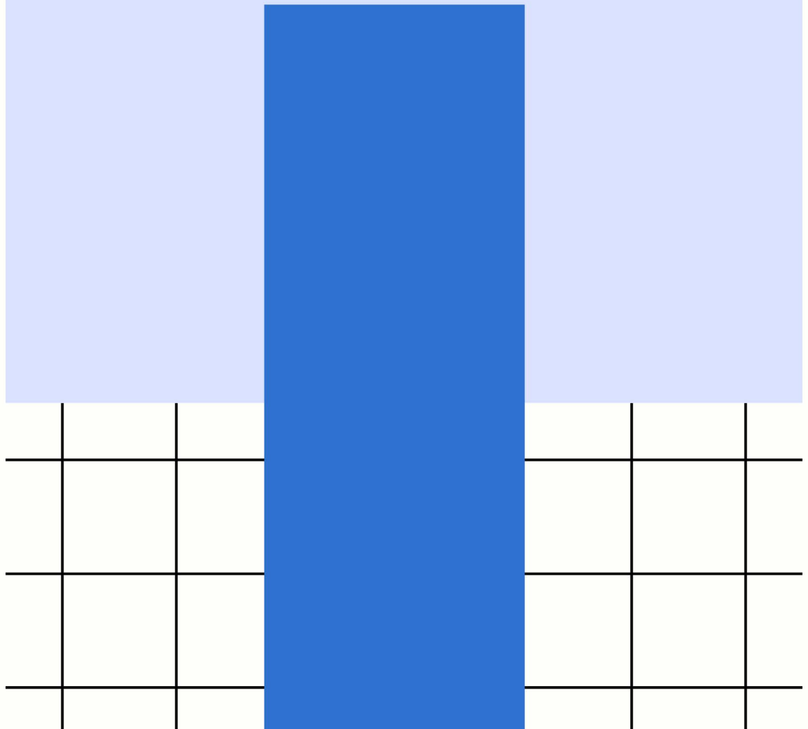
CLUSTERING ALGORITHMS

– Matee Vadrukchid –



Supervised vs. Unsupervised Learning

Supervised Learning

In supervised learning, we are provided with a set of training pairs $(\mathbf{x}^{(i)}, y^{(i)})$, where each $\mathbf{x}^{(i)}$ is an input vector belonging to the input space $\mathcal{X}$ and each $y^{(i)}$ is the corresponding label or target value from the output space $\mathcal{Y}$. Here, $\mathcal{X}$ typically represents $\mathbb{R}^n$ (an n-dimensional real vector space), and $\mathcal{Y}$ could either be $\mathbb{R}$ for regression tasks or a set of discrete categories $\{1, \ldots, k\}$ for classification tasks. The goal in supervised learning is to learn a function that, given a new input $\mathbf{x} \in \mathcal{X}$, predicts the corresponding output $\hat{y} \in \mathcal{Y}$ or estimates the probabilities $p(y = y_i \mid \mathbf{x})$ for each category y_i based on the learned model.

Unsupervised Learning

In unsupervised learning, the training data consist of a set of input vectors $\mathbf{x}^{(i)} \in \mathcal{X}$ without any associated target values or labels. The primary objective is to discover the underlying structure of the data or to identify patterns that characterize the distribution of $\mathbf{x}^{(i)}$ in the input space. Given a new instance $\mathbf{x} \in \mathcal{X}$, the task might involve determining how this instance relates to or differs from previously observed examples.

Among the most prevalent goals in unsupervised learning is **clustering**, which involves organizing the data into K distinct groups such that data points in the same group (or cluster) are more similar to each other than to those in other groups. When a new data point $\mathbf{x}$ is presented, the model identifies the cluster $c \in \{1, \ldots, K\}$ to which $\mathbf{x}$ most closely belongs.

K-means Clustering

Research on k-means Clustering Algorithm

An Improved k-means Clustering Algorithm

Shi Na
College of Information Engineering, Capital Normal University
CNU
Beijing, China
shina8237140@126.com

Liu Xumin
College of Information Engineering, Capital Normal University
CNU
Beijing, China
hellosn@126.com

Guan yong
College of Information Engineering, Capital Normal University
CNU
Beijing, China
whwqd@126.com

***Abstract*—Clustering analysis method is one of the main analytical methods in data mining,the method of clustering algorithm will influence the clustering results directly. This paper discusses the standard k-means clustering algorithm and analyzes the shortcomings of standard k-means algorithm, such as the k-means clustering algorithm has to calculate the distance between each data object and all cluster centers in each iteration, which makes the efficiency of clustering is not high. This paper proposes an improved k-means algorithm in order to solve this question, requring a simple data structure to store some information in every iteration,which is to be used in the next interation.The improved method avoids computing the distance of each data object to the cluster centers repeatly, saving the running time. Experimental results show that the improved method can effectively improve the speed of clustering and accuracy, reducing the computational complexity of the k-means.**

Keywords-clustering analysis; k-means algorithm; distance; computational complexity

I. INTRODUCTION

Clustering is a way that classify the raw data reasonably and searchs the hidden patterns that may exist in datasets [7]. It is a process of grouping data objects into disjointed clusters so that the datas in the same cluster are similar, yet datas belonging to different cluster differ.The demand for organizing the sharp

K-means is a numerical, unsupervised, non-deterministic, iterative method. It is simple and very fast , so in many practical applications, the method is proved to be a very effective way that can produce good clustering results. But it is very suitable for producing globular clusters. Several attemps were made by researchers to improve efficiency of the k-means algorithms [5]. In the literature [3], there is an improved k-means algorithm based on weights. This is a new partitioning clustering algorithm, which can handle the datas of numerical attribute, and it also can handle the datas of symbol attribute. Meanwhile, this method reduces the impact of isolated points and the “noise”, so it enhances the efficiency of clustering. However, this method has no improvement on the complexity of time. In the literature [1], it proposed a systematic method to find the initial cluster centers, This centers obtained by this method are consistant with the distribution of datas. Hence this method can produce more accurate clustering results than the standard k-means algorithm, but this method does not have any improvements on the executive time and the time complexity of algorithm. This paper presents an improved k-means algorithm. Although this algorithm can generate the same clustering results as that of the standard k-means algorithm, the algorithm of this paper proposed is superior to the standard k-means method on running time and accuracy, thus enhancing the speed of clustering and improving the time complexity of algorithm. By comparing the experimental

k-means Clustering

k-means is a cornerstone algorithm in unsupervised learning specifically tailored for clustering. It is particularly applicable when $\mathcal{X} = \mathbb{R}^n$ and the Euclidean distance is a meaningful measure of dissimilarity between data points.

The k-means Algorithm

The k-means clustering algorithm proceeds as follows:

1. **Initialization**: Start by randomly selecting k initial points from the dataset to serve as the initial centroids $\mu_1, \ldots, \mu_k$ of the clusters.
2. **Iterative Optimization**:
   - **Assignment Step**: For each data point $\mathbf{x}^{(i)}$, determine the nearest centroid and assign $\mathbf{x}^{(i)}$ to that cluster. This is mathematically represented as:

     $$c^{(i)} \leftarrow \text{argmin}_j \|\mathbf{x}^{(i)} - \mu_j\|^2$$

     where $\|\cdot\|$ denotes the Euclidean distance.
   - **Update Step**: Adjust each centroid to be the mean of the points assigned to it, ensuring that it accurately represents its cluster:

     $$\mu_j \leftarrow \frac{\sum_{i=1}^{m} \delta(c^{(i)} = j)\mathbf{x}^{(i)}}{\sum_{i=1}^{m} \delta(c^{(i)} = j)}$$

     In this formula, δ is an indicator function that equals 1 if the condition is true and 0 otherwise. This function effectively counts and sums only those $\mathbf{x}^{(i)}$ that belong to the jth cluster.

Imagine you had some data that you could plot on a line, and you knew you needed to put it into 3 clusters. Maybe they are measurements from 3 different types of tumors or other cell types.

In this case the data make three, relatively obvious, clusters.

But, rather than rely on our eye, let's see if we can get a computer to identify the same 3 clusters.

To do this, we'll use K-means clustering.

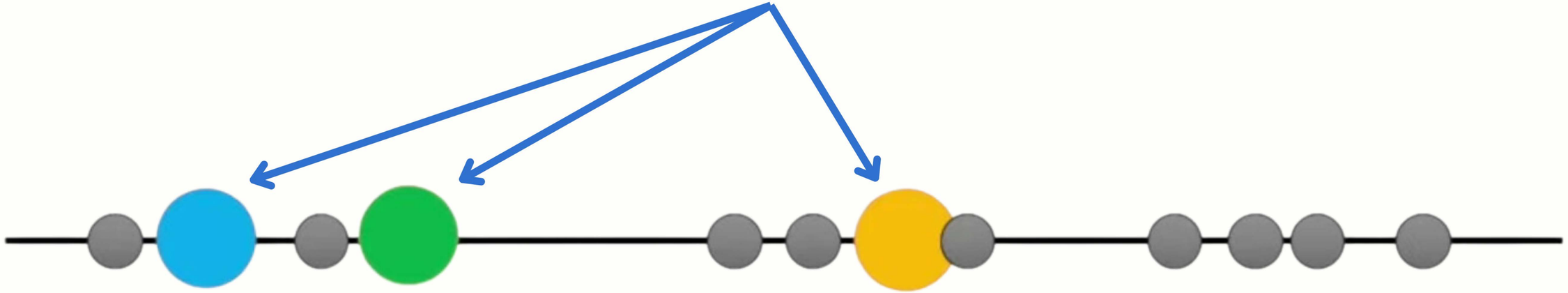
Step 1: Select the number of clusters you want to identify in your data. This is the “K” in “K-means clustering

In this case, we’ll select K=3. That is to say, we want to identify 3 clusters.

There is a fancier way to select a value for “K”, but we’ll talk about that later.

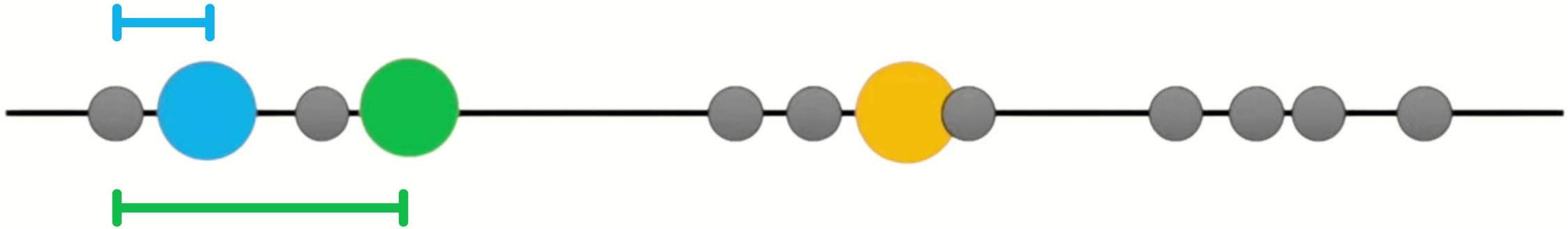
Step 2: Randomly select 3 distinct data points

These are the initial clusters.



Step 3: Measure the distance between the 1st point and the three initial clusters.

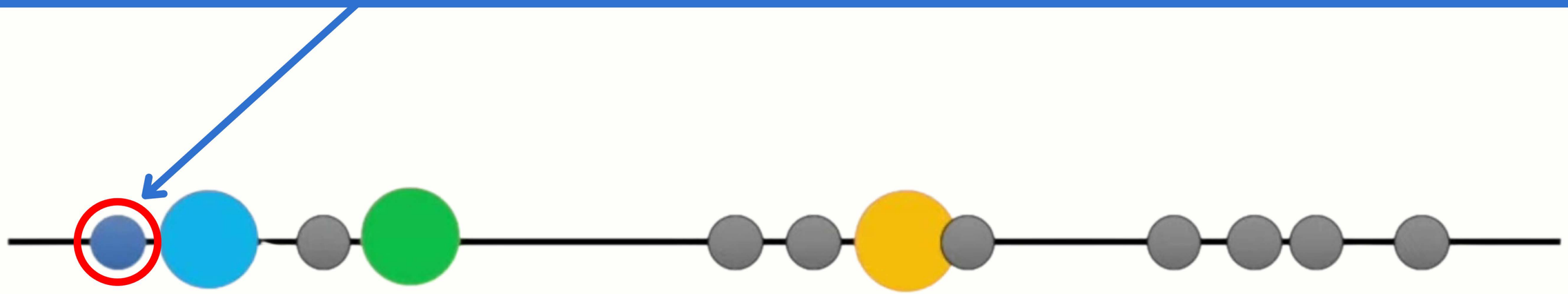
Distance from the 1st point to the **blue** cluster



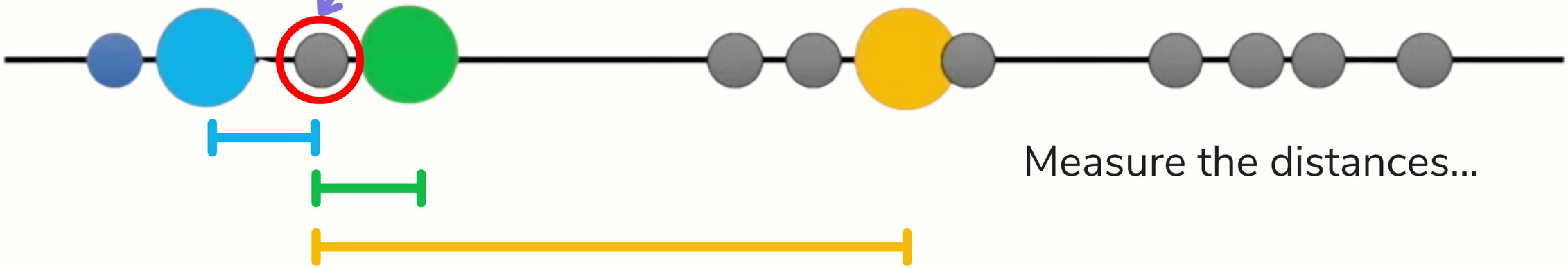
Distance from the 1st point to the **green** cluster

Distance from the 1st point to the **orange** cluster

Step 4: Assign the 1st point to the nearest cluster. In this case, the nearest cluster is the blue cluster.

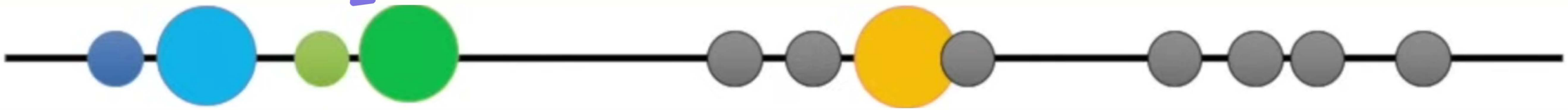


Now do the same things for the next point

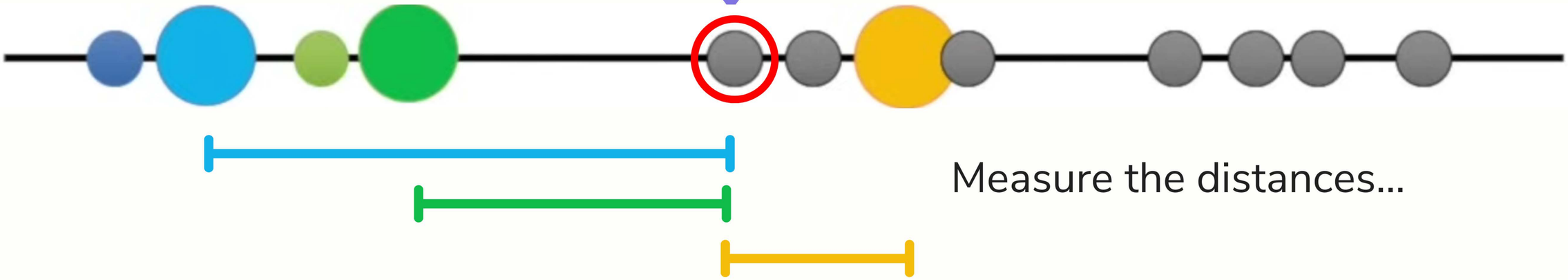


Measure the distances...

Assign the point to the nearest cluster.

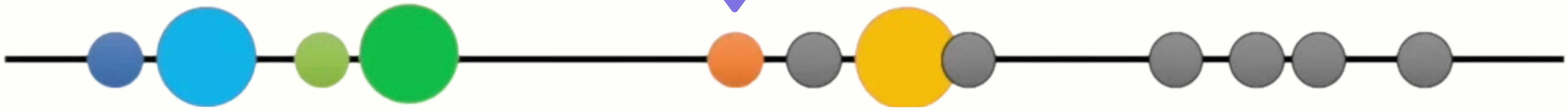


Now figure out which cluster the 3rd point belongs to

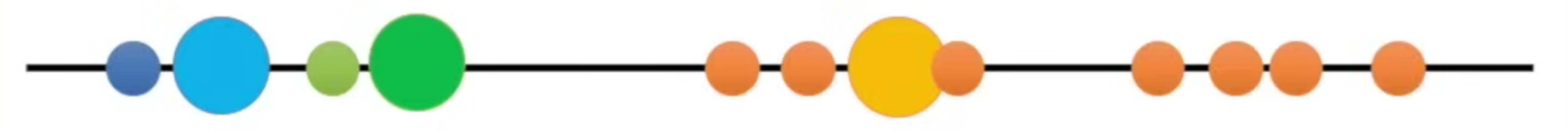


Measure the distances...

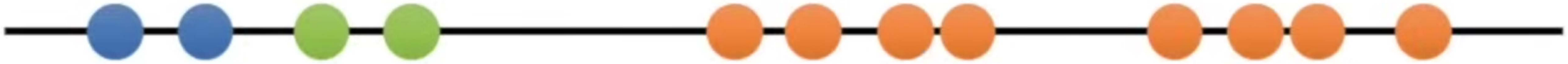
Assign the point to the nearest cluster.



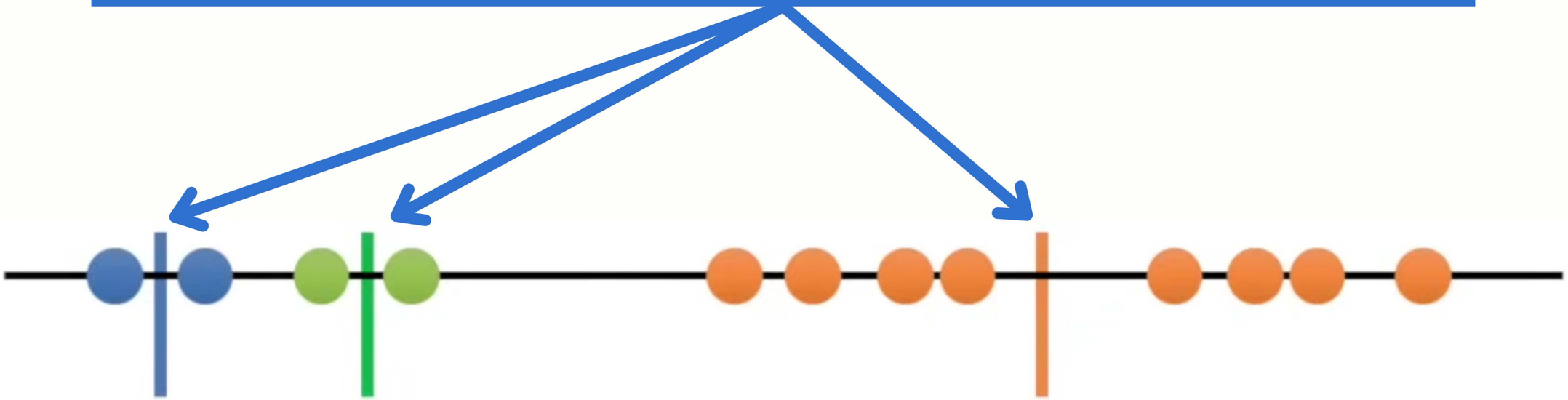
The rest of these points are closest to the orange cluster, so they'll go in that one, too.



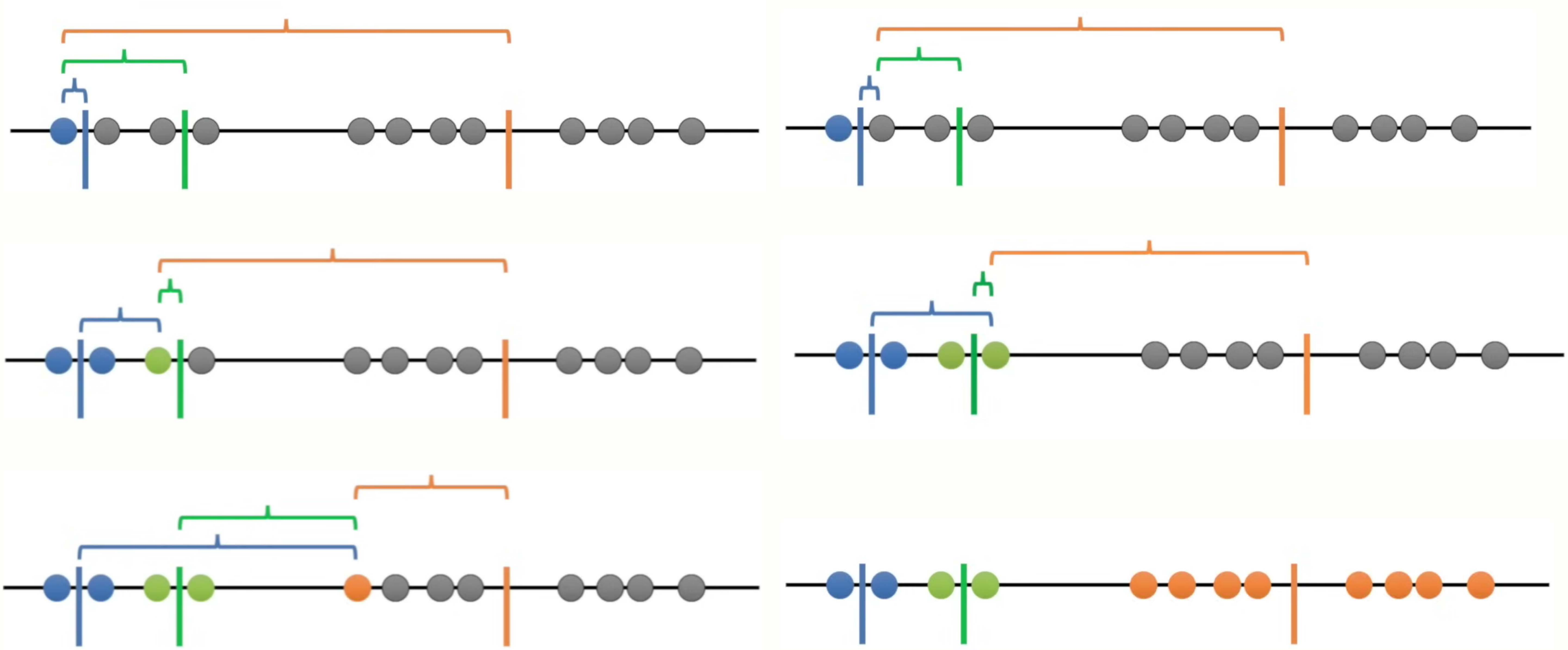
Now that all of the points are in clusters, we go on to...



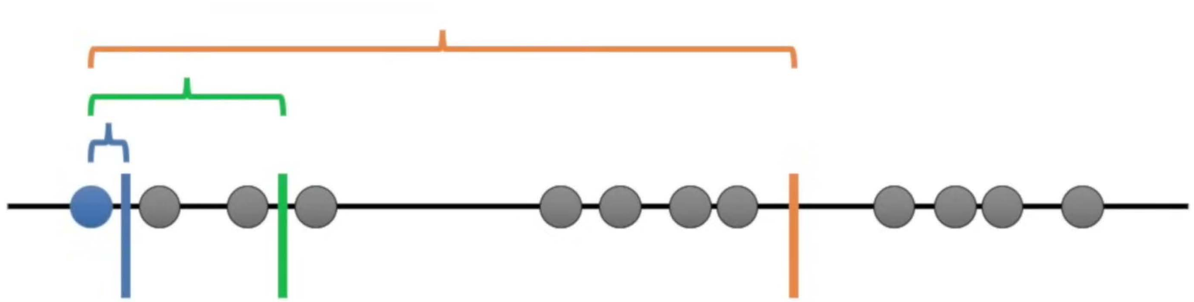
Step 5: calculate the mean of each cluster.

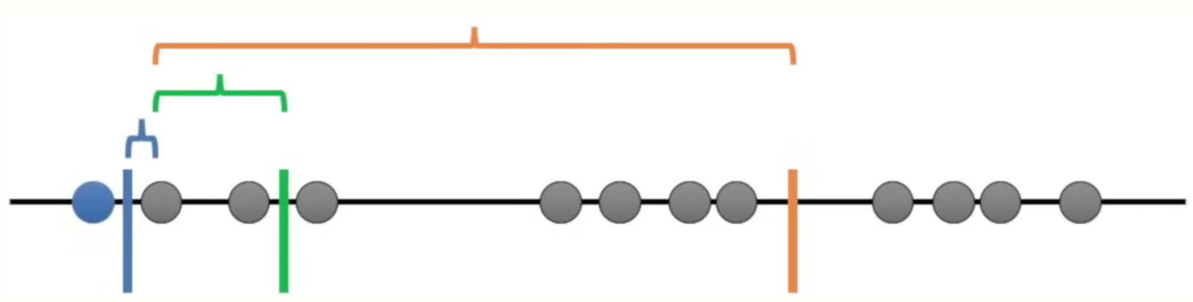


Then we repeat what we just did (measure and cluster) using the mean values.

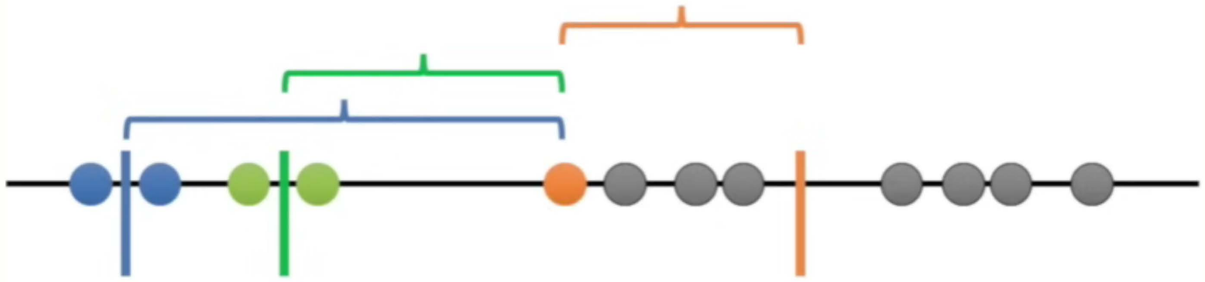


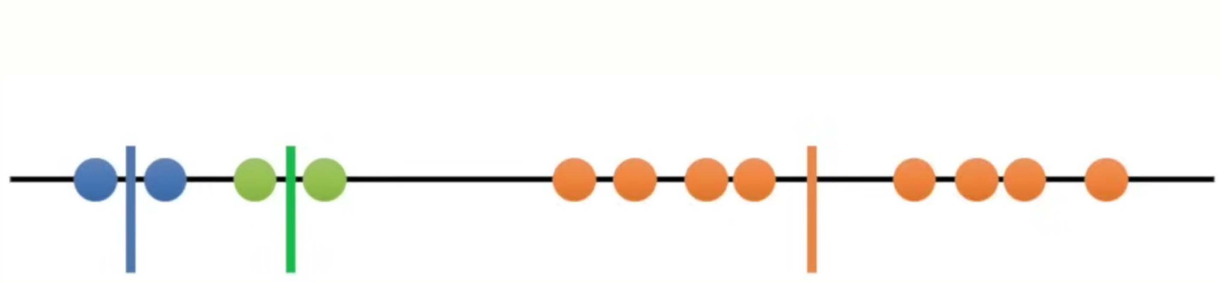
Then we repeat what we just did (measure and cluster) using the mean values.

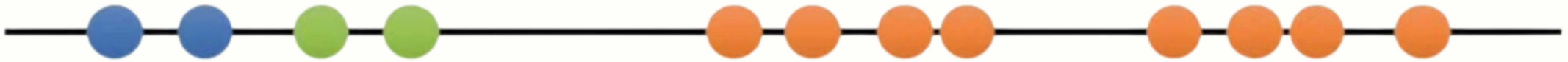




Since the clustering did not change at all during the last iteration, we're done...

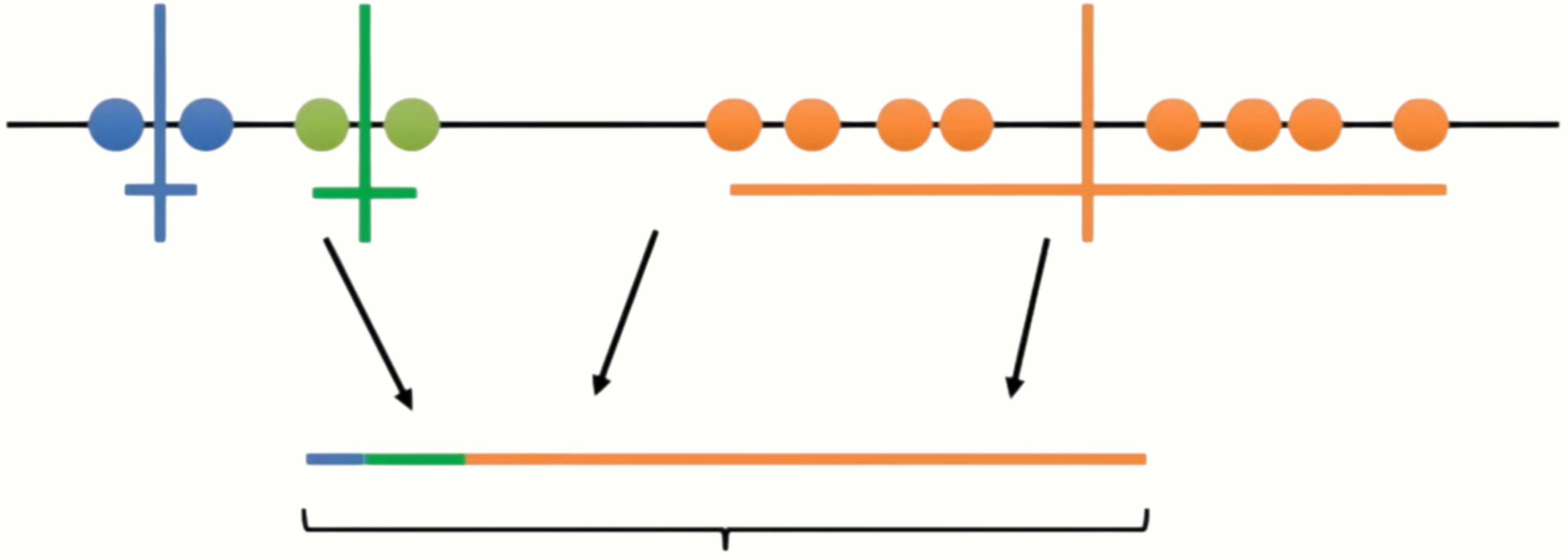






The K-means clustering is pretty terrible compared to what we did by eye.

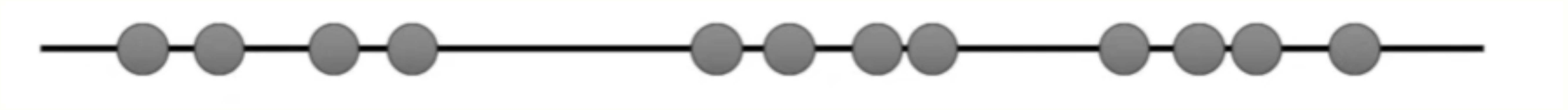
We can assess the quality of the clustering by adding up the variation within each cluster.



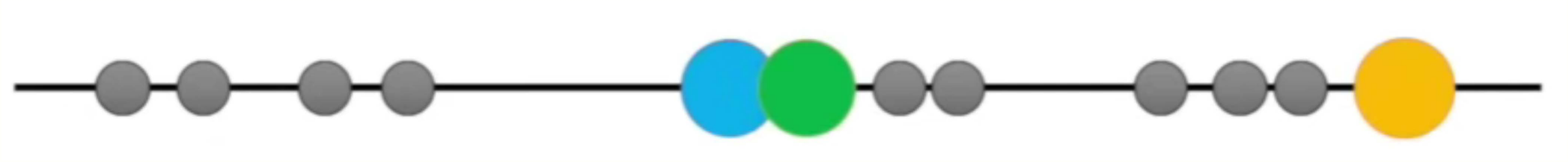
Total variation within the clusters

Since K-means clustering can't "see" the best clustering, its only option is to keep track of these clusters, and their total variance, and do the whole thing over again with different starting points.

So, here we are again, back at the beginning.

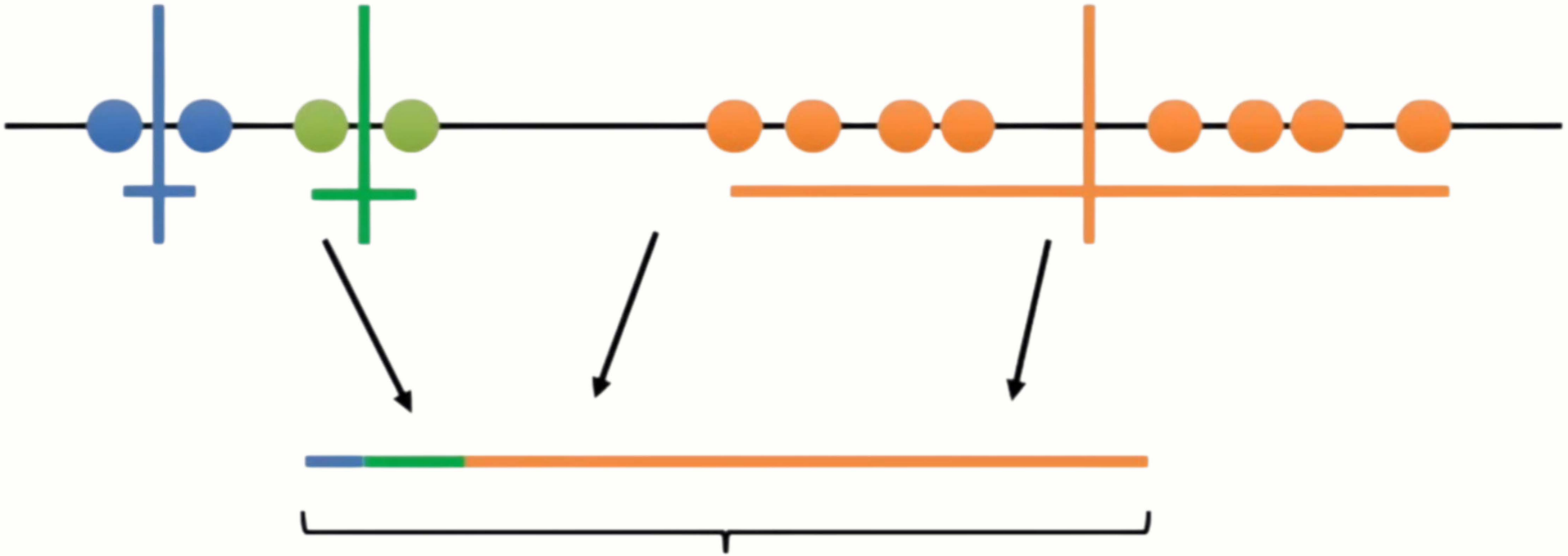


K-means clustering picks 3 initial clusters...



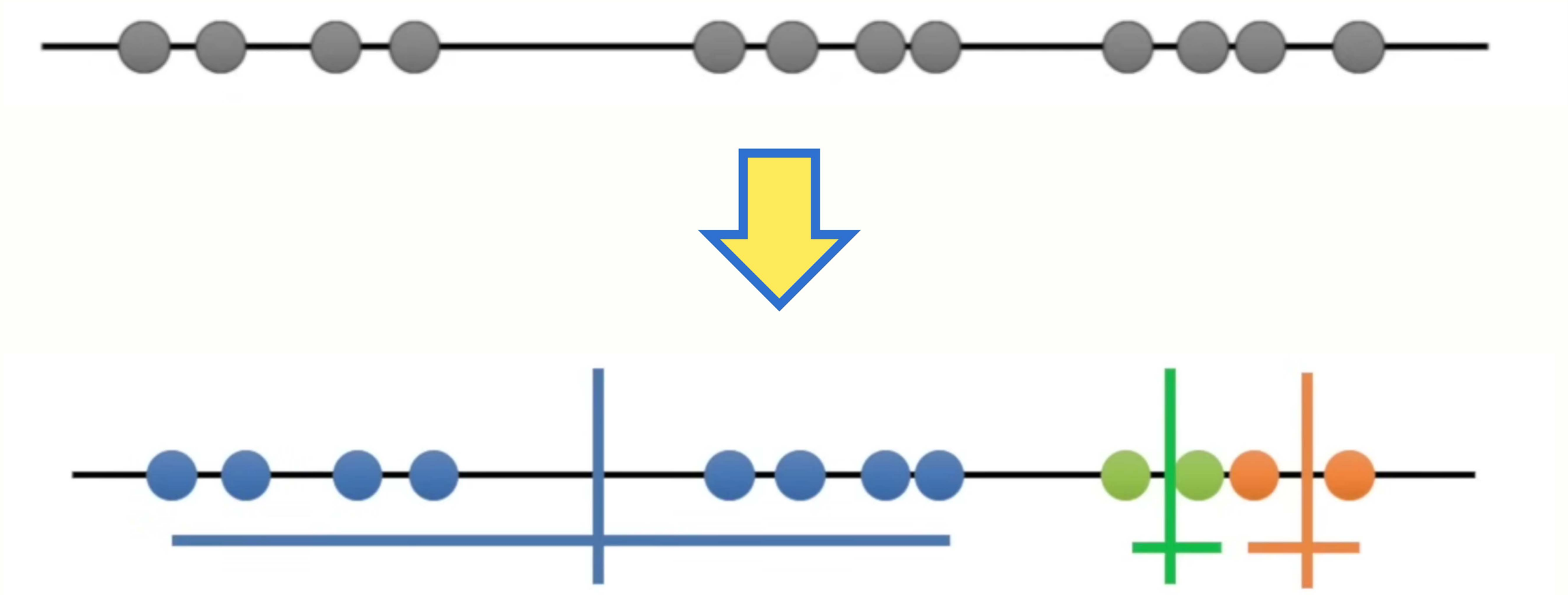
...and then clusters all the remaining points, calculates the mean of each cluster and then reclusters based on the new means. It repeats until the clusters no longer change.

Now that the data are clustered, we sum the variation within each cluster.

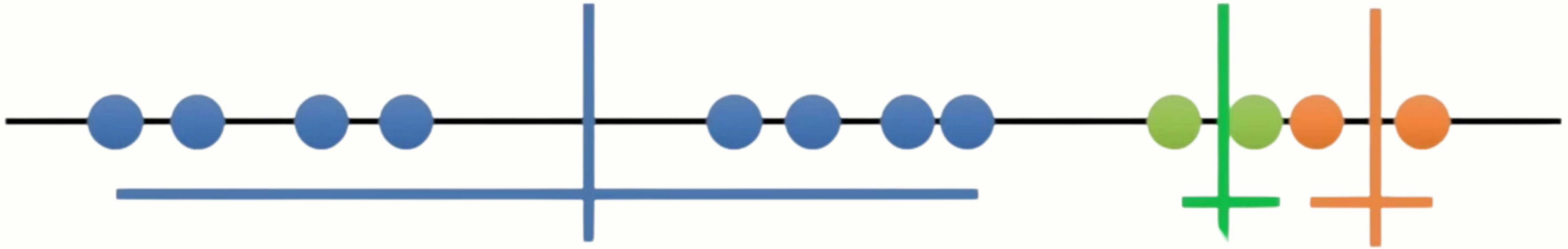


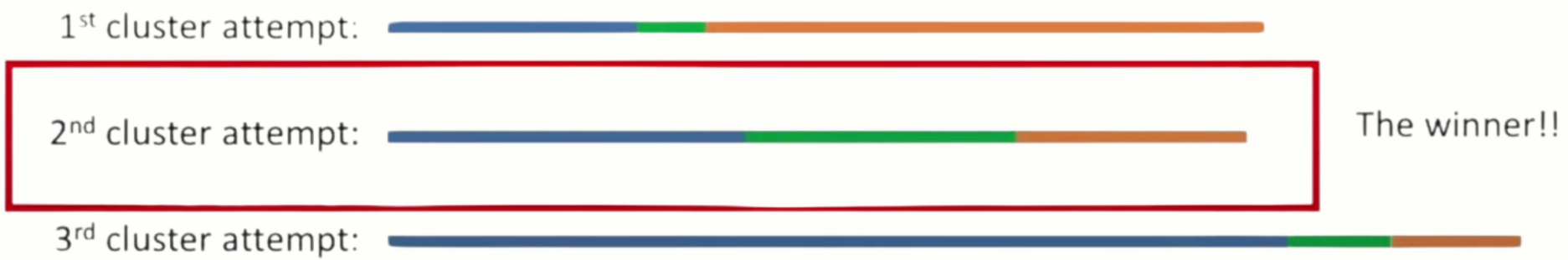
Total variation within the clusters

And then do it all again...

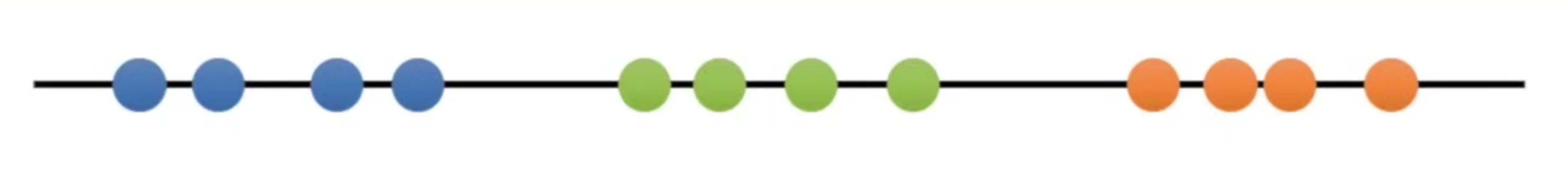


At this point, K-means clustering knows that the 2nd clustering is the best clustering so far. But it doesn't know if it's the best overall, so it will do a few more clusters (it does as many as you tell it to do) and then come back and return that one if it is still the best.





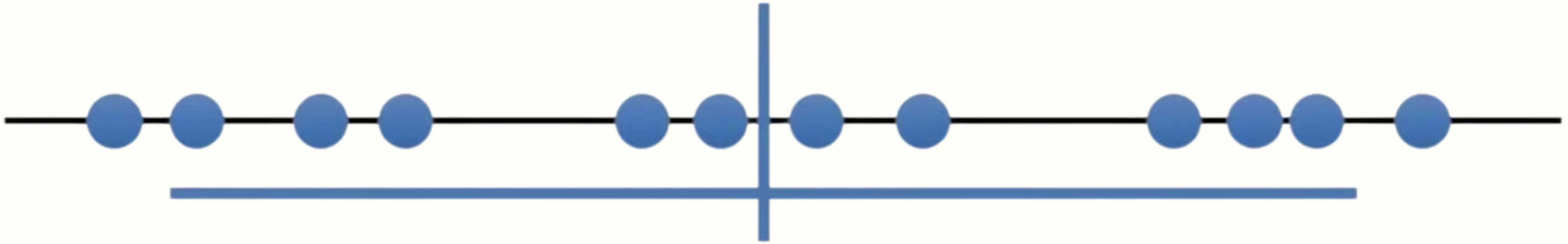
Question: How do you figure out what value to use for 'K'?



With this data, it’s obvious that we should set K to 3,
but other times it is not so clear.

One way to decide is to just try different values for K.

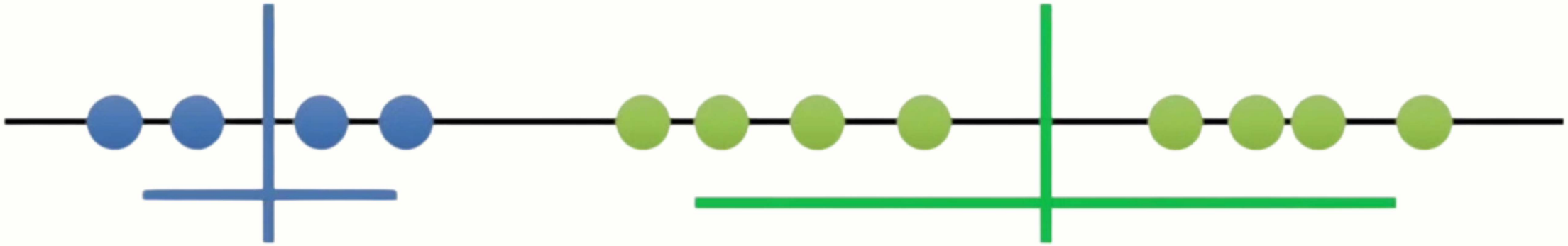
Start with K = 1



K = 1 is the worst case scenario.

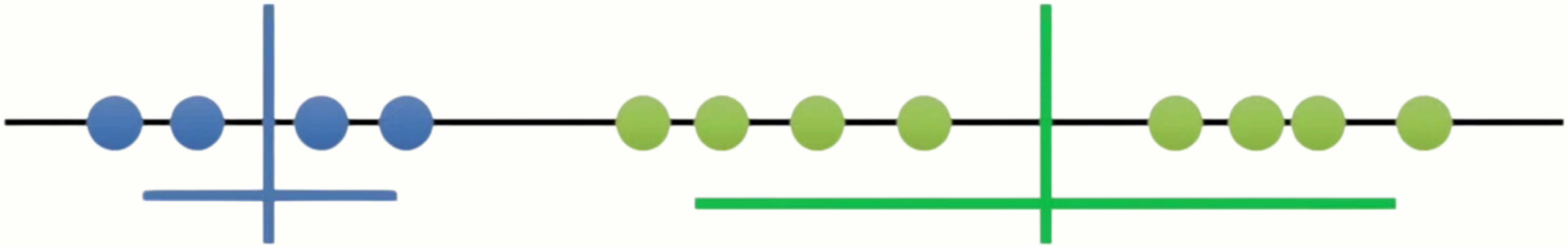
We can quantify its 'badness' with the total variation.

Start with K = 2



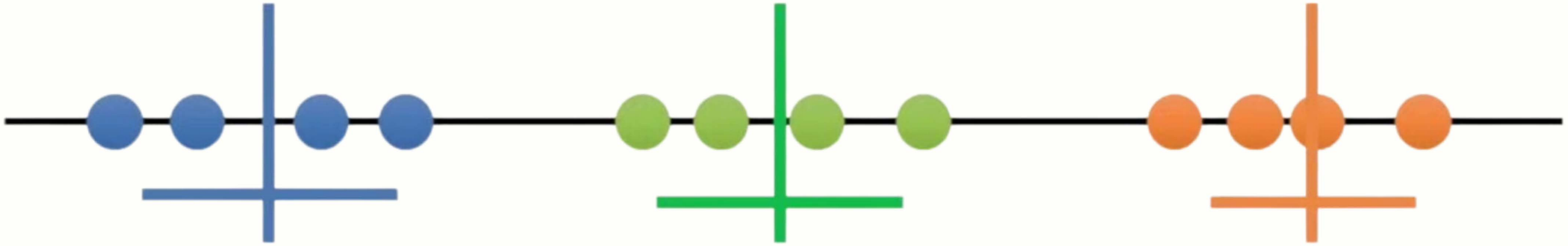
K = 2 is better, and we can quantify how much better by comparing the total variation within the 2 clusters to K = 1

Start with K = 2

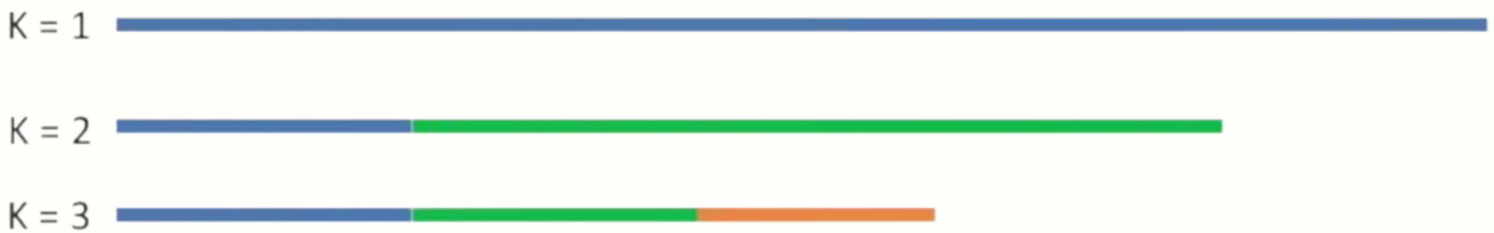


K = 2 is better, and we can quantify how much better by comparing the total variation within the 2 clusters to K = 1

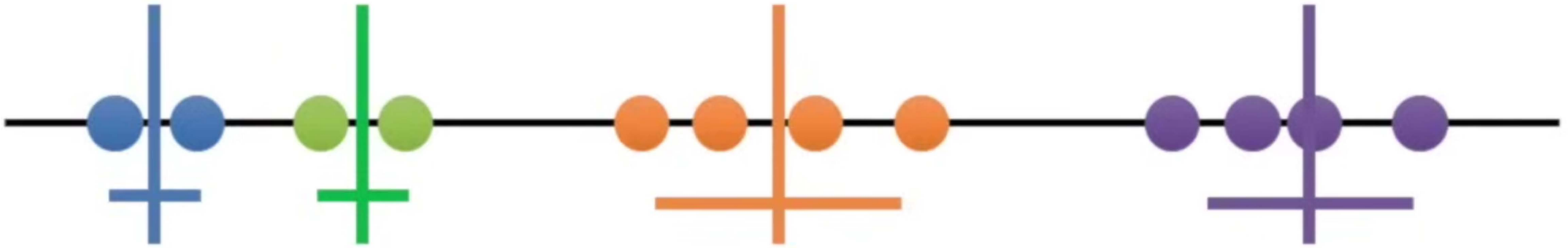
Start with K = 3



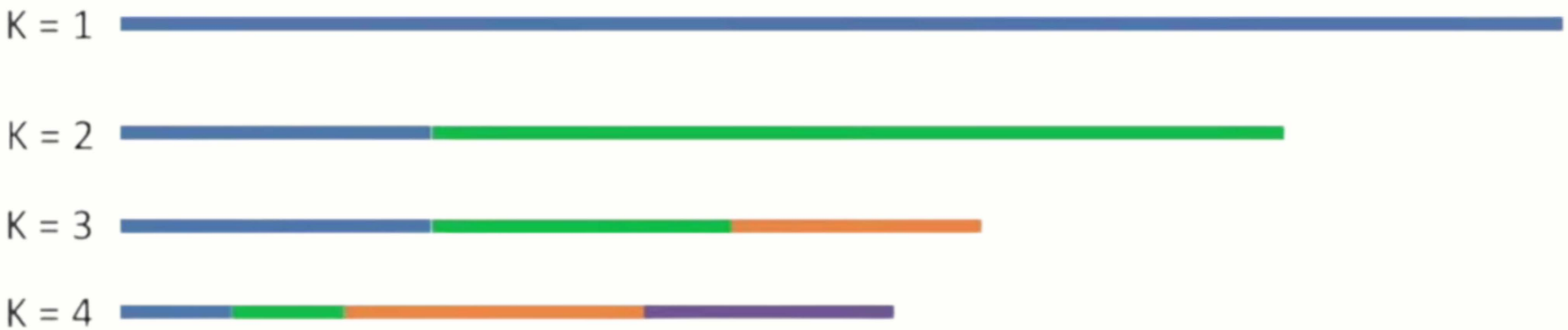
K = 3 is even better! We can quantify how much better by comparing the total variation within the 3 clusters to K = 2.



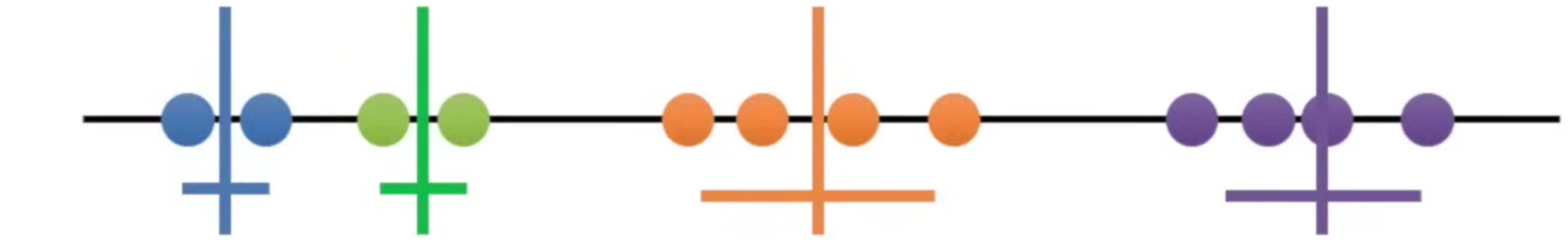
Start with K = 4



The total variation within each cluster is less than when K=3



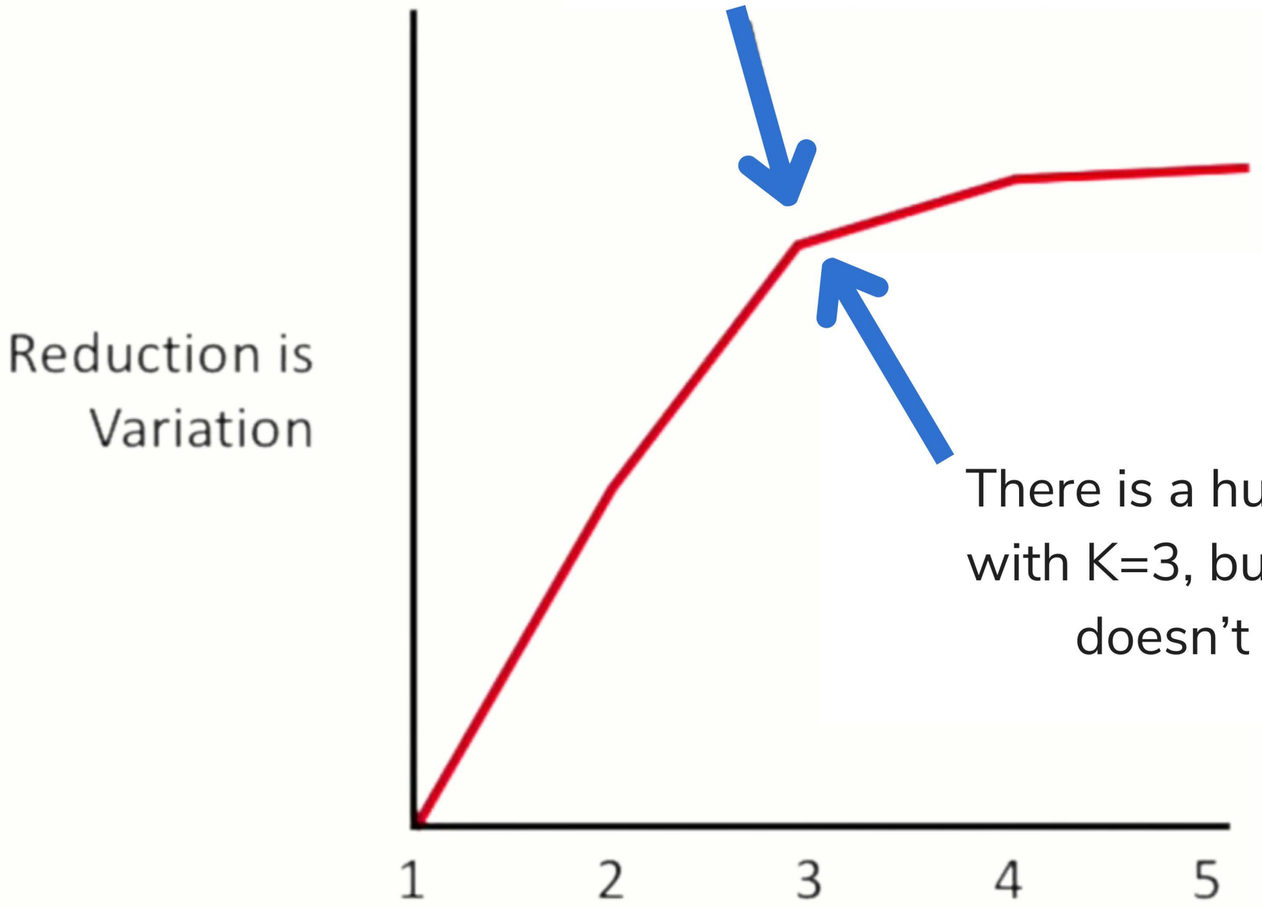
Start with K = 4



Each time we add a new cluster, the total variation within each cluster is smaller than before. And when there is only one point per cluster, the variation = 0.

However, if we plot the reduction in variance per value for K...

This is called an 'elbow plot', and you can pick 'K' by finding the 'elbow' in the plot.

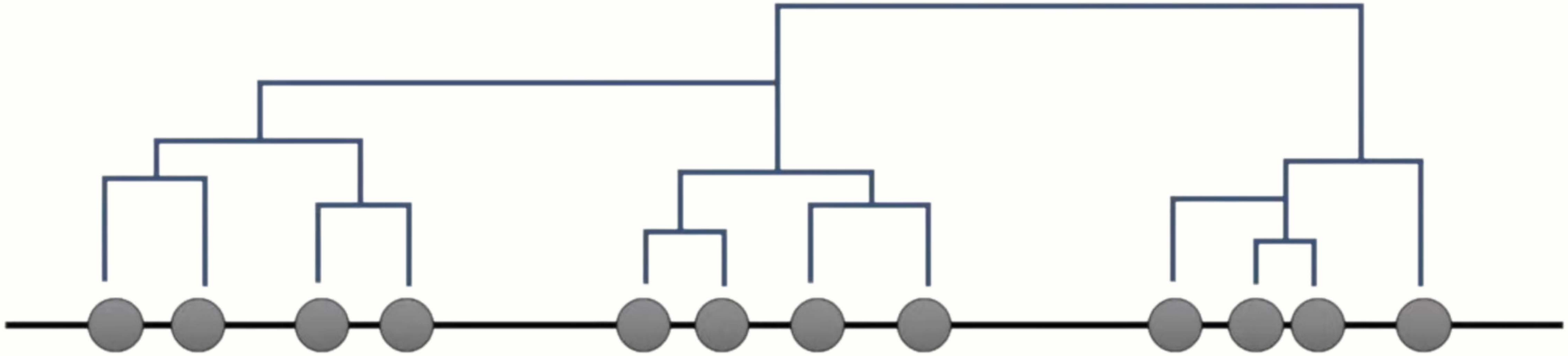


There is a huge reduction in variation with K=3, but after that, the variation doesn't go down as quickly.

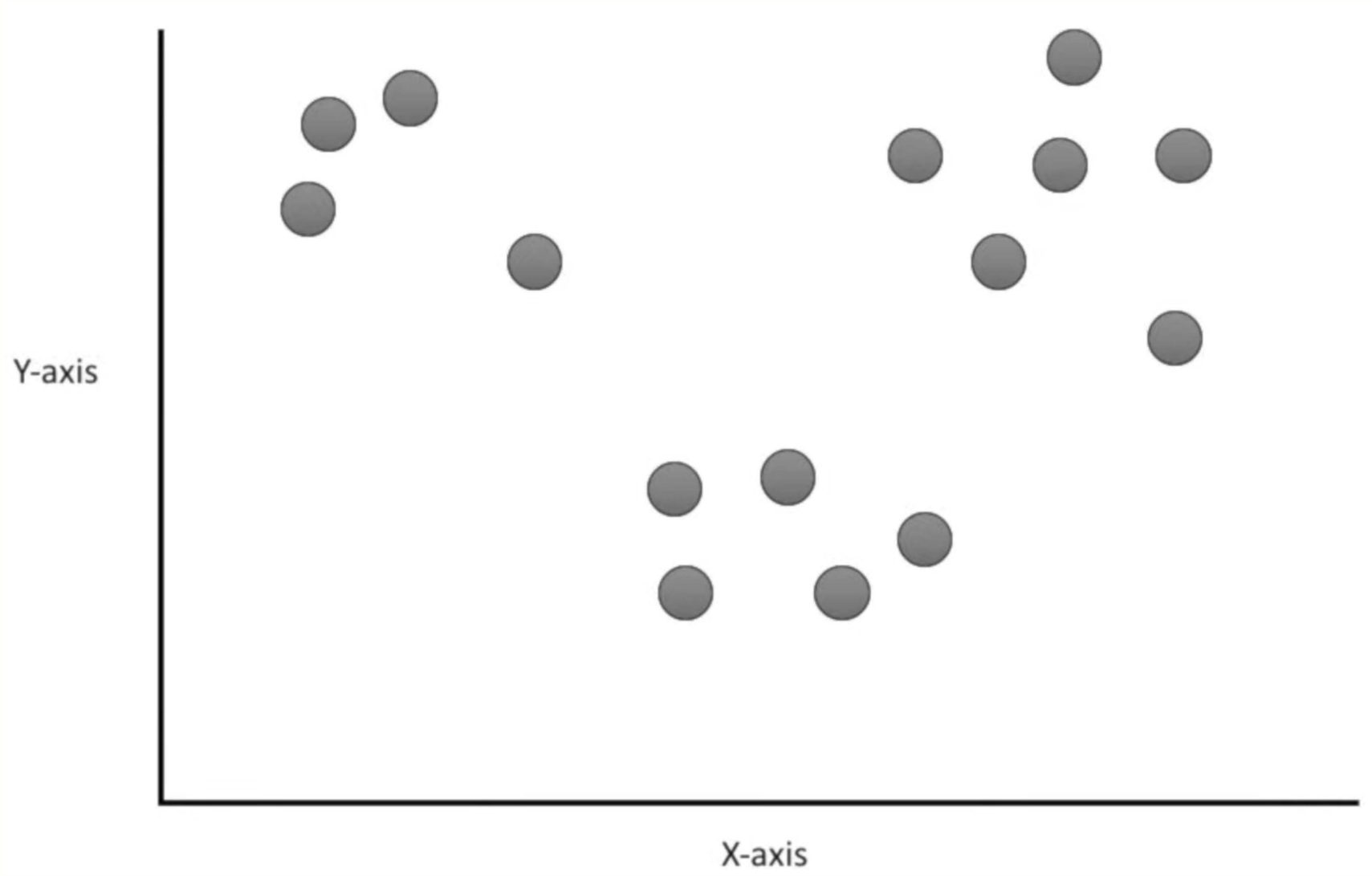
Question: How is K-means clustering different from hierarchical clustering?

K-means clustering specifically tries to put the data into the number of clusters you tell it to.

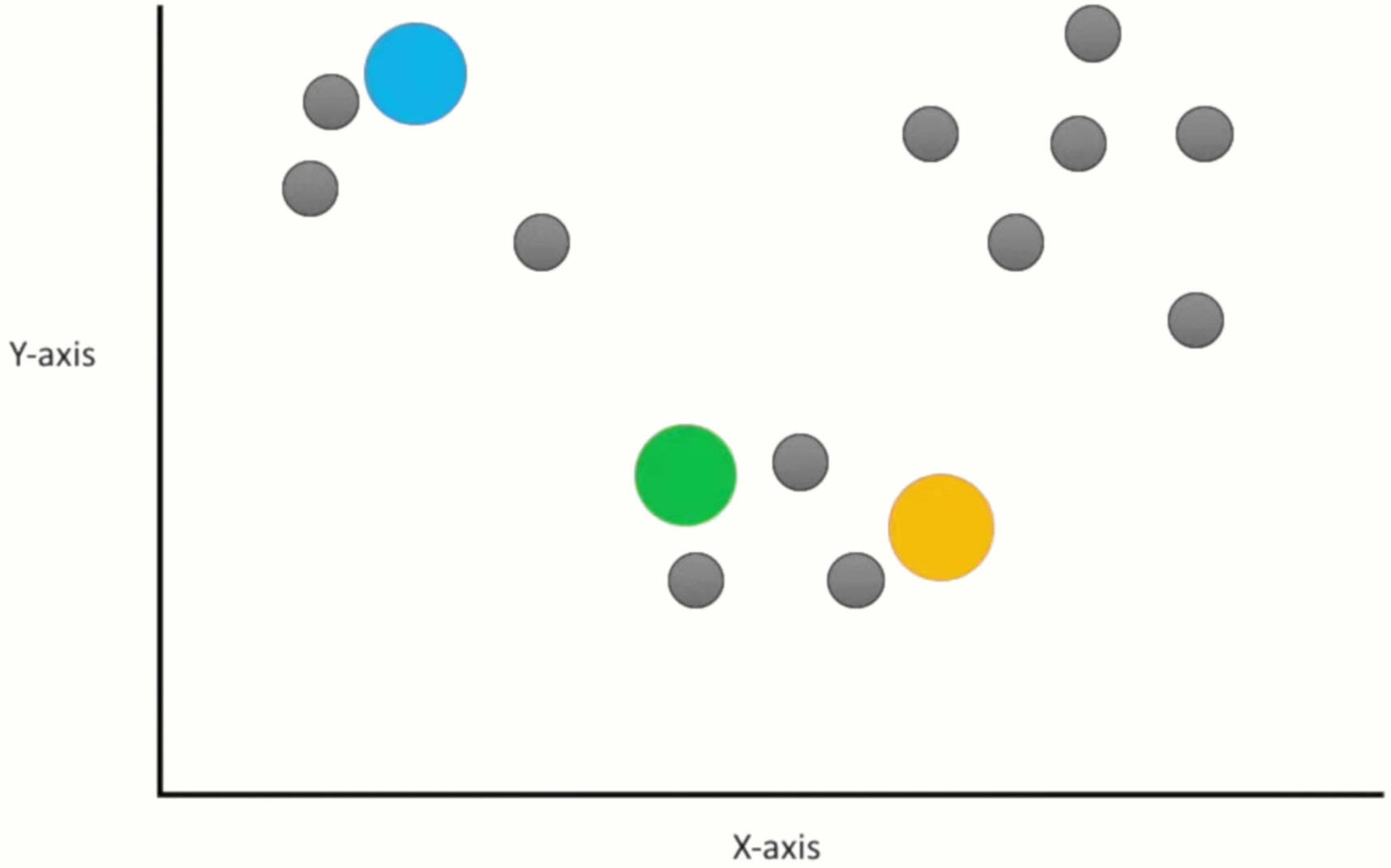
Hierarchical clustering just tells you, pairwise, what two things are most similar.



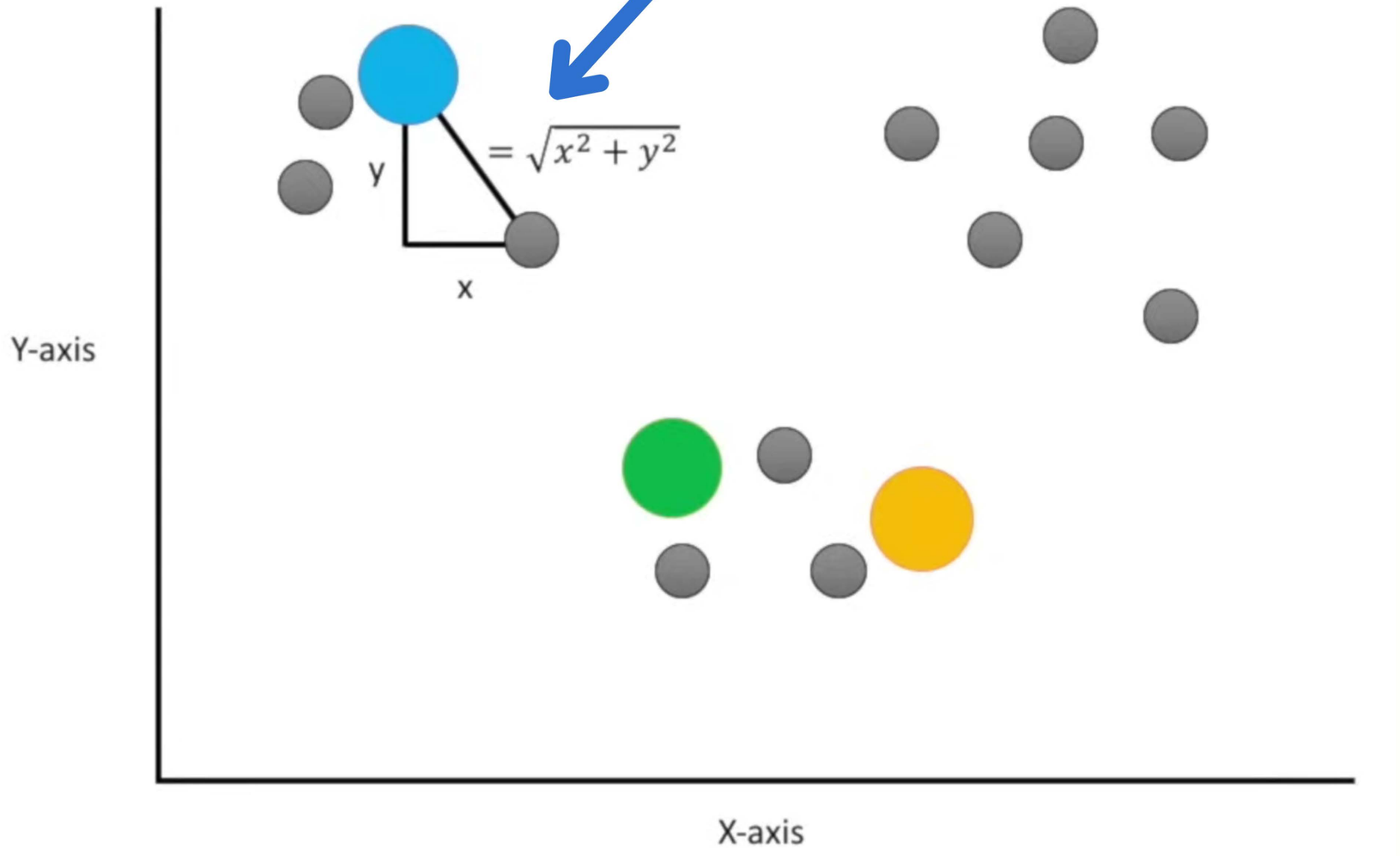
Question: What if our data isn't plotted on a number line?



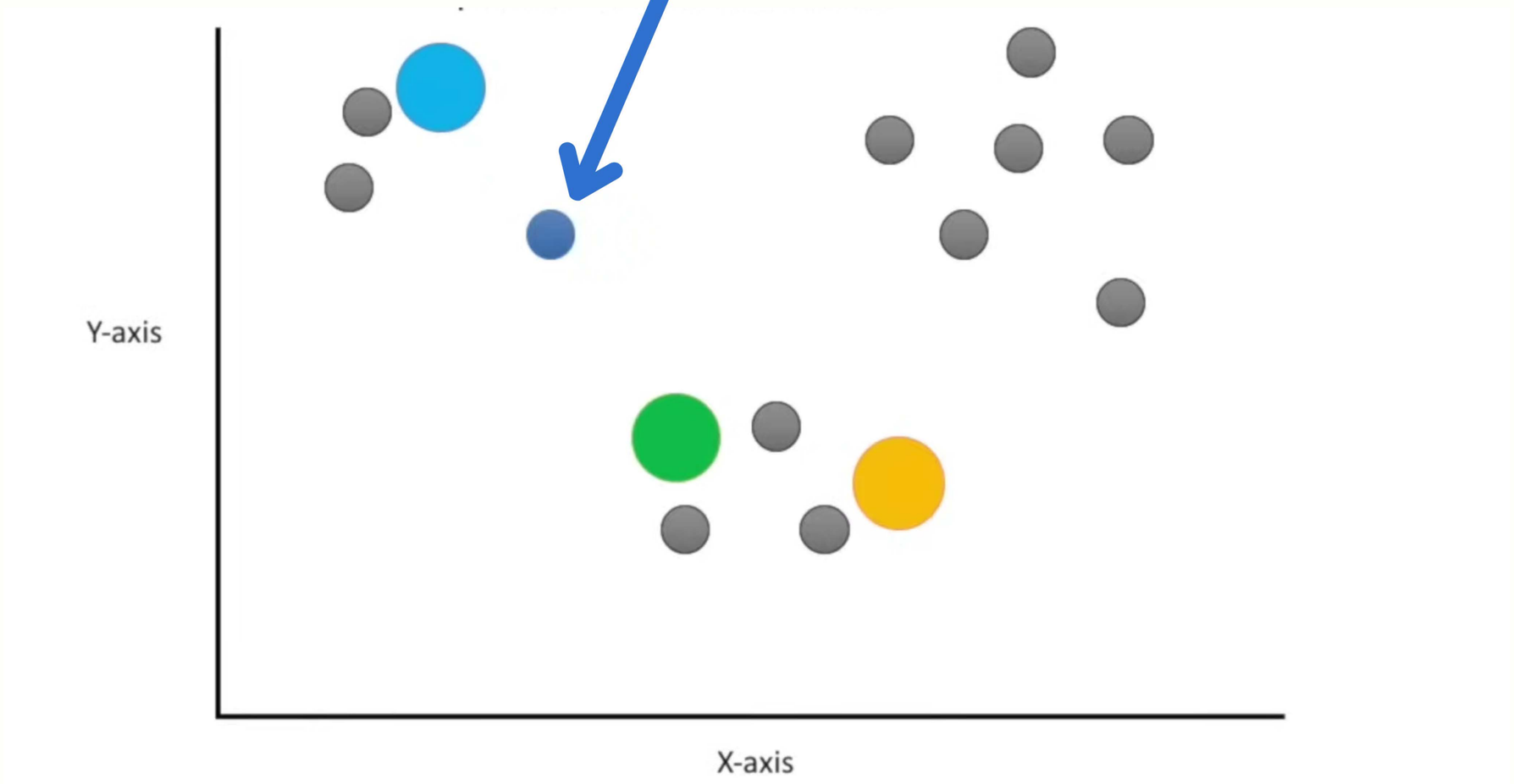
Just like before, you pick three random points...



