assignment2

March 16, 2021

[34]:
```
import pandas as pd
import numpy as np
from collections import Counter
```

1 2D data

[64]:
```
n = 100 #data points
X1 = np.random.normal(loc=-2.0, scale=2.0, size=int(n/2))
X2 = np.random.normal(loc=2.0, scale=2.0, size=int(n/2))

Y1 = np.random.normal(loc=0, scale=1.0, size=int(n/2))
Y2 = np.random.normal(loc=0, scale=1.0, size=int(n/2))
X = np.concatenate((X1, X2), axis=0)
Y = np.concatenate((Y1, Y2), axis=0)
```

[65]:
```
l1 = [0] * int(n/2)
l2 = [1] * int(n/2)
labels = l1 + l2
```

[66]:
```
df = pd.DataFrame({'X':X, 'Y':Y, 'target':labels}, columns=['X', 'Y', 'target'])
df.head()
```

[66]:

	X	Y	target
0	-4.523820	0.073254	0
1	-5.265537	-0.495205	0
2	-2.221905	-2.076712	0
3	-5.471477	1.024543	0
4	-3.181710	0.850003	0

2 Eulidean Distance

[67]:
```
def distance(a, b):
    dim = len(a)
    distance = 0
    p = 2
    for d in range(dim):
        distance += abs(a[d] - b[d])**p
    distance = distance**(1/p)
    return distance
```

3 function for KNN

[68]:
```
def knn(newObservation, referenceData, k=3):
    X_train = referenceData.iloc[:, :-1]
    y_train = referenceData['target']
    X_test = newObservation

    y_hat_test = []

    for test_point in X_test.values:
        distances = []

        for train_point in X_train.values:
            dis = distance(test_point, train_point)
            distances.append(dis)

        df_dists = pd.DataFrame(data=distances, columns=['dist'], index=y_train.
 ↪index)
        df_nn = df_dists.sort_values(by=['dist'], axis=0)[:k]

        counter = Counter(y_train[df_nn.index])
        prediction = counter.most_common()[0][0]

        y_hat_test.append(prediction)
    return y_hat_test
```

4 Splitting the dataset and making predictions

[73]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df[['X', 'Y']],␣
 ↪df['target'], test_size = 0.2)
```

```
data = pd.concat([pd.DataFrame(X_train), pd.DataFrame(y_train)], axis=1)

y_hat_test = knn(X_test, data, k=3)
```

5 Evaluation

[82]:
```
from sklearn.metrics import accuracy_score

accuracy_score(y_test, y_hat_test)
```

[82]: 0.91

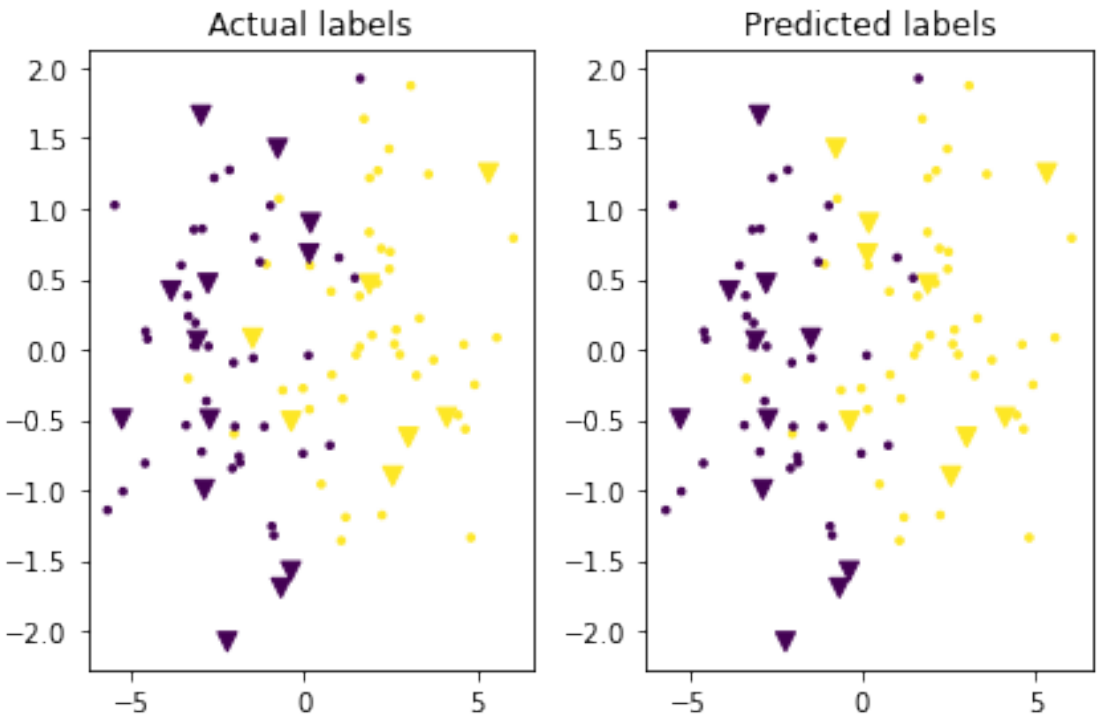
6 Scatterplot

[76]:
```
import matplotlib.pyplot as plt

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_hat_test, marker="v")
plt.title("Predicted labels")

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



7 3D data

[77]:
```
n = 1000 #data points

X1 = np.random.normal(loc=0, scale=3, size=int(n/4))
X2 = np.random.normal(loc=0, scale=3, size=int(n/4))
X3 = np.random.normal(loc=0, scale=3, size=int(n/4))
X4 = np.random.normal(loc=0, scale=3, size=int(n/4))

Y1 = np.random.normal(loc=-3, scale=1.0, size=int(n/4))
Y2 = np.random.normal(loc=1, scale=2.0, size=int(n/4))
Y3 = np.random.normal(loc=3, scale=1.0, size=int(n/4))
Y4 = np.random.normal(loc=5, scale=3, size=int(n/4))

Z1 = np.random.normal(loc=-1, scale=1.0, size=int(n/4))
Z2 = np.random.normal(loc=1, scale=1.0, size=int(n/4))
Z3 = np.random.normal(loc=4, scale=1.0, size=int(n/4))
Z4 = np.random.normal(loc=-3, scale=1.0, size=int(n/4))

X = np.concatenate((X1, X2, X3, X4), axis=0)
Y = np.concatenate((Y1, Y2, Y3, Y4), axis=0)
Z = np.concatenate((Z1, Z2, Z3, Z4), axis=0)
```

[78]:
```
l1 = [0] * int(n/4)
l2 = [1] * int(n/4)
l3 = [2] * int(n/4)
l4 = [3] * int(n/4)
labels = l1 + l2 + l3 + l4
```

[79]:
```
df = pd.DataFrame({'X':X, 'Y':Y, 'Z':Z,'target':labels}, columns=['X', 'Y',␣
↪'Z', 'target'])
df.head()
```

[79]:

	X	Y	Z	target
0	-4.355604	-2.854703	-1.342923	0
1	-4.440185	-4.921913	-1.362204	0
2	-3.844392	-3.074317	0.023136	0
3	-3.892565	-1.590245	-2.863647	0
4	1.428272	-2.019032	-3.261340	0

8 Splitting the dataset

[84]:
```
X_train, X_test, y_train, y_test = train_test_split(df[['X', 'Y', 'Z']],␣
↪df['target'], test_size = 0.2)
data = pd.concat([pd.DataFrame(X_train), pd.DataFrame(y_train)], axis=1)
```

9 Evaluation

[85]:
```
y_hat_test = knn(X_test, data, k=3)
accuracy_score(y_test, y_hat_test)
```

[85]:
```
0.905
```

[ ]:
```

```