Clustering

Let’s learn something!

Python and Spark

- We’ve seen how to deal with labeled data, but what about unlabeled data?
- Often you’ll find yourself trying to create groups from data, instead of trying to predict classes or values.

Python and Spark

- This sort of problem is known as clustering, you can think of it as an attempt to create labels.
- You input some unlabeled data, and the **unsupervised learning** algorithm returns back possible clusters of the data.

Python and Spark

- This means you have data that only contains features and you want to see if there are patterns in the data that would allow you to create groups or clusters.

Python and Spark

- This is a key distinction from our previous **supervised learning** tasks, where we had historical labeled data.
- Now we will have unlabeled data, and attempt to “discover” possible labels, through clustering.

Python and Spark

- By the nature of this problem, it can be difficult to evaluate the groups or clusters for “correctness”.
- A large part of being able to interpret the clusters assigned comes down to domain knowledge!

Python and Spark

- Maybe you have some customer data, and then cluster them into distinct groups.
- It will be up to you to decide what the groups actually represent.
- Sometimes this is easy, sometimes it’s really hard!

Python and Spark

- For example, you could cluster tumors into two groups, hoping to separate between benign and malignant.
- But there is no guarantee that the clusters will fall along those lines, it will just split into the two most separable groups.

Python and Spark

- Also depending on the clustering algorithm, it may be up to you to decide beforehand how many clusters you expect to create!

Python and Spark

- A lot of clustering problems have no 100% correct approach or answer, that is the nature of unsupervised learning!
- Let’s continue by discussing K-means clustering.

K Means Clustering

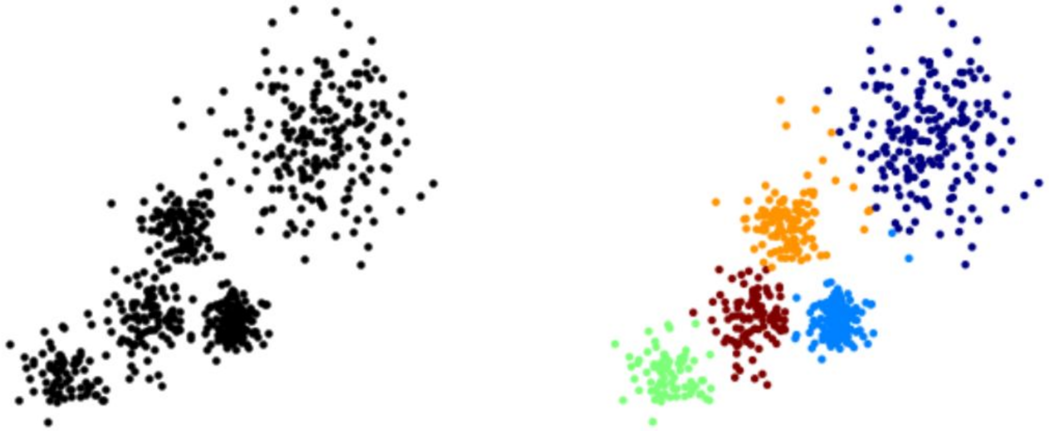
K Means Clustering is an unsupervised learning algorithm that will attempt to group similar clusters together in your data.

So what does a typical clustering problem look like?

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

K Means Clustering

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

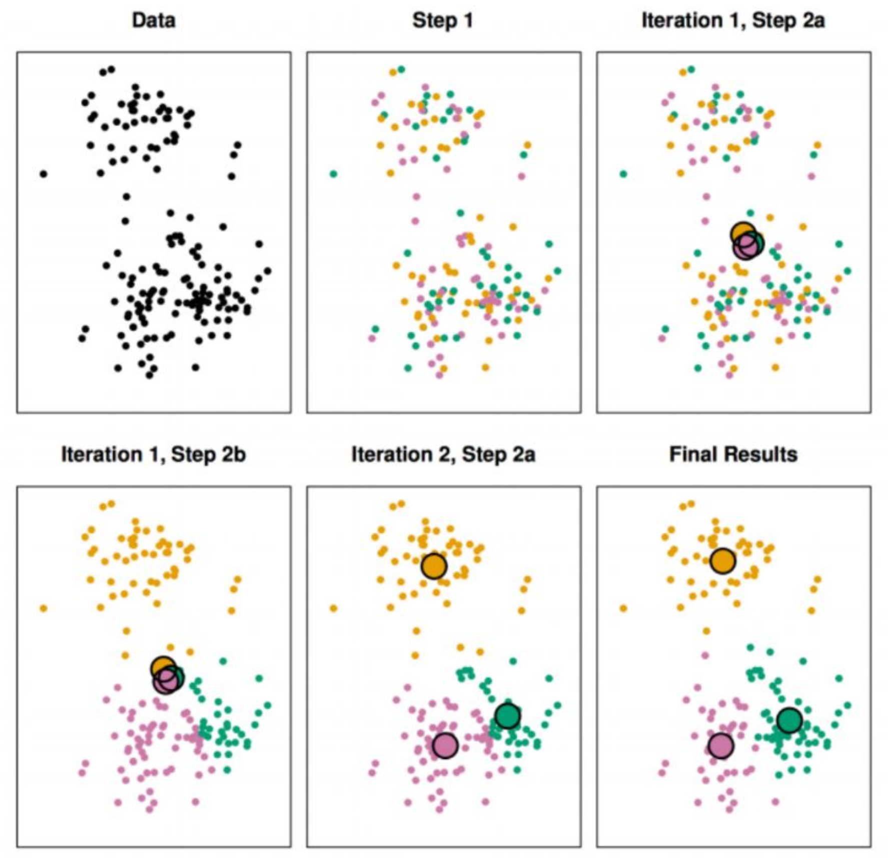


K Means Clustering

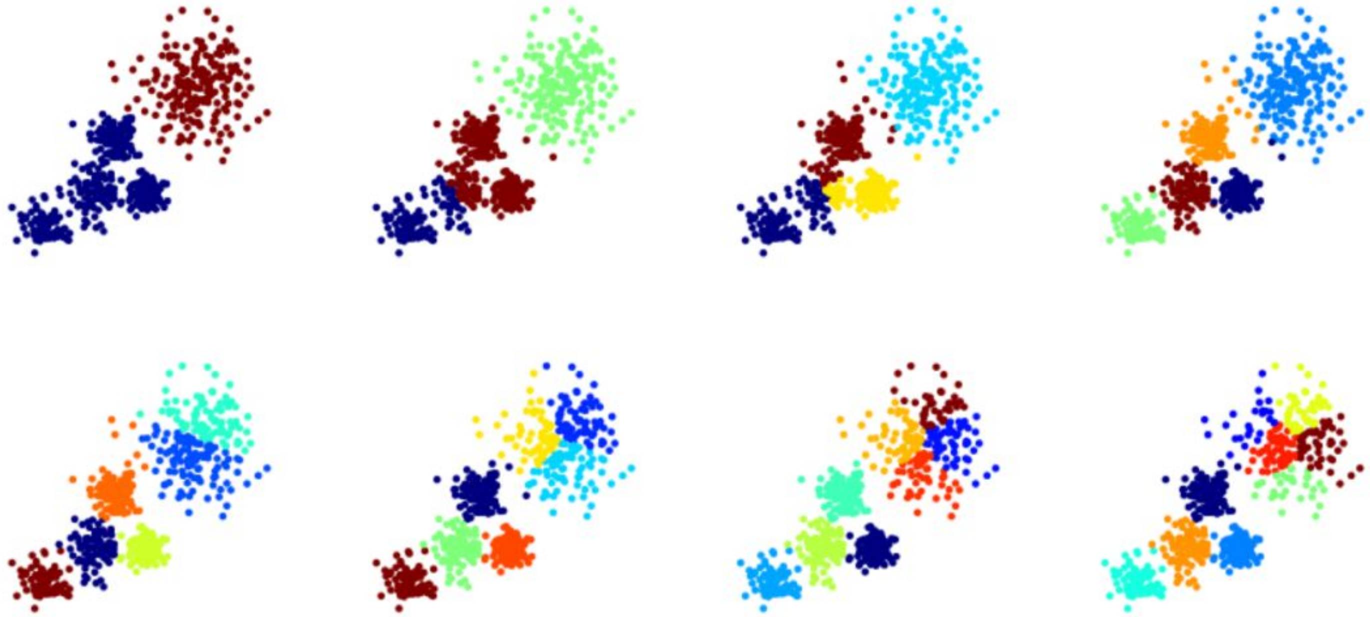
The K Means Algorithm

- 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



Choosing a K Value



Choosing a K Value

- There is no easy answer for choosing a “best” K value
- One way is the elbow method

First of all, compute the sum of squared error (SSE) for some values of k (for example 2, 4, 6, 8, etc.).

The SSE is defined as the sum of the squared distance between each member of the cluster and its centroid.

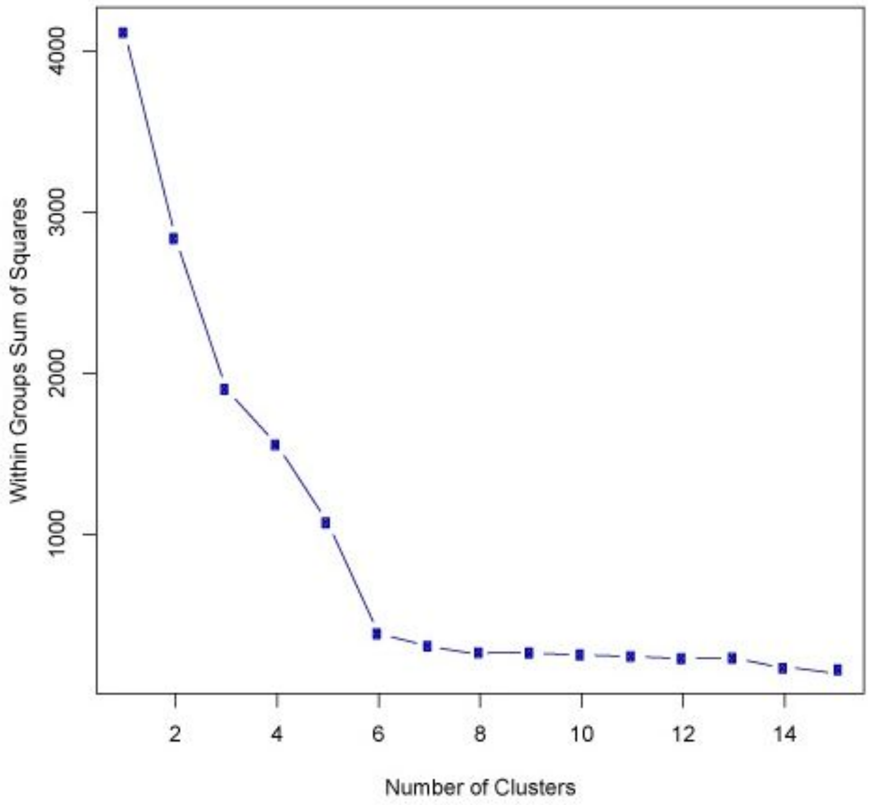
Choosing a K Value

If you plot k against the SSE, you will see that *the error decreases as k gets larger*; this is because when the number of clusters increases, they should be smaller, so distortion is also smaller.

The idea of the elbow method is to choose the k at which the SSE decreases abruptly.

This produces an "elbow effect" in the graph, as you can see in the following picture:

Choosing a K Value



Choosing a K Value

- Pyspark by itself doesn’t support a plotting mechanism, but you could use collect() and then plot the results with matplotlib or other visualization libraries.

Choosing a K Value

- But don’t take this as a strict rule when choosing a K value!
- A lot of depends more on the context of the exact situation (domain knowledge)
- We’ll try our best to get a feel for this with the examples and consulting projects!