1. kNN & k-means

March 28, 2022

[1]:
```
import numpy as np
import scipy
```

[2]:
```
# kNN
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split

data = load_iris(as_frame=True)
X=data.data
y=data.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)

k=5
y_pred=[]

# find euclidean distance of each point in test set with each point in the␣
 ↪training set
for test in np.array(X_test):
    distances=[]
    for train in np.array(X_train):
        distance = np.linalg.norm(test-train)
        distances.append(distance)
    distances = np.array(distances)

    # find index of k points in training set which are closest to test point
    knn_ids = np.argsort(distances)[:k]
    # find the label associated with these k points
    knn_labels = y_train.iloc[knn_ids]
    # find the label having highest frequency
    label = scipy.stats.mode(knn_labels)[0][0]
    # add it to the list of labels
    y_pred.append(label)

from sklearn.metrics import classification_report
cr = classification_report(y_test,y_pred)
print(cr)
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11
           1       0.88      1.00      0.93         7
           2       1.00      0.92      0.96        12

    accuracy                           0.97        30
   macro avg       0.96      0.97      0.96        30
weighted avg       0.97      0.97      0.97        30
```

[3]:
```
# k-means
from scipy.spatial.distance import cdist
from sklearn.datasets import load_iris
data = load_iris()
X=data.data

 # randomly choose centroids from the dataset itself
idx = np.random.choice(len(X), k, replace=False)
centroids = X[idx, :]

#find the distance between centroids and all the data points
distances = cdist(X, centroids ,'euclidean')

# for each point in dataset find centroid with the minimum distance
points = np.array([np.argmin(i) for i in distances])

# Repeat the above steps for a defined number of iterations
for _ in range(100):
    centroids = []
    for idx in range(k):
        # update centroids by taking mean of cluster it belongs to
        temp_cent = X[points==idx].mean(axis=0)
        centroids.append(temp_cent)

    distances = cdist(X, centroids ,'euclidean')
    labels = np.array([np.argmin(i) for i in distances])
print(labels)
```

```
[4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 2 3 3 3 2 3 2 2 3 2 3 2 3 0 2 3 2 0 3 3 3
 3 3 3 3 3 2 2 2 2 0 0 3 3 3 2 2 2 3 2 2 2 2 2 3 2 2 1 0 1 0 1 1 2 1 1 1 0
 0 1 0 0 1 1 1 1 0 1 0 1 0 1 1 0 0 1 1 1 1 1 3 0 1 1 1 0 1 1 1 0 1 1 1 0 0
 1 0]
```

[ ]:

[ ]: