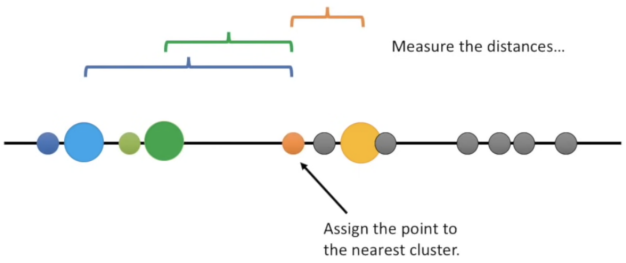
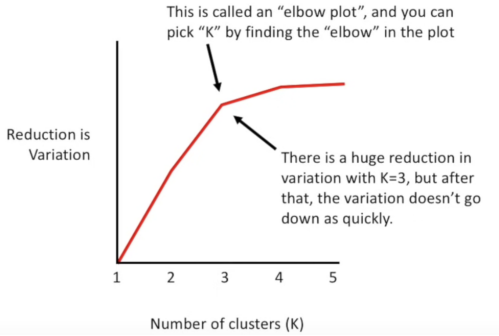
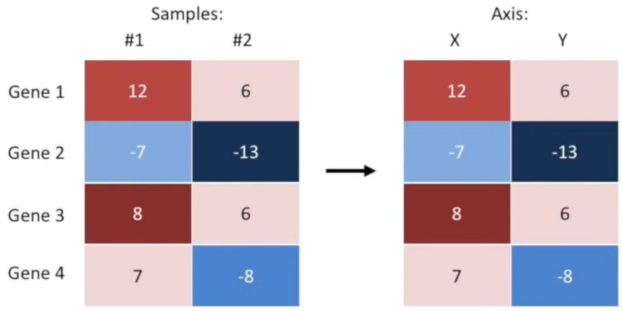


⇨ K-Means (StatQuest)

- Start with raw, unclustered data, and select the number of clusters you would like to identify in your data, K.
- Randomly select K distinct data points ⇨ the initial clusters
- Measure the distance between the first point and the three initial cluster centers
- Assign the first point to the nearest cluster.
- Do the same thing for the following data points, measuring and assigning
- Calculate the average of each cluster, then measure and cluster using the average values
- When no more changes occur, no reclustering, the clustering is done.
- The quality of the clustering is found by adding up the variance within each cluster
- The clustering process will repeat as many times as instructed, and the best clustering will be chosen.
- How to find the best value for K:
  - Try different values for K and evaluate resulting variation (choose K at elbow of curve)



- This varies from hierarchical clustering in that you tell the program K number of clusters rather than the program deciding.
- When data is plottable on a coordinate plane, the Euclidean (Pythagorean) distance is used
- Heatmap example ⇨



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:

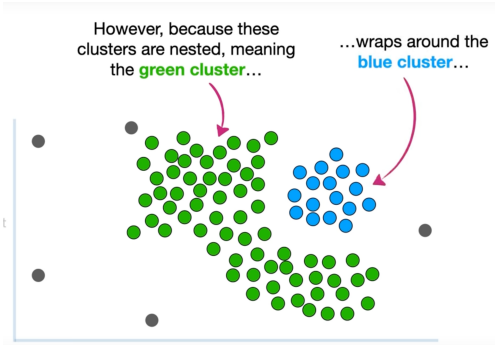
$$\sqrt{x^2 + y^2 + z^2}$$

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

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

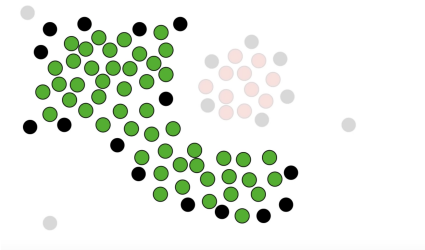
⇨ DBSCAN (StatQuest)

- Outperforms K-Means when data is nested
- Good at identifying clusters in data with many dimensions
- It identifies like we do by eye, finding high-density regions as clusters and outliers in the low-density regions
- The user must experiment with the radius.
- The overlap within the radius can be partial
- A core point is one that is close to at least n other points, defined by user.



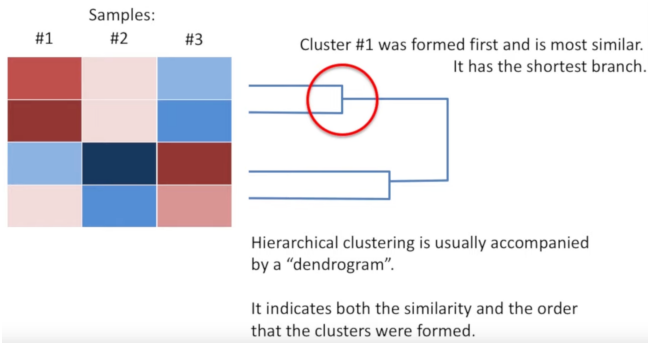
- Points that can originally be called core points are distinguished from outliers, or those that cannot be. The remaining are considered non-core.
- Next step: randomly choose a core point, and all points within the radius belong to that cluster
- The core points that are within the first core point's cluster join it and extend it to other close-by core points, at first ONLY adding points which have been labeled as core points.

- Once all core points join the first cluster, any non-core points that are close to clustered core points will be added to the cluster.
- Non-core points can only join a cluster and cannot extend it further
- Clusters are created sequentially, meaning a point that may be close to two resulting clusters becomes ineligible for clustering once an early cluster claims it.



⇨ Hierarchical Clustering (StatQuest)

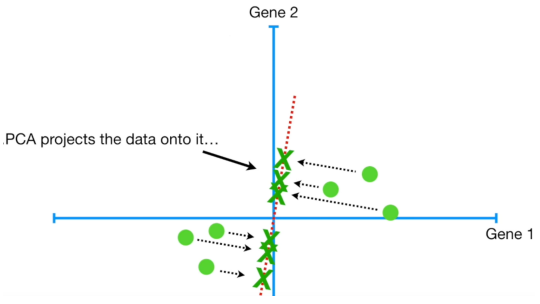
- Often associated with heatmaps, where the columns typically represent different samples (data points), and the rows represent the features, and colors represent the measurements. (Note: rows and columns are usually switched from this configuration.
- Orders the rows and / or the columns based on similarity, making correlations in the data easy to see.
- First step: figure out similarities of the rows given the columns, and group them accordingly, with a hierarchy based on how much they correlate to one another.
    - This process goes by one data point at a time, looking for correlations with others.
    - Once all data points have been considered, the ones that are most similar get merged into a cluster.
    - The first step repeats, now treating the clustered data as one and comparing it to others, clustering.
    - **Dendrograms**: represent both the similarity and the order in which clusters were formed. The height of the branches show what is most similar (short = more similarity, long = less.)
    - Method for determining similarity is arbitrary, but Euclidean distance is often used.



- The Manhattan distance is also used, the sum of all the differences in distances.
- Ways to compare clusters when incorporating further data into them:
    - Compare new data to the average of the cluster, the centroid
    - Compare to the nearest point in the cluster, called single-linkage
    - Compare to the furthest point in the clusters, complete-linkage

⇨ Principal Component Analysis (StatQuest)

- **SVD** = singular value decomposition ⇨ measurements scaled to 1, but the ratio remains the same
- Each feature considered against the data represents an additional axis on "the graph"
- PCA can take 4 or more dimensions of data and make a 2D plot
- PCA can also tell us what the most valuable feature/label is for clustering the data
- It can also tell us how accurate the 2D graphing is
- With the average values for the data, the center can be calculated
- The x and y axes are shifted so that the origin is in the center of all the data
    - This does not affect the relative position of the data points to one another
- With the data centered on the origin, we try to fit a line to it, random but running through the origin
- The line is rotated until it fits the data as well as it possibly can, while still going through the origin
- How does PCA decide if the line fits the data well:
    - It projects the data onto it and measures the distances of the data from the line, either minimizing the distances or maximizing the distances to the origin from the projected points



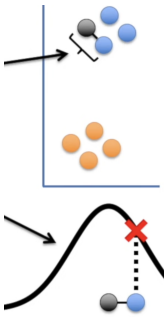
↳ These two measurements work together to come out the same (Pythagorean)
↳ PCA actually performs the latter operation; the first is easier to comprehend

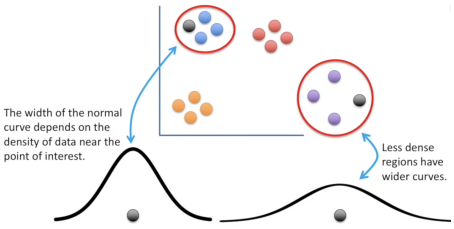
- Then it sums all the squared distances as it rotates, repeating until it finds the largest sum of squared distances between the projected points and the origin.
- The slope of the resulting line represents the correlation between two of the variables. This ratio tells how important the variables are in describing how the data are spread out (if the rise is higher than the run, it is more influential.)
- This is called a **linear combination** between variables

- The 1 unit scaled vector consisting of the specific ratio of variable to variable is called the **singular vector**, or the **Eigenvector** for the line (PC1, when it is the first line worked out by PCA.)
- The proportions of each gene are called **loading scores**
- The sum of the squares of the distances for the best fit version of a line is called its **Eigenvalue**.
- The square root of the Eigenvalue is called the **singular value**
- **PC1** is the first line calculated on the data, for one variable. **PC2**, if it is only 2D data are simply the line that runs perpendicular to PC1 through the origin, with no further optimization.
- These relationships tell us how important a feature is overall and how important they are in comparison to one another.
- To plot the final PCA plot, we just rotate the graph so that PC1 is horizontal, becoming the x axis.
- Then the projected points on the axes are put back into their respective positions on the graph
- **Variation for a PC** is the sum of distances of data to origin divided by the total samples minus 1. This gives the percentage of total variation around the PCs that that particular PC accounts for
- **Scree Plot** = a graphical representation of the percentages of variation that each PC accounts for
- When there are more than 2 dimensions, it follows the same procedure just with more principal components represented by lines that are rotated until best fit is achieved.
- Once all the PCs are found, the scree plot of their percentages of variation (Eigenvalues) can be used to filter out the less influential variables
- The less important PCs can be eliminated, and the points can be projected onto the influential PCs, rotated horizontally again, and then put back in their places on the coordinate plane.
- If the scree plot has a more evenly distributed level of influence among the PCs, then eliminating the lower ones can result in inaccuracy
  - PCA can still be used to find clusters of data.

⇨ t-SNE and T-Distributed Stochastic Neighbor Embedding

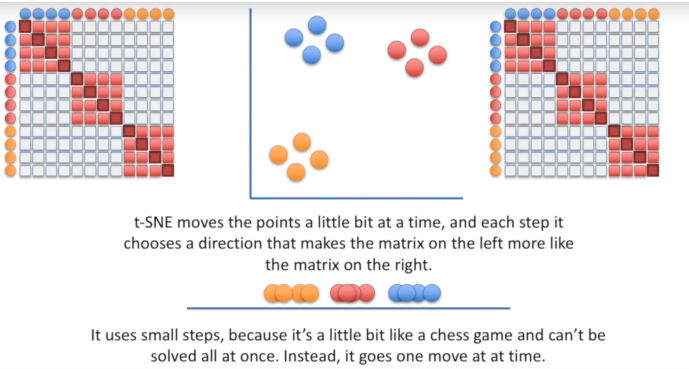
- Takes a high dimensional dataset and reduces it to a low dimensional graph that retains a lot of the original information
- The data cannot be projected onto just one of the axes, because it will overlap and lose accuracy
- t-SNE finds a way to project the data onto a 1-dimensional line in such a way that it preserves the multi-dimensional information
- Take original multi-dimensional data, and project them onto a line in a random order
- Then moves the points a little bit at a time until it has clustered them according to the multidimensional relationships between the points
  - Similar points attract (similarity is based on original data), while different points repel
  - The movement is based on what is stronger: attraction vs repulsion
- More Detail ⇨ FIrst: Determine the similarity of all the data points by measuring the distance between them
  - Then plot the distance on a normal distribution curve centered on the data point of interest
  - Calculate the unscaled similarity between the point in question and other points to see which it should be clustered with.

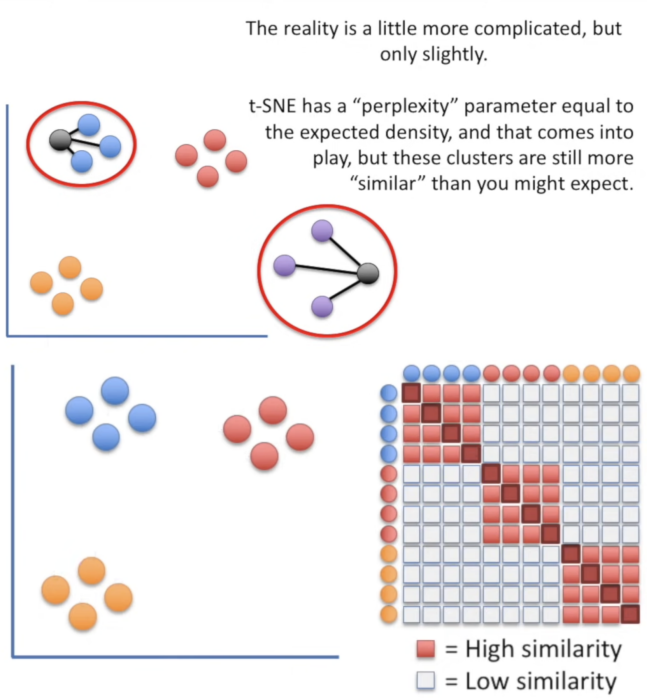


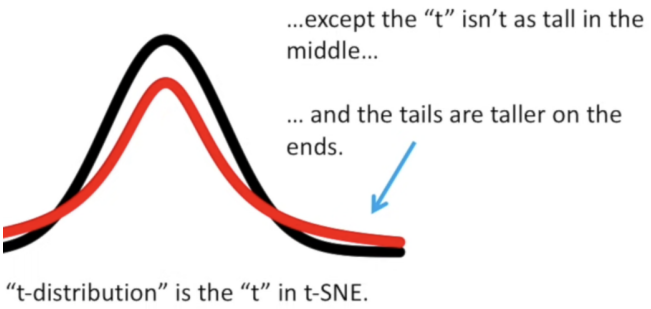


- Scale data distances to equal 1 ⇨ divide score by the sum of all scores = scaled score
- This makes the relative correlation between points in a cluster more easily comparable.
- The width of the distribution is based on the density of the surrounding data points.
- You end up with a matrix of similarity scores, each row and column representing the similarity scores calculated from the point of interest.

- Next, randomly project the data onto a number line and calculate similarity scores for the points on the numberline, as we did before picking a point, measuring the distance, and then connecting the point to a curve, but this time with a **t-distribution**
- We follow relatively similar steps to the steps before.
- It moves the points a little bit at a time.







- Without the t-distribution, all the data points would bunch up in the middle.