NAYANA DAVIS

TUNING CLUSTERS

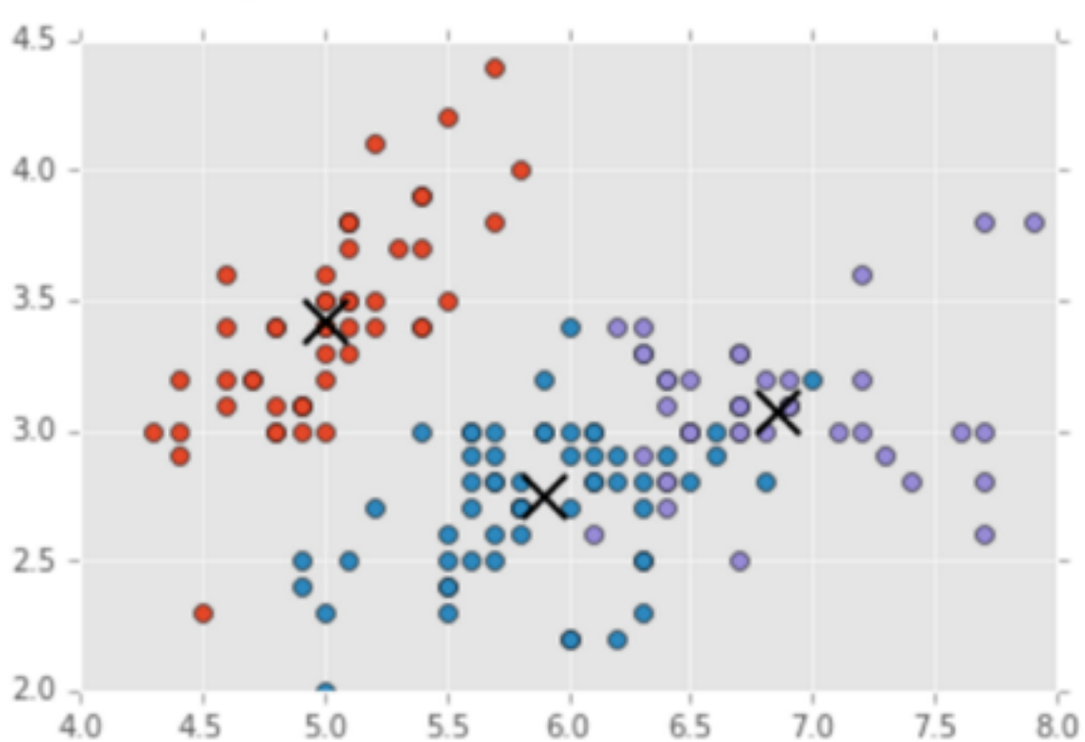
WE'VE PERFORMED CLUSTER ANALYSIS—NOW WHAT?

- The key to understanding your clustering analysis are the visual evaluation of your clusters, the measurement of their characteristics, and the computation of metrics that can measure how good your analysis is and how to interpret it

- So what constitutes a good cluster versus a bad cluster? Largely based on the accuracy and precision of the analysis, we can explore how well we've characterized our data.

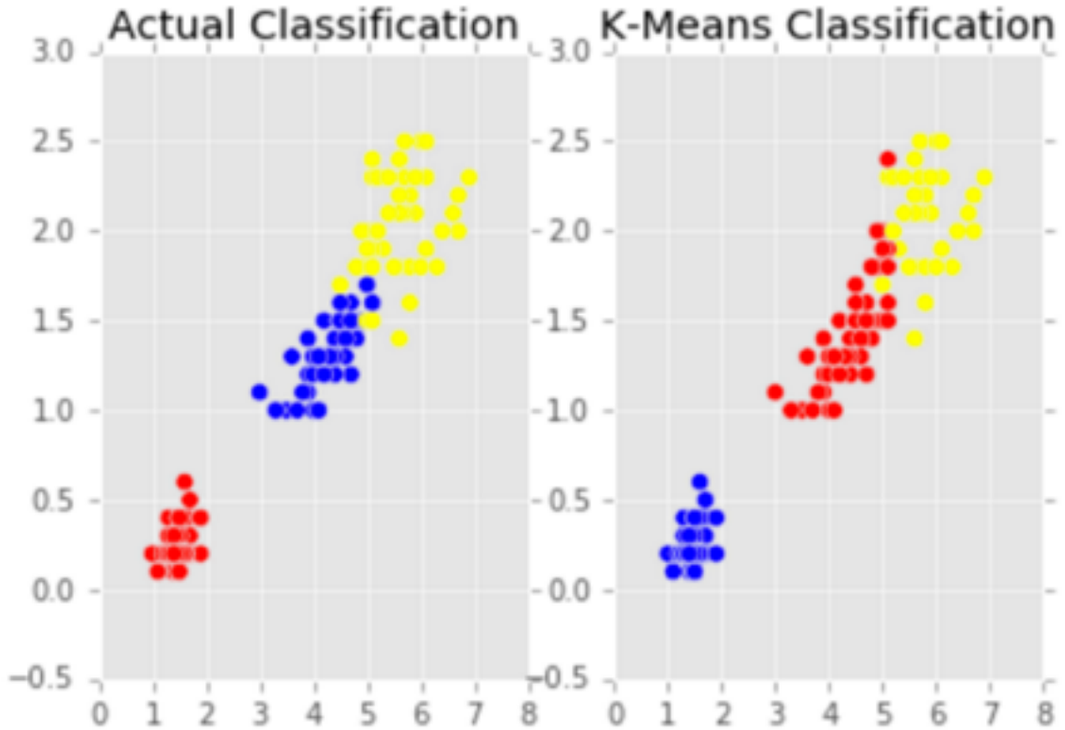
VISUALIZATION

- After we run the algorithm and calculate the centroids as we did in the previous lesson, we can plot the resulting clusters to see where the centroids are based and how the clusters are grouping.



VISUALIZATION

- We can also compare the classification of the original data to the classification given by our analysis - this gives us a first, primary look at how our analysis is performing in comparison to the original classifications.



ACCURACY SCORE

- In multilabel classification, this function computes subset accuracy: the set of labels predicted for a sample must exactly match the corresponding set of labels in y_true.

```
>>> import numpy as np
>>> from sklearn.metrics import accuracy_score
>>> y_pred = [0, 2, 1, 3]
>>> y_true = [0, 1, 2, 3]
>>> accuracy_score(y_true, y_pred)
0.5
>>> accuracy_score(y_true, y_pred, normalize=False)
2
```

SILHOUETTE SCORES

- The silhouette score, or silhouette coefficient, is the measure of how closely related a point is to members of its cluster rather than members of other clusters.

`sklearn.metrics.` **`silhouette_score`** (*X*, *labels*, *metric='euclidean'*, *sample_size=None*, *random_state=None*, ***kwds*) [source]

F-MEASURE

- The F Measure, sometimes known as the F1 Score, is used to measure the test's accuracy by measuring the number of correct positive results versus the positive results that should have been returned.

```
>>> from sklearn.metrics import classification_report
>>> y_true = [0, 1, 2, 2, 2]
>>> y_pred = [0, 0, 2, 2, 1]
>>> target_names = ['class 0', 'class 1', 'class 2']
>>> print(classification_report(y_true, y_pred, target_names=target_names))
             precision    recall  f1-score   support

    class 0       0.50      1.00      0.67         1
    class 1       0.00      0.00      0.00         1
    class 2       1.00      0.67      0.80         3

avg / total       0.70      0.60      0.61         5
```

CONFUSION MATRIX

- A simple quadrant graph with metrics that looks like this:

0	1	2	
0	28	22	Iris-setosa
47	3	0	Iris-versicolor
50	0	0	Iris-virginica

CONFUSION MATRIX

- When used in clustering, the matrix has the predicted class labels on the top x axis, and the actual class labels on the y axis. Each number within the matrix represents how many of the true classes were classified as each of the predicted classes.

```
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

```
>>> y_true = ["cat", "ant", "cat", "cat", "ant", "bird"]
>>> y_pred = ["ant", "ant", "cat", "cat", "ant", "cat"]
>>> confusion_matrix(y_true, y_pred, labels=["ant", "bird", "cat"])
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

POSSIBLE APPLICATIONS

- Clustering algorithms can be applied in many fields, for instance:
- Marketing: finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records;
- Biology: classification of plants and animals given their features;
- Libraries: book ordering;
- Insurance: identifying groups of motor insurance policy holders with a high average claim cost; identifying frauds;
- City-planning: identifying groups of houses according to their house type, value and geographical location;
- Earthquake studies: clustering observed earthquake epicenters to identify dangerous zones;
- WWW: document classification; clustering weblog data to discover groups of similar access patterns.

REQUIREMENTS CLUSTERING ALGORITHM SHOULD SATISFY

- scalability;
- dealing with different types of attributes;
- discovering clusters with arbitrary shape;
- minimal requirements for domain knowledge to determine input parameters;
- ability to deal with noise and outliers;
- insensitivity to order of input records;
- high dimensionality;
- interpretability and usability.

GUIDED PRACTICE

INDEPENDENT PRACTICE