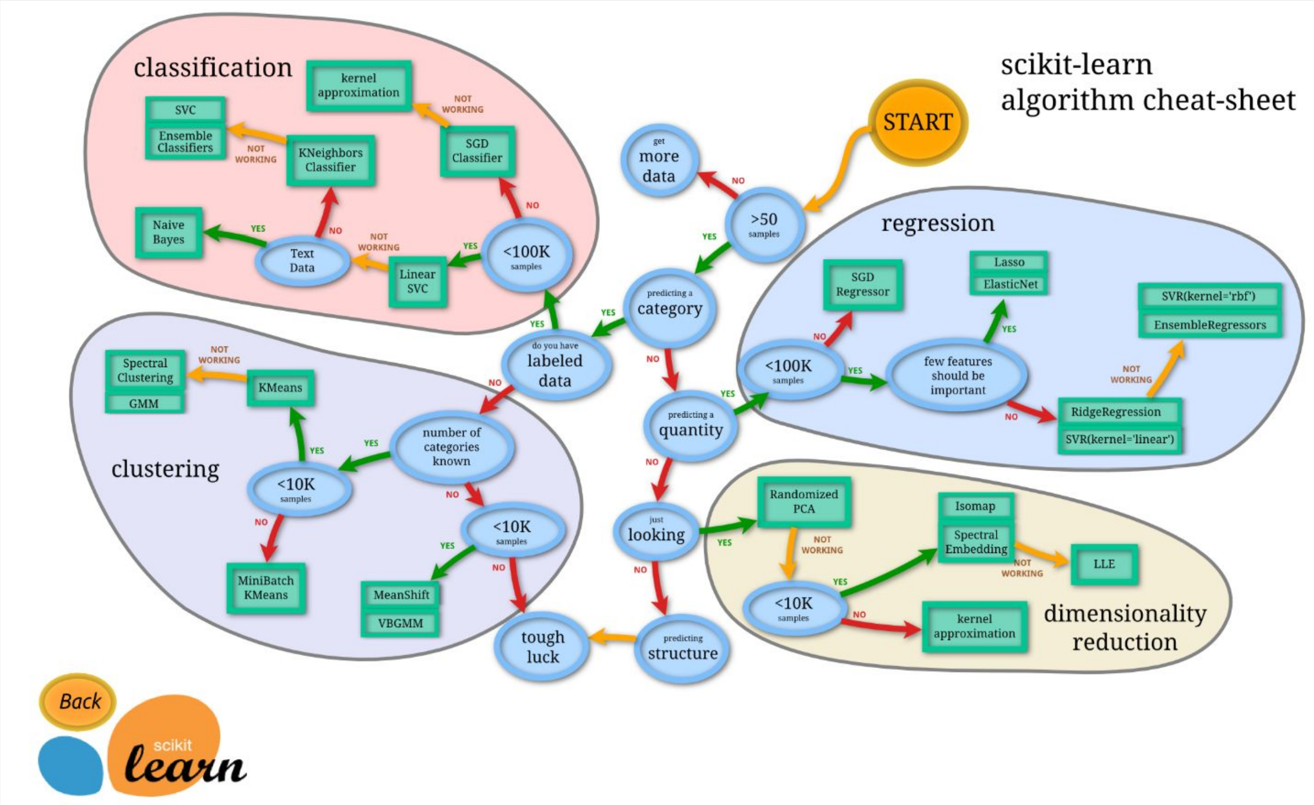
K-MEANS CLUSTERING

- What it is?
- python implementation
- Use cases

SCIKIT-LEARN – CHEAT SHEET



WHAT IT IS

- K-means clustering is a simple unsupervised learning algorithm that is used to group/ cluster n objects based on certain attributes into k partitions, where k < n.

- Follows a simple procedure of classifying a given data set into a number of clusters, defined by the letter "k," which is fixed beforehand.

- Groups data using a "top-down" approach since it starts with a predefined number of clusters and assigns all observations to each of them.

- no overlaps in the groups; each observation is assigned only to a single group.

