Fruits Classification Using K-NN

In [1]:
```
%matplotlib notebook
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split

fruits = pd.read_table(r'E:\Protfolio\Fruits Classification Using K-NN/fruit_data_with_colors.txt')
```

In [2]:
```
fruits.head(10)
```

Out[2]:

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn	172	7.4	7.0	0.89

In [3]:
```
# create a mapping from fruit label value to fruit name to make results easier to interpret
lookup_fruit_name = dict(zip(fruits.fruit_label.unique(), fruits.fruit_name.unique()))
lookup_fruit_name
```

Out[3]:
```
{1: 'apple', 2: 'mandarin', 3: 'orange', 4: 'lemon'}
```

The file contains the mass, height, and width of a selection of oranges, lemons and apples. The heights were measured along the core of the fruit. The widths were the widest width perpendicular to the height.

Examining the data

plotting a scatter matrix

from matplotlib import cm from pandas.plotting import scatter_matrix X = fruits[['height', 'width', 'mass', 'color_score']] y = fruits['fruit_label'] X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)

cmap = cm.get_cmap('gnuplot') scatter = pd.scatter_matrix(X_train, c= y_train, marker = 'o', s=40, hist_kwds={'bins':15}, figsize=(9,9), cmap=cmap)

plotting a 3D scatter plot

from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure() ax = fig.add_subplot(111, projection = '3d') ax.scatter(X_train['width'], X_train['height'], X_train['color_score'], c = y_train, marker = 'o', s=100) ax.set_xlabel('width') ax.set_ylabel('height') ax.set_zlabel('color_score') plt.show()

Create train-test split

In [4]:
```
# For this example, we use the mass, width, and height features of each fruit instance
X = fruits[['mass', 'width', 'height','color_score']]
y = fruits['fruit_label']

# default is 75% / 25% train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
```

Create classifier object

In [5]:
```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors = 5)
```

Train the classifier (fit the estimator) using the training data

In [6]:
```
knn.fit(X_train, y_train)
```

Out[6]:
```
KNeighborsClassifier()
```

Estimate the accuracy of the classifier on future data, using the test data

In [7]:
```
knn.score(X_test, y_test)
```

Out[7]:
```
0.5333333333333333
```

Use the trained k-NN classifier model to classify new, previously unseen objects

In [8]:
```
# first example: a small fruit with mass 20g, width 4.3 cm, height 5.5 cm
fruit_prediction = knn.predict([[20, 4.3, 5.5,.70]])
lookup_fruit_name[fruit_prediction[0]]
```

Out[8]:
```
'mandarin'
```

In [9]:
```
# second example: a larger, elongated fruit with mass 100g, width 6.3 cm, height 8.5 cm
fruit_prediction = knn.predict([[100, 6.3, 8.5,.56]])
lookup_fruit_name[fruit_prediction[0]]
```

Out[9]:
```
'lemon'
```

In [ ]:
```

```