In [1]:

```
import pandas as pd
df=pd.read_csv('census.csv')
```

In [2]:

```
df.isin(['?']).sum(axis=0)
```

Out[2]:

```
age                  0
workclass         2799
fnlwgt               0
education            0
education-num        0
marital-status       0
occupation        2809
relationship         0
race                 0
sex                  0
capital-gain         0
capital-loss         0
hours-per-week       0
native-country     857
income               0
dtype: int64
```

In [3]:

```
import numpy as np
```

In [4]:

```
df['native-country'] = df['native-country'].replace('?',np.nan)
df['workclass'] = df['workclass'].replace('?',np.nan)
df['occupation'] = df['occupation'].replace('?',np.nan)
```

In [5]:

```
df.describe()
```

Out[5]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week
count	48842.000000	4.884200e+04	48842.000000	48842.000000	48842.000000	48842.000000
mean	38.643585	1.896641e+05	10.078089	1079.067626	87.502314	40.422382
std	13.710510	1.056040e+05	2.570973	7452.019058	403.004552	12.391444
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.175505e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.781445e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.376420e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.490400e+06	16.000000	99999.000000	4356.000000	99.000000

In [6]:

```
#statistical sumaary -categorical features
df.describe(include=object)
```

Out[6]:

	workclass	education	marital-status	occupation	relationship	race	sex	native-country	income

	workclass	education	marital-status	occupation	relationship	race	sex	native-country	income
count	46043	48842	48842	46033	48842	48842	48842	47985	48842
unique	8	16	7	14	6	5	2	41	2
top	Private	HS-grad	Married-civ-spouse	Prof-specialty	Husband	White	Male	United-States	<=50K
freq	33906	15784	22379	6172	19716	41762	32650	43832	37155

In [7]:

```
num_fea = df.select_dtypes(include=['int64', 'float64']).columns
```

In [8]:

```
#married male with higher income
df.loc[(df['sex'] == 'Male') &
      (df['marital-status'].str.startswith('Married')), 'income'].value_counts()
```

Out[8]:

```
<=50K    11318
>50K      8917
Name: income, dtype: int64
```

In [9]:

```
#married female with higher income
df.loc[(df['sex'] == 'Female') &
      (df['marital-status'].str.startswith('Married')), 'income'].value_counts()
```

Out[9]:

```
<=50K    1670
>50K     1139
Name: income, dtype: int64
```

In [10]:

```
df_drop= pd.DataFrame(df, columns=df.columns,index=df.index)
```

In [11]:

```
df_drop.dropna(how='any',inplace=True)
```

In [12]:

```
X=df_drop.drop('income', axis = 1)
```

In [13]:

```
Y=pd.DataFrame(df_drop["income"], columns=['income'])
```

In [14]:

```
cat_iX = X.select_dtypes(include=['object', 'bool']).columns
```

In [15]:

```
cat_iX
```

Out[15]:

```
Index(['workclass', 'education', 'marital-status', 'occupation',
       'relationship', 'race', 'sex', 'native-country'],
      dtype='object')
```

In [16]:

```
data1= X[cat_iX]
```

In [17]:

```
data1
```

```
data1
```

Out[17]:

	workclass	education	marital-status	occupation	relationship	race	sex	native-country
0	Private	11th	Never-married	Machine-op-inspct	Own-child	Black	Male	United-States
1	Private	HS-grad	Married-civ-spouse	Farming-fishing	Husband	White	Male	United-States
2	Local-gov	Assoc-acdm	Married-civ-spouse	Protective-serv	Husband	White	Male	United-States
3	Private	Some-college	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	United-States
5	Private	10th	Never-married	Other-service	Not-in-family	White	Male	United-States
...	...	...	...	...	...	...	...	...
48837	Private	Assoc-acdm	Married-civ-spouse	Tech-support	Wife	White	Female	United-States
48838	Private	HS-grad	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	United-States
48839	Private	HS-grad	Widowed	Adm-clerical	Unmarried	White	Female	United-States
48840	Private	HS-grad	Never-married	Adm-clerical	Own-child	White	Male	United-States
48841	Self-emp-inc	HS-grad	Married-civ-spouse	Exec-managerial	Wife	White	Female	United-States

