pract-nb

June 7, 2023

[13]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import␣
 ↪classification_report,confusion_matrix,accuracy_score
import matplotlib.pyplot as plt
from sklearn import preprocessing
```

[2]:
```
data=pd.read_csv('car_evaluation.csv')
```

[3]:
```
print("Descriptive statistics:\n",data.describe())
```

```
Descriptive statistics:
        Buying Maintenance Doors Persons Luggage_Boot Safety Target
count    1728        1728  1728    1728         1728   1728   1728
unique      4           4     4       3            3      3      4
top     vhigh       vhigh     2       2        small    low  unacc
freq      432         432   432     576          576    576   1210
```

[4]:
```
print(data)
```

```
     Buying Maintenance  Doors Persons Luggage_Boot Safety Target
0     vhigh       vhigh      2       2        small    low  unacc
1     vhigh       vhigh      2       2        small    med  unacc
2     vhigh       vhigh      2       2        small   high  unacc
3     vhigh       vhigh      2       2          med    low  unacc
4     vhigh       vhigh      2       2          med    med  unacc
…     …           …          …       …          …      …    …
1723    low         low  5more    more          med    med   good
1724    low         low  5more    more          med   high  vgood
1725    low         low  5more    more          big    low  unacc
1726    low         low  5more    more          big    med   good
1727    low         low  5more    more          big   high  vgood

[1728 rows x 7 columns]
```

[6]:
```
le=preprocessing.LabelEncoder()
data['Buying']=le.fit_transform(data['Buying'])
data['Maintenance']=le.fit_transform(data['Maintenance'])
data['Doors']=le.fit_transform(data['Doors'])
data['Persons']=le.fit_transform(data['Persons'])
data['Luggage_Boot']=le.fit_transform(data['Luggage_Boot'])
data['Safety']=le.fit_transform(data['Safety'])
```

[8]:
```
data.head()
```

[8]:

	Buying	Maintenance	Doors	Persons	Luggage_Boot	Safety	Target
0	3	3	0	0	2	1	unacc
1	3	3	0	0	2	2	unacc
2	3	3	0	0	2	0	unacc
3	3	3	0	0	1	1	unacc
4	3	3	0	0	1	2	unacc

[9]:
```
x=data.drop('Target',axis=1)
y=data['Target']
```

[11]:
```
import numpy as np
test_size=np.linspace(0.1,0.9,9)
test_size_values=[]
accuracy_values=[]

for test_size in test_size:


  ␣
 ↪x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=test_size,random_state=5)
```

[14]:
```
model=GaussianNB()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
print("Naive Bayes Classification Report:\n")
print(classification_report(y_test,y_pred))
cm=confusion_matrix(y_test,y_pred)
print(cm)
accuracy=accuracy_score(y_test,y_pred)
print("accuracy score:\n",accuracy)
```

```
Naive Bayes Classification Report:

              precision    recall  f1-score   support

         acc       0.41      0.21      0.27       346
        good       0.00      0.00      0.00        64
       unacc       0.86      0.82      0.84      1088
```

```
       vgood       0.17      1.00      0.29        58

    accuracy                           0.66      1556
   macro avg       0.36      0.51      0.35      1556
weighted avg       0.70      0.66      0.66      1556

[[ 71   0 125 150]
 [ 21   0  16  27]
 [ 81   0 894 113]
 [  0   0   0  58]]
accuracy score:
 0.6574550128534704
```

```
C:\Users\pksef\anaconda3\lib\site-
packages\sklearn\metrics\_classification.py:1318: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\pksef\anaconda3\lib\site-
packages\sklearn\metrics\_classification.py:1318: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\pksef\anaconda3\lib\site-
packages\sklearn\metrics\_classification.py:1318: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

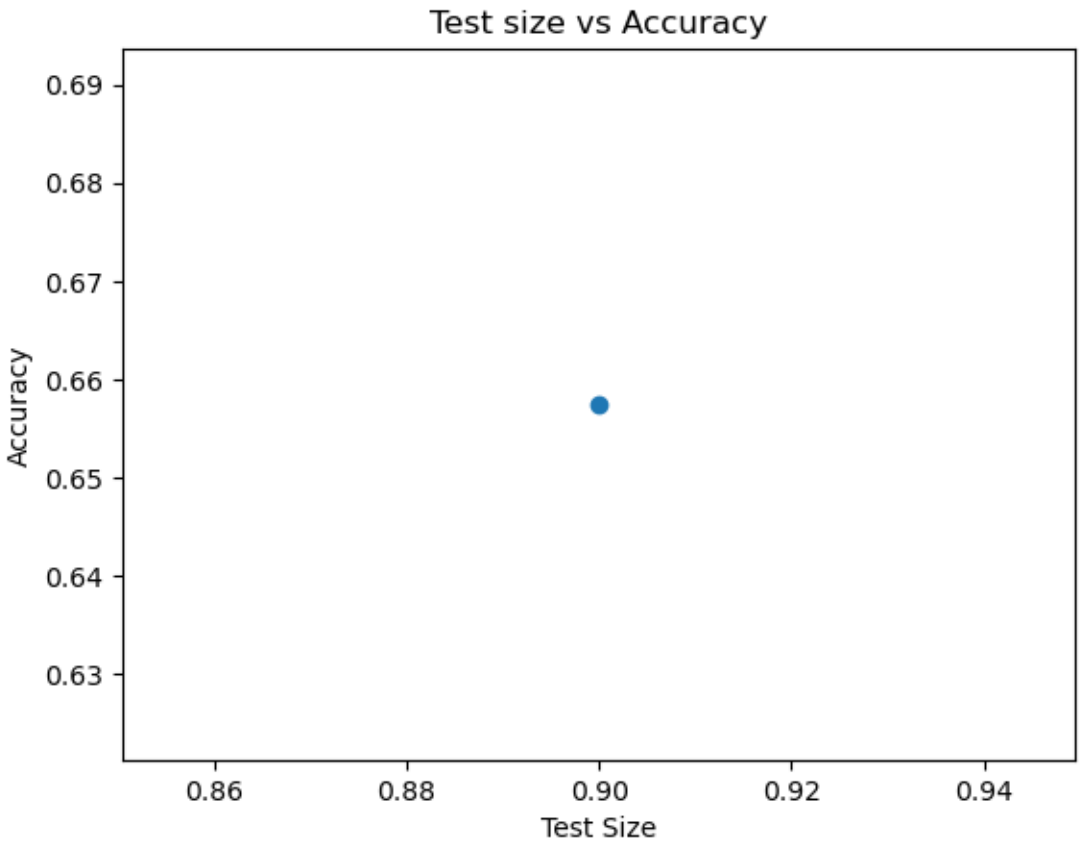
[15]:
```
model1=DecisionTreeClassifier()
model1.fit(x_train,y_train)
y_pred=model.predict(x_test)
print("Decision Tree Classification Report:\n")
print(classification_report(y_test,y_pred))
cm=confusion_matrix(y_test,y_pred)
```

```
Decision Tree Classification Report:

              precision    recall  f1-score   support

         acc       0.41      0.21      0.27       346
        good       0.00      0.00      0.00        64
       unacc       0.86      0.82      0.84      1088
       vgood       0.17      1.00      0.29        58

    accuracy                           0.66      1556
   macro avg       0.36      0.51      0.35      1556
weighted avg       0.70      0.66      0.66      1556
```

```
C:\Users\pksef\anaconda3\lib\site-
packages\sklearn\metrics\_classification.py:1318: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\pksef\anaconda3\lib\site-
packages\sklearn\metrics\_classification.py:1318: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\pksef\anaconda3\lib\site-
packages\sklearn\metrics\_classification.py:1318: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[16]:

```
test_size_values.append(test_size)
accuracy_values.append(accuracy)
plt.plot(test_size,accuracy,marker='o')
plt.xlabel('Test Size')
plt.ylabel('Accuracy')
plt.title('Test size vs Accuracy')
plt.show()
```


Test size vs Accuracy
0.69
0.68
0.67
0.66
0.65
0.64
0.63
Accuracy
0.86
0.88
0.90
0.92
0.94
Test Size

[ ]: