Lesson 1: Introduction to Clustering

Lesson Objectives

By the end of this lesson, you will be able to:

* Distinguish between supervised learning and unsupervised learning
* Explain the concept of clustering
* Implement k-means clustering algorithm using built-in Python packages

Introduction

“What does this data mean?”

Have you ever been asked to take a look at some data and came up empty handed? Maybe you’re not familiar with the data set, or maybe you don’t even know where to start. This may have been extremely frustrating and even embarrassing at times depending on who asked you to take care of the task.

You are not alone, and interestingly enough there are many times the data itself is simply too confusing to be made sense of. As you try and figure out what all those numbers in your spreadsheet mean, you’re most likely mimicking what many unsupervised algorithms do when they try to find meaning in data. The reality is that many datasets in the real world don’t have any rhyme or reason to them. You will be tasked with analyzing them with little background preparation. Don’t fret however – this course will prepare you to never be frustrated again when dealing with data exploration asks.

Through this course we have developed some of the best in class content to help you understand how unsupervised algorithms work and where to use them. We cover some of the foundations of finding clusters in your data, how to reduce the size of your data so it’s easier to understand, and how each of these sides of unsupervised learning can be applied in the real world. Our hope is that you will come away from this course with a strong real world understanding of unsupervised learning and what problems it can solve and those it cannot.

Thanks for joining us and we hope you enjoy the ride!

Discussion: If you were asked to understand a dataset, how would you do it? How many people in this class are familiar with supervised learning? What would you do if you got a dataset without labels, or targets, to train on?

Unsupervised Learning vs Supervised Learning

Unsupervised learning is one of the most exciting areas of development in machine learning today. If you have explored machine learning coursework before you are probably familiar with the common breakout of problems into either supervised or unsupervised learning. Supervised learning encompasses the problem set of having a labeled dataset that can be used to either classify (for example, predicting smoker or non-smoker if you’re looking at a lung health data set) or fit a regression line on (for example, predicting the sale price of a home based on how many bedrooms it has). This model most closely mirrors an intuitive human approach to learning.

If you wanted to learn how to not burn your food with a basic understanding of cooking, you can build a dataset by putting your food on the burner and seeing how long it takes (input) for your food to burn (output). Eventually as you continue to burn your food you will build a mental model of when burning will occur and avoid it in the future. Development in supervised learning has been fast-paced in valuable, but it has since simmered down in recent years – many of the obstacles in the world of knowing your data have already been tackled.



Figure 1.1: Differences between Unsupervised and Supervised Learning

Conversely, unsupervised learning encompasses the problem set of having a tremendous amount of data that is unlabeled. Labeled data in this case would be data that has a supplied “target” outcome that you are trying to find the correlation towards with supplied data (you know that you are looking for whether your food was burned in the previous example). Unlabeled data is when you do not know what the “target” outcome is and you only have supplied input data.

Building upon the previous example, imagine you were just dropped on planet Earth with zero knowledge of how cooking works. You are given 100 days, a stove, and a fridge full of food without any instructions on what to do. Your initial exploration of a kitchen can go in infinite directions – on day 10 you may finally learn how to open the fridge, on day 30 you may learn that food can go on the stove, and after many days in between you may have unwittingly made an edible meal. As you can see, trying to find meaning in a kitchen devoid of adequate informational structure leads to very noisy data that is completely irrelevant to actually preparing a meal.

Unsupervised learning can be an answer to this problem. By looking back at your 100 days of data, clustering can be used to find patterns of similar days where a meal was produced, and you can easily review what you did those days. However, unsupervised learning isn’t a magical answer – by simply finding clusters it can just as likely help you find pockets of similar yet ultimately useless data.

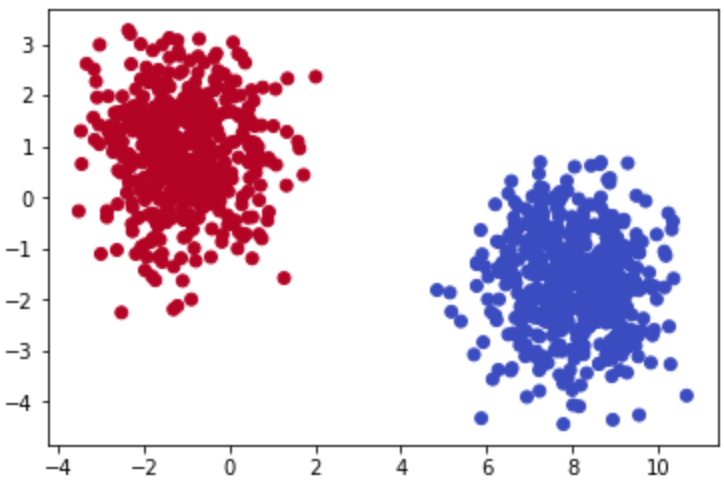
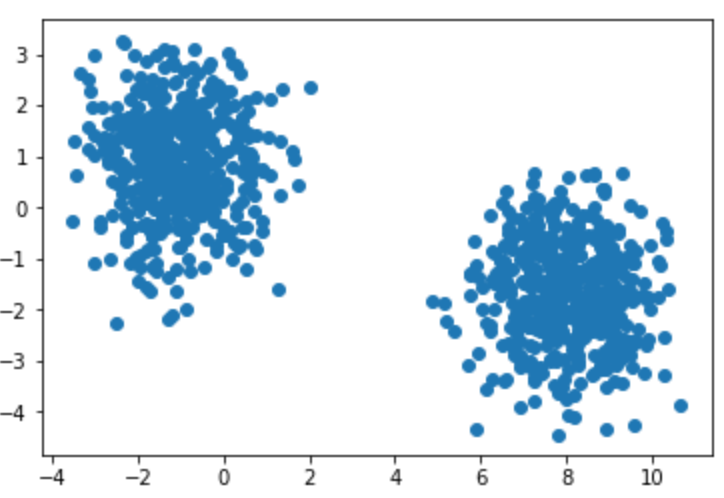
This challenge is what makes unsupervised learning so exciting. How can we find smarter techniques to speed the process to finding clusters of information that are beneficial to our end goals?

Clustering

Being able to find groups of similar data that exist in your data set can be extremely valuable if you are trying to find its underlying meaning. If you were a store owner and you wanted to understand which of your customers are more valuable without a set idea of what “valuable” is, clustering is a great place to start to find patterns in your data. You may have a few high-level ideas of what denotes a valuable customer, but you aren’t entirely sure in the face of a large mountain of available data. Through clustering you can find commonalities among similar groups in your data. If you look deeper at a cluster of similar people you may learn that everyone in this group visits your website for longer periods of time than others. This can show you what value is and also provides a clean sample size for future supervised learning experiments.

Discussion: Who here is familiar with clustering? Can you give examples from past work experience where clustering has proven valuable? If you haven’t used it before, can you provide some examples of why knowing what records are similar in your data set may be valuable?

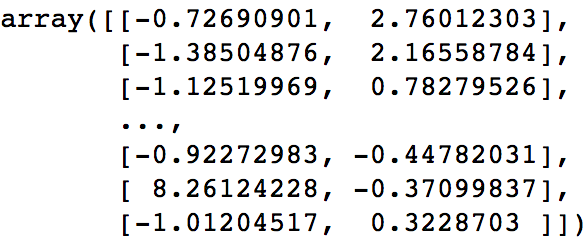
Below figures provide an extremely oversimplified example of what clustering is. Both figures display randomly generated number pairs (x,y coordinates) pulled from a gaussian distribution. Just by simply glancing at Figure 1.2 it should be plainly obvious where the clusters exist in your data - in real life it will never be this easy. Now that you know that the data can be clearly separated into 2 clusters, you can start to understand what differences exist between the two groups.



Figures 1.2 (left) and 1.3 (right): Scatter plots clearly showing clusters that exist among a provided dataset.

Rewinding a bit from where unsupervised learning fits into the larger machine learning environment, let’s begin by understanding the building blocks of clustering. The most basic definition finds clusters simply as groupings of similar data as subsets of a larger data set. Finding clusters can be immensely valuable in identifying items that are more similar to, and on the other end, quite different from each other.

To understand this idea further imagine that you were given a simple 1,000-row dataset by your employer that had 2 columns of numerical data (Figure 1.4). At first glance this dataset provides no real structure or understanding – confusing to say the least!



Figures 1.4: 2-dimensional raw data in a NumPy array

You begin by plotting the first column against the second column to get a better idea of what the data structure looks like. There will be plenty of times where the cause of differences between groups will prove to be underwhelming, however the cases that have differences that you can take action on are extremely rewarding!

Activity 1: Identifying the clusters in data

Scenario: You are given two-dimensional plots. Please look at the provided two-dimensional graphs and identify the clusters, to drive the point home that machine learning is important

Aim: To identify where the clusters exist in the data.

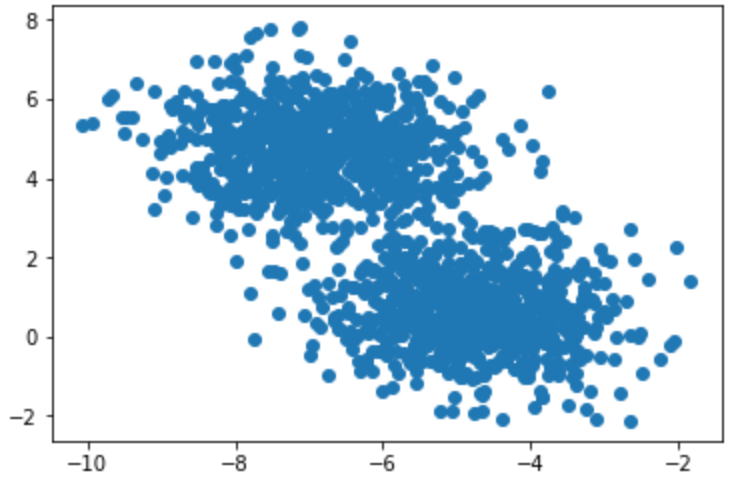


Figure1.5 Two-dimensional scatter-plot

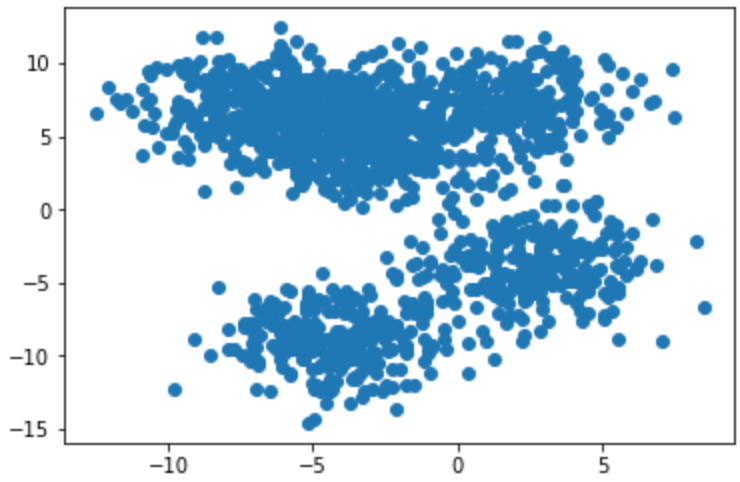


Figure1.6 Two-dimensional scatter-plot

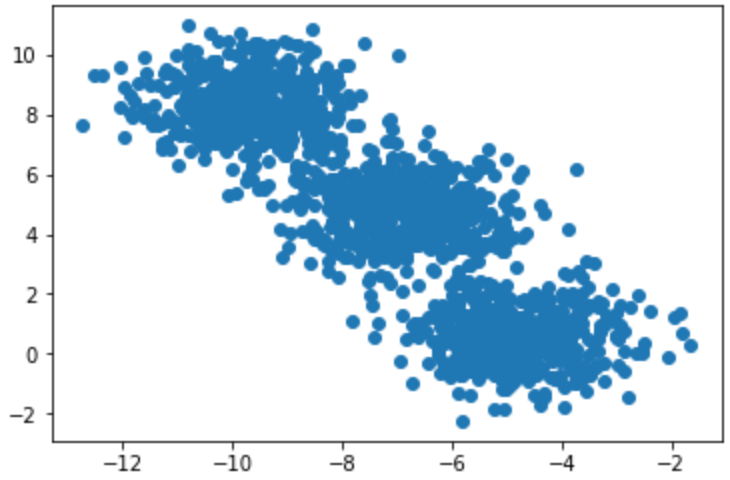


Figure1.7 Two-dimensional scatter-plot

Introduction to K-Means Clustering

Hopefully by now you can see that finding clusters is extremely valuable in a machine learning workflow. However, how can you actually find these clusters? One of the most basic yet popular approaches is by using a cluster analysis called K-Means Clustering. The workflow behind K-Means is actually quite intuitive – here is the no-math introduction to the clustering analysis:

1. Pick K centroids (K = expected distinct # of clusters)
2. Randomly place K centroids anywhere amongst your existing training data
3. Calculate the Euclidean distance from each centroid to all the points in your training data
4. Training data points get grouped in with their nearest centroid
5. Amongst data points grouped into each centroid, calculate the mean data point and move your centroid to that location.
6. Repeat the above process until convergence, or when membership in each group no longer changes.

And that’s it! Here is the above process laid out step by step with the previous simple cluster examples:

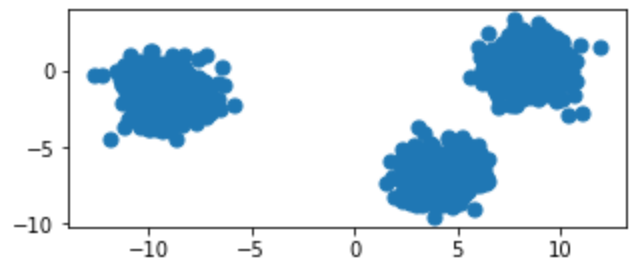


Figure 1.8 Original raw data charted on x,y coordinates

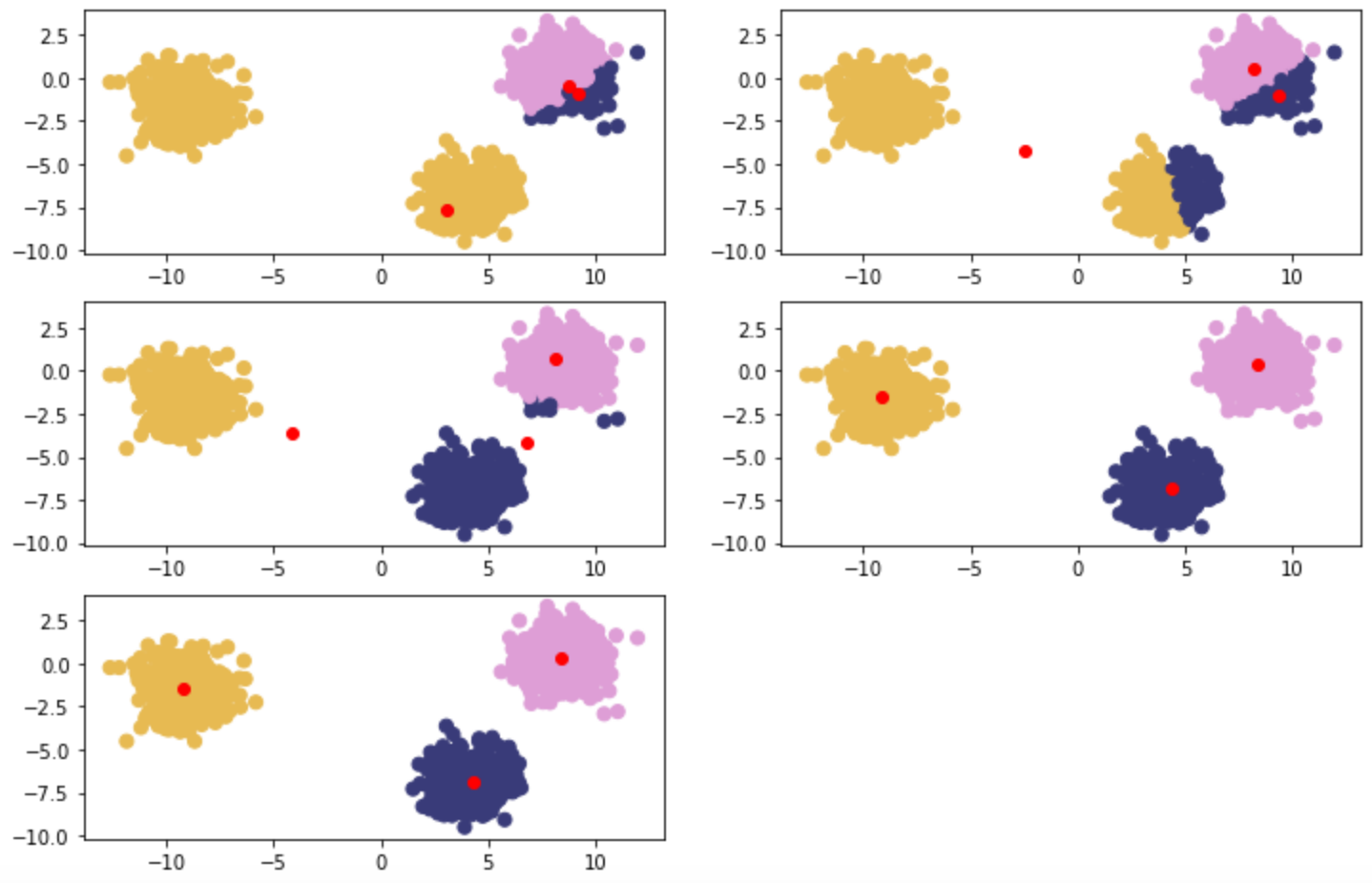


Figure 1.9: Clockwise from top left – Red points are randomly initialized centroids, and closest data points are assigned to groupings of each centroid (denoted by distinct color).

K-Means Clustering Step-by-Step Walkthrough

Preceding examples are clear to visualize when your data is only two-dimensional. This is for convenience to help drive the point home of how K-Means works and could lead you into a false understanding regarding how easy clustering is. In many of your own applications, your data will likely be orders of magnitude larger to the point that it cannot be perceived by visualization (anything beyond 3 dimensions will be imperceptible to human understanding). In the above examples you could mentally work out a few 2-dimensional lines to separate the data into their own groups. At higher dimensions you will need to be aided by a computer to find an n-dimensional hyperplane that adequately separates the dataset. In practice, this is where clustering methods such as K-Means provide significant value.

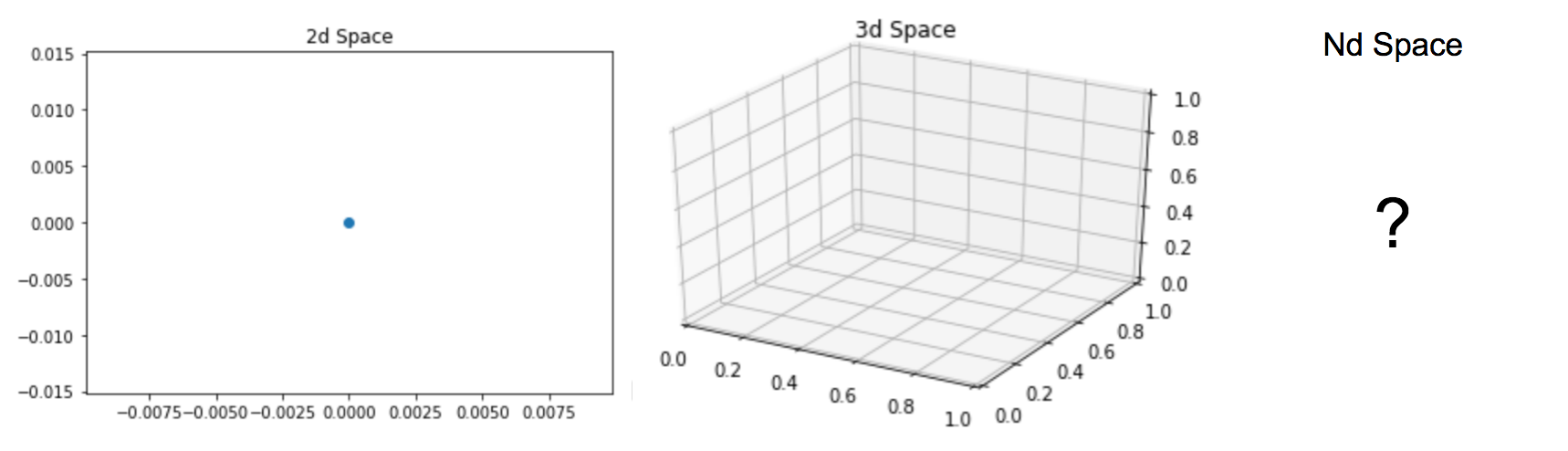


Figure 1.10: 2d plot, 3d plot, nd plot

To add more detail let’s walk through the example given in the introductory section again with some of the math that supports K-Means. The key component at play is the Euclidean distance formula:

Discussion: Does anyone have an idea of what a centroid is? (A centroid is basically just a fancy way to say the geometric center of a figure. This is the mean of all points in the plane. When randomly set at first they are simply random points in the space)

Centroids are randomly set at the beginning as points in your n-dimensional space. Each of these centers is fed into the above formula as (a,b), and a point in your space is fed in as (x,y). Distances are calculated between each point and the coordinates of every centroid, with the centroid of smallest distance away chosen as the point’s group. This process can be seen below:

1. Random Centroids: [ (2,5) , (8,3) , (4, 5) ]
2. Arbitrary point x: (0, 8)
3. Distance from point to each centroid: [ 3.61, 9.43, 5.00 ]
4. Point x is assigned to Centroid 1.

This process is repeated for every point in the dataset until each point is assigned to a cluster. After each point is assigned, the mean point is calculated among all of the points within each cluster. Calculation of mean among these points is the same as calculating a mean between single integers. An example of this can be seen here:

1. Cluster 1 Membership: [ (0,8), (3,8), (3,4) ]
2. Mean point calculation: [ (0+3+3)/3 , (8+8+4)/3 ] = (2, 6.67)
3. Centroid for Cluster one becomes (2, 6.67)
4. Repeat process until distance between new and old centroids = 0

Activity 2: Implementing K-Means Clustering

Scenario: You are asked in an interview to implement a K-Means clustering algorithm from scratch to prove you understand how it works.

Aim: To truly understand how something works you need to build it from scratch. Take what you have learned in the above sections and implement K-Means from scratch in Python.

Please open your favorite editing platform and try the following:

1. Using NumPy and the above Euclidean distance formula, write a function that calculates the distance between two coordinates.
2. Write a function that calculates the distance from centroids to each of the points in your dataset and returns the cluster membership.
3. Write a K-Means function that takes in a dataset and number of clusters (K) and returns final cluster centroids, as well as the data points that make up that cluster’s membership.

After implementing K-Means from scratch, apply your custom algorithm to the Iris dataset located here: <https://archive.ics.uci.edu/ml/datasets/iris>

Remove the classes supplied in this dataset and see if your K-Means algorithm can group the different Iris species into their proper groups just based on plant characteristics!

Summary

In this lesson we have explored what clustering is and why it could be important in a variety of data challenges. Building upon this foundation of clustering knowledge you implemented K-Means, which is one of the simplest yet most popular methods of unsupervised learning. If you have reached this summary and can repeat what K-Means does step-by-step to your fellow classmate, good job! If not please go back and review the material above – the content only grows in complexity from here.

Practice Questions

1. How are supervised and unsupervised learning different?
2. What is a cluster?
3. Are clusters always valuable in a dataset?
4. How are dimensions expressed in a dataset?
5. How many dimensions can be easily visualized by humans?
6. What does the “K” represent in K-Means clustering?
7. T/F – The starting points for K-Means are determined by taking the mean of all the points in the space.
8. What is the formula that underpins K-Means clustering?
9. When does a K-Means clustering algorithm finish running?
10. T/F – K-Means will still find clusters in a dataset even if all of the data is fairly similar.