



Python Programming

Clustering

Feedback is greatly appreciated!

Clustering

Clustering

Clustering is the grouping of objects together so that objects belonging in the same group (cluster) are more similar to each other than those in other groups (clusters)



Clustering

Clustering is an unsupervised learning algorithm that will attempt to group similar clusters together in your data.

So what does a typical clustering problem look like?

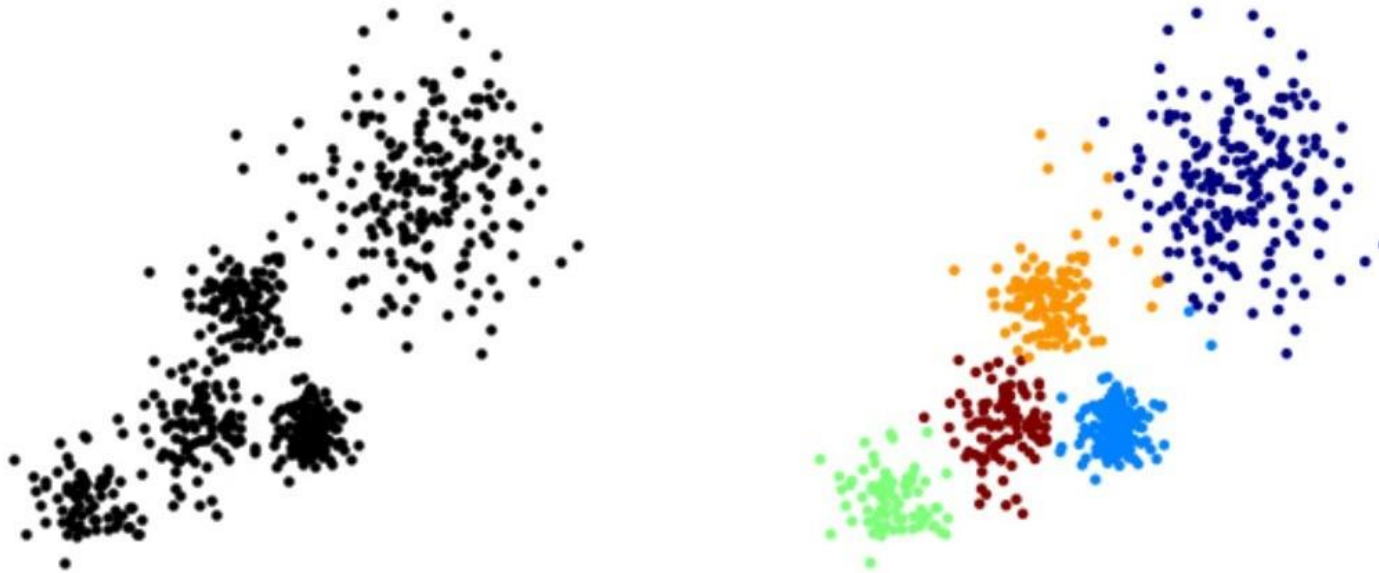
- Cluster Similar Documents
- Cluster Customers based on Features
- Market Segmentation
- Identify similar physical groups

Contd.

Method name	Parameters	Scalability	Use case	Geometry (metric used)
K-Means	number of clusters	Very large n_samples, medium n_clusters with MiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Distances between points
Spectral clustering	number of clusters	Medium n_samples, small n_clusters	Few clusters, even cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters	Large n_samples and n_clusters	Many clusters, possibly connectivity constraints	Distances between points
Agglomerative clustering	number of clusters, linkage type, distance	Large n_samples and n_clusters	Many clusters, possibly connectivity constraints, non Euclidean distances	Any pairwise distance
DBSCAN	neighborhood size	Very large n_samples, medium n_clusters	Non-flat geometry, uneven cluster sizes	Distances between nearest points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation	Mahalanobis distances to centers
Birch	branching factor, threshold, optional global clusterer.	Large n_clusters and n_samples	Large dataset, outlier removal, data reduction.	Euclidean distance between points

Clustering

The overall goal is to divide data into distinct groups such that observations within each group are similar



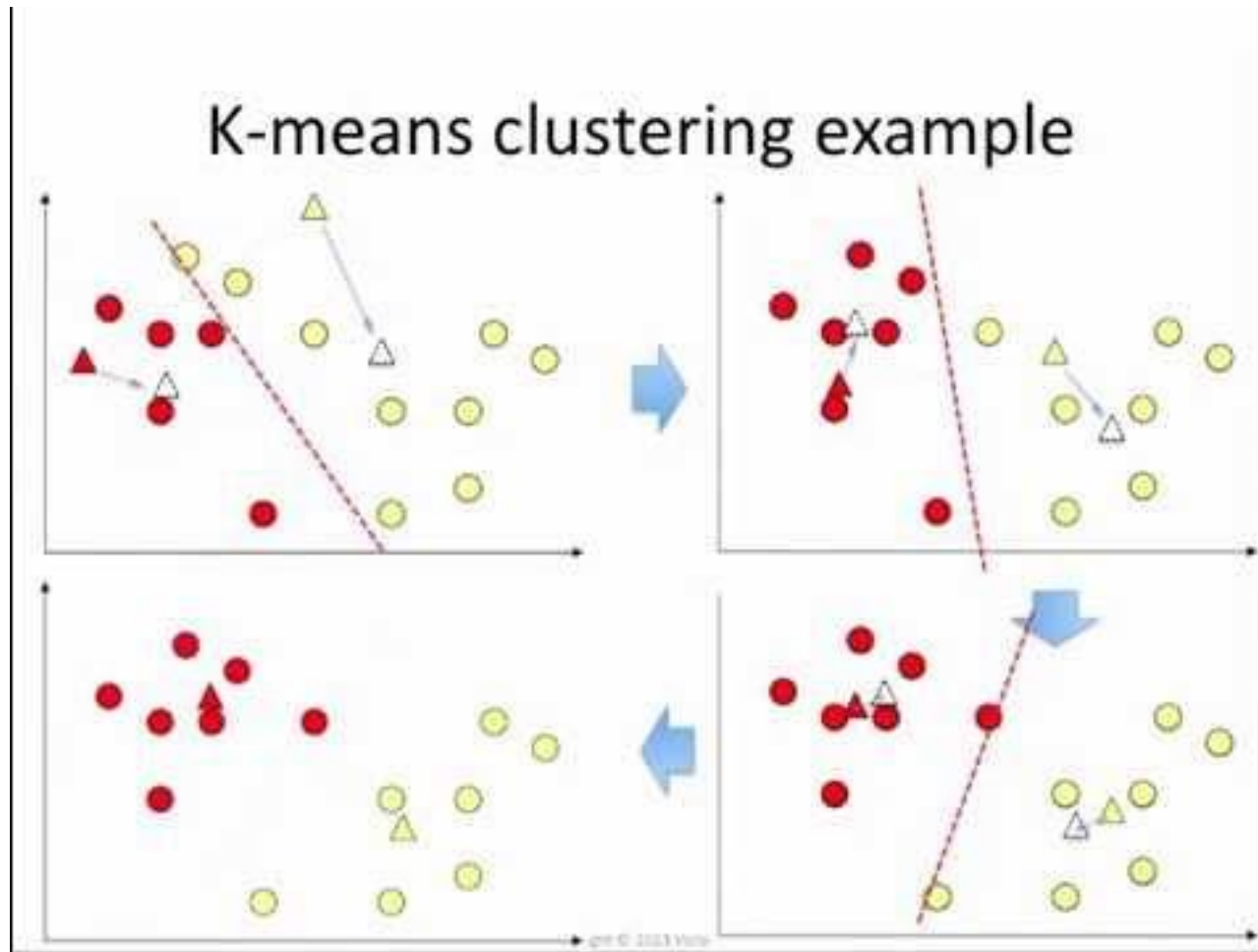
K Means Clustering

K Means Clustering

The K Means Algorithm

- Choose a number of Clusters “K”
- Randomly assign each point to a cluster
- Until clusters stop changing, repeat the following:
 - For each cluster, compute the cluster centroid by taking the mean vector of points in the cluster
 - Assign each data point to the cluster for which the centroid is the closest

K Means Clustering



K-Means

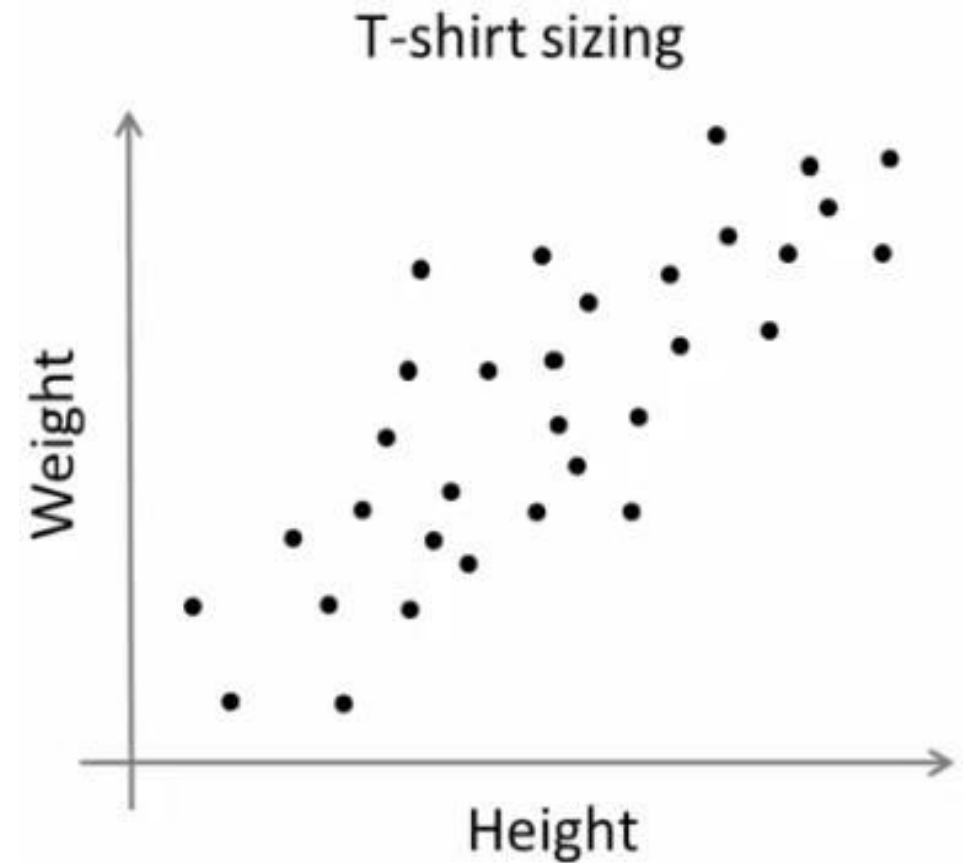
T-shirt size problem

Consider a company, which is going to release a new model of T-shirt to market.

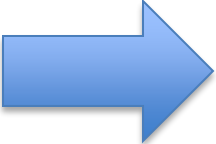
Obviously they will have to manufacture models in different sizes to satisfy people of all sizes.

So the company make a data of people's height and weight, and plot them on to a graph.

Company can't create t-shirts with all the sizes. Instead, they divide people to Small, Medium and Large, and manufacture only these 3 models which will fit into all the people.



As a first step in finding a sensible initial partition($k=2$)

Subject	A	B			
1	1.0	1.0			
2	1.5	2.0			
3	3.0	4.0			
4	5.0	7.0		Group 1	1
5	3.5	5.0		Group 2	4
6	4.5	5.0			
7	3.5	4.5			
				Individual	Mean Vector (centroid)
					(1.0, 1.0)
					(5.0, 7.0)

Steps to do clustering

Step	Cluster 1		Cluster 2	
	Individual	Mean Vector (centroid)	Individual	Mean Vector (centroid)
1	1	(1.0, 1.0)	4	(5.0, 7.0)
2	1, 2	(1.2, 1.5)	4	(5.0, 7.0)
3	1, 2, 3	(1.8, 2.3)	4	(5.0, 7.0)
4	1, 2, 3	(1.8, 2.3)	4, 5	(4.2, 6.0)
5	1, 2, 3	(1.8, 2.3)	4, 5, 6	(4.3, 5.7)
6	1, 2, 3	(1.8, 2.3)	4, 5, 6, 7	(4.1, 5.4)

- The remaining individuals are now examined in sequence and allocated to the cluster to which they are **closest**, in terms of **Euclidean distance** to the cluster **mean**.
- The mean vector is recalculated each time a new member is added.

Steps to do clustering

- Now the initial partition has changed, and the two clusters at this stage having the following characteristics

	Individual	Mean Vector (centroid)
Cluster 1	1, 2, 3	(1.8, 2.3)
Cluster 2	4, 5, 6, 7	(4.1, 5.4)

Steps to do clustering

Individual	Distance to mean (centroid) of Cluster 1	Distance to mean (centroid) of Cluster 2
1	1.5	5.4
2	0.4	4.3
3	2.1	1.8
4	5.7	1.8
5	3.2	0.7
6	3.8	0.6
7	2.8	1.1

- But we cannot yet be sure that each individual has been assigned to the right cluster.
- So, we compare each individual's distance to its own cluster mean and to that of the opposite cluster.

Steps to do clustering

	Individual	Mean Vector (centroid)
Cluster 1	1, 2	(1.3, 1.5)
Cluster 2	3, 4, 5, 6, 7	(3.9, 5.1)

- Only individual 3 is nearer to the mean of the opposite cluster (Cluster 2) than its own (Cluster 1).
- In other words, each individual's distance to its own cluster mean should be smaller than the distance to the other cluster's mean (which is not the case with individual 3).
- Thus, individual 3 is relocated to Cluster 2

Web tools for kmeans clustering

- <https://www.naftaliharris.com/blog/visualizing-k-means-clustering/>

Evaluation

- Silhouette score can be used to study the separation distance between the resulting clusters
- The silhouette plot displays a measure of **how close each point in one cluster is to points in the neighboring clusters** and thus provides a way to assess parameters like number of clusters visually
- This measure has a range **of $[-1, 1]$**
- Silhouette coefficients near **+1** indicate that the sample is **far away from the neighboring clusters**
- A value of **0** indicates that the sample is on or **very close to the decision boundary** between two neighboring clusters
- **Negative** values indicate that samples might have been assigned to the **wrong cluster**.

Dimensionality reduction

- dimension reduction is the process of reducing the number of random variables under consideration by obtaining a set of principal variables.
- It can be divided into [feature selection](#) and [feature extraction](#)

Principal Component Analysis(PCA)

- The main linear technique for dimensionality reduction
- principal component analysis, performs a linear mapping of the data to a lower-dimensional space in such a way that the variance of the data in the low-dimensional representation is maximized.

Feature scaling

- It refers to putting the values in the same range or same scale so that no variable is dominated by the other.
- It is mostly used in the **categorical data** where the categories are assigned simple integers such as 0,1,2...which might represent different categories.
- But the problem is that when we put this through a machine learning model, it may interpret it as the **weightage** or something else .
- For instance, it may use **0** as the **least preference** and **2** as a high preference

Various methods of feature scaling

- Standardization: It is also called Z-score normalization
- Min-Max Scaling: It is also referred to as Normalization The features are scaled between 0 and 1

Use Case: Clustering and Dimension Reduction on Iris data set



Iris Versicolor



Iris Setosa



Iris Virginica

Clustering

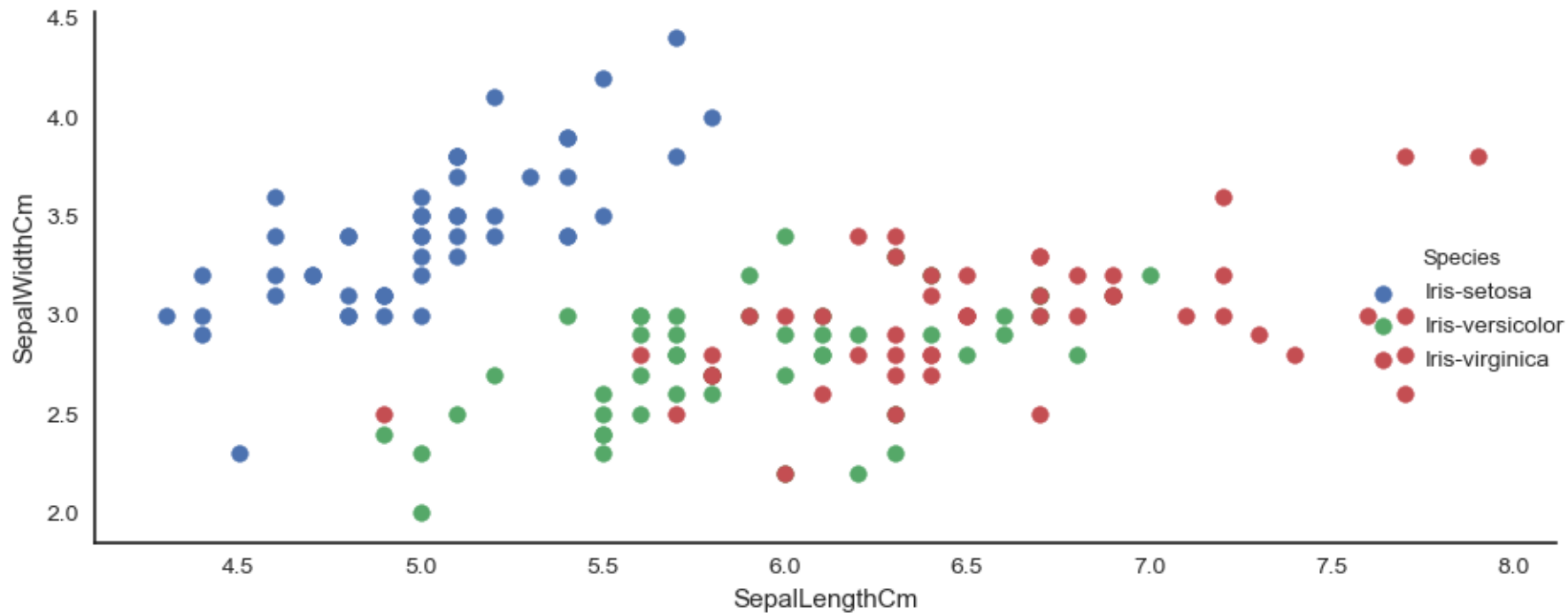
- Reading data:

```
dataset = pd.read_csv('Iris.csv')
x = dataset.iloc[:, [1, 2, 3, 4]]
y = dataset.iloc[:, -1]
# see how many samples we have of each species
dataset["Species"].value_counts()
```

Iris-versicolor	50
Iris-setosa	50
Iris-virginica	50

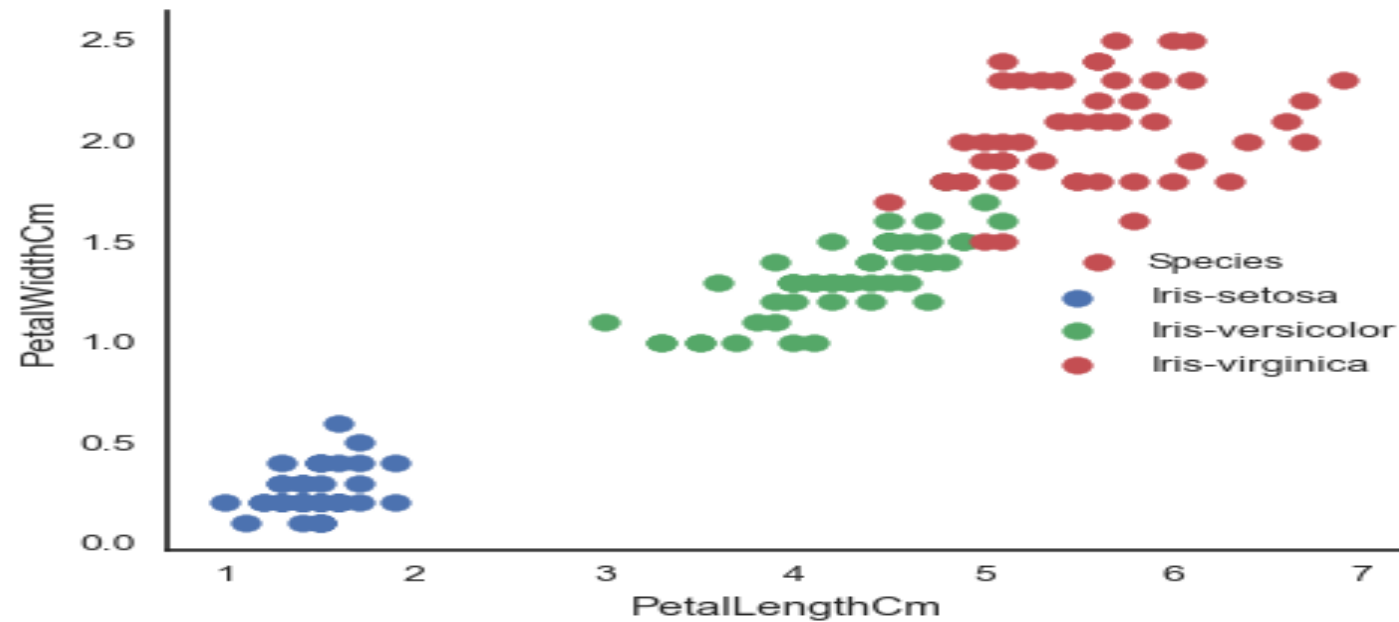
Visualize data

```
sns.FacetGrid(dataset, hue="Species", size=4).map  
(plt.scatter, "SepalLengthCm", SepalWidthCm).add_legend()
```



Visualize data

- *# do same for petals*
`sns.FacetGrid(dataset, hue="Species", size=4).map(plt.scatter,
"PetalLengthCm", "PetalWidthCm").add_legend()`



observations

- Note that the **species** are nearly linearly separable with **petal size**, but **sepal** sizes are more mixed.
- A clustering algorithm might have a hard time realizing that there were three separate species

Standardization

- The data is unbalanced, so should do feature scaling
- Otherwise larger feature will dominate the other

```
from sklearn import preprocessing
scaler = preprocessing.StandardScaler()
scaler.fit(X)
X_scaled_array = scaler.transform(X)
X_scaled = pd.DataFrame(X_scaled_array, columns = X.columns)
```

K-Means Clustering

```
from sklearn.cluster import Kmeans  
nclusters = 3 # this is the k in kmeans  
seed = 0  
km = KMeans(n_clusters=nclusters, random_state=seed)  
km.fit(X_scaled) # predict the cluster for each data point  
y_cluster_kmeans = km.predict(X_scaled)
```

Silhouette score

predict the cluster for each data point

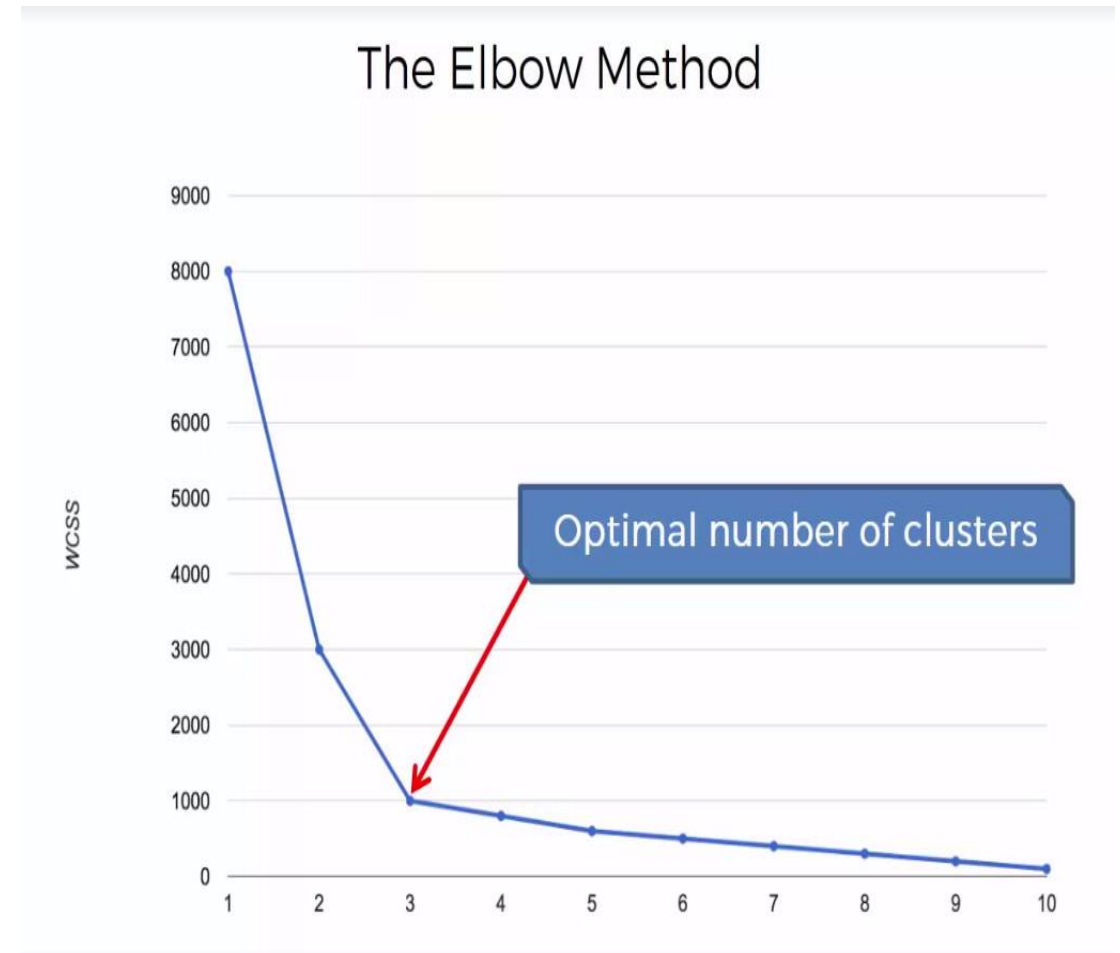
```
y_cluster_kmeans = km.predict(x)
```

```
from sklearn import metrics
```

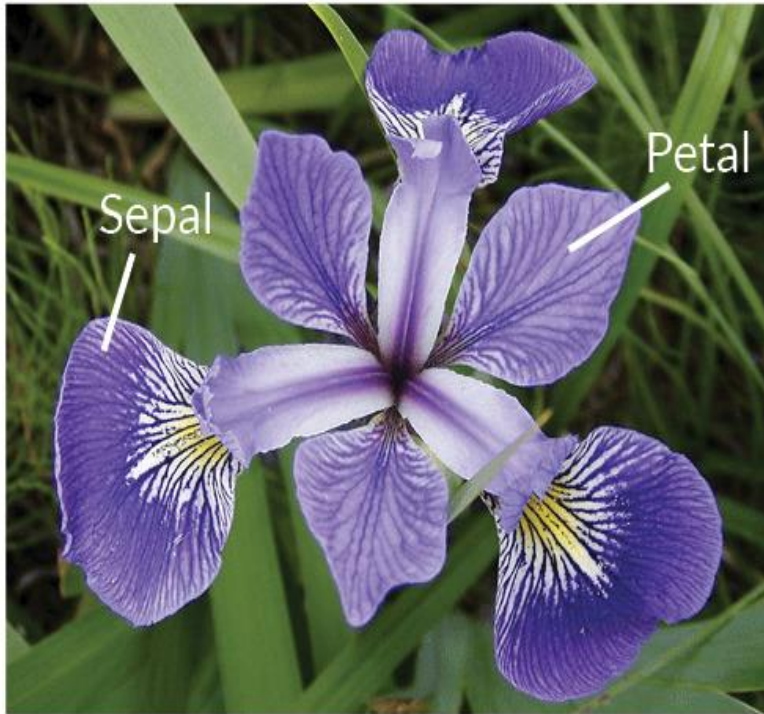
```
score = metrics.silhouette_score(x, y_cluster_kmeans)
```

Elbow plot: number of clusters

```
wcss = []  
##elbow method to know the number of clusters  
for i in range(1,11):  
    kmeans = KMeans(n_clusters=i,  
max_iter=300,random_state=0)  
    kmeans.fit(x)  
    wcss.append(kmeans.inertia_)  
  
plt.plot(range(1,11),wcss)  
plt.title('the elbow method')  
plt.xlabel('Number of Clusters')  
plt.ylabel('Wcss')  
plt.show()
```



Dimensionality reduction on IRIS data



Iris Versicolor



Iris Setosa



Iris Virginica

Read data and Standardize

- PCA is effected by scale
- you need to scale the features in the data before applying PCA
- You can transform the data onto unit scale (mean = 0 and variance = 1)

```
dataset = pd.read_csv('Iris.csv')
x = dataset.iloc[:,[1,2,3,4]]
y = dataset.iloc[:,-1]
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
# Fit on training set only.
scaler.fit(x)
# Apply transform to both the training set and the test set.
x_scaler = scaler.transform(x)
```

Apply PCA

- Notice the code below has **2** for the number of components parameter.
- It means that scikit-learn choose 2 of principal components

```
from sklearn.decomposition import PCA # Make an instance of the Model  
pca = PCA(2)  
X_pca = pca.fit_transform(x_scaler)
```

References

<http://www-bcf.usc.edu/~gareth/ISL/ISLR%20Sixth%20Printing.pdf>
<https://web.stanford.edu/~schmit/cme193/lec/lec5.pdf>
<http://machinelearningmastery.com/machine-learning-in-python-step-by-step/>
http://scikit-learn.org/stable/auto_examples/linear_model/plot_ols.html
<https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/>
<http://scikit-learn.org/stable/modules/clustering.html>
http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html
<http://www.kdnuggets.com/2015/11/seven-steps-machine-learning-python.html/2>
<http://beancoder.com/linear-regression-stock-prediction/>
https://github.com/tarlen5/coursera_ml/blob/master/unit6/ex2_sklearn
<https://www.kaggle.com/bburns/iris-exploration-pca-k-means-and-gmm-clustering>
<https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60>