45222 rows × 8 columns

In [18]:

```
from sklearn.preprocessing import OneHotEncoder
from sklearn.impute import SimpleImputer
```

In [19]:

```
enc_missing = SimpleImputer(strategy="most_frequent",fill_value="missing")
```

In [20]:

```
enc_missing.fit(data1)
```

Out[20]:

```
SimpleImputer(fill_value='missing', strategy='most_frequent')
```

In [21]:

```
data1 = pd.DataFrame(enc_missing.transform(data1),columns=data1.columns,index=data1.inde
x)
```

In [22]:

```
data1
```

Out[22]:

	workclass	education	marital-status	occupation	relationship	race	sex	native-country
0	Private	11th	Never-married	Machine-op-inspct	Own-child	Black	Male	United-States
1	Private	HS-grad	Married-civ-spouse	Farming-fishing	Husband	White	Male	United-States
2	Local-gov	Assoc-acdm	Married-civ-spouse	Protective-serv	Husband	White	Male	United-States
3	Private	Some-college	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	United-States
5	Private	10th	Never-married	Other-service	Not-in-family	White	Male	United-States
...	...	...	...	...	...	...	...	...
48837	Private	Assoc-acdm	Married-civ-spouse	Tech-support	Wife	White	Female	United-States
48838	Private	HS-grad	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	United-States
48839	Private	HS-grad	Widowed	Adm-clerical	Unmarried	White	Female	United-States
48840	Private	HS-grad	Never-married	Adm-clerical	Own-child	White	Male	United-States

	workclass	education	marital-status	occupation	relationship	race	sex	native-country
48841	Self-emp-inc	HS-grad	Married-civ-spouse	Exec-managerial	Wife	White	Female	United-States

45222 rows × 8 columns

In [23]:

```
ohe = OneHotEncoder(sparse=False, handle_unknown='ignore')
```

In [24]:

```
ohe
```

Out[24]:

```
OneHotEncoder(handle_unknown='ignore', sparse=False)
```

In [25]:

```
enc_x=ohe.fit_transform(data1)
```

In [26]:

```
enc_x
```

Out[26]:

```
array([[0., 0., 1., ..., 1., 0., 0.],
       [0., 0., 1., ..., 1., 0., 0.],
       [0., 1., 0., ..., 1., 0., 0.],
       ...,
       [0., 0., 1., ..., 1., 0., 0.],
       [0., 0., 1., ..., 1., 0., 0.],
       [0., 0., 0., ..., 1., 0., 0.]])
```

In [27]:

```
col_name= ohe.get_feature_names(['workclass','education','marital-status','occupation','
relationship','race','sex','native-country'])
```

In [28]:

```
col_name
```

Out[28]:

```
array(['workclass_Federal-gov', 'workclass_Local-gov',
       'workclass_Private', 'workclass_Self-emp-inc',
       'workclass_Self-emp-not-inc', 'workclass_State-gov',
       'workclass_Without-pay', 'education_10th', 'education_11th',
       'education_12th', 'education_1st-4th', 'education_5th-6th',
       'education_7th-8th', 'education_9th', 'education_Assoc-acdm',
       'education_Assoc-voc', 'education_Bachelors',
       'education_Doctorate', 'education_HS-grad', 'education_Masters',
       'education_Preschool', 'education_Prof-school',
       'education_Some-college', 'marital-status_Divorced',
       'marital-status_Married-AF-spouse',
       'marital-status_Married-civ-spouse',
       'marital-status_Married-spouse-absent',
       'marital-status_Never-married', 'marital-status_Separated',
       'marital-status_Widowed', 'occupation_Adm-clerical',
       'occupation_Armed-Forces', 'occupation_Craft-repair',
       'occupation_Exec-managerial', 'occupation_Farming-fishing',
       'occupation_Handlers-cleaners', 'occupation_Machine-op-inspct',
       'occupation_Other-service', 'occupation_Priv-house-serv',
       'occupation_Prof-specialty', 'occupation_Protective-serv',
       'occupation_Sales', 'occupation_Tech-support',
       'occupation_Transport-moving', 'relationship_Husband',
       'relationship_Not-in-family', 'relationship_Other-relative',
       'relationship_Own-child', 'relationship_Unmarried',
       'relationship_Wife', 'race_Amer-Indian-Eskimo',
       'race_Asian-Pac-Islander', 'race_Black', 'race_Other',
       'race_White', 'sex_Female', 'sex_Male', 'native-country_Cambodia',
       'native-country_Canada', 'native-country_China',
       'native-country_Columbia', 'native-country_Cuba',
       'native-country_Dominican-Republic', 'native-country_Ecuador',
       'native-country_El-Salvador', 'native-country_England',
       'native-country_France', 'native-country_Germany',
       'native-country_Greece', 'native-country_Guatemala',
       'native-country_Haiti', 'native-country_Holand-Netherlands',
       'native-country_Honduras', 'native-country_Hong',
       'native-country_Hungary', 'native-country_India',
       'native-country_Iran', 'native-country_Ireland',
       'native-country_Italy', 'native-country_Jamaica',
       'native-country_Japan', 'native-country_Laos',
       'native-country_Mexico', 'native-country_Nicaragua',
       'native-country_Outlying-US(Guam-USVI-etc)', 'native-country_Peru',
       'native-country_Philippines', 'native-country_Poland',
       'native-country_Portugal', 'native-country_Puerto-Rico',
       'native-country_Scotland', 'native-country_South',
       'native-country_Taiwan', 'native-country_Thailand',
       'native-country_Trinadad&Tobago', 'native-country_United-States',
       'native-country_Vietnam', 'native-country_Yugoslavia'],
      dtype=object)
```

In [29]:

```
encoded_x= pd.DataFrame(enc_x,columns=col_name, index=data1.index)
```

In [30]:

```
encoded_x
```

Out[30]:

	workclass_Federal-gov	workclass_Local-gov	workclass_Private	workclass_Self-emp-inc	workclass_Self-emp-not-inc	workclass_State-gov	workclas
0	0.0	0.0	1.0	0.0	0.0	0.0	
1	0.0	0.0	1.0	0.0	0.0	0.0	
2	0.0	1.0	0.0	0.0	0.0	0.0	
3	0.0	0.0	1.0	0.0	0.0	0.0	
5	0.0	0.0	1.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	
48837	0.0	0.0	1.0	0.0	0.0	0.0	
48838	0.0	0.0	1.0	0.0	0.0	0.0	
48839	0.0	0.0	1.0	0.0	0.0	0.0	
48840	0.0	0.0	1.0	0.0	0.0	0.0	
48841	0.0	0.0	0.0	1.0	0.0	0.0	

45222 rows × 98 columns

In [31]:

```
num_ix = X.select_dtypes(include=['int64','float64']).columns
```

In [32]:

```
num_ix
```

Out[32]:

```
Index(['age', 'fnlwgt', 'education-num', 'capital-gain', 'capital-loss',
       'hours-per-week'],
      dtype='object')
```

In [33]:

```
dat=pd.DataFrame(X[num_ix],columns=['age','education-num', 'capital-gain', 'capital-loss
', 'hours-per-week'], index= data1.index)
```

In [34]:

```
finl_x=pd.concat([dat,encoded_x],axis=1)
```

In [35]:

```
finl_x
```

Out[35]:

	age	education-num	capital-gain	capital-loss	hours-per-week	workclass_Federal-gov	workclass_Local-gov	workclass_Private	workclass_Self-emp-inc
0	25	7	0	0	40	0.0	0.0	1.0	0.0
1	38	9	0	0	50	0.0	0.0	1.0	0.0
2	28	12	0	0	40	0.0	1.0	0.0	0.0
3	44	10	7688	0	40	0.0	0.0	1.0	0.0
5	34	6	0	0	30	0.0	0.0	1.0	0.0
...	...	...	...	...	...	...	...	...	...
48837	27	12	0	0	38	0.0	0.0	1.0	0.0
48838	40	9	0	0	40	0.0	0.0	1.0	0.0
48839	58	9	0	0	40	0.0	0.0	1.0	0.0
48840	22	9	0	0	20	0.0	0.0	1.0	0.0
48841	52	9	15024	0	40	0.0	0.0	0.0	1.0

45222 rows × 103 columns

In [36]:

```
from sklearn.model_selection import train_test_split
```

In [37]:

```
x_train,x_test,y_train,y_test=train_test_split(finl_x,Y,test_size=0.30, random_state=42,
shuffle=True)
```

In [38]:

```
from sklearn.preprocessing import MinMaxScaler
```

In [39]:

```
num_ix = x_train.select_dtypes(include=['int64', 'float64']).columns
```

In [40]:

```
num_ix
```

Out[40]:

```
Index(['age', 'education-num', 'capital-gain', 'capital-loss',
       'hours-per-week', 'workclass_Federal-gov', 'workclass_Local-gov',
       'workclass_Private', 'workclass_Self-emp-inc',
       'workclass_Self-emp-not-inc',
       ...
       'native-country_Portugal', 'native-country_Puerto-Rico',
       'native-country_Scotland', 'native-country_South',
       'native-country_Taiwan', 'native-country_Thailand',
       'native-country_Trinadad&Tobago', 'native-country_United-States',
       'native-country_Vietnam', 'native-country_Yugoslavia'],
      dtype='object', length=103)
```

In [41]:

```
data=x_train[num_ix]
```

In [42]:

```
data
```

Out[42]:

	age	education-num	capital-gain	capital-loss	hours-per-week	workclass_Federal-gov	workclass_Local-gov	workclass_Private	workclass_Self-emp-inc	v
28065	38	14	7298	0	60	0.0	0.0	0.0	0.0	
43124	47	9	0	1977	45	0.0	0.0	1.0	0.0	
41428	43	13	0	0	50	0.0	0.0	1.0	0.0	
12494	42	16	0	0	40	0.0	0.0	1.0	0.0	
7756	26	9	0	0	40	0.0	0.0	1.0	0.0	
...	...	...	...	...	...	...	...	...	...	
12183	39	6	0	0	40	0.0	0.0	0.0	0.0	
48310	36	14	0	0	50	0.0	1.0	0.0	0.0	
41203	53	5	0	0	40	0.0	1.0	0.0	0.0	
930	57	11	0	0	40	0.0	0.0	1.0	0.0	
17081	43	12	0	0	40	0.0	0.0	1.0	0.0	

31655 rows × 103 columns

In [43]:

```
scaler= MinMaxScaler()
```

In [44]:

```
scaled_train_x=pd.DataFrame(scaler.fit_transform(data),columns=data.columns)
```

In [45]:

```
scaled_train_x
```

Out[45]:

	age	education-num	capital-gain	capital-loss	hours-per-week	workclass_Federal-gov	workclass_Local-gov	workclass_Private	workclas e
0	0.287671	0.866667	0.072981	0.000000	0.602041	0.0	0.0	0.0	
1	0.410959	0.533333	0.000000	0.453857	0.448980	0.0	0.0	1.0	
2	0.356164	0.800000	0.000000	0.000000	0.500000	0.0	0.0	1.0	
3	0.342466	1.000000	0.000000	0.000000	0.397959	0.0	0.0	1.0	
4	0.123288	0.533333	0.000000	0.000000	0.397959	0.0	0.0	1.0	
...	...	...	...	...	...	...	...	...	
31650	0.301370	0.333333	0.000000	0.000000	0.397959	0.0	0.0	0.0	
31651	0.260274	0.866667	0.000000	0.000000	0.500000	0.0	1.0	0.0	
31652	0.493151	0.266667	0.000000	0.000000	0.397959	0.0	1.0	0.0	

	age	education-num	capital-gain	capital-loss	hours-per-week	workclass_Federal-gov	workclass_Local-gov	workclass_Private	workclas e
31653	0.547945	0.666667	0.000000	0.000000	0.397959	0.0	0.0	1.0	
31654	0.356164	0.733333	0.000000	0.000000	0.397959	0.0	0.0	1.0	

31655 rows × 103 columns

In [46]:

```
from sklearn.preprocessing import LabelEncoder
```

In [47]:

```
y_train=pd.DataFrame(LabelEncoder().fit_transform(y_train),columns=y_train.columns)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
```

In [48]:

```
y_train
```

Out[48]:

	income
0	1
1	1
2	1
3	1
4	0
...	...
31650	0
31651	1
31652	0
31653	0
31654	0

31655 rows × 1 columns

In [49]:

```
num_ix = x_test.select_dtypes(include=['int64', 'float64']).columns
```

In [50]:

```
data=x_test[num_ix]
```

In [51]:

```
data
```

Out[51]:

	age	education-num	capital-gain	capital-loss	hours-per-week	workclass_Federal-gov	workclass_Local-gov	workclass_Private	workclass_Self-emp-inc
21762	19	6	0	0	40	0.0	0.0	0.0	0.0
21701	45	9	0	0	40	0.0	0.0	1.0	0.0
42663	47	11	0	0	40	0.0	0.0	1.0	0.0

	age	education-num	capital-gain	capital-loss	hours-per-week	workclass_Federal-gov	workclass_Local-gov	workclass_Private	workclass_Self-emp-inc	w
42694	23	10	0	0	40	0.0	0.0	1.0	0.0	
13590	53	9	0	0	40	0.0	1.0	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	
37210	40	10	0	0	40	0.0	0.0	1.0	0.0	
23727	33	9	0	0	40	0.0	0.0	1.0	0.0	
19600	20	9	0	1590	40	0.0	0.0	1.0	0.0	
34019	36	5	0	0	40	0.0	0.0	1.0	0.0	
40039	75	7	0	0	50	0.0	0.0	0.0	1.0	

13567 rows × 103 columns

In [52]:

```
scaler= MinMaxScaler()
```

In [53]:

```
scaler
```

Out[53]:

```
MinMaxScaler()
```

In [54]:

```
scaled_x_test=pd.DataFrame(scaler.fit_transform(data),columns=data.columns)
```

In [55]:

```
scaled_x_test
```

Out[55]:

	age	education-num	capital-gain	capital-loss	hours-per-week	workclass_Federal-gov	workclass_Local-gov	workclass_Private	workclass en
0	0.027397	0.333333	0.0	0.000000	0.397959	0.0	0.0	0.0	
1	0.383562	0.533333	0.0	0.000000	0.397959	0.0	0.0	1.0	
2	0.410959	0.666667	0.0	0.000000	0.397959	0.0	0.0	1.0	
3	0.082192	0.600000	0.0	0.000000	0.397959	0.0	0.0	1.0	
4	0.493151	0.533333	0.0	0.000000	0.397959	0.0	1.0	0.0	
...	...	...	...	...	...	...	...	...	
13562	0.315068	0.600000	0.0	0.000000	0.397959	0.0	0.0	1.0	
13563	0.219178	0.533333	0.0	0.000000	0.397959	0.0	0.0	1.0	
13564	0.041096	0.533333	0.0	0.407692	0.397959	0.0	0.0	1.0	
13565	0.260274	0.266667	0.0	0.000000	0.397959	0.0	0.0	1.0	
13566	0.794521	0.400000	0.0	0.000000	0.500000	0.0	0.0	0.0	

13567 rows × 103 columns

In [56]:

```
# Label encoding target in test data
y_test=pd.DataFrame(LabelEncoder().fit_transform(y_test), columns=y_test.columns)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
```

```
  return f(**kwargs)
```

In [57]:

```
y_test
```

Out[57]:

	income
0	0
1	0
2	1
3	0
4	0
...	...
13562	1
13563	0
13564	0
13565	0
13566	0

13567 rows × 1 columns

In [58]:

```
from sklearn.linear_model import LogisticRegression
```

In [59]:

```
log_reg=LogisticRegression()
```

In [60]:

```
log_reg.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:764: Co
nvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Out[60]:

```
LogisticRegression()
```

In [61]:

```
y_pred = log_reg.predict(x_test)
```

In [62]:

```
y_pred
```

Out[62]:

```
array([0, 0, 0, ..., 0, 0, 0])
```

In [63]:

```
from sklearn.metrics import accuracy_score
```

In [64]:

```
accuracy_score(y_pred,y_test)
```

Out[64]:

```
0.8323137023660352
```

support vector classification

In [65]:

```
from sklearn.ensemble import BaggingClassifier
```

In [66]:

```
bagcl=BaggingClassifier()
```

In [67]:

```
bagcl.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
```

Out[67]:

```
BaggingClassifier()
```

In [68]:

```
from sklearn.metrics import accuracy_score
```

In [69]:

```
bagcl_pred=bagcl.predict(x_test)
```

In [70]:

```
accuracy_score(bagcl_pred,y_test)
```

Out[70]:

```
0.8386526129579126
```

In [71]:

```
from sklearn.svm import SVC
```

In [72]:

```
#model_bagging_svc=BaggingClassifier(base_estimator=LogisticRegression(),n_estimators=50,
random_state=0).fit(x_train,y_train)
```

hyperparameter tunning

bagging clssifer regressor

In [73]:

```
from sklearn.ensemble import BaggingRegressor
```

In [74]:

```
bag_reg_model=BaggingRegressor()
```

In [75]:

```
bag_reg_model.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
```

Out[75]:

```
BaggingRegressor()
```

In [76]:

```
bag_reg_pred=bag_reg_model.predict(x_test)
```

In [77]:

```
bag_reg_pred
```

Out[77]:

```
array([0.1 , 0.1 , 0.35, ..., 0.  , 0.  , 0.3 ])
```

In [78]:

```
from sklearn.metrics import mean_squared_error
```

In [79]:

```
mean_squared_error(y_test,bag_reg_pred)
```

Out[79]:

```
0.11806785415953722
```

In [80]:

```
#extra tree classifer
```

In [81]:

```
from sklearn.ensemble import ExtraTreesClassifier
```

In [82]:

```
extree=ExtraTreesClassifier()
```

In [83]:

```
extree.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: DataConversionWa
rning: A column-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  """Entry point for launching an IPython kernel.
```

Out[83]:

```
ExtraTreesClassifier()
```

In [84]:

```
extre_pred=extree.predict(x_test)
```

In [85]:

```
extre_pred
```

Out[85]:

```
array([0, 0, 0, ..., 0, 0, 0])
```

In [86]:

```
accuracy_score(extre_pred,y_test)
```

Out[86]:

```
0.8244269182575367
```

In [87]:

```
from sklearn.ensemble import ExtraTreesRegressor
```

In [88]:

```
extree_reg=ExtraTreesRegressor()
```

In [89]:

```
extree_reg.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: DataConversionWa
rning: A column-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  """Entry point for launching an IPython kernel.
```

Out[89]:

```
ExtraTreesRegressor()
```

In [90]:

```
extre_reg_pred=extree_reg.predict(x_test)
```

In [91]:

```
mean_squared_error(extre_reg_pred,y_test)
```

Out[91]:

```
0.12931816276025845
```

voting classifer

In [92]:

```
from sklearn.ensemble import VotingClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
svc=SVC()
rfc=RandomForestClassifier()
#log_reg
```

In [93]:

```
label=['Support vector classifier','Random Forest Classifier','Logistic Regreesion']
```

In [94]:

```
from sklearn import model_selection
```

In [95]:

```
"""
```

```
for clf ,label in zip([svc,rfc,log_reg],label):
    scores=model_selection.cross_val_score(clf,finl_x,Y,cv=5,scoring='accuracy')

    print("accuracy:%0.2f (+/- %0.2f) [%s]"
          % (score.mean(),score.std(),label)) """
```

Out[95]:

```
'\nfor clf ,label in zip([svc,rfc,log_reg],label):\n    scores=model_selection.cross_val_
score(clf,finl_x,Y,cv=5,scoring=\'accuracy\')\n    \n    print("accuracy:%0.2f (+/- %0.2f
) [%s]" \n          % (score.mean(),score.std(),label)) '
```

In [96]:

```
vot_class=VotingClassifier(estimators=
                 [(label[0],svc),
                  (label[1],rfc),
                  (label[2],log_reg)])
```

In [97]:

```
vot_class.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:764: Co
nvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Out[97]:

```
VotingClassifier(estimators=[('Support vector classifier', SVC()),
                             ('Random Forest Classifier',
                              RandomForestClassifier()),
                             ('Logistic Regreesion', LogisticRegression())])
```

In [98]:

```
vot_class_pred=vot_class.predict(x_test)
```

In [99]:

```
vot_class_pred
```

Out[99]:

```
array([0, 0, 0, ..., 0, 0, 0])
```

In [100]:

```
accuracy_score(vot_class_pred,y_test)
```

Out[100]:

```
0.8413061104149775
```

In [101]:

```
from sklearn.ensemble import VotingRegressor
```

In [106]:

```
from sklearn.svm import SVR
svr=SVR()
```

```
from sklearn.ensemble import RandomForestRegressor
rfr=RandomForestRegressor()
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

In [ ]:

```
```

In [107]:

```
label2=['Support vector regressor','Random Forest regressor','linear Regreesion']
```

In [108]:

```
vot_reg=VotingRegressor(estimators=
                [(label2[0],svr),
                 (label2[1],rfr),
                 (label2[2],lr)])
```

In [109]:

```
vot_reg.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\sklearn\utils\validation.py:72: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
```

Out[109]:

```
VotingRegressor(estimators=[('Support vector regressor', SVR()),
                            ('Random Forest regressor',
                             RandomForestRegressor()),
                            ('linear Regreesion', LinearRegression())])
```

In [110]:

```
vot_reg_pred=vot_reg.predict(x_test)
```

In [112]:

```
mean_squared_error(vot_reg_pred,y_test)
```

Out[112]:

```
0.10836269201716557
```

Random forest classifier

In [113]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [114]:

```
RFC=RandomForestClassifier()
```

In [115]:

```
RFC.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: DataConversionWa
rning: A column-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  """Entry point for launching an IPython kernel.
```

Out[115]:

```
RandomForestClassifier()
```

In [119]:

```
RFC_PRED=RFC.predict(x_test)
RFC_PRED
```

Out[119]:

```
array([0, 0, 0, ..., 0, 0, 0])
```

In [120]:

```
accuracy_score(RFC_PRED,y_test)
```

Out[120]:

```
0.8426328591435099
```

hyper paramter random forest classifier

In [121]:

```
rf_model_with_best_params2=RandomForestClassifier(criterion='gini',max_depth=14,max_featu
res='log2',min_samples_leaf=1,min_samples_split=2,n_estimators=114)
```

In [122]:

```
rf_model_with_best_params2.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: DataConversionWa
rning: A column-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  """Entry point for launching an IPython kernel.
```

Out[122]:

```
RandomForestClassifier(max_depth=14, max_features='log2', n_estimators=114)
```

In [123]:

```
rf_model_best_pred=rf_model_with_best_params2.predict(x_test)
```

In [124]:

```
rf_model_best_pred
```

Out[124]:

```
array([0, 0, 0, ..., 0, 0, 0])
```

In [125]:

```
accuracy_score(rf_model_best_pred,y_test)
```

Out[125]:

```
0.8603228421906096
```

Radom forest regressor

In [126]:

```
from sklearn.ensemble import RandomForestRegressor
```

In [127]:

```
rfr3=RandomForestRegressor()
```

In [128]:

```
rfr3.fit(x_train,y_train)
```

```
C:\Users\rav smarty\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: DataConversionWa
rning: A column-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  """Entry point for launching an IPython kernel.
```

Out[128]:

```
RandomForestRegressor()
```

In [130]:

```
rfr_pred=rfr3.predict(x_test)
```

In [131]:

```
mean_squared_error(rfr_pred,y_test)
```

Out[131]:

```
0.1108228986591736
```

In [ ]: