capstonecreditcard

June 26, 2024

[54]:
```python
import pandas as pd
```

[56]:
```python
data=pd.read_csv(r'C:\Users\DELL\Documents\credit card.csv')
```

[58]:
```python
pd.options.display.max_columns = None
```

[60]:
```python
data.head()
```

[60]:
```
   Time        V1        V2        V3        V4        V5        V6        V7  \
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         V8        V9       V10       V11       V12       V13       V14  \
0  0.098698  0.363787  0.090794 -0.551600 -0.617801 -0.991390 -0.311169
1  0.085102 -0.255425 -0.166974  1.612727  1.065235  0.489095 -0.143772
2  0.247676 -1.514654  0.207643  0.624501  0.066084  0.717293 -0.165946
3  0.377436 -1.387024 -0.054952 -0.226487  0.178228  0.507757 -0.287924
4 -0.270533  0.817739  0.753074 -0.822843  0.538196  1.345852 -1.119670

        V15       V16       V17       V18       V19       V20       V21  \
0  1.468177 -0.470401  0.207971  0.025791  0.403993  0.251412 -0.018307
1  0.635558  0.463917 -0.114805 -0.183361 -0.145783 -0.069083 -0.225775
2  2.345865 -2.890083  1.109969 -0.121359 -2.261857  0.524980  0.247998
3 -0.631418 -1.059647 -0.684093  1.965775 -1.232622 -0.208038 -0.108300
4  0.175121 -0.451449 -0.237033 -0.038195  0.803487  0.408542 -0.009431

        V22       V23       V24       V25       V26       V27       V28  \
0  0.277838 -0.110474  0.066928  0.128539 -0.189115  0.133558 -0.021053
1 -0.638672  0.101288 -0.339846  0.167170  0.125895 -0.008983  0.014724
2  0.771679  0.909412 -0.689281 -0.327642 -0.139097 -0.055353 -0.059752
3  0.005274 -0.190321 -1.175575  0.647376 -0.221929  0.062723  0.061458
4  0.798278 -0.137458  0.141267 -0.206010  0.502292  0.219422  0.215153

   Amount  Class
```

```
0  149.62      0
1    2.69      0
2  378.66      0
3  123.50      0
4   69.99      0
```

[62]:
```python
data.shape
```

[62]: (284807, 31)

[64]:
```python
print("Number of Rows",data.shape[0])
print("Number of Cloumns",data.shape[1])
```

```
Number of Rows 284807
Number of Cloumns 31
```

[66]:
```python
data.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
```

[68]:
```
data.isnull().sum()
```

[68]:
```
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
```

[70]:
```
data.head()
```

[70]:
```
   Time        V1        V2        V3        V4        V5        V6        V7  \
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         V8        V9       V10       V11       V12       V13       V14  \
0  0.098698  0.363787  0.090794 -0.551600 -0.617801 -0.991390 -0.311169
1  0.085102 -0.255425 -0.166974  1.612727  1.065235  0.489095 -0.143772
2  0.247676 -1.514654  0.207643  0.624501  0.066084  0.717293 -0.165946
3  0.377436 -1.387024 -0.054952 -0.226487  0.178228  0.507757 -0.287924
4 -0.270533  0.817739  0.753074 -0.822843  0.538196  1.345852 -1.119670

        V15       V16       V17       V18       V19       V20       V21  \
0  1.468177 -0.470401  0.207971  0.025791  0.403993  0.251412 -0.018307
1  0.635558  0.463917 -0.114805 -0.183361 -0.145783 -0.069083 -0.225775
2  2.345865 -2.890083  1.109969 -0.121359 -2.261857  0.524980  0.247998
3 -0.631418 -1.059647 -0.684093  1.965775 -1.232622 -0.208038 -0.108300
4  0.175121 -0.451449 -0.237033 -0.038195  0.803487  0.408542 -0.009431

        V22       V23       V24       V25       V26       V27       V28  \
0  0.277838 -0.110474  0.066928  0.128539 -0.189115  0.133558 -0.021053
1 -0.638672  0.101288 -0.339846  0.167170  0.125895 -0.008983  0.014724
2  0.771679  0.909412 -0.689281 -0.327642 -0.139097 -0.055353 -0.059752
3  0.005274 -0.190321 -1.175575  0.647376 -0.221929  0.062723  0.061458
4  0.798278 -0.137458  0.141267 -0.206010  0.502292  0.219422  0.215153

   Amount  Class
0  149.62      0
1    2.69      0
2  378.66      0
3  123.50      0
4   69.99      0
```

