CS156 (Introduction to AI), Spring 2022

Homework 2 submission

Roster Name: Austin Rivard

Preferred Name (if different):

Student ID: 015044445

Email address: austin.rivard@sjsu.edu

Any special notes or anything you would like to communicate to me about this homework submission goes in here.

References and sources

List all your references and sources here. This includes all sites/discussion boards/blogs/posts/etc. where you grabbed some code examples.

Solution

Load libraries and set random number generator seed

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from scipy.spatial import distance
from sklearn.metrics import accuracy_score
```

In [2]:
```
np.random.seed(42)
```

Code the solution

1) 2-D data

In [3]:
```
def knn(newObservation, referenceData: pd.DataFrame, k=3):
  # 1. create array of distances from each reference data point to new observati
  # scipy.spatial.distance.euclidean()

  distances = distance.cdist([newObservation], referenceData.values[:, 0:-1], me

  # 2. nearestNeighbors are the points with least smallest distance, found using
```

```
    nearestNeighbors = referenceData.assign(dist=distances[0]).nsmallest(k, 'dist'

    # 3. the new observation is classified as the most common classification of it
    # if there is a tie, take the first (but perhaps random would be better?)

    return nearestNeighbors['classification'].mode().iloc[0]
```

In [4]:

```
#generate 2d data
n = 100

X1 = np.random.normal(loc=-2, scale=2, size=n//2)
X2 = np.random.normal(loc=2, scale=2, size=n//2)
X = np.concatenate((X1, X2))
Y = np.random.normal(size=n)
```

In [5]:

```
num_labels = 2
labels = np.concatenate([[i] * (n // num_labels) for i in range(num_labels)])
print(labels)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```

In [6]:

```
df = pd.DataFrame({'X': X, 'Y': Y})
df.head()
```

Out[6]:

	X	Y
0	-1.006572	-1.415371
1	-2.276529	-0.420645
2	-0.704623	-0.342715
3	1.046060	-0.802277
4	-2.468307	-0.161286

In [7]:

```
X_train, X_test, Y_train, Y_test = train_test_split(df,labels, test_size=0.2, ra
```

In [8]:

```
ref_data = X_train.assign(classification = Y_train)
print(ref_data.head())
```

```
           X         Y  classification
43 -2.602207  0.184634               0
62 -0.212670  1.158596               1
3   1.046060 -0.802277               0
71  5.076073 -0.815810               1
45 -3.439688  0.781823               0
```

In [9]:

```
pred = knn(X_test.iloc[0], ref_data, k=3)
print(f'prediction = {pred}, actual = {Y_test[0]}')
```

```
prediction = 0, actual = 0
```

In [10]:
```
Y_pred = X_test.apply(knn, axis=1, referenceData=ref_data, k=3)
print(Y_pred.head())
```

```
26    0
86    1
2     0
55    1
75    1
dtype: int64
```

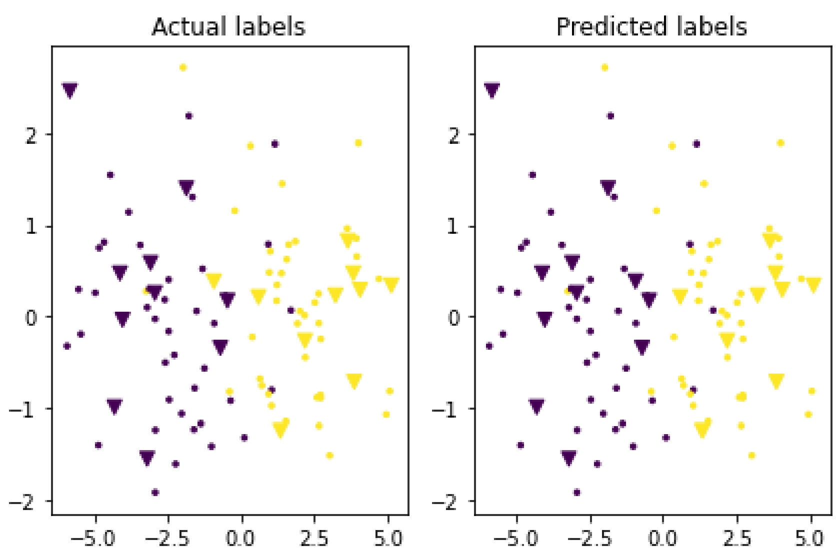
In [11]:
```
print(f'Accuracy of the predictions on the test data set is {accuracy_score(Y_te

plt.subplot(1, 2, 1)
plt.scatter(X_train.iloc[:,0],X_train.iloc[:,1], s=25, c=Y_train, marker=".")
plt.scatter(X_test.iloc[:,0],X_test.iloc[:,1], s=50, c=Y_test, marker="v")
plt.title("Actual labels")

plt.subplot(1, 2, 2)
plt.scatter(X_train.iloc[:,0],X_train.iloc[:,1], s=25, c=Y_train, marker=".")
plt.scatter(X_test.iloc[:,0],X_test.iloc[:,1], s=50, c=Y_pred, marker="v")
plt.title("Predicted labels")

plt.tight_layout()
plt.show()
```

```
Accuracy of the predictions on the test data set is 0.95
```



2) 3-D data

In [12]:
```
#generate 3d data
n = 1000
num_classes = 4

X = np.random.normal(loc=0, scale=3, size=n)

Y1 = np.random.normal(loc=-3, scale=1, size=n//num_classes)
Y2 = np.random.normal(loc=1, scale=2, size=n//num_classes)
Y3 = np.random.normal(loc=3, scale=1, size=n//num_classes)
Y4 = np.random.normal(loc=5, scale=3, size=n//num_classes)
```

```
Y = np.concatenate((Y1, Y2, Y3, Y4))

Z1 = np.random.normal(loc=-1, scale=1, size=n//num_classes)
Z2 = np.random.normal(loc=1, scale=1, size=n//num_classes)
Z3 = np.random.normal(loc=4, scale=1, size=n//num_classes)
Z4 = np.random.normal(loc=-3, scale=1, size=n//num_classes)
Z = np.concatenate((Z1, Z2, Z3, Z4))

labels = np.concatenate([[i] * (n // num_classes) for i in range(num_classes)])
```

In [13]:

```
df = pd.DataFrame({'X': X, 'Y': Y, 'Z': Z})
df.head()
```

Out[13]:

	X	Y	Z
0	1.073362	-2.874775	0.804348
1	1.682354	-3.429406	-1.190904
2	3.249154	-2.877702	-0.280242
3	3.161406	-2.456702	-2.293273
4	-4.133008	-2.951140	-1.956436

In [14]:

```
X_train, X_test, Y_train, Y_test = train_test_split(df,labels, test_size=0.2, ra
```

In [15]:

```
ref_data = X_train.assign(classification = Y_train)
ref_data.head()
```

Out[15]:

	X	Y	Z	classification
687	-1.077876	3.424061	5.388338	2
500	-1.568169	3.350630	4.386809	2
332	-0.198239	1.646335	1.555513	1
979	-3.310768	5.020400	-2.484372	3
817	-1.451658	-1.011587	-5.832156	3

In [16]:

```
Y_pred = X_test.apply(knn, axis=1, referenceData=ref_data, k=3)
print(Y_pred)
```

```
993    3
859    3
298    1
553    2
672    1
      ..
679    2
722    2
215    0
653    2
```

```
150    0
Length: 200, dtype: int64
```

In [17]:

```
print(f'Accuracy of the predictions on the test data set is {accuracy_score(Y_te

fig = plt.figure(figsize=(18, 15))

ax = plt.subplot(2, 3, 1, projection='3d')
ax.scatter(X_train.iloc[:,0],X_train.iloc[:,1], X_train.iloc[:,2], s=25, c=Y_tra
ax.scatter(X_test.iloc[:,0],X_test.iloc[:,1], X_test.iloc[:,2], s=50, c=Y_test,
plt.title("Actual labels")

ax = plt.subplot(2, 3, 2, projection='3d')
ax.scatter(X_train.iloc[:,0],X_train.iloc[:,1], X_train.iloc[:,2], s=25, c=Y_tra
ax.scatter(X_test.iloc[:,0],X_test.iloc[:,1], X_test.iloc[:,2], s=50, c=Y_pred,
plt.title("Predicted labels")

plt.tight_layout()
plt.show()
```

```
Accuracy of the predictions on the test data set is 0.915
```