In [2]:
```
import numpy as np
import pandas as pd
```

In [3]:
```
df=pd.read_csv("C:/Users/DELL/Desktop/india.csv")
```

In [4]:
```
df.head()
```

Out[4]:

	HSCode	Commodity	value	country	year
0	5	PRODUCTS OF ANIMAL ORIGIN, NOT ELSEWHERE SPECI...	0.00	AFGHANISTAN TIS	2018
1	7	EDIBLE VEGETABLES AND CERTAIN ROOTS AND TUBERS.	12.38	AFGHANISTAN TIS	2018
2	8	EDIBLE FRUIT AND NUTS; PEEL OR CITRUS FRUIT OR...	268.60	AFGHANISTAN TIS	2018
3	9	COFFEE, TEA, MATE AND SPICES.	35.48	AFGHANISTAN TIS	2018
4	11	PRODUCTS OF THE MILLING INDUSTRY; MALT; STARCH...	NaN	AFGHANISTAN TIS	2018

In [5]:
```
df_cat=df.select_dtypes(include=[object])
df_num=df.select_dtypes(include=[np.number])
```

In [6]:
```
df_cat.head()
```

Out[6]:

	Commodity	country
0	PRODUCTS OF ANIMAL ORIGIN, NOT ELSEWHERE SPECI...	AFGHANISTAN TIS
1	EDIBLE VEGETABLES AND CERTAIN ROOTS AND TUBERS.	AFGHANISTAN TIS
2	EDIBLE FRUIT AND NUTS; PEEL OR CITRUS FRUIT OR...	AFGHANISTAN TIS
3	COFFEE, TEA, MATE AND SPICES.	AFGHANISTAN TIS
4	PRODUCTS OF THE MILLING INDUSTRY; MALT; STARCH...	AFGHANISTAN TIS

In [7]:
```
df_num.head()
```

Out[7]:

	HSCode	value	year
0	5	0.00	2018
1	7	12.38	2018
2	8	268.60	2018
3	9	35.48	2018
4	11	NaN	2018

In [8]:
```
df_cat.isnull().sum()
```

Out[8]:
```
Commodity    0
country      0
dtype: int64
```

In [9]:
```
df_num.isnull().sum()
```

Out[9]:
```
HSCode        0
value     14027
year          0
dtype: int64
```

In [10]:
```
df_num.value.fillna(df_num.value.mean(),inplace=True)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py:6130: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  self._update_inplace(new_data)
```

In [11]:
```
df_num.head()
```

Out[11]:

	HSCode	value	year
0	5	0.000000	2018
1	7	12.380000	2018
2	8	268.600000	2018
3	9	35.480000	2018
4	11	63.289855	2018

In [12]:
```
from sklearn.preprocessing import LabelEncoder
```

In [13]:
```
le=LabelEncoder()
```

In [14]:
```
df_cat=df_cat.apply(le.fit_transform)
```

In [15]:
```
df=pd.concat([df_cat,df_num],axis=1)
```

In [16]:
```
df.head()
```

Out[16]:

	Commodity	country	HSCode	value	year
0	71	0	5	0.000000	2018
1	22	0	7	12.380000	2018
2	21	0	8	268.600000	2018
3	16	0	9	35.480000	2018
4	72	0	11	63.289855	2018

In [17]:
```
from sklearn.linear_model import LogisticRegression
```

In [18]:
```
lr=LogisticRegression()
```

In [19]:
```
x=df.iloc[:,[0,1,2,3,4]].values
y=df.iloc[:,2].values
```

In [20]:
```
from sklearn.model_selection import train_test_split
```

In [21]:
```
x_train,x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, rando
m_state=0) #shift+tab
```

In [22]:
```
lr.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
33: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
60: FutureWarning: Default multi_class will be changed to 'auto' in 0.22. Spe
cify the multi_class option to silence this warning.
  "this warning.", FutureWarning)
```

Out[22]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

In [23]:
```
pre=lr.predict(x_test)
```

In [24]:
```
pre
```

Out[24]:
```
array([42, 17, 17, ..., 84, 39, 72], dtype=int64)
```

In [25]:
```
from sklearn.metrics import accuracy_score
```

In [26]:
```
a=accuracy_score(pre,y_test)
```

In [27]:
```
a
```

Out[27]:
```
0.4576199466180587
```

In [28]:
```
fit=lr.fit(x,y)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
33: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
60: FutureWarning: Default multi_class will be changed to 'auto' in 0.22. Spe
cify the multi_class option to silence this warning.
  "this warning.", FutureWarning)
```

In [29]:
```
fit.score(x,y)
```

Out[29]:
```
0.4674579730382942
```

decision tree

In [30]:
```
df.head()
```

Out[30]:

	Commodity	country	HSCode	value	year
0	71	0	5	0.000000	2018
1	22	0	7	12.380000	2018
2	21	0	8	268.600000	2018
3	16	0	9	35.480000	2018
4	72	0	11	63.289855	2018

In [31]:
```
x1=df.iloc[:,[0,1,2,3,4]].values
y=df.iloc[:,2].values
```

In [32]:
```
from sklearn.model_selection import train_test_split
```

In [33]:
```
x1_train, x1_test, y_train, y_test = train_test_split(x1, y, test_size=0.33, r
andom_state=0)
```

In [34]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [35]:
```
dt=DecisionTreeClassifier()
```

In [36]:
```
dt.fit(x1_train,y_train)
```

Out[36]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

In [37]:
```
pre=dt.predict(x1_test)
```

In [38]:
```
pre
```

Out[38]:
```
array([42, 17, 17, ..., 75, 49, 54], dtype=int64)
```

In [39]:
```
from sklearn.metrics import accuracy_score
```

In [40]:
```
b=accuracy_score(y_test,pre)
```

In [42]:
```
b
```

Out[42]:
```
1.0
```

random forest

In [43]:
```
df.head()
```

Out[43]:

	Commodity	country	HSCode	value	year
0	71	0	5	0.000000	2018
1	22	0	7	12.380000	2018
2	21	0	8	268.600000	2018
3	16	0	9	35.480000	2018
4	72	0	11	63.289855	2018

In [44]:
```
x2=df.iloc[:,[0,1,2,3,4]].values
y=df.iloc[:,2].values
```

In [45]:
```
from sklearn.model_selection import train_test_split
```

In [46]:
```
x2_train, x2_test, y_train, y_test = train_test_split(x2, y, test_size=0.33, r
andom_state=0)
```

In [47]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [48]:
```
rf=RandomForestClassifier()
```

In [49]:
```
rf.fit(x2_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: Fu
tureWarning: The default value of n_estimators will change from 10 in version
0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

Out[49]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

In [51]:
```
pre=rf.predict(x2_test)
```

In [52]:
```
pre
```

Out[52]:
```
array([42, 17, 17, ..., 75, 49, 54], dtype=int64)
```

In [53]:
```
from sklearn.metrics import accuracy_score
```

In [54]:
```
c=accuracy_score(y_test,pre)
```

In [55]:
```
c
```

Out[55]:
```
0.9996419503938546
```

knn(k nearest neighbour)

In [56]:
```
df.head()
```

Out[56]:

	Commodity	country	HSCode	value	year
0	71	0	5	0.000000	2018
1	22	0	7	12.380000	2018
2	21	0	8	268.600000	2018
3	16	0	9	35.480000	2018
4	72	0	11	63.289855	2018

In [58]:
```
x3=df.iloc[:,[0,1,2,3,4]].values
y=df.iloc[:,2].values
```

In [59]:
```
from sklearn.model_selection import train_test_split
```

In [60]:
```
x3_train, x3_test, y_train, y_test = train_test_split(x3, y, test_size=0.33, r
andom_state=0)
```

In [61]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [62]:
```
knn=KNeighborsClassifier()
```

In [63]:
```
knn.fit(x3_train,y_train)
```

Out[63]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=None, n_neighbors=5, p=2,
           weights='uniform')
```

In [64]:
```
pre=knn.predict(x3_test)
```

In [65]:
```
pre
```

Out[65]:
```
array([42, 26, 17, ..., 75, 49, 54], dtype=int64)
```

In [66]:
```
from sklearn.metrics import accuracy_score
```

In [67]:
```
d=accuracy_score(y_test,pre)
```

In [68]:
```
d
```

Out[68]:
```
0.9014061584532257
```

svm(support vector machine)

In [69]:
```
df.head()
```

Out[69]:

	Commodity	country	HSCode	value	year
0	71	0	5	0.000000	2018
1	22	0	7	12.380000	2018
2	21	0	8	268.600000	2018
3	16	0	9	35.480000	2018
4	72	0	11	63.289855	2018

In [70]:
```
x4=df.iloc[:,[0,1,2,3,4]].values
y=df.iloc[:,2].values
```

In [71]:
```
from sklearn.model_selection import train_test_split
```

In [72]:
```
x4_train, x4_test, y_train, y_test = train_test_split(x4, y, test_size=0.33, r
andom_state=0)
```

In [73]:
```
from sklearn.svm import LinearSVC
```

In [74]:
```
svm=LinearSVC()
```

In [75]:
```
svm.fit(x4_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:931: Convergen
ceWarning: Liblinear failed to converge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
```

Out[75]:
```
LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
     intercept_scaling=1, loss='squared_hinge', max_iter=1000,
     multi_class='ovr', penalty='l2', random_state=None, tol=0.0001,
     verbose=0)
```

In [76]:
```
pre=svm.predict(x4_test)
```

In [77]:
```
pre
```

Out[77]:
```
array([54, 54, 54, ..., 54, 54, 54], dtype=int64)
```

In [78]:
```
from sklearn.metrics import accuracy_score
```

In [79]:
```
e=accuracy_score(y_test,pre)
```

In [80]:
```
e
```

Out[80]:
```
0.019009179089903003
```

naive baiyes(nb)

In [81]:
```
df.head()
```

Out[81]:

	Commodity	country	HSCode	value	year
0	71	0	5	0.000000	2018
1	22	0	7	12.380000	2018
2	21	0	8	268.600000	2018
3	16	0	9	35.480000	2018
4	72	0	11	63.289855	2018

In [82]:
```
x5=df.iloc[:,[0,1,2,3,4]].values
y=df.iloc[:,2].values
```

In [83]:
```
from sklearn.model_selection import train_test_split
```

In [84]:
```
x5_train, x5_test, y_train, y_test = train_test_split(x5, y, test_size=0.33, r
andom_state=0)
```

In [85]:
```
from sklearn.naive_bayes import GaussianNB
```

In [86]:
```
nb=GaussianNB()
```

In [87]:
```
nb.fit(x5_train,y_train)
```

Out[87]:
```
GaussianNB(priors=None, var_smoothing=1e-09)
```

In [92]:
```
pre=nb.predict(x5_test)
```

In [93]:
```
pre
```

Out[93]:
```
array([42, 17, 17, ..., 75, 49, 54], dtype=int64)
```

In [94]:
```
from sklearn.metrics import accuracy_score
```

In [95]:
```
f=accuracy_score(y_test,pre)
```

In [96]:
```
f
```

Out[96]:
```
1.0
```

In [ ]:
```

```