# Clustering

Let’s learn something!

PIERIAN DATA

- 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!

- 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.

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

- 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.

# Reading Assignment

Chapter 10 of

**Introduction to Statistical Learning**

By Gareth James, et al.

# K Means Clustering

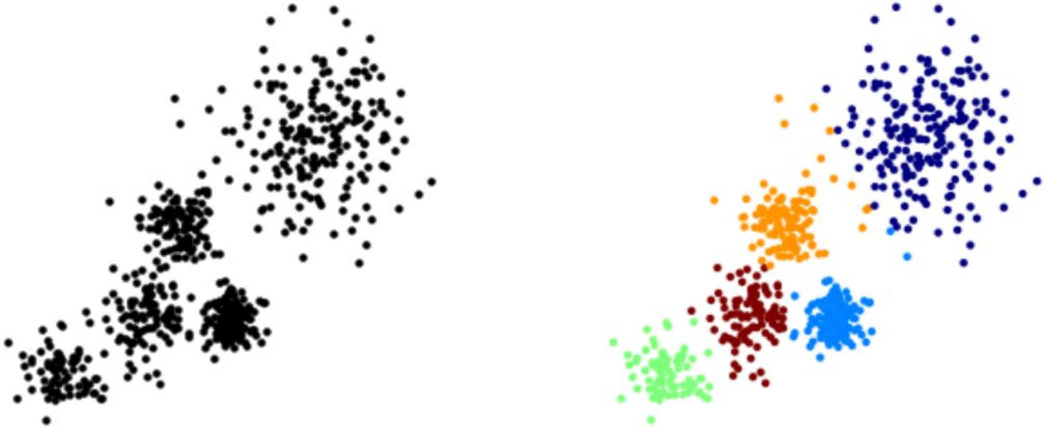
K Means Clustering is an unsupervised learning algorithm that will attempt to group similar clusters together in your data.

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 similar

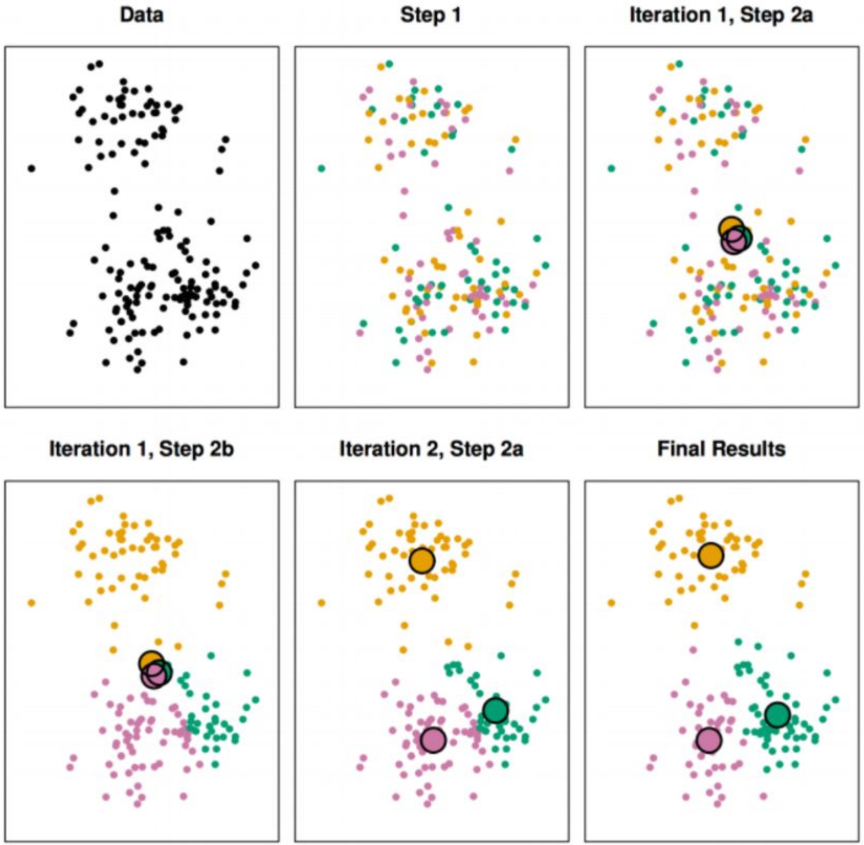


# K Means Clustering

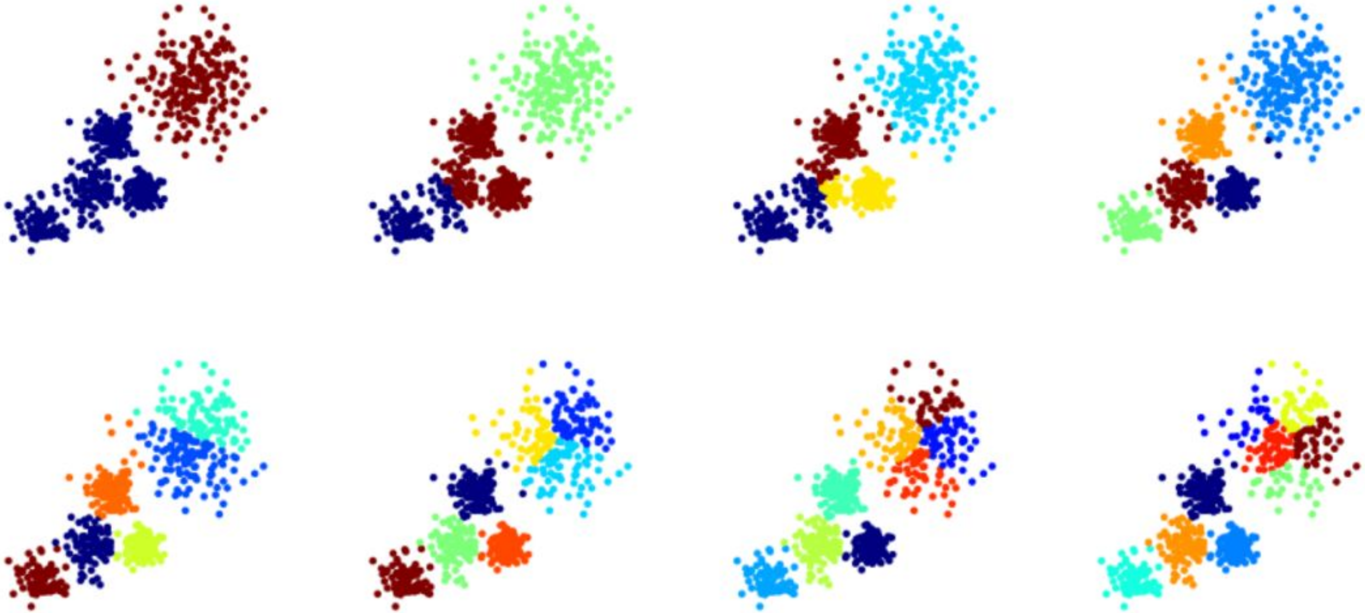
## The K Means Algorithm

- 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.

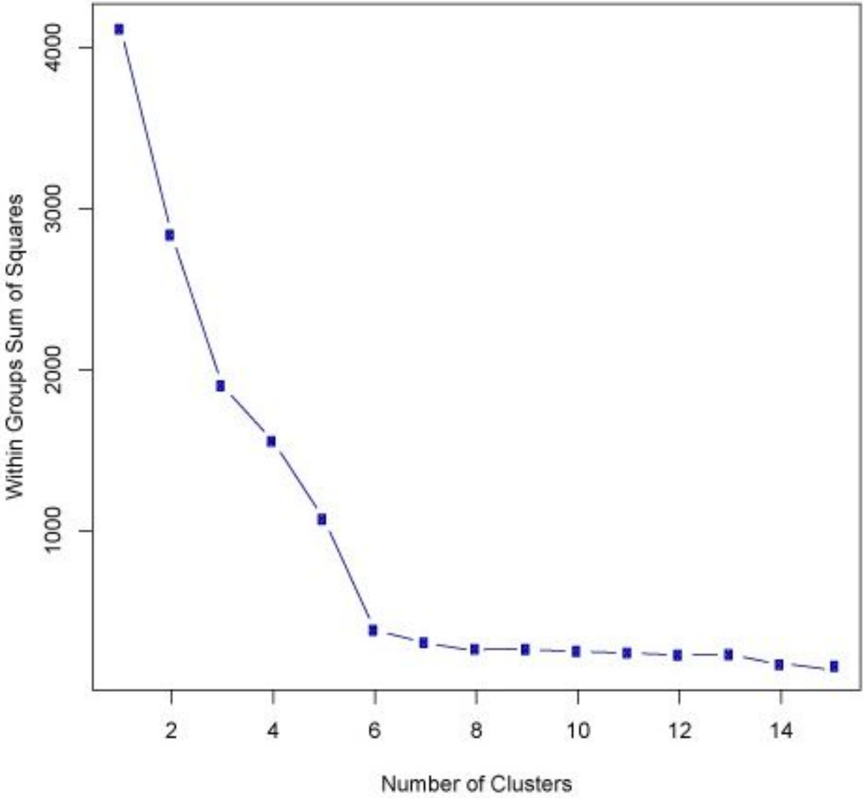
# Choosing a K Value

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!

# K-Means Clustering Documentation Example

Let’s learn something!

# Python and Spark

- Let’s work through the documentation example for clustering.
- Pay close attention to how we don’t need the label column (which makes sense given clustering)

# Python and Spark

- The documentation’s example is a bit peculiar in its choice of data set, but we’ll explain it along the way.
- Hopefully our own custom code along will clarify things further!
- Let’s get started!

# K-Means Clustering Code Along

# Python and Spark

- We'll work through a real data set containing some data on three distinct seed types.
- Notebook: **Clustering Code Along.ipynb**

# Python and Spark

- For certain Machine Learning algorithms, it is a good idea to scale your data.
- Drops in model performance can occur with highly dimensional data, so we’ll practice scaling features using PySpark!

# Python and Spark

- Remember, there won’t be any confusion matrix or classification test results.
- This is **unsupervised learning**!
- Meaning we don’t have the original labels to actually perform some sort of test against!

# Python and Spark

- This is a common point of confusion for beginners, you can't easily check to see how well your clustering algorithm performed, this is the difficulty of all unsupervised tasks!
- Let's get started!

# K-Means Clustering Consulting Project

# Python and Spark

- You’re becoming world famous due to your machine learning skills!
- A technology start-up in California needs your help!

# Python and Spark

- It’s time for you to go to San Francisco to help out a tech startup!

# Python and Spark

- They’ve been recently hacked and need your help finding out about the hackers!

# Python and Spark

- Luckily their forensic engineers have grabbed valuable data about the hacks, including information like session time,locations, wpm typing speed, etc.

# Python and Spark

- The forensic engineer relates to you what she has been able to figure out so far, she has been able to grab meta-data of each session that the hackers used to connect to their servers.
- These are the features of the data…

# Python and Spark

- 'Session_Connection_Time': How long the session lasted in minutes
- 'Bytes Transferred': Number of MB transferred during session
- 'Kali_Trace_Used': Indicates if the hacker was using Kali Linux
- 'Servers_Corrupted': Number of server corrupted during the attack
- 'Pages_Corrupted': Number of pages illegally accessed
- 'Location': Location attack came from (Probably useless because the hackers used VPNs)
- 'WPM_Typing_Speed': Their estimated typing speed based logs.

# Python and Spark

- The technology firm has 3 potential hackers that perpetrated the attack.
- They are certain of the first two hackers but they aren't very sure if the third hacker was involved or not.
- They have requested your help!

# Python and Spark

- Can you help figure out whether or not the third suspect had anything to do with the attacks, or was it just two hackers?
- It's probably not possible to know for sure, but maybe what you've just learned about Clustering can help!

# Python and Spark

- One last key fact, the forensic engineer knows that the hackers trade off attacks.
- Meaning they should each have roughly the same amount of attacks.

- For example if there were 100 total attacks, then in a 2 hacker situation each should have about 50 hacks, in a three hacker situation each would have about 33 hacks.

# Python and Spark

- The engineer believes this is the key element to solving this, but doesn't know how to distinguish this unlabeled data into groups of hackers.

# Python and Spark

- Best of luck with this project, it should be a fun one!
- If you get stuck, feel free to go straight to the solution lecture.
- Enjoy!

# K-Means Clustering Consulting Project Solutions