

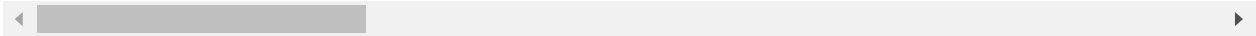
```
In [1]: import pandas as pd
import numpy as np
```

```
In [2]: a=pd.read_csv(r"C:\All Datasets\train.csv")
a
```

Out[2]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Mo
0	5634	3392	1	Aaron Maashoh	23.0	821000265.0	Scientist	19114.12	
1	5635	3392	2	Aaron Maashoh	23.0	821000265.0	Scientist	19114.12	
2	5636	3392	3	Aaron Maashoh	23.0	821000265.0	Scientist	19114.12	
3	5637	3392	4	Aaron Maashoh	23.0	821000265.0	Scientist	19114.12	
4	5638	3392	5	Aaron Maashoh	23.0	821000265.0	Scientist	19114.12	
...	...	...	...	...	...	...	...	...	...
99995	155625	37932	4	Nicks	25.0	78735990.0	Mechanic	39628.99	
99996	155626	37932	5	Nicks	25.0	78735990.0	Mechanic	39628.99	
99997	155627	37932	6	Nicks	25.0	78735990.0	Mechanic	39628.99	
99998	155628	37932	7	Nicks	25.0	78735990.0	Mechanic	39628.99	
99999	155629	37932	8	Nicks	25.0	78735990.0	Mechanic	39628.99	

100000 rows × 28 columns



```
In [3]: a.isnull().sum()
```

```
Out[3]: ID                                0
        Customer_ID                       0
        Month                             0
        Name                              0
        Age                               0
        SSN                               0
        Occupation                        0
        Annual_Income                     0
        Monthly_Inhand_Salary             0
        Num_Bank_Accounts                  0
        Num_Credit_Card                    0
        Interest_Rate                      0
        Num_of_Loan                        0
        Type_of_Loan                       0
        Delay_from_due_date                0
        Num_of_Delayed_Payment             0
        Changed_Credit_Limit               0
        Num_Credit_Inquiries               0
        Credit_Mix                         0
        Outstanding_Debt                   0
        Credit_Utilization_Ratio           0
        Credit_History_Age                 0
        Payment_of_Min_Amount              0
        Total_EMI_per_month                0
        Amount_invested_monthly            0
        Payment_Behaviour                  0
        Monthly_Balance                    0
        Credit_Score                       0
        dtype: int64
```

```
In [4]: from sklearn.preprocessing import LabelEncoder
        l=LabelEncoder()
        a["Name"]=l.fit_transform(a["Name"])
        a["Occupation"]=l.fit_transform(a["Occupation"])
        a["Credit_Mix"]=l.fit_transform(a["Credit_Mix"])
        a["Payment_of_Min_Amount"]=l.fit_transform(a["Payment_of_Min_Amount"])
        a["Payment_Behaviour"]=l.fit_transform(a["Payment_Behaviour"])
        a["Type_of_Loan"]=l.fit_transform(a["Type_of_Loan"])
```

```
In [5]: a["Credit_Score"].unique()
```

```
Out[5]: array(['Good', 'Standard', 'Poor'], dtype=object)
```

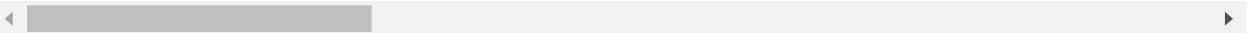
```
In [6]: a["Credit_Score"]=a["Credit_Score"].map({"Good":10,"Standard":20,"Poor":30})
```

In [7]: a

Out[7]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Month
0	5634	3392	1	11	23.0	821000265.0	12	19114.12	
1	5635	3392	2	11	23.0	821000265.0	12	19114.12	
2	5636	3392	3	11	23.0	821000265.0	12	19114.12	
3	5637	3392	4	11	23.0	821000265.0	12	19114.12	
4	5638	3392	5	11	23.0	821000265.0	12	19114.12	
...	...	...	...	...	...	...	...	...	...
99995	155625	37932	4	6508	25.0	78735990.0	9	39628.99	
99996	155626	37932	5	6508	25.0	78735990.0	9	39628.99	
99997	155627	37932	6	6508	25.0	78735990.0	9	39628.99	
99998	155628	37932	7	6508	25.0	78735990.0	9	39628.99	
99999	155629	37932	8	6508	25.0	78735990.0	9	39628.99	

100000 rows × 28 columns



```
In [8]: x=a.iloc[:, :-1].values
        y=a.iloc[:, -1].values
```

```
In [9]: from sklearn.preprocessing import StandardScaler
        s=StandardScaler()
        x1=s.fit_transform(x)
```

```
In [10]: a["Credit_Score"].value_counts()
```

```
Out[10]: 20    53174
         30    28998
         10    17828
         Name: Credit_Score, dtype: int64
```

```
In [11]: from imblearn.over_sampling import SMOTE
         sm=SMOTE()
         X,Y=sm.fit_resample(x1,y)
```

```
In [12]: from sklearn.model_selection import train_test_split
         x_train,x_test,y_train,y_test=train_test_split(X,Y,random_state=20,test_size=0.2)
```

```
In [13]: from sklearn.neighbors import KNeighborsClassifier
kn=KNeighborsClassifier(n_neighbors=5,metric="minkowski",p=2)
kn.fit(x_train,y_train)
```

Out[13]: KNeighborsClassifier()

```
In [14]: from sklearn.tree import DecisionTreeClassifier
d=DecisionTreeClassifier()
d.fit(x_train,y_train)
```

Out[14]: DecisionTreeClassifier()

```
In [15]: from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

Out[15]: LogisticRegression()

```
In [16]: from sklearn.ensemble import VotingClassifier
vc=VotingClassifier(estimators=[("Knn",kn),("Dec.Tree",d),("log.reg",lr)])
vc.fit(x_train,y_train)
```

Out[16]: VotingClassifier(estimators=[('Knn', KNeighborsClassifier()),  
('Dec.Tree', DecisionTreeClassifier()),  
('log.reg', LogisticRegression())])

```
In [17]: p1=vc.predict(x_test)
```

```
In [18]: p1
```

Out[18]: array([20, 30, 10, ..., 30, 20, 20], dtype=int64)

```
In [19]: from sklearn.metrics import accuracy_score
accuracy_score(p1,y_test)*100
```

Out[19]: 82.37266886068014

```
In [20]: from sklearn.model_selection import StratifiedKFold
sk=StratifiedKFold(n_splits=5,random_state=20,shuffle=True)
sk.get_n_splits(x_train,y_train)
```

Out[20]: 5

```
In [21]: from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import accuracy_score
```

```
In [22]: scores=cross_val_score(vc,x_train,y_train)
pred=cross_val_predict(vc,x_test,y_test)
print((scores)*100)
print(pred)
pred2=accuracy_score(pred,y_test)*100
print(pred2)
```

```
[81.69565899 81.37047485 81.99271246 81.828155    81.62049916]
[20 30 10 ... 30 20 20]
75.17943895941075
```

```
In [ ]:
```