- Approach is computationally faster and can handle greater numbers of observations than agglomerative hierarchical clustering

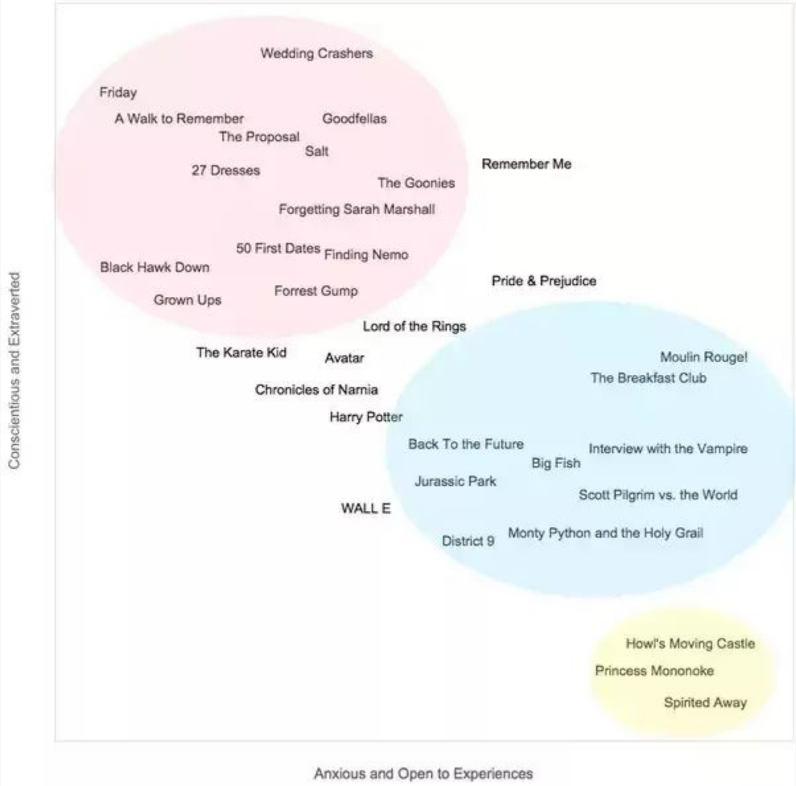
- K-means clustering has uses in search engines, market segmentation, statistics and even astronomy.

EXAMPLE

Different movie genres appeal to people of different personalities. To confirm this, construct a plot of movie titles along personality dimensions:

There appears to be 3 clusters:

- Red: Conscientious extraverts who like action and romance genres
- Blue: Anxious and open people who like advant-garde and fantasy genres
- Yellow: Introverts with social anxieties who like Japanese animations (otaku culture)
- Movies in the center seem to be general household favorites.



K-MEANS ALGORITHM PROPERTIES

- There are always K clusters.
- There is always at least **one** item in each cluster.
- The clusters **are non-hierarchical** and they **do not overlap**.
- Every member of a cluster is closer to its cluster than any other cluster

THE K-MEANS ALGORITHM PROCESS

1. The dataset is partitioned into K clusters and the data points are randomly assigned to the clusters.
2. For each data point:

- Calculate the distance from the data point to each cluster.
- If the data point is closest to its own cluster, leave it where it is. If the data point is not closest to its own cluster, move it into the closest cluster.
- Repeat the above step until a complete pass through all the data points results in no data point moving from one cluster to another. At this point the clusters are stable and the clustering process ends.
- The choice of initial partition can greatly affect the final clusters that result, in terms of inter-cluster and intra-cluster distances

DISTANCE CALCULATION

- The Euclidean distance function measures the 'as-the-crow-flies' distance. The formula for this distance between a point X (X1, X2, etc.) and a point Y (Y1, Y2, etc.) is:

$$d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- Let observations $u = (u_1, u_2, \ldots, u_q)$ and $v = (v_1, v_2, \ldots, v_q)$ each comprise measurements of q variables.
- The Euclidean distance between observations u and v is
- $d_{u,v} = \sqrt{(u_1 - v_1)^2 + (u_2 - v_2)^2 + \cdots + (u_q - v_q)^2}$

HOW IT WORKS

Data Point		
A1	2	10
A2	2	5
A3	8	4
A4	5	8
A5	7	5
A6	6	4
A7	1	2
A8	4	9

Group the data points into 3 clusters

Randomly decide on starting center points of the 3 clusters

HOW IT WORKS

Initial centroid assigned (randomly)

Distance calculation using rectilinear method and assignment of cluster based on min distance

New centroid for each cluster calculated (averaging of data points)

Iterations -- 1

			Cluster C1		Cluster C2		Cluster C3		
			2	10	5	8	1	2	
Data Point			Distance from c1		Distance from c2		Distance from c1		Cluster Assignment
A1	2	10	0		5		9		c1
A2	2	5	5		6		4		c3
A3	8	4	12		7		9		c2
A4	5	8	5		0		10		c2
A5	7	5	10		5		9		c2
A6	6	4	10		5		7		c2
A7	1	2	9		10		0		c3
A8	4	9	3		2		10		c2

Cluster C1		Cluster C2		Cluster C3	
2	10	8	4	2	5
		5	8	1	2
		7	5		
		6	4		
		4	9		
2	10	6	6	1.5	3.5

HOW IT WORKS

Centroid adjusted from previous iteration

Distance calculation using rectilinear method and assignment of cluster based on min distance

New centroid for each cluster calculated (averaging of data points)

Iterations -- 2									
			Cluster C1		Cluster C2		Cluster C3		
			2	10	6	6	1.5	3.5	
Data Point			Distance from c1		Distance from c2		Distance from c1		Cluster Assignment
A1	2	10	0		8		7		c1
A2	2	5	5		5		2		c3
A3	8	4	12		4		7		c2
A4	5	8	5		3		8		c2
A5	7	5	10		2		7		c2
A6	6	4	10		2		5		c2
A7	1	2	9		9		2		c3
A8	4	9	3		5		8		c1

Cluster C1		Cluster C2		Cluster C3	
2	10	8	4	2	5
4	9	5	8	1	2
		7	5		
		6	4		
3	9.5	6.5	5.25	1.5	3.5

HOW IT WORKS

Centroid adjusted from previous iteration

Distance calculation using rectilinear method and assignment of cluster based on min distance

New centroid for each cluster calculated (averaging of data points)

Iterations -- 3									
			Cluster C1		Cluster C2		Cluster C3		
			3	9.5	6.5	5.25	1.5	3.5	
Data Point			Distance from c1		Distance from c2		Distance from c1		Cluster Assignment
A1	2	10	0		8		7		c1
A2	2	5	5		5		2		c3
A3	8	4	12		4		7		c2
A4	5	8	5		3		8		c2
A5	7	5	10		2		7		c2
A6	6	4	10		2		5		c2
A7	1	2	9		9		2		c3
A8	4	9	3		5		8		c1

Cluster Assignment
c1
c3
c2
c2
c2
c2
c3
c1

Cluster C1		Cluster C2		Cluster C3	
2	10	8	4	2	5
4	9	5	8	1	2
		7	5		
		6	4		
3	9.5	6.5	5.25	1.5	3.5

No change in the cluster assignment, Hence STOP

PRACTICAL APPLICATIONS

- Customer Segmentation:

Pricing Segmentation	
Loyalty	
Spend Behaviour	
Branch Geo	
Customer Need	needs, channel of preferences, service expectations.
Category	Who are these customers? Why are they behaving the way to?
Customer Service	Customer Value in last 6/12/18/24 months Customer Type – Individuals and Small Businesses Product type (e.g. Gas, Electricity etc) Length of Relationship Overall consumption Number of complains
News Article Clustering	

DISADVANTAGES

- Fixed number of clusters can make it difficult to predict what K should be.
- Does not work well with non-globular clusters.
- Different initial partitions can result in different final clusters.
- Even outliers become part of some cluster!

A Globular clusters are very tightly bound, which gives them their spherical shapes and relatively high stellar densities toward their centers.

USE CASES

Behavioral segmentation	Inventory categorization	Sorting sensor measurements	Detecting bots or anomalies
• Segment by purchase history • Segment by activities on application, website, or platform • Define personas based on interests • Create profiles based on activity monitoring	• Group inventory by sales activity • Group inventory by manufacturing metrics	• Detect activity types in motion sensors • Group images • Separate audio • Identify groups in health monitoring	• Separate valid activity groups from bots • Group valid activity to clean up outlier detection

KMEANS – SCITKIT LEARN

- KMeans(n_clusters=8, init='k-means++', n_init=10, max_iter=300, tol=0.0001, precompute_distances='auto', verbose=0, random_state=None, copy_x=True, n_jobs=None, algorithm='auto')[source]

- **Attributes:**
  - cluster_centers_ : array, [n_clusters, n_features] Coordinates of cluster centers. If the algorithm stops before fully converging
  - labels_ : Labels of each point
  - inertia_ : float, Sum of squared distances of samples to their closest cluster center.
  - n_iter_ : int, Number of iterations run.

SPECIFYING DIFFERENT DISTANCE CALCULATION METHOD

- SCITKIT learn implementation only supports 'Euclidean' distance measure