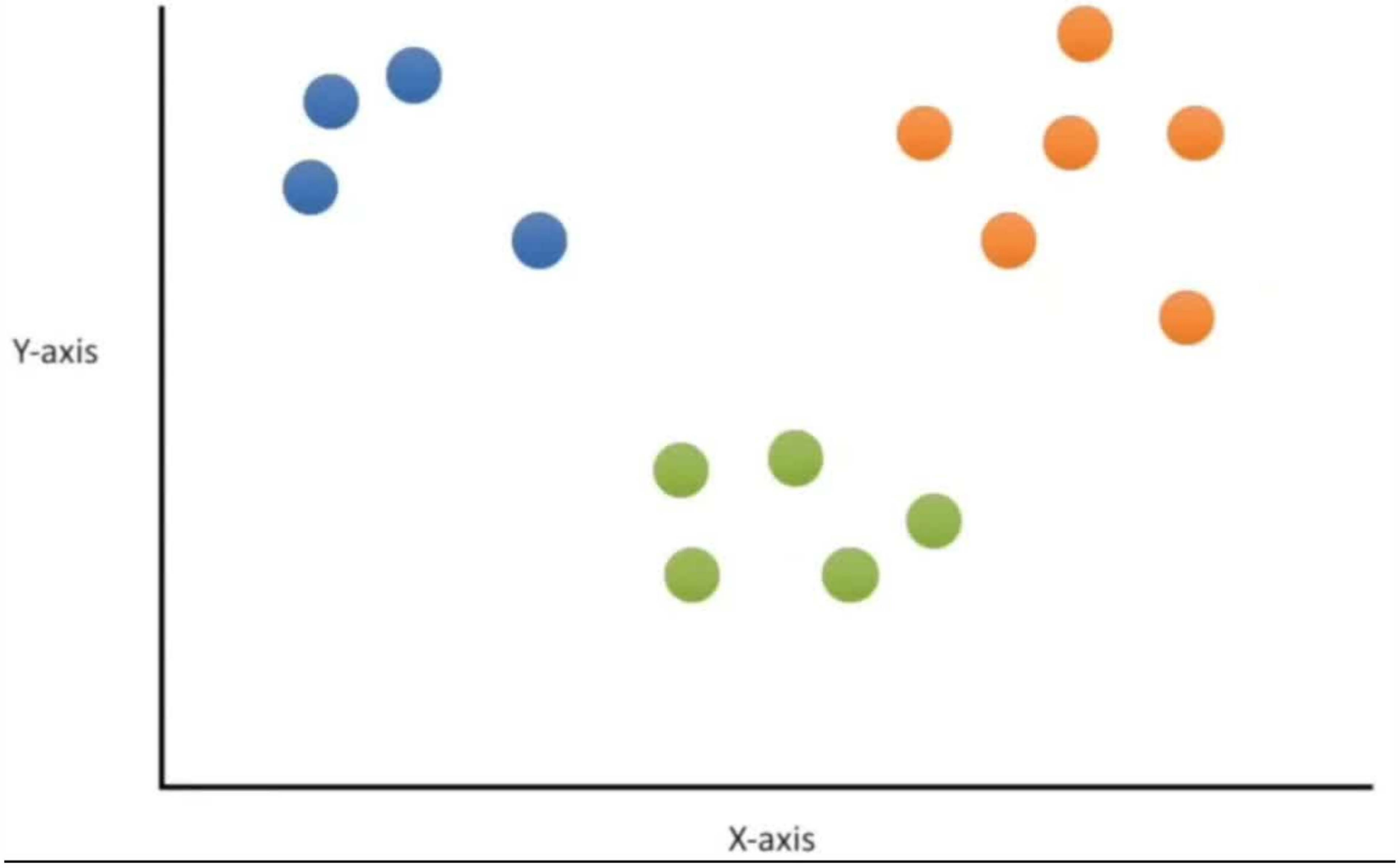
And we use the Euclidean distance. In 2 dimensions, the Euclidean distance is the same thing as the Pythagorean theorem.



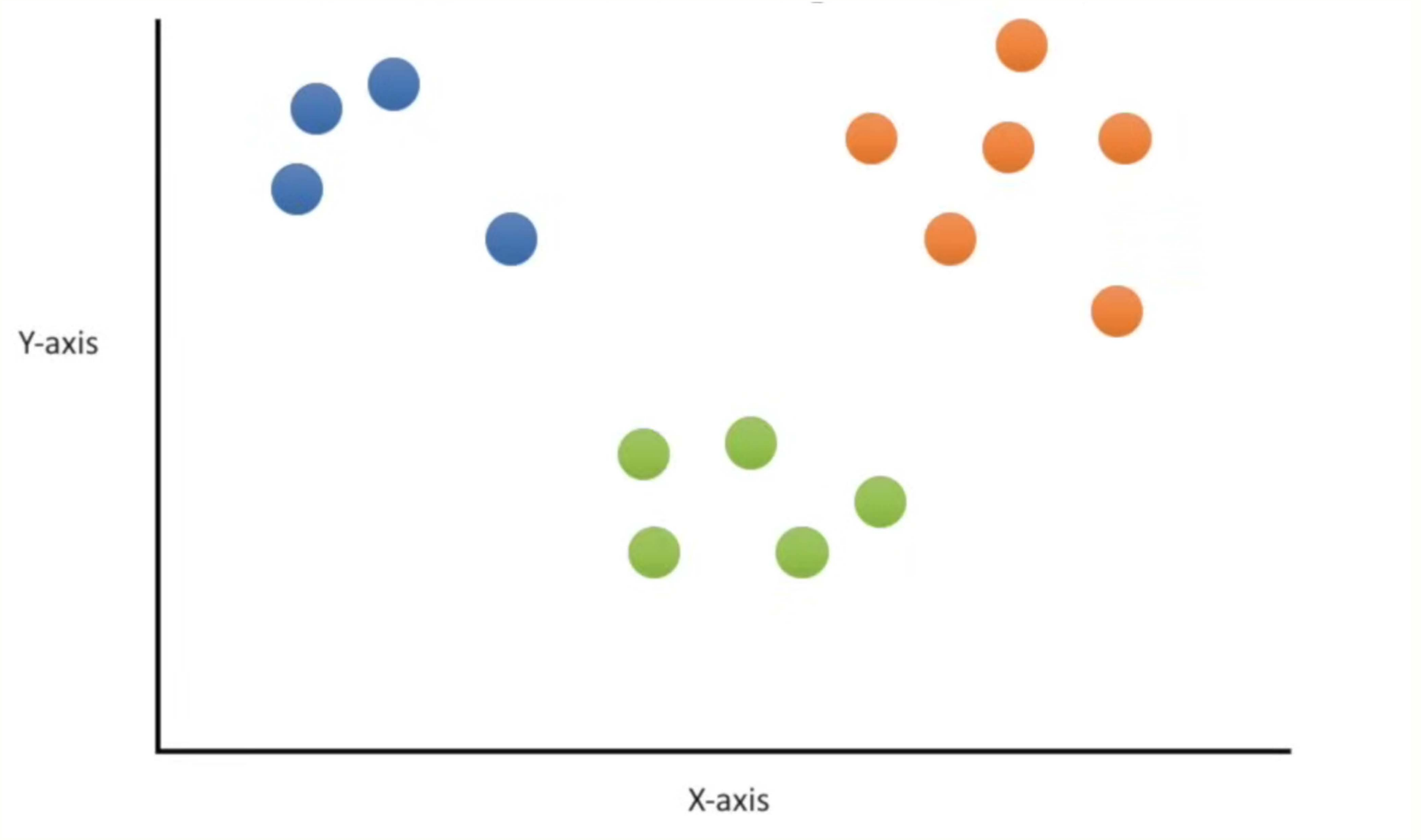
Then, just like before, we assign the point to the nearest cluster.



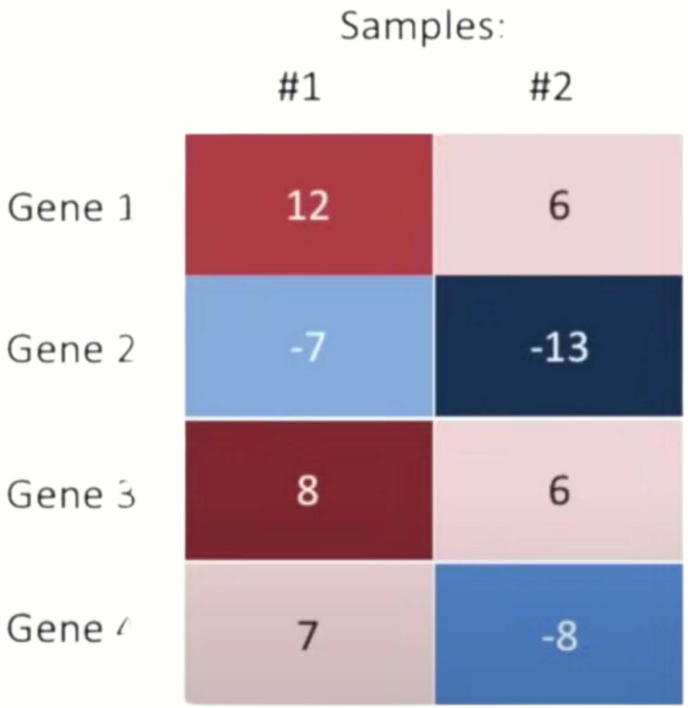
And, just like before, we then calculate the center of each cluster and recluster...



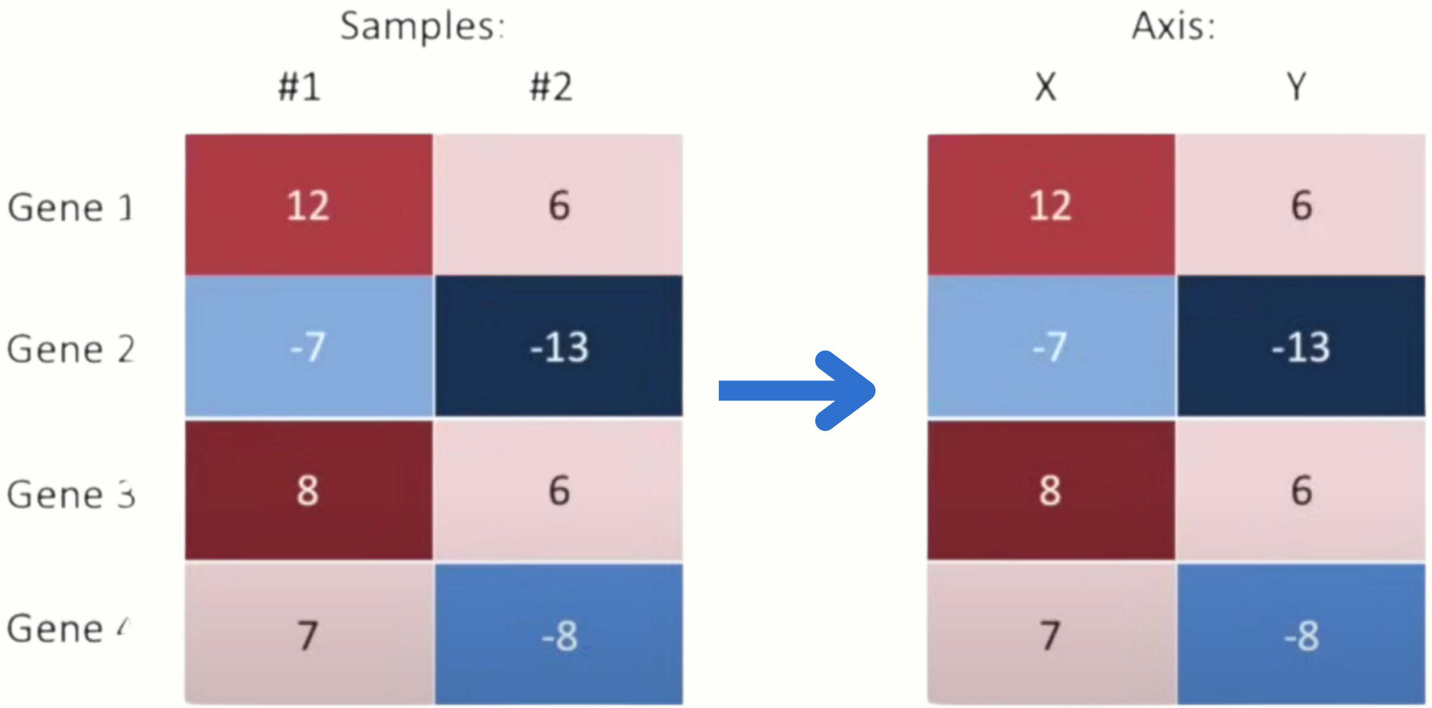
And, just like before, we then calculate the center of each cluster and recluster...



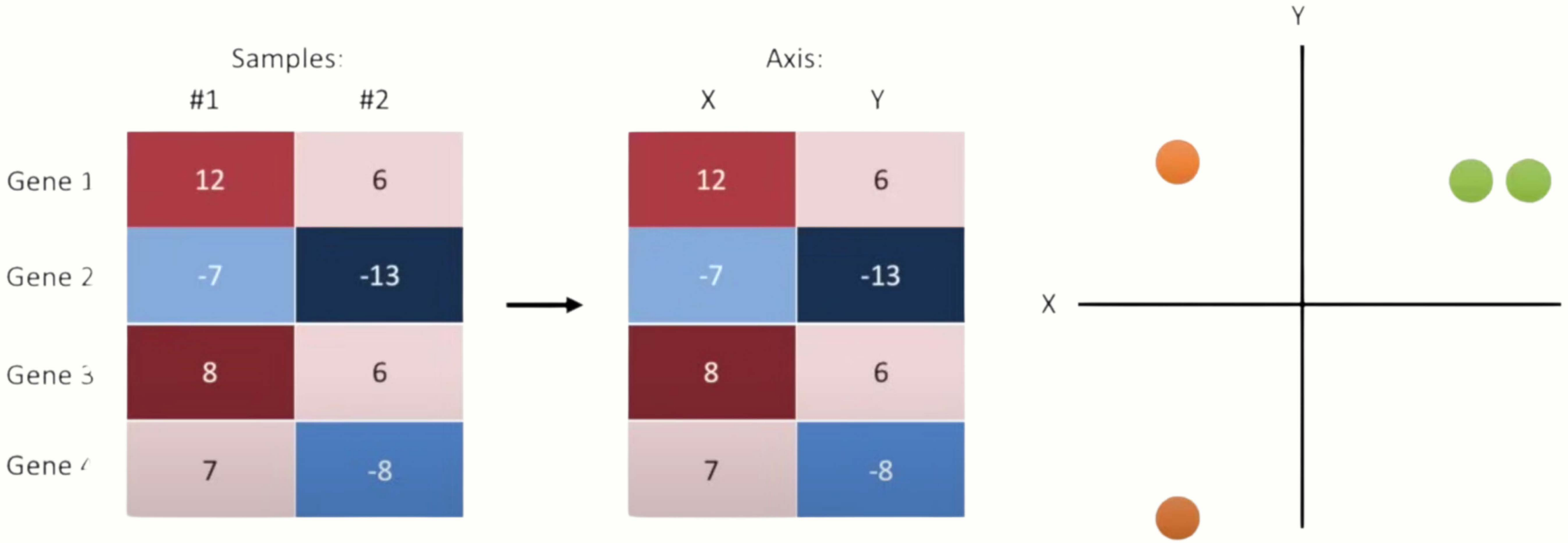
Question: What if my data is a heatmap?



Well, if we just have 2 samples, we can rename them 'X' and 'Y'



And we then plot the data in an X/Y graph

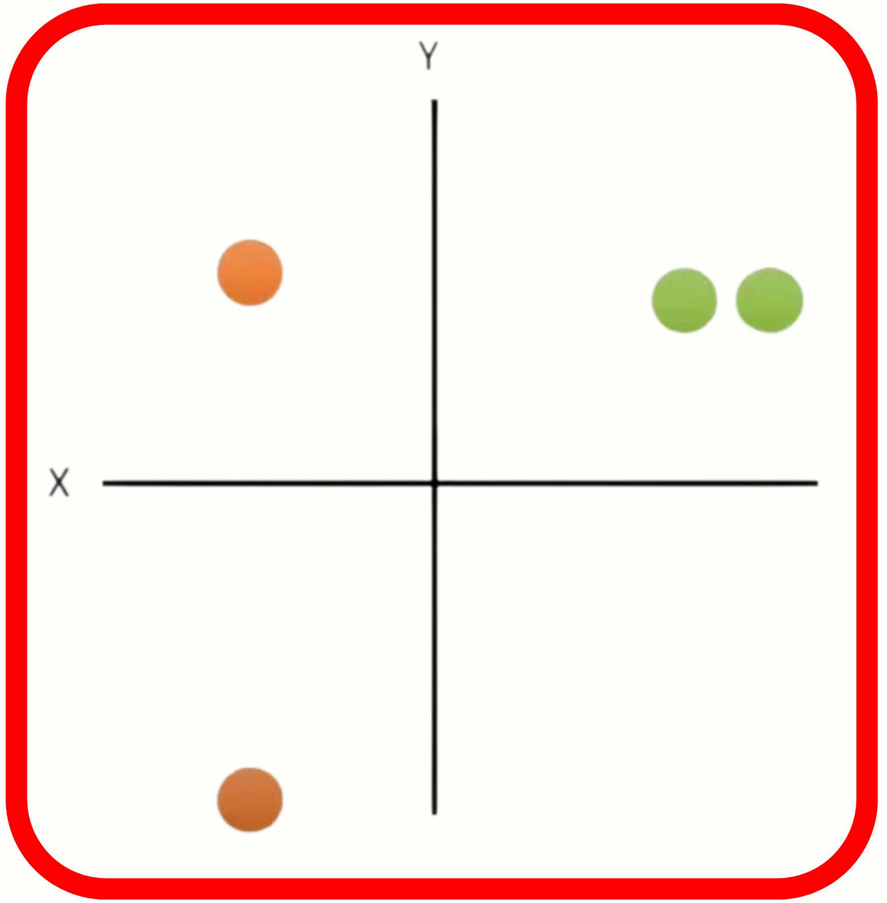


And we then plot the data in an X/Y graph

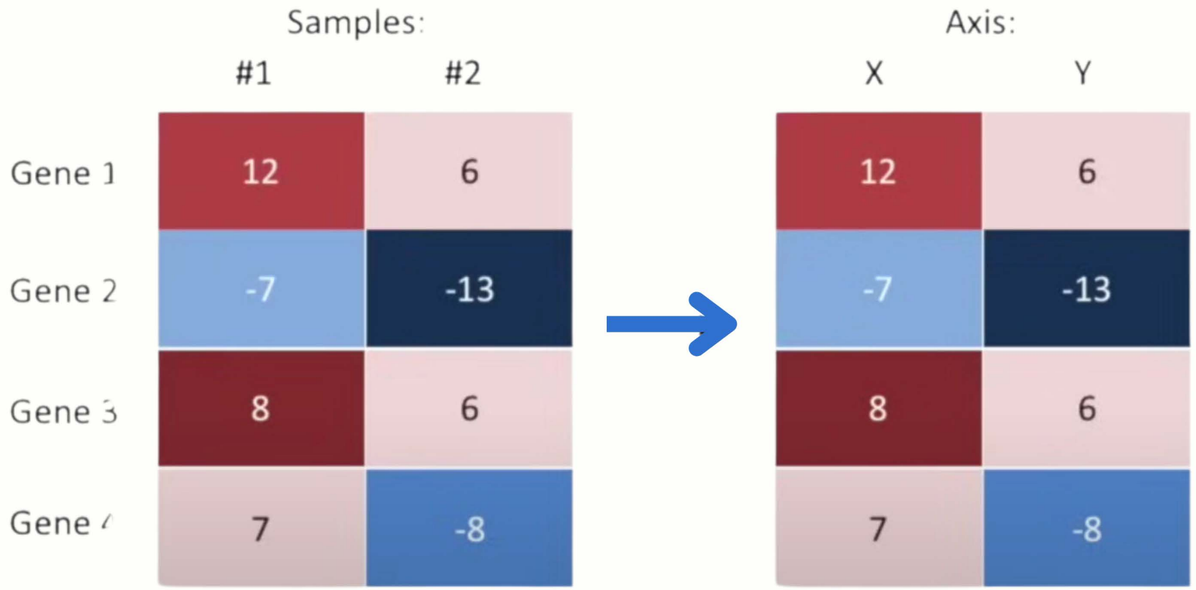
	Samples: #1	#2
Gene 1	12	6
Gene 2	-7	-13
Gene 3	8	6
Gene 4	7	-8

→

	Axis: X	Y
Gene 1	12	6
Gene 2	-7	-13
Gene 3	8	6
Gene 4	7	-8



Then we can cluster just like before!



Note: We don't actually need to plot the data in order to cluster it. We just need to calculate the distances between things.

When we have **2** samples, or **2** axes, the Euclidean distance is:

$$\sqrt{x^2 + y^2}$$

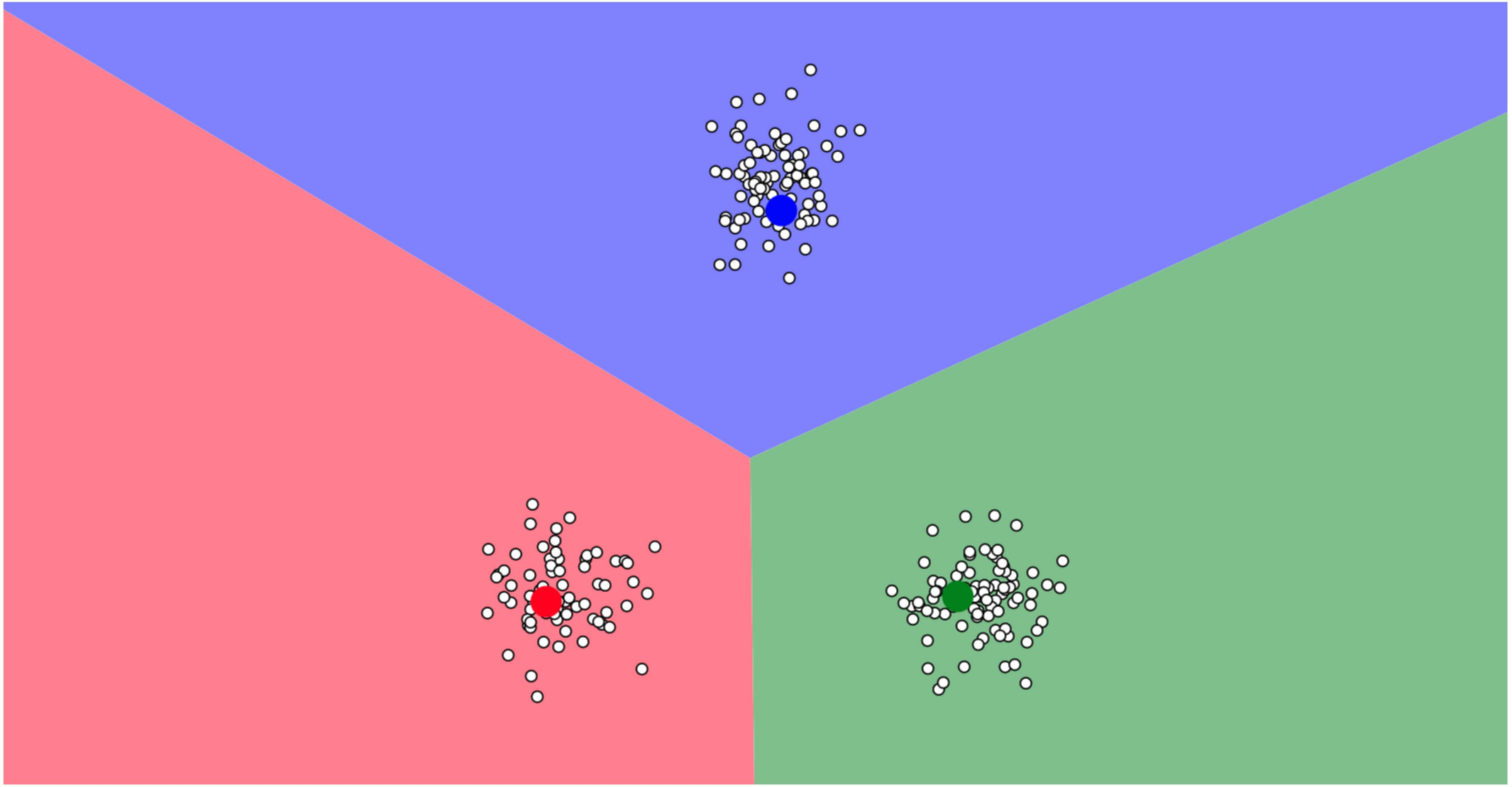
When we have **3** samples, or **3** axes, the Euclidean distance is:

$$\sqrt{x^2 + y^2 + z^2}$$

When we have **4** samples, or **4** axes, the Euclidean distance is:

$$\sqrt{x^2 + y^2 + z^2 + a^2}$$

LINK: HTTPS://WWW.NAFTALIHARRIS.COM/BLOG/VISUALIZING-K-MEANS-CLUSTERING/



C O D I N G T I M E ! ! ! ! ! !