HW2&3_question10_FeatureSelection

March 25, 2021

[130]:
```
import pandas as pd

# read the breast-cancer-wisconsin dataset, also named bcw
bcw = pd.read_csv("breast-cancer-wisconsin.data", header=None)
```

[132]:
```
# drop first column--code number
bcw = bcw.iloc[:,1:11]
# imputated by mean property
for i in range(len(bcw)):
    for j in range(len(bcw.columns)):
        if(bcw.iloc[i,j]=='?'):
            bcw.iloc[i,j]=None
            bcw.iloc[i,j]=int(bcw.iloc[i].mean(skipna=True))
            break
# dataset features
bcw_f = bcw.iloc[:,1:10]
# dataset label
bcw_l = bcw.iloc[:,-1]
```

[133]:
```
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest, chi2

# first split 0.6% train and 0.4% other datasets
train_data, test_data, train_label, test_label = train_test_split(bcw_f, bcw_l,
 ↪random_state=None, train_size=0.6)
# then split 0.2% feature and 0.2% test datasets
feature_data, test_data, feature_label, test_label =␣
 ↪train_test_split(test_data, test_label,
 ↪random_state=None, train_size=0.5)
# extract top 3 best features
selected_feature = SelectKBest(score_func=chi2, k=3).fit(feature_data,␣
 ↪feature_label)
scores = pd.DataFrame(selected_feature.scores_)
columns = pd.DataFrame(bcw_f.columns)
feature_scores= pd.concat([columns,scores],axis=1)
```

```
feature_scores.columns= ['feature_idx','Score']
print(feature_scores.nlargest(3,'Score'))
# reconstruct the train and test data with selected features
train_selected_data = selected_feature.transform(train_data)
test_selected_data = selected_feature.transform(test_data)
```

```
   feature_idx       Score
3            6  344.411260
0            3  274.509074
5            8  253.880282
```

[134]:
```
from sklearn.neighbors import KNeighborsClassifier

# avg of score
avg = 0

# run 10 times
for i in range(10):
    knn_3 = KNeighborsClassifier(n_neighbors = 3)
    knn_3.fit(train_selected_data, train_label)
    avg = avg + knn_3.score(test_selected_data, test_label)

# get average of score
print('avg:',avg/10)
```

```
avg: 0.9714285714285713
```