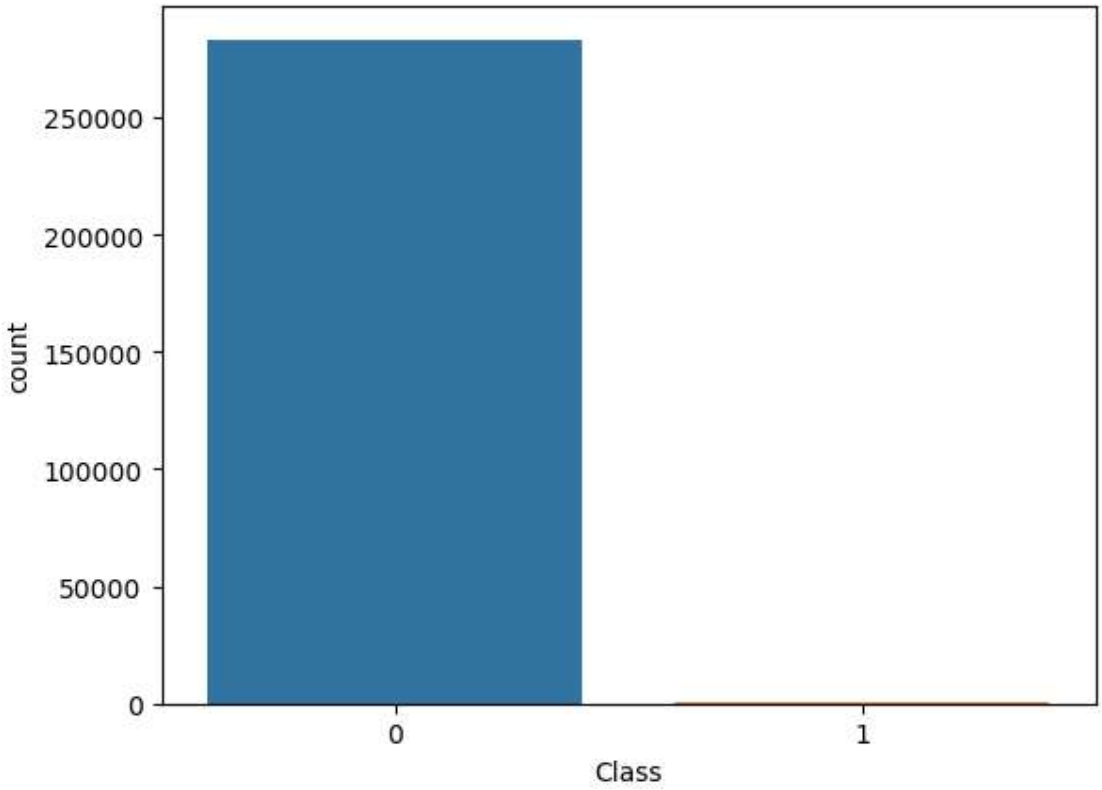
Out[96]:
```
0         False
1         False
2         False
3         False
4         False
          ...
284802    False
284803    False
284804    False
284805    False
284806    False
Length: 284807, dtype: bool
```

In [97]:
```
df.drop_duplicates(inplace=True)
```

Data Visualization

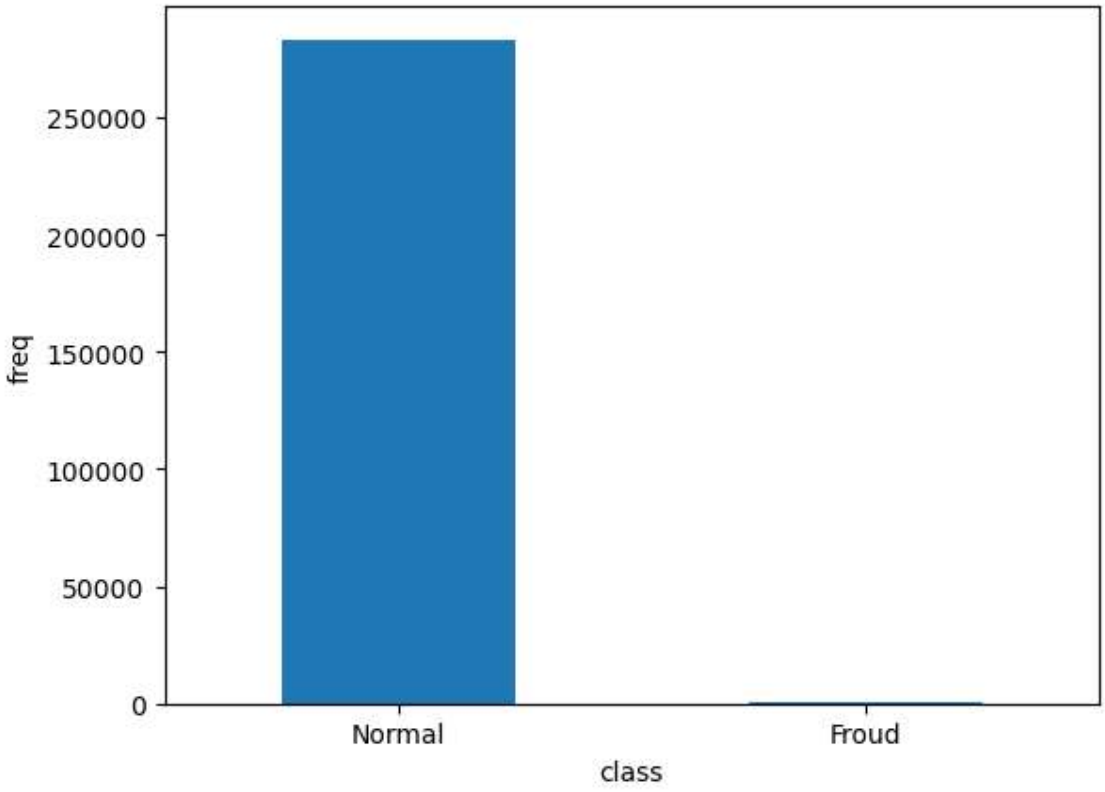
In [98]:
```
sns.countplot('Class',data=df)
```

Out[98]: <AxesSubplot:xlabel='Class', ylabel='count'>



In [99]:
```
LABELS='Normal','Froud'
```

In [100]:
```
Class = pd.value_counts(df['Class'],sort=True)
Class.plot(kind='bar',rot=0)
plt.title=('Class_Distribustion')
plt.xticks(range(2),LABELS)
plt.xlabel('class')
plt.ylabel('freq')
plt.show()
```



In [101]:
```
Froud=df[df['Class']==1]
Normal=df[df['Class']==0]
```

In [102]:
```
Froud.shape
```

Out[102]: (473, 31)

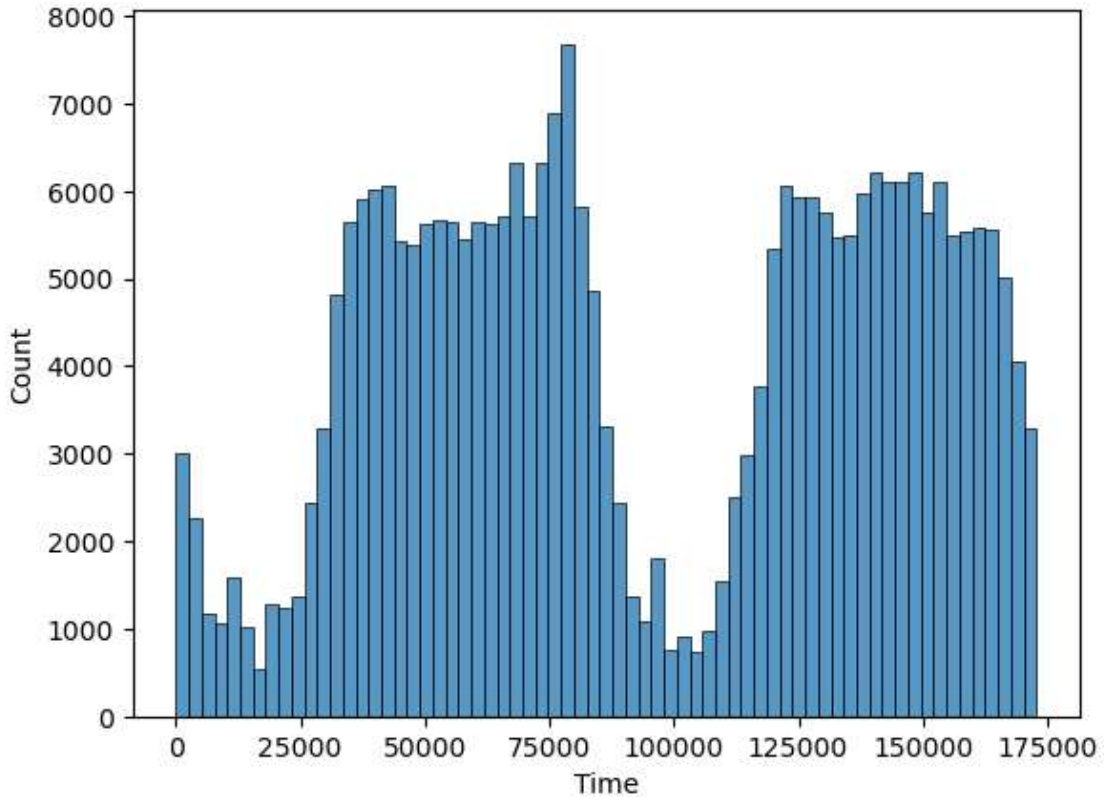
In [103]:
```
Normal.shape
```

Out[103]: (283253, 31)

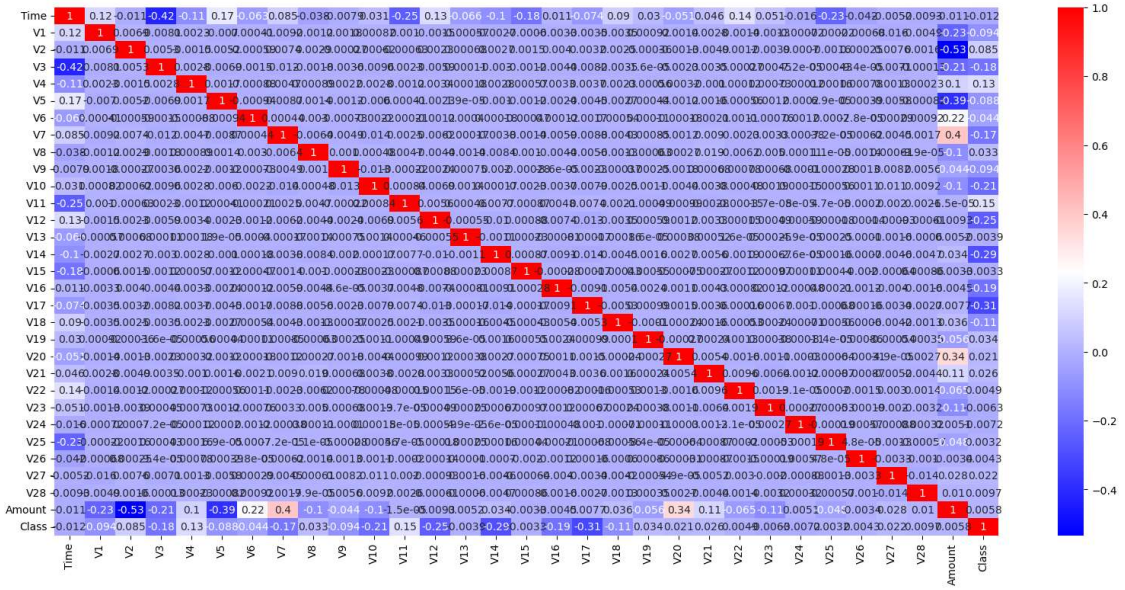
In [149]:
```
sns.histplot(df.Time)

plt.show
```

Out[149]: <function matplotlib.pyplot.show(close=None, block=None)>



In [106]:
```
plt.figure(figsize=(20,9));
sns.heatmap(df.corr(),annot=True,cmap='bwr')
plt.show()
```



Finding Correlation using Heatmap

Divide Dataset into(independent & denpendent)

In [107]:
```
x=df.iloc[:,:-1]
y=df.iloc[:,-1]
```

In [137]:
```
x
```

Out[137]:

	Time	V1	V2	V3	V4	V5	V6	V
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.2395
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.0788
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.7914
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.2376
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.5929
...	...	...	...	...	...	...	...	...
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.9182
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.0243
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.2968
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.6861
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.5770

283726 rows × 30 columns

In [138]:
```
y
```

Out[138]:
```
0         0
1         0
2         0
3         0
4         0
         ..
284802    0
284803    0
284804    0
284805    0
284806    0
Name: Class, Length: 283726, dtype: int64
```

In [108]:
```
x.shape
```

Out[108]:
```
(283726, 30)
```

In [109]:
```
y.shape
```

Out[109]: (283726,)

Split dataset into Train & Test Data

In [110]:
```
from sklearn.model_selection import train_test_split
```

In [111]:
```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_state=
```

In [112]:
```
xtrain.shape
xtest.shape
ytrain.shape
ytest.shape
```

Out[112]: (56746,)

Logistic Regression

In [113]:
```
from sklearn.linear_model import LogisticRegression
```

In [114]:
```
from sklearn.metrics import accuracy_score,classification_report,confusion
```

In [115]:
```
LR=LogisticRegression()
```

In [116]:
```
LR.fit(xtrain,ytrain)
```

Out[116]: LogisticRegression()

In [117]:
```
ypred=LR.predict(xtest)
```

In [118]:
```
ypred
```

Out[118]: array([0, 0, 0, ..., 0, 0, 0], dtype=int64)

In [119]:
```
print(accuracy_score(ypred,ytest))
```

0.9989779015260988

In [120]:
```
print(classification_report(ypred,ytest))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56664
           1       0.63      0.70      0.66        82

    accuracy                           1.00     56746
   macro avg       0.82      0.85      0.83     56746
weighted avg       1.00      1.00      1.00     56746
```

Random-Over Sample

In [121]:
```
pip install imblearn
```

```
Requirement already satisfied: imblearn in c:\users\disha__computers\ana
conda3\lib\site-packages (0.0)
Requirement already satisfied: imbalanced-learn in c:\users\disha__compu
ters\anaconda3\lib\site-packages (from imblearn) (0.11.0)
Requirement already satisfied: scipy>=1.5.0 in c:\users\disha__computers
\anaconda3\lib\site-packages (from imbalanced-learn->imblearn) (1.9.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\disha__c
omputers\anaconda3\lib\site-packages (from imbalanced-learn->imblearn)
(2.2.0)
Requirement already satisfied: scikit-learn>=1.0.2 in c:\users\disha__co
mputers\anaconda3\lib\site-packages (from imbalanced-learn->imblearn)
(1.0.2)
Requirement already satisfied: numpy>=1.17.3 in c:\users\disha__computer
s\anaconda3\lib\site-packages (from imbalanced-learn->imblearn) (1.21.5)
Requirement already satisfied: joblib>=1.1.1 in c:\users\disha__computer
s\anaconda3\lib\site-packages (from imbalanced-learn->imblearn) (1.3.1)
Note: you may need to restart the kernel to use updated packages.
```

In [122]:
```
from imblearn.over_sampling import RandomOverSampler
```

In [123]:
```
ROS=RandomOverSampler()
```

In [124]:
```
R1,R2=ROS.fit_resample(x,y)
```

In [125]:
```
R1.shape
```

Out[125]: (566506, 30)

In [126]:
```
R2.shape
```

Out[126]: (566506,)

Hyperparameter

In [127]:
```
from collections import Counter
```

In [128]:
```
print('orignale dataset{}'.format(Counter(y)))
```

```
orignale datasetCounter({0: 283253, 1: 473})
```

In [129]:
```
print('resample dataset{}'.format(Counter(R2)))
```

```
resample datasetCounter({0: 283253, 1: 283253})
```

Apply Classificaton algorithm

In [130]:
```
x1train,x1test,y1train,y1test=train_test_split(R1,R2,test_size=0.2,random_
```

In [131]:
```
LR.fit(x1train,y1train)
```

Out[131]:
```
LogisticRegression()
```

In [132]:
```
ypred=LR.predict(x1test)
```

In [133]:
```
print(accuracy_score(ypred,y1test))
```

```
0.9441139609186069
```

In [134]:
```
print(classification_report(ypred,y1test))
```

```
              precision    recall  f1-score   support

           0       0.96      0.93      0.95     58713
           1       0.92      0.96      0.94     54589

    accuracy                           0.94    113302
   macro avg       0.94      0.94      0.94    113302
weighted avg       0.94      0.94      0.94    113302
```

SMOTE

In [140]:
```
from imblearn.over_sampling import SMOTE
```

In [141]:
```
SME=SMOTE()
```

In [142]:
```
y1,y2=SME.fit_resample(x,y)
```

In [143]:
```
y1.shape
```

Out[143]: (566506, 30)

In [144]:
```
y2.shape
```

Out[144]: (566506,)

Hyperparameter of SMOTE

In [145]:
```
print('Original dataset shape {}'.format(Counter(y)))
print('Resampled dataset shape {}'.format(Counter(y2)))
```

```
Original dataset shape Counter({0: 283253, 1: 473})
Resampled dataset shape Counter({0: 283253, 1: 283253})
```

In [146]:
```
X_train1, X_test1, Y_train1, Y_test1 = train_test_split(y1,y2,test_size=0.

from sklearn.linear_model import LogisticRegression
clf = LogisticRegression().fit(X_train1,Y_train1)
predy=clf.predict(X_test1)
```

In [147]:
```
print(accuracy_score(Y_test1,predy))
```

```
0.9706006954863992
```

In [148]:
```
print(classification_report(Y_test1,predy))
```

```
              precision    recall  f1-score   support

           0       0.96      0.98      0.97     56463
           1       0.98      0.96      0.97     56839

    accuracy                           0.97    113302
   macro avg       0.97      0.97      0.97    113302
weighted avg       0.97      0.97      0.97    113302
```