Department of Computer Engineering
University of Peradeniya

Data Mining and Machine Learning
Lab 07 - Clustering

August 29, 2017

1 K-means Clustering

k-means clustering is a data mining/machine learning algorithm used to cluster observations into groups of related observations without any prior knowledge of those relationships.

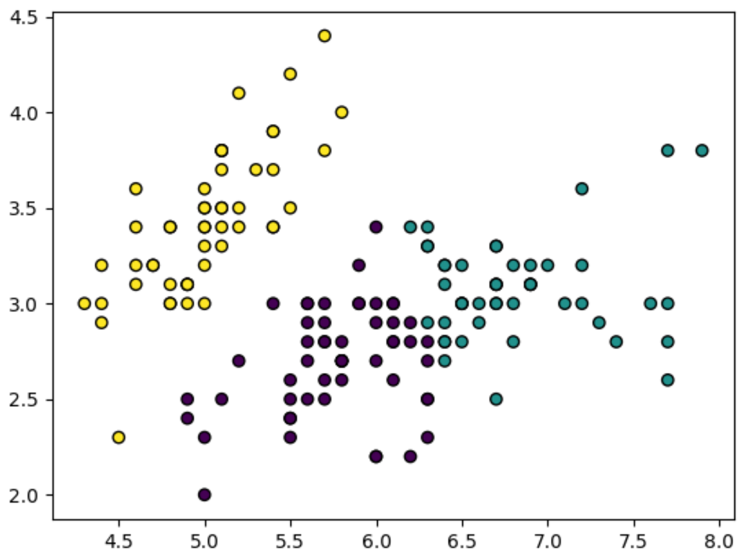

Figure 1: K means with k=3 for Iris data set

2 Hierarchical clustering

Hierarchical clustering is a general family of clustering algorithms that build nested clusters by merging or splitting them successively. This hierarchy of clusters is represented as a tree or dendrogram. The following criteria are used in hierarchical clustering.

- Ward : This minimizes the sum of squared differences within all clusters. It is a variance-minimizing approach.

- Complete : Complete linkage method minimizes the maximum distance between observations of pairs of clusters.
- Average : Average linkage minimizes the average of the distances between all observations of pairs of clusters.

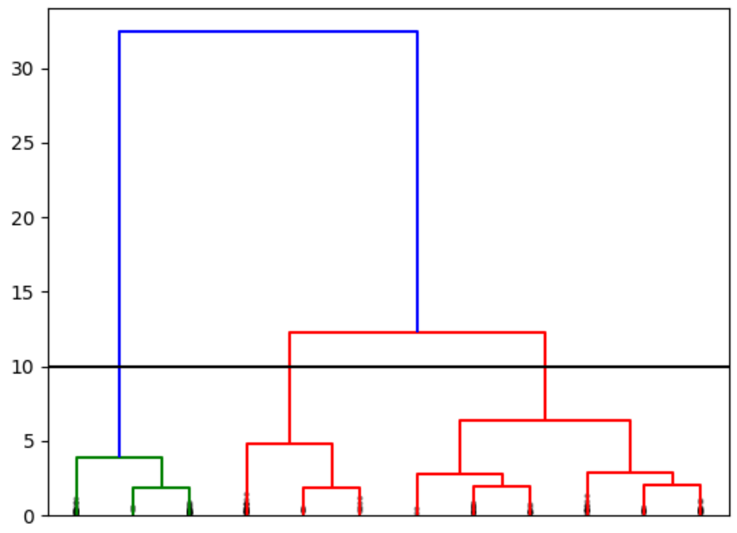

Figure 2: Dendrogram for Iris data set

3 Try Out

3.1 Sklearn.cluster.KMeans

Clustering of unlabeled data can be performed with the module *sklearn.cluster*. Each clustering algorithm comes in two variants: a class, that implements the fit method to learn the clusters on train data, and a function, that, given train data, returns an array of integer labels corresponding to the different clusters. For the class, the labels over the training data can be found in the labels-attribute.

1. First, import the necessary libraries that you require for the analysis.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn import datasets
```

2. Load iris dataset by loading the data set as given.

```
iris = datasets.load_iris()
X = iris.data
```

3. Let's consider first two features of the Iris data set. Estimate three clusters using k means clustering algorithm.

```
z=X[:,:2]
est=KMeans(n_clusters=3,random_state=0).fit(z)
labels = est.labels_
```

4. Using the estimated class labels, plot the data set.

```
x=X[:, 0]
y=X[:, 1]
plt.scatter(x,y,edgecolors='k', c=labels)
plt.show()
```

   - **x, y** : array-like, shape `(n, )` input data
   - **c** :color, sequence, or sequence of color, optional, default: 'b'
   - **edgecolors** : color or sequence of color, optional, default: None

5. Repeat the above steps for the whole Iris data set and observe the result.

3.2 Hierarchical clustering

One of the benefits of hierarchical clustering is that you don't need to already know the number of clusters k in your data in advance.

1. First, import the necessary libraries that you require for the analysis.

```
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
import numpy as np
```

2. Load iris dataset by loading the data set as given.

```
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data
```

3. Perform the Hierarchical Clustering and generate the linkage matrix.'**ward**' is one of the methods that can be used to calculate the distance between newly formed clusters. Also, there are some other common linkage methods like '**single**', '**complete**', '**average**'.

```
Z = linkage(X, 'ward')
```

4. A dendrogram is a visualization in form of a tree showing the order and distances of merged instances during the hierarchical clustering. The following code snippet used to plot dendrogram without truncation.

```
plt.figure(figsize=(25, 10))
dendrogram( Z, leaf_rotation=90.# rotates the x axis labels
                ,leaf_font_size=8.,# font size for the x axis labels
        )
```

5. This code snippet shows how to truncate the dendrogram into p clusters.

```
dendrogram( Z,truncate_mode='lastp' # show only the last p merged
clusters ,
                   p=12 # show only the last p merged clusters ,
                   show_leaf_counts=False  # otherwise numbers in brackets
are counts ,
                   leaf_rotation=90.,
                   leaf_font_size=12.,
                   show_contracted=True # to get a distribution impression
in truncated branches ,
                 )
```

6. Show the final plot with cut-off line which determines the number of clusters.

```
plt.axhline(y=10, c='k') # Selecting a Distance Cut−Off line
Determining the Number of Clusters
plt.show()
```

4 Lab Exercise

1. Load the breast-cancer dataset. Observe attributes and their values.
2. Briefly describe what is meant by the term ‘random-state’ in KMeans function.
3. Input three as the number of clusters and keep other fields with default values in `KMeans`. Apply KMeans using the first three features of the data set and comment on the result.
4. Perform hierarchical cluster analysis for the whole data set. Use **average** linkage method. Plot the dendrogram.
5. Visualize cluster assignments in KMeans and Hierarchical then comment on the results.

5 Submission

Submit a single .py file as [12|13]xxxlab07.py where xxx is your registration number. Add answers for questions 4 and 5 as comments in the same file.

6 Important

Make sure that you have the basic understanding of Clustering. This lab is really important for successive labs. If you do not understand any concepts, make sure you get some help from instructors.

7 Deadline

September 12, 23:59:59 GMT+5:30.