[72]:
```python
from sklearn.preprocessing import StandardScaler
```

[74]:
```python
sc = StandardScaler()
data['Amount']=sc.fit_transform(pd.DataFrame(data['Amount']))
```

[76]:
```python
data.head()
```

[76]:
```
   Time        V1        V2        V3        V4        V5        V6        V7  \
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
```

```
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         V8        V9       V10       V11       V12       V13       V14  \
0  0.098698  0.363787  0.090794 -0.551600 -0.617801 -0.991390 -0.311169
1  0.085102 -0.255425 -0.166974  1.612727  1.065235  0.489095 -0.143772
2  0.247676 -1.514654  0.207643  0.624501  0.066084  0.717293 -0.165946
3  0.377436 -1.387024 -0.054952 -0.226487  0.178228  0.507757 -0.287924
4 -0.270533  0.817739  0.753074 -0.822843  0.538196  1.345852 -1.119670

        V15       V16       V17       V18       V19       V20       V21  \
0  1.468177 -0.470401  0.207971  0.025791  0.403993  0.251412 -0.018307
1  0.635558  0.463917 -0.114805 -0.183361 -0.145783 -0.069083 -0.225775
2  2.345865 -2.890083  1.109969 -0.121359 -2.261857  0.524980  0.247998
3 -0.631418 -1.059647 -0.684093  1.965775 -1.232622 -0.208038 -0.108300
4  0.175121 -0.451449 -0.237033 -0.038195  0.803487  0.408542 -0.009431

        V22       V23       V24       V25       V26       V27       V28  \
0  0.277838 -0.110474  0.066928  0.128539 -0.189115  0.133558 -0.021053
1 -0.638672  0.101288 -0.339846  0.167170  0.125895 -0.008983  0.014724
2  0.771679  0.909412 -0.689281 -0.327642 -0.139097 -0.055353 -0.059752
3  0.005274 -0.190321 -1.175575  0.647376 -0.221929  0.062723  0.061458
4  0.798278 -0.137458  0.141267 -0.206010  0.502292  0.219422  0.215153

     Amount  Class
0  0.244964      0
1 -0.342475      0
2  1.160686      0
3  0.140534      0
4 -0.073403      0
```

[78]:
```
data = data.drop(['Time'],axis=1)
```

[80]:
```
data.head()
```

[80]:
```
         V1        V2        V3        V4        V5        V6        V7  \
0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
4 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         V8        V9       V10       V11       V12       V13       V14  \
0  0.098698  0.363787  0.090794 -0.551600 -0.617801 -0.991390 -0.311169
1  0.085102 -0.255425 -0.166974  1.612727  1.065235  0.489095 -0.143772
2  0.247676 -1.514654  0.207643  0.624501  0.066084  0.717293 -0.165946
3  0.377436 -1.387024 -0.054952 -0.226487  0.178228  0.507757 -0.287924
4 -0.270533  0.817739  0.753074 -0.822843  0.538196  1.345852 -1.119670
```

```
        V15       V16       V17       V18       V19       V20       V21  \
0  1.468177 -0.470401  0.207971  0.025791  0.403993  0.251412 -0.018307
1  0.635558  0.463917 -0.114805 -0.183361 -0.145783 -0.069083 -0.225775
2  2.345865 -2.890083  1.109969 -0.121359 -2.261857  0.524980  0.247998
3 -0.631418 -1.059647 -0.684093  1.965775 -1.232622 -0.208038 -0.108300
4  0.175121 -0.451449 -0.237033 -0.038195  0.803487  0.408542 -0.009431

        V22       V23       V24       V25       V26       V27       V28  \
0  0.277838 -0.110474  0.066928  0.128539 -0.189115  0.133558 -0.021053
1 -0.638672  0.101288 -0.339846  0.167170  0.125895 -0.008983  0.014724
2  0.771679  0.909412 -0.689281 -0.327642 -0.139097 -0.055353 -0.059752
3  0.005274 -0.190321 -1.175575  0.647376 -0.221929  0.062723  0.061458
4  0.798278 -0.137458  0.141267 -0.206010  0.502292  0.219422  0.215153

     Amount  Class
0  0.244964      0
1 -0.342475      0
2  1.160686      0
3  0.140534      0
4 -0.073403      0
```

[82]:
```
data.shape
```

[82]: (284807, 30)

[84]:
```
data.duplicated().any()
```

[84]: True

[86]:
```
data = data.drop_duplicates()
```

[88]:
```
data.shape
```

[88]: (275663, 30)

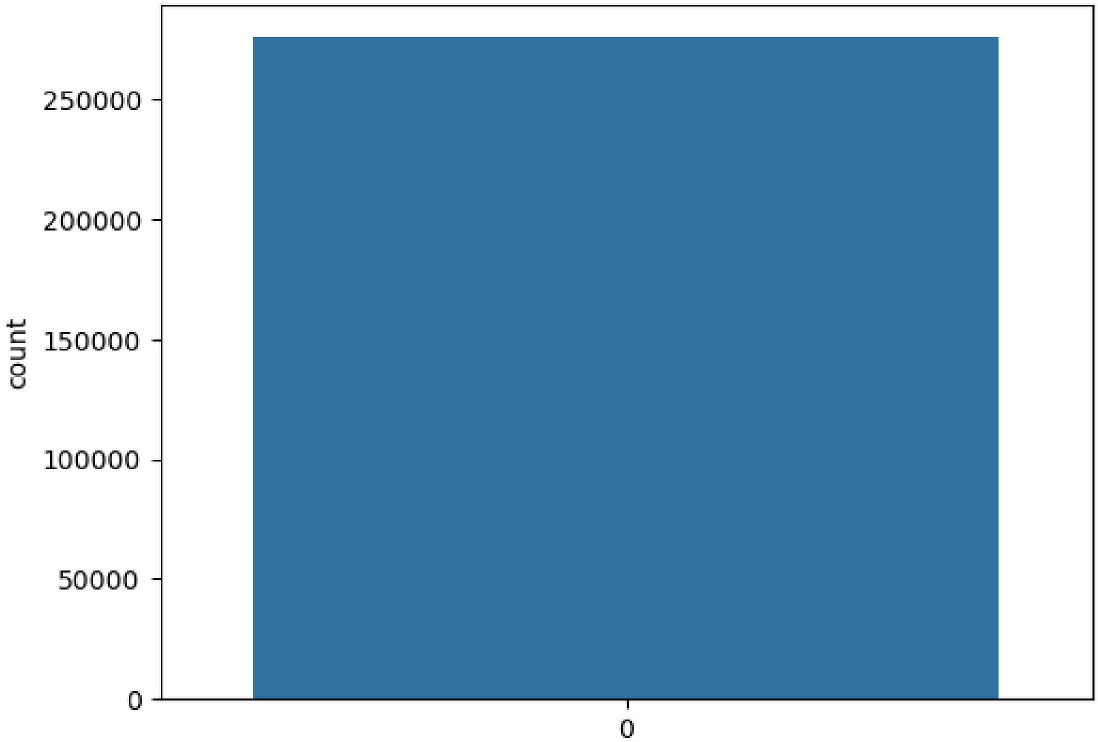
[90]:
```
data['Class'].value_counts()
```

[90]:
```
Class
0    275190
1       473
Name: count, dtype: int64
```

[92]:
```
import seaborn as sns
```

[94]:
```
sns.countplot(data['Class'])
```

[94]: <Axes: ylabel='count'>



[96]:
```
x=data.drop('Class',axis=1)
y=data['Class']
```

[98]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.
↪20,random_state=42)
```

[100]:
```
normal = data[data['Class']==0]
fraud = data[data['Class']==1]
```

[102]:
```
normal.shape
```

[102]: (275190, 30)

[104]:
```
fraud.shape
```

[104]: (473, 30)

[106]:
```
normal_sample=normal.sample(n=473)
```

[110]:
```
normal_sample.shape
```

[110]: (473, 30)

[112]:
```
new_data = pd.concat([normal_sample,fraud],ignore_index=True)
```

[114]:
```
new_data['Class'].value_counts()
```

[114]:
```
Class
0    473
1    473
Name: count, dtype: int64
```

[116]:
```
new_data.head()
```

[116]:
```
         V1        V2        V3        V4        V5        V6        V7  \
0  1.115257 -1.779067  0.692283 -1.295945 -2.019082 -0.120768 -1.372727
1 -0.530174  0.135782  1.558311 -1.458473 -0.222240 -0.680797  0.681516
2 -0.869921  0.634924  2.039045 -1.254002 -0.196762  0.224561 -0.050755
3 -3.091510  2.852625 -0.727117 -0.013360 -1.890684 -1.233278 -1.113311
4 -0.940893  1.074155  1.759398 -0.601446  0.101693 -0.188520  0.455756

         V8        V9       V10       V11       V12       V13       V14  \
0  0.208498 -1.585965  1.542702  1.245757 -1.129232 -1.536584  0.171966
1 -0.242224 -1.127463 -0.471296  0.224039 -0.772260 -0.103609 -1.668418
2 -0.660373  0.497608 -1.106703 -1.516634  0.378012  0.880850 -0.789697
3  2.123149 -0.414949 -0.195267 -1.109742  0.597555 -0.016520  1.517442
4 -3.460682  0.441525  0.917818  0.877285  0.338269  0.646250 -0.828662

        V15       V16       V17       V18       V19       V20       V21  \
0  0.615714 -0.191430  0.594151  0.563718 -0.620012 -0.178500  0.097289
1 -0.334266  0.797692  1.346722 -1.726595 -0.360249  0.239301  0.038616
2 -0.454402  0.578837 -0.652759 -0.127053 -0.124200 -0.195734  0.668493
3  0.763747  0.756659  0.431783 -0.181157 -0.131151  0.040078 -0.143839
4  1.464952 -0.915167  0.171665 -1.054133  0.276801 -0.209018  2.270069

        V22       V23       V24       V25       V26       V27       V28  \
0  0.263326 -0.062267  0.181284  0.089065 -0.112896  0.024124  0.036898
1  0.158680 -0.034544  0.308940 -0.003244 -0.474407 -0.079171 -0.096014
2 -0.368430 -0.199768 -0.368441 -0.006672  0.887622  0.073190  0.085501
3 -0.784700  0.238355  0.336640  0.106969  0.099980  0.068592  0.028134
4 -0.143518  0.153908  0.700927 -0.413235  1.374031 -0.996161 -0.836301

     Amount  Class
0  0.278468      0
1 -0.084078      0
2 -0.279705      0
```

```
3 -0.297296      0
4 -0.313289      0
```

[118]:
```
x=new_data.drop('Class',axis=1)
y=new_data['Class']
```

[120]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.
 ↪20,random_state=42)
```

[122]:
```
from sklearn.linear_model import LogisticRegression
```

[124]:
```
log = LogisticRegression()
```

[126]:
```
log.fit(x_train,y_train)
```

[126]:
```
LogisticRegression()
```

[128]:
```
y_pred1 = log.predict(x_test)
```

[38]:
```
from sklearn.metrics import accuracy_score
```

[130]:
```
accuracy_score(y_test,y_pred1)
```

[130]:
```
0.9421052631578948
```

[132]:
```
accuracy_score(y_test,y_pred1)
```

[132]:
```
0.9421052631578948
```

[134]:
```
from sklearn.metrics import precision_score, recall_score, f1_score
```

[169]:
```
precision_score(y_test,y_pred1)
```

[169]:
```
0.8870967741935484
```

[136]:
```
precision_score(y_test,y_pred1)
```

[136]:
```
0.9690721649484536
```

[171]:
```
recall_score(y_test,y_pred1)
```

[171]:
```
0.6043956043956044
```

[138]:
```
recall_score(y_test,y_pred1)
```

[138]:
```
0.9215686274509803
```

[173]:
```
f1_score(y_test,y_pred1)
```

[173]: 0.718954248366013

[140]:
```
f1_score(y_test,y_pred1)
```

[140]: 0.9447236180904522

[142]:
```
from sklearn.tree import DecisionTreeClassifier
```

[144]:
```
dt = DecisionTreeClassifier()
```

[146]:
```
dt.fit(x_train, y_train)
```

[146]: DecisionTreeClassifier()

[148]:
```
y_pred2 = dt.predict(x_test)
```

[150]:
```
accuracy_score(y_test, y_pred2)
```

[150]: 0.9315789473684211

[152]:
```
precision_score(y_test,y_pred2)
```

[152]: 0.9238095238095239

[154]:
```
recall_score(y_test,y_pred2)
```

[154]: 0.9509803921568627

[156]:
```
f1_score(y_test,y_pred2)
```

[156]: 0.9371980676328502

[158]:
```
from sklearn.ensemble import RandomForestClassifier
```

[160]:
```
rf = RandomForestClassifier()
```

[162]:
```
rf.fit(x_train,y_train)
```

[162]: RandomForestClassifier()

[164]:
```
y_pred3 = rf.predict(x_test)
```

[166]:
```
accuracy_score(y_test, y_pred3)
```

[166]: 0.9421052631578948

[168]:
```
precision_score(y_test,y_pred3)
```

[168]: 0.9789473684210527

[170]:
```
recall_score(y_test,y_pred3)
```

[170]: 0.9117647058823529

[172]:
```
f1_score(y_test,y_pred3)
```

[172]: 0.9441624365482234

[174]:
```
final_data = pd.DataFrame({'Models':['LR','DT','RF'],
              "ACC":[accuracy_score(y_test,y_pred1)*100,
                     accuracy_score(y_test,y_pred2)*100,
                     accuracy_score(y_test,y_pred3)*100]})
```

[310]:
```
final_data
```

[310]:

	Models	ACC
0	LR	94.736842
1	DT	91.578947
2	RF	94.736842

[184]:
```
acc_lr = accuracy_score(y_test, y_pred1) * 100
acc_dt = accuracy_score(y_test, y_pred2) * 100
acc_rf = accuracy_score(y_test, y_pred3) * 100
```

[186]:
```
final_data = pd.DataFrame({
    'Models': ['LR', 'DT', 'RF'],
    'ACC': [acc_lr, acc_dt, acc_rf]
})
```

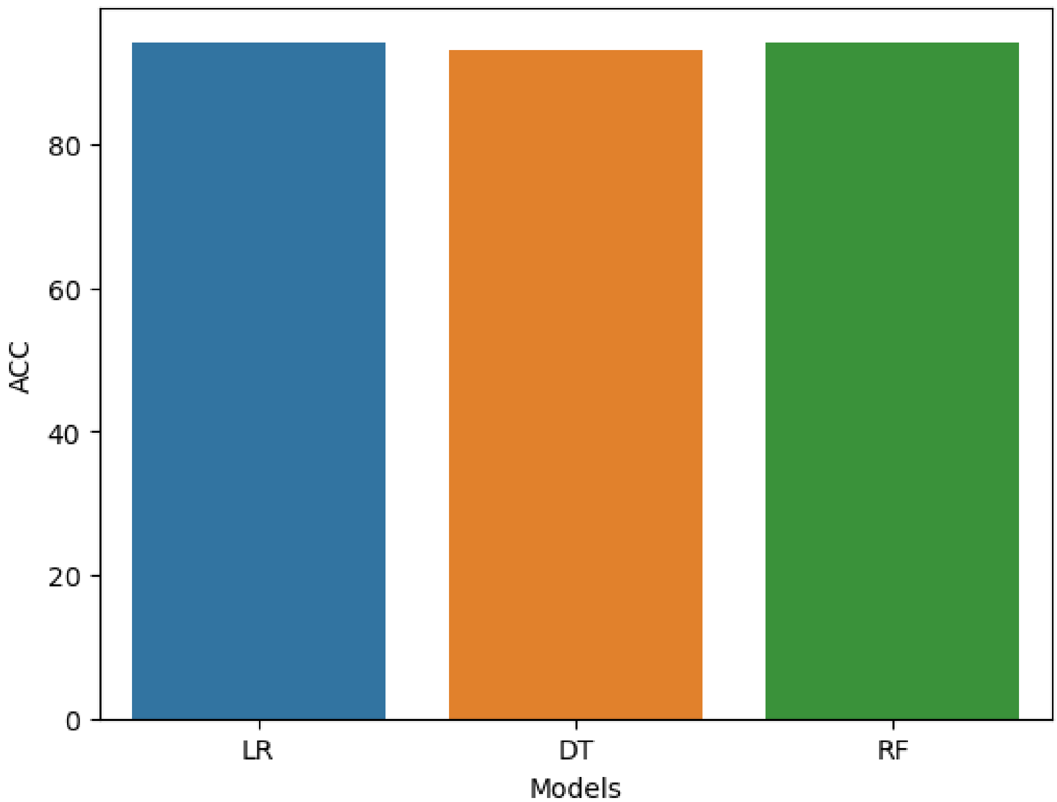
[189]:
```
final_data
```

[189]:

	Models	ACC
0	LR	94.210526
1	DT	93.157895
2	RF	94.210526

[191]:
```
sns.barplot(x='Models', y='ACC', data=final_data)
```

[191]: <Axes: xlabel='Models', ylabel='ACC'>



[193]:
```
x=new_data.drop('Class',axis=1)
y=new_data['Class']
```

[195]:
```
x.shape
```

[195]: (946, 29)

[197]:
```
y.shape
```

[197]: (946,)

[210]:
```
data.rename(columns={'Class': 'class'}, inplace=True)
```

[212]:
```
fraud = data.loc[data['class']==1]
normal = data.loc[data['class']==0]
```

[214]:
```
fraud
```

[214]:
```
              Time        V1        V2        V3        V4        V5        V6  \
541          406.0 -2.312227  1.951992 -1.609851  3.997906 -0.522188 -1.426545
623          472.0 -3.043541 -3.157307  1.088463  2.288644  1.359805 -1.064823
```

```
4920       4462.0 -2.303350  1.759247 -0.359745  2.330243 -0.821628 -0.075788
6108       6986.0 -4.397974  1.358367 -2.592844  2.679787 -1.128131 -1.706536
6329       7519.0  1.234235  3.019740 -4.304597  4.732795  3.624201 -1.357746
…            …        …        …        …        …        …        …
279863   169142.0 -1.927883  1.125653 -4.518331  1.749293 -1.566487 -2.010494
280143   169347.0  1.378559  1.289381 -5.004247  1.411850  0.442581 -1.326536
280149   169351.0 -0.676143  1.126366 -2.213700  0.468308 -1.120541 -0.003346
281144   169966.0 -3.113832  0.585864 -5.399730  1.817092 -0.840618 -2.943548
281674   170348.0  1.991976  0.158476 -2.583441  0.408670  1.151147 -0.096695

               V7        V8        V9       V10       V11        V12       V13  \
541     -2.537387  1.391657 -2.770089 -2.772272  3.202033  -2.899907 -0.595222
623      0.325574 -0.067794 -0.270953 -0.838587 -0.414575  -0.503141  0.676502
4920     0.562320 -0.399147 -0.238253 -1.525412  2.032912  -6.560124  0.022937
6108    -3.496197 -0.248778 -0.247768 -4.801637  4.895844 -10.912819  0.184372
6329     1.713445 -0.496358 -1.282858 -2.447469  2.101344  -4.609628  1.464378
…            …        …        …        …        …         …         …
279863  -0.882850  0.697211 -2.064945 -5.587794  2.115795  -5.417424 -1.235123
280143  -1.413170  0.248525 -1.127396 -3.232153  2.858466  -3.096915 -0.792532
280149  -2.234739  1.210158 -0.652250 -3.463891  1.794969  -2.775022 -0.418950
281144  -2.208002  1.058733 -1.632333 -5.245984  1.933520  -5.030465 -1.127455
281674   0.223050 -0.068384  0.577829 -0.888722  0.491140   0.728903  0.380428

              V14       V15       V16        V17       V18       V19       V20  \
541     -4.289254  0.389724 -1.140747  -2.830056 -0.016822  0.416956  0.126911
623     -1.692029  2.000635  0.666780   0.599717  1.725321  0.283345  2.102339
4920    -1.470102 -0.698826 -2.282194  -4.781831 -2.615665 -1.334441 -0.430022
6108    -6.771097 -0.007326 -7.358083 -12.598419 -5.131549  0.308334 -0.171608
6329    -6.079337 -0.339237  2.581851   6.739384  3.042493 -2.721853  0.009061
…            …        …        …         …         …        …        …
279863  -6.665177  0.401701 -2.897825  -4.570529 -1.315147  0.391167  1.252967
280143  -5.210141 -0.613803 -2.155297  -3.267116 -0.688505  0.737657  0.226138
280149  -4.057162 -0.712616 -1.603015  -5.035326 -0.507000  0.266272  0.247968
281144  -6.416628  0.141237 -2.549498  -4.614717 -1.478138 -0.035480  0.306271
281674  -1.948883 -0.832498  0.519436   0.903562  1.197315  0.593509 -0.017652

              V21       V22       V23       V24       V25       V26       V27  \
541      0.517232 -0.035049 -0.465211  0.320198  0.044519  0.177840  0.261145
623      0.661696  0.435477  1.375966 -0.293803  0.279798 -0.145362 -0.252773
4920    -0.294166 -0.932391  0.172726 -0.087330 -0.156114 -0.542628  0.039566
6108     0.573574  0.176968 -0.436207 -0.053502  0.252405 -0.657488 -0.827136
6329    -0.379068 -0.704181 -0.656805 -1.632653  1.488901  0.566797 -0.010016
…            …        …        …        …        …        …        …
279863   0.778584 -0.319189  0.639419 -0.294885  0.537503  0.788395  0.292680
280143   0.370612  0.028234 -0.145640 -0.081049  0.521875  0.739467  0.389152
280149   0.751826  0.834108  0.190944  0.032070 -0.739695  0.471111  0.385107
281144   0.583276 -0.269209 -0.456108 -0.183659 -0.328168  0.606116  0.884876
```

```
281674 -0.164350 -0.295135 -0.072173 -0.450261  0.313267 -0.289617  0.002988

             V28  Amount  class
541    -0.143276    0.00      1
623     0.035764  529.00      1
4920   -0.153029  239.93      1
6108    0.849573   59.00      1
6329    0.146793    1.00      1
…            …       …     …
279863  0.147968  390.00      1
280143  0.186637    0.76      1
280149  0.194361   77.89      1
281144 -0.253700  245.00      1
281674 -0.015309   42.53      1

[492 rows x 31 columns]
```

[216]:

```
fraud.count()
```

[216]:

```
Time      492
V1        492
V2        492
V3        492
V4        492
V5        492
V6        492
V7        492
V8        492
V9        492
V10       492
V11       492
V12       492
V13       492
V14       492
V15       492
V16       492
V17       492
V18       492
V19       492
V20       492
V21       492
V22       492
V23       492
V24       492
V25       492
V26       492
V27       492
```

```
V28        492
Amount     492
class      492
dtype: int64
```

[218]:
```
fraud.sum()
```

[218]:
```
Time      3.972743e+07
V1       -2.347799e+03
V2        1.782899e+03
V3       -3.460374e+03
V4        2.234678e+03
V5       -1.550403e+03
V6       -6.876865e+02
V7       -2.739816e+03
V8        2.807529e+02
V9       -1.269912e+03
V10      -2.793026e+03
V11       1.869685e+03
V12      -3.079621e+03
V13      -5.379224e+01
V14      -3.430088e+03
V15      -4.572094e+01
V16      -2.036853e+03
V17      -3.279592e+03
V18      -1.105184e+03
V19       3.348844e+02
V20       1.831811e+02
V21       3.510855e+02
V22       6.912050e+00
V23      -1.983152e+01
V24      -5.172411e+01
V25       2.039285e+01
V26       2.541088e+01
V27       8.392280e+01
V28       3.722831e+01
Amount    6.012797e+04
class     4.920000e+02
dtype: float64
```

[220]:
```
len(fraud)
```

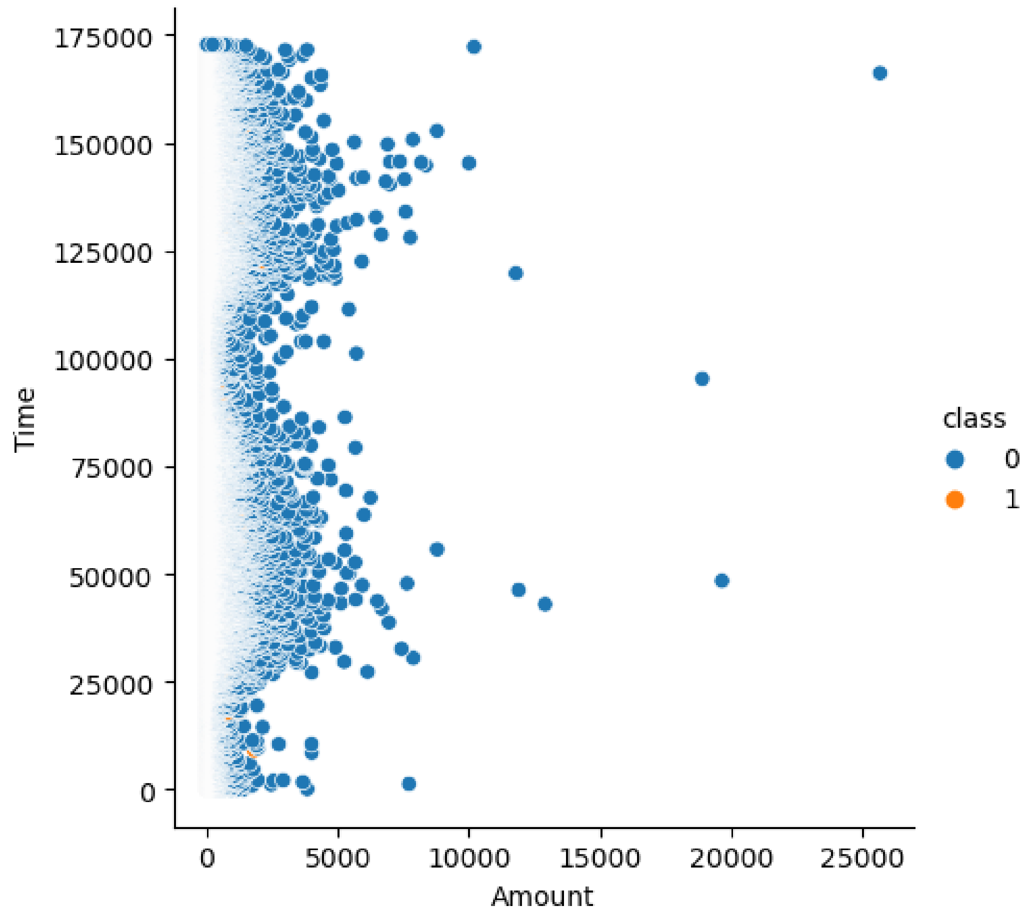
[220]:
```
492
```

[222]:
```
len(normal)
```

[222]:
```
284315
```

[228]:
```
sns.relplot(x= 'Amount',y="Time",hue="class",data=data)
```

[228]: <seaborn.axisgrid.FacetGrid at 0x1a79b271550>



[240]:
```
from imblearn.over_sampling import SMOTE
```

[242]:
```
x_res,y_res = SMOTE().fit_resample(x,y)
```

[244]:
```
y_res.shape
```

[244]: (946,)

[ ]:
```

```

[284]:
```
from sklearn.model_selection import GridSearchCV
```

[286]:
```
model = RandomForestClassifier()
```

[288]:
```python
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

[290]:
```python
grid_search = GridSearchCV(estimator=model, param_grid=param_grid, cv=3,␣
 ↪n_jobs=-1, verbose=2)
grid_search.fit(x_train, y_train)
```

```
Fitting 3 folds for each of 108 candidates, totalling 324 fits
```

[290]:
```
GridSearchCV(cv=3, estimator=RandomForestClassifier(), n_jobs=-1,
             param_grid={'max_depth': [None, 10, 20, 30],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10],
                         'n_estimators': [50, 100, 200]},
             verbose=2)
```

[291]:
```python
best_model = grid_search.best_estimator_
y_pred_best = best_model.predict(x_test)
```

[292]:
```python
"Best Parameters:", grid_search.best_params_
"Best Score:", grid_search.best_score_
```

[292]:
```
('Best Score:', 0.9444444444444445)
```

[294]:
```python
accuracy = accuracy_score(y_test, y_pred_best)
f'Accuracy: {accuracy * 100:.2f}%'
```

[294]:
```
'Accuracy: 94.21%'
```

[317]:
```python
from sklearn.metrics import accuracy_score, confusion_matrix,␣
 ↪classification_report
```

[319]:
```python
conf_matrix = confusion_matrix(y_test, y_pred_best)
"Confusion Matrix (Tuned):"
conf_matrix
```

[319]:
```
array([[86,  2],
       [ 9, 93]], dtype=int64)
```

[321]:
```python
"Confusion Matrix (Tuned):"
confusion_matrix(y_test, y_pred_best)
```

[321]: array([[86, 2],
       [9, 93]], dtype=int64)

[327]:
```
class_report = classification_report(y_test, y_pred_best)
"\nClassification Report (Tuned):"
class_report
```

[327]:
```
'              precision    recall  f1-score   support\n\n           0
0.91      0.98      0.94        88\n           1       0.98      0.91      0.94
102\n\n    accuracy                           0.94       190\n   macro avg
0.94      0.94      0.94       190\nweighted avg       0.94      0.94      0.94
190\n'
```

[329]:
```
"\nClassification Report (Tuned):"
classification_report(y_test, y_pred_best)
```

[329]:
```
'              precision    recall  f1-score   support\n\n           0
0.91      0.98      0.94        88\n           1       0.98      0.91      0.94
102\n\n    accuracy                           0.94       190\n   macro avg
0.94      0.94      0.94       190\nweighted avg       0.94      0.94      0.94
190\n'
```

[337]:
```
accuracy_score(y_test, y_pred_best)
```

[337]: 0.9421052631578948

[403]:
```
# save the model
```

[345]:
```
!pip install joblib
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: joblib in c:\programdata\anaconda3\lib\site-
packages (1.2.0)
```

[346]:
```
import joblib
```

[420]:
```
joblib.dump(model, 'credit_card_model')
```

[420]: ['credit_card_model']

[434]:
```
model1=joblib.load("credit_card_model")
```

[460]:

```
pred1=model1.predict([[-1.359807,        -0.072781,        2.536347,        1.
↪378155,        -0.338321,        0.462388,        0.239599,        0.
↪098698,        0.363787,        0.090794,        -0.551600,        -0.
↪617801,        -0.991390,        -0.311169,        1.468177,        -0.
↪470401,        0.207971,        0.025791,        0.403993,        0.
↪251412,        -0.018307,        0.277838,        -0.110474,        0.
↪066928,        0.128539,        -0.189115,        0.133558,        -0.
↪021053,  0.244964]])
```

[456]:
```
import warnings
warnings.filterwarnings('ignore')
```

[462]:
```
if pred1 ==0:
    print("normal transaction")
else:
    print("fraud transaction")
```

```
normal transaction
```

[ ]: