PCA using SVD

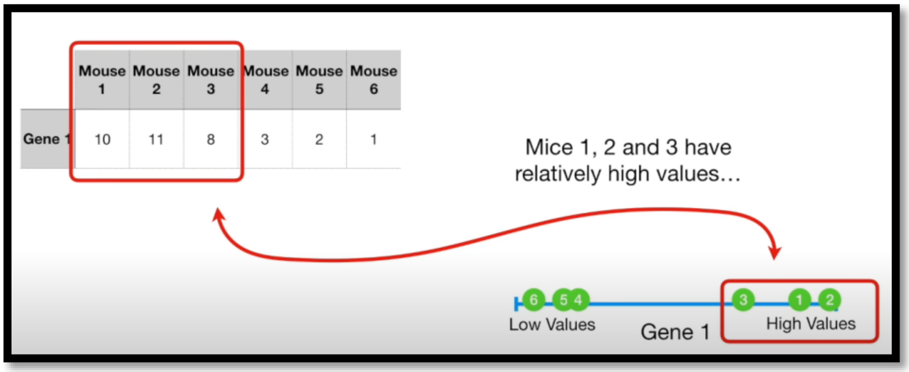
Table of Content

Motivation

Let’s consider this dataset

	Gene1	Gene2
Mouse 1	10	6
Mouse 2	11	4
Mouse 3	8	5
Mouse 4	3	3
Mouse 5	2	2.8
Mouse 6	1	1

If we only measure **one gene**, we can plot the data on **a number line**

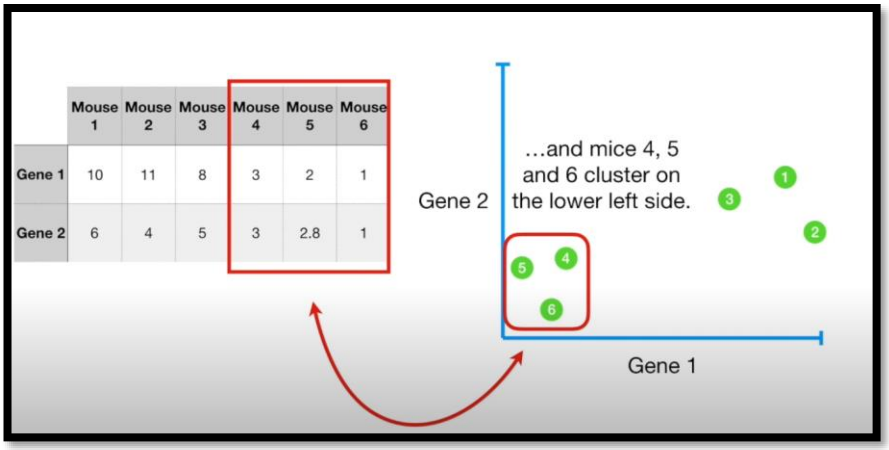


	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1

Mice 1, 2 and 3 have relatively high values…

Although it is a simple graph, but it shows that mice 1, 2, and 3 are more similar to each other than they are to mice 4, 5, and 6

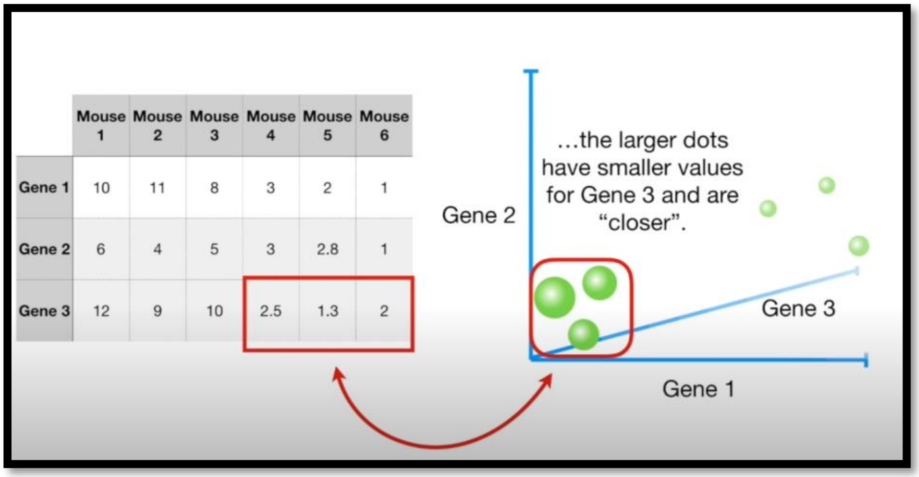
If we measure **two genes**, we can plot the data on **a 2D x/y graph**



	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1

…and mice 4, 5 and 6 cluster on the lower left side.

If we measure **three genes**, we would add another axis to the graph (3D)



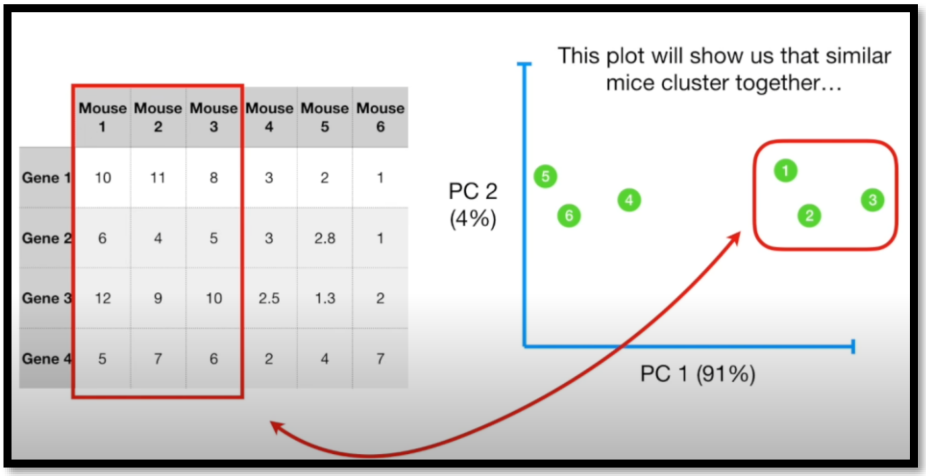
	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1
Gene 3	12	9	10	2.5	1.3	2

…the larger dots have smaller values for Gene 3 and are “closer”.

Purpose of PCA

If we measure **four genes**, we can no longer plot the data because it requires **4D** graph. therefore, we are going to talk about how PCA can take 4 or more gene measurements and make a 2D plot.

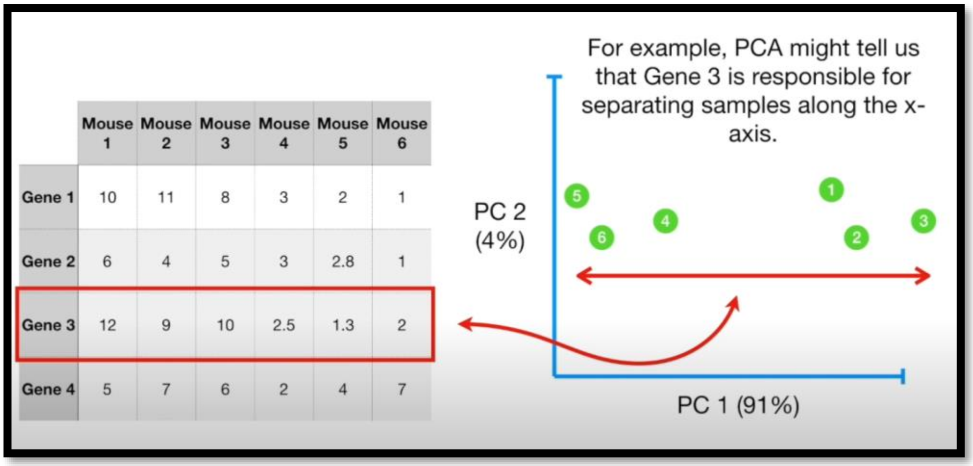
The purpose of PCA is to reduce the dimensionality of the data while still show us that similar mice cluster together!



	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1
Gene 3	12	9	10	2.5	1.3	2
Gene 4	5	7	6	2	4	7

This plot will show us that similar mice cluster together…

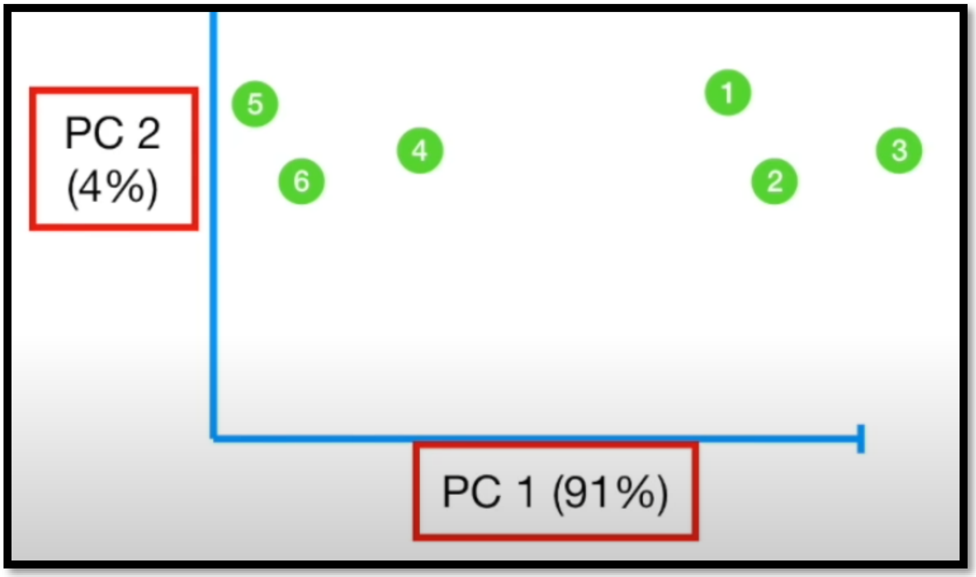
We will also talk about how PCA can tell us which gene (feature) is the most valuable for clustering the data. For example, PCA might tell us that gene 3 is responsible for separating samples along the x-axis.



	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1
Gene 3	12	9	10	2.5	1.3	2
Gene 4	5	7	6	2	4	7

For example, PCA might tell us that Gene 3 is responsible for separating samples along the x-axis.

Lastly, we will talk about how PCA can tell us how accurate the 2D graph is

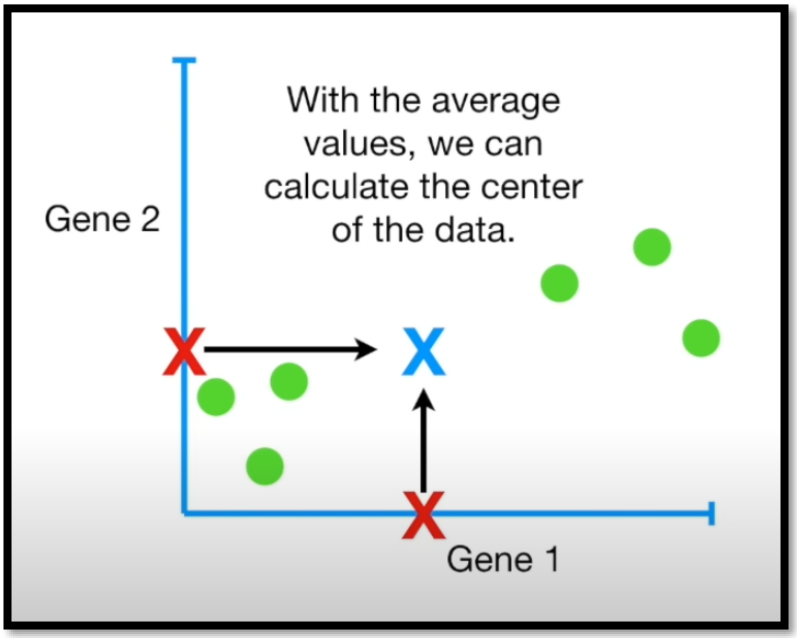


PCA for 2D

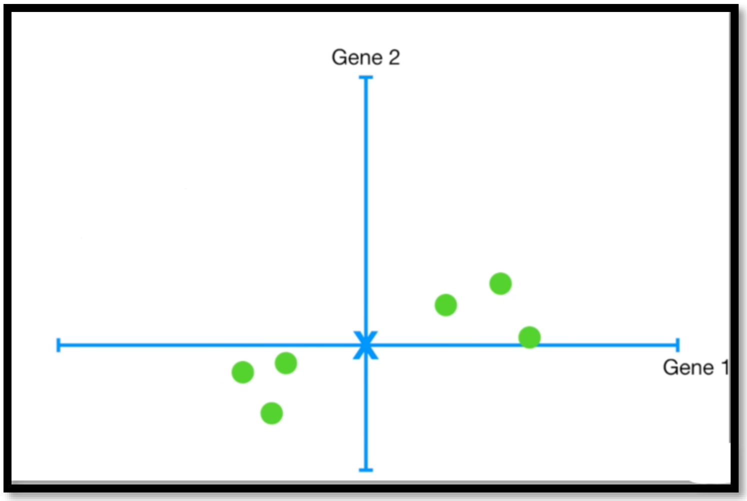
To understand what PCA does and how it works, let’s go back to the original dataset.

	Gene1	Gene2
Mouse 1	10	6
Mouse 2	11	4
Mouse 3	8	5
Mouse 4	3	3
Mouse 5	2	2.8
Mouse 6	1	1

We will start by plotting the data, then we’ll calculate the average measurement for Gene 1 and the average measurement for Gene 2. With the average values we can calculate the center of the data.



Now we will shift the data so that the center is on top of the origin in the graph.

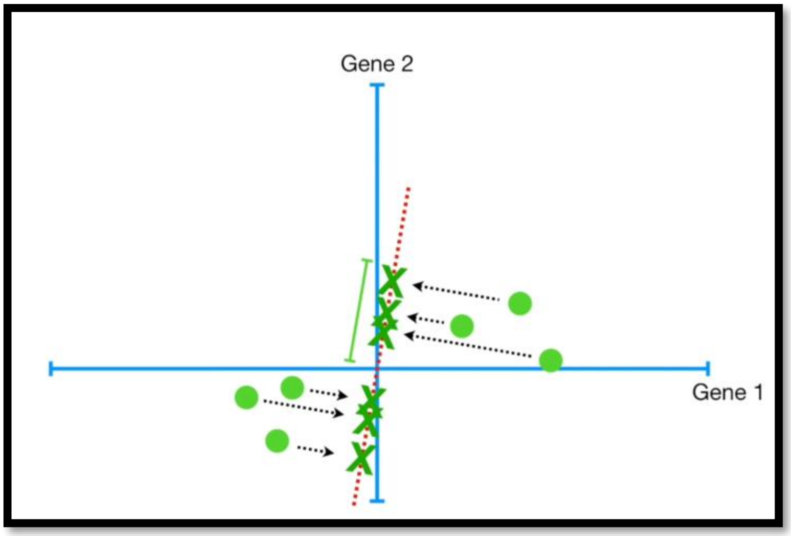


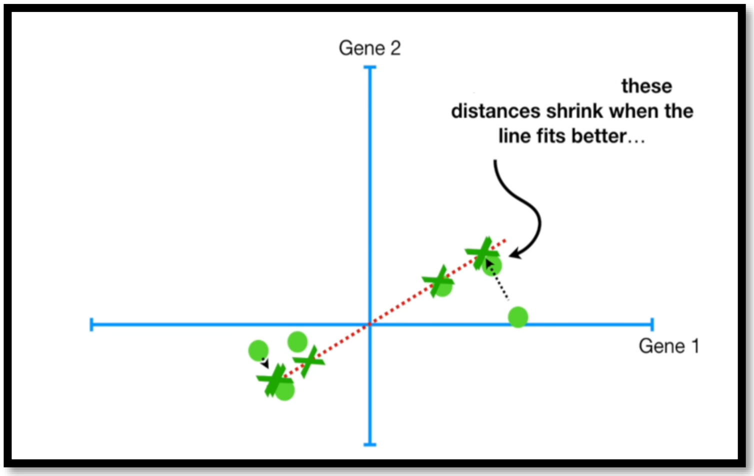
Now we will try to fit a line to it. To do this we start by drawing a random line the goes through the origin, then we rotate the line until it fits the data as well as it can, given that it has to go through the origin.

<u>How PCA decides what is the best line?</u>

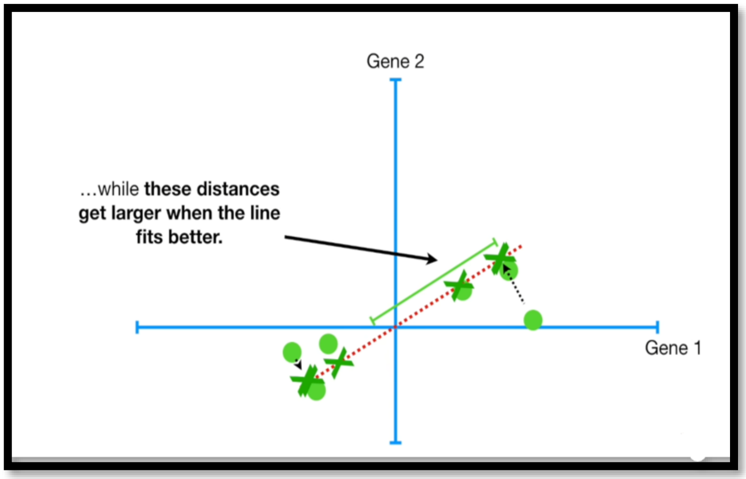
PCA **project** the data point onto the line and then it can either:

- Measure the distance from the data to the line and try to find the line that minimizes those distances.
- Or it can try to find the line that maximizes the distance from the **projected** points to the origin.



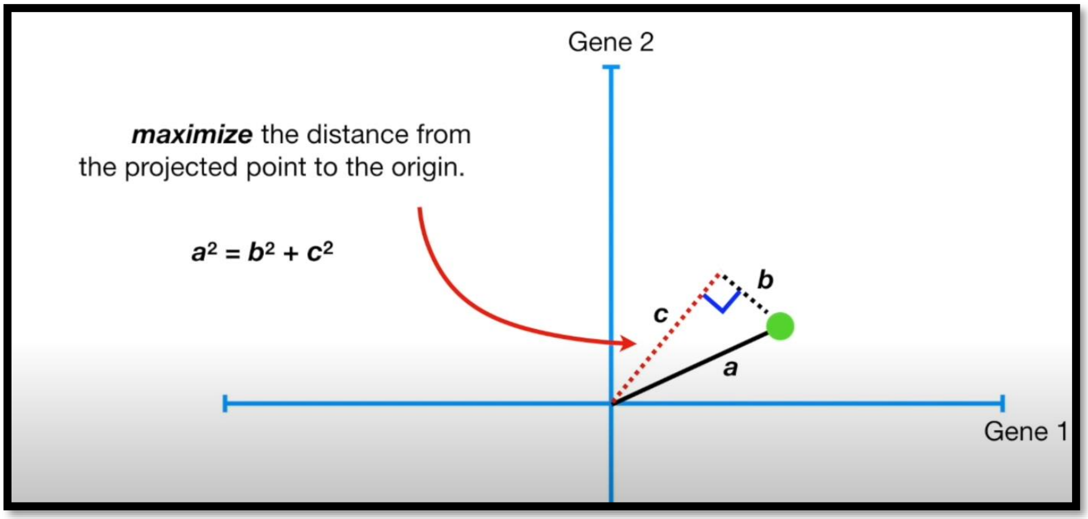


these distances shrink when the line fits better…



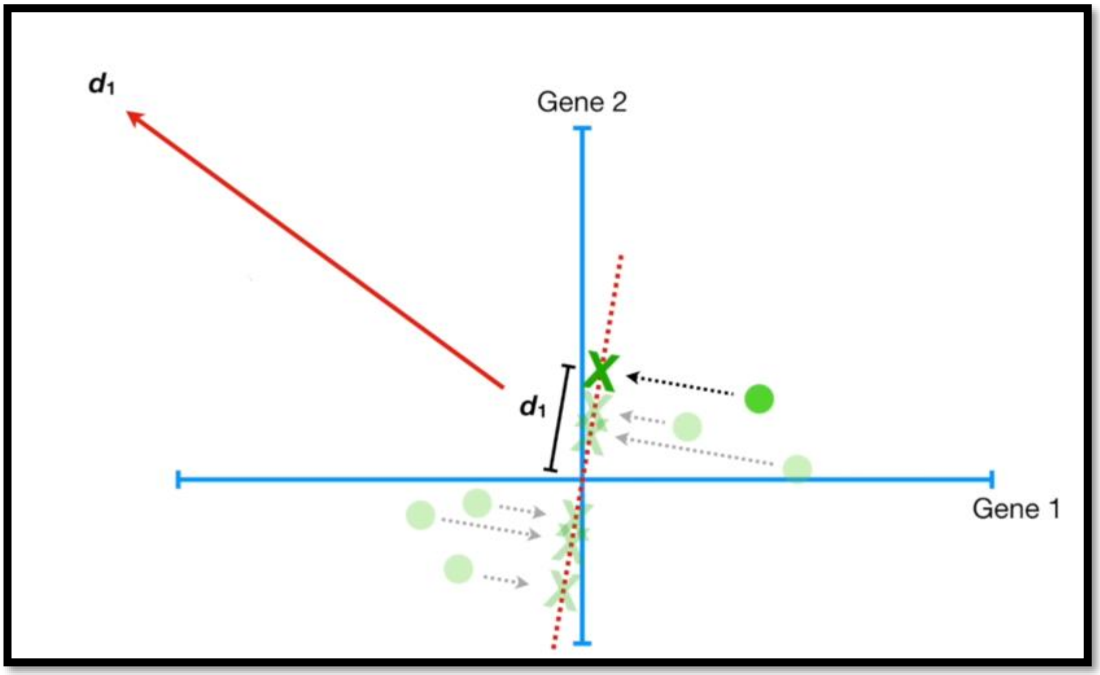
…while **these distances get larger when the line fits better.**

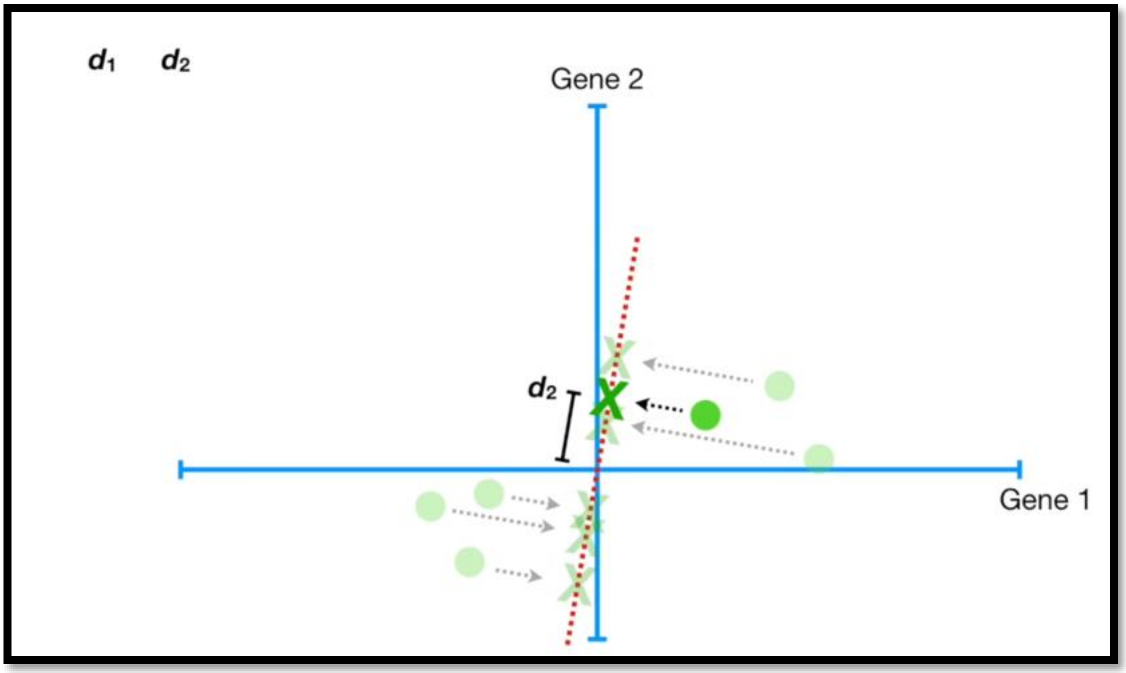
It is actually easier to calculate the distance from the projected point to the origin (c), so PCA finds the best fitting line by maximizing the sum of the squared distances from projected points to the origin.

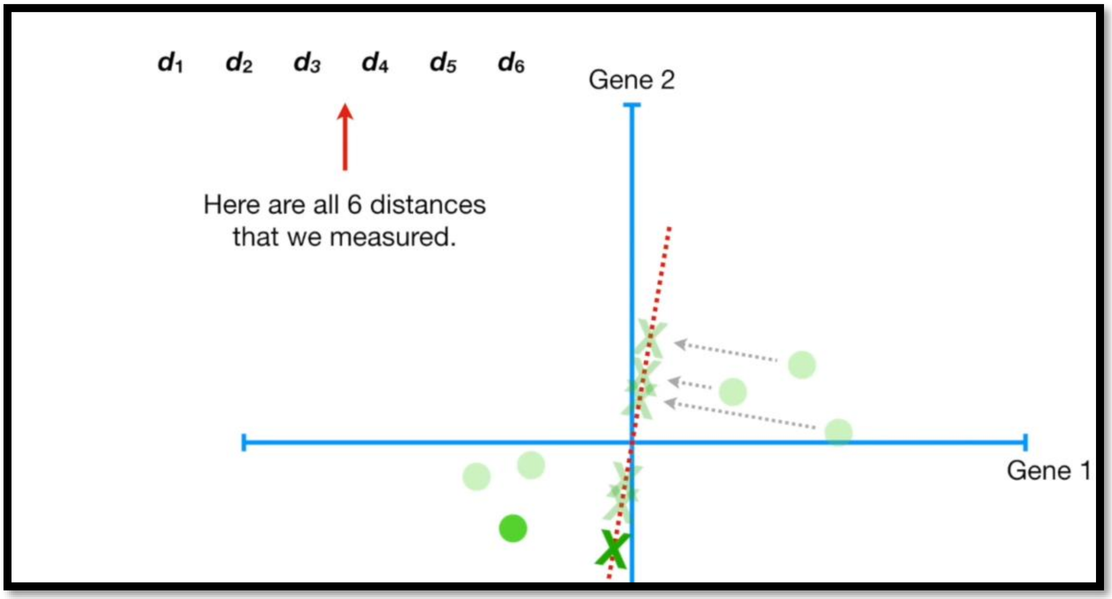


maximize the distance from the projected point to the origin.

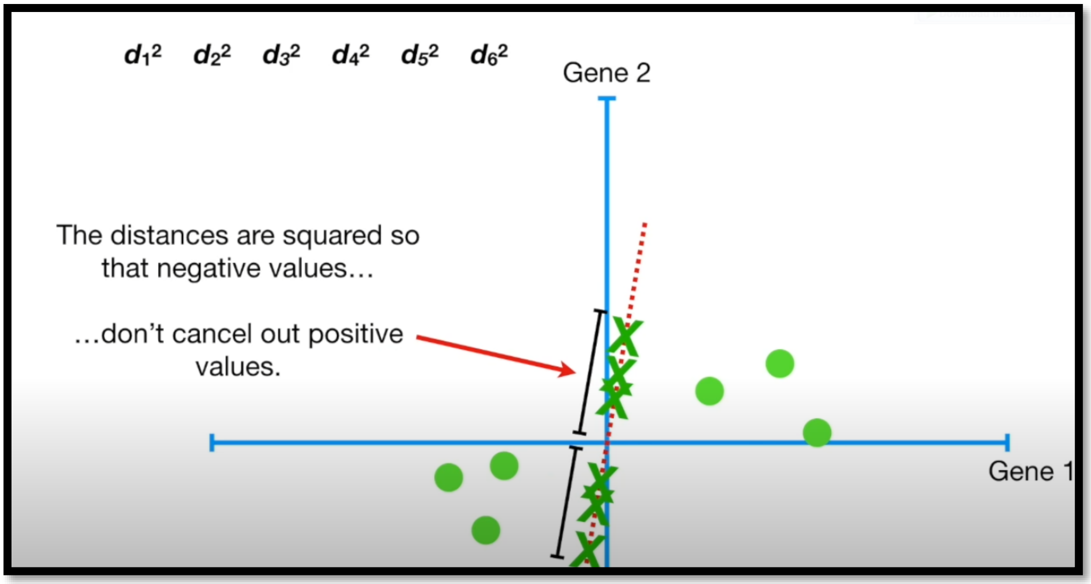
$a^2 = b^2 + c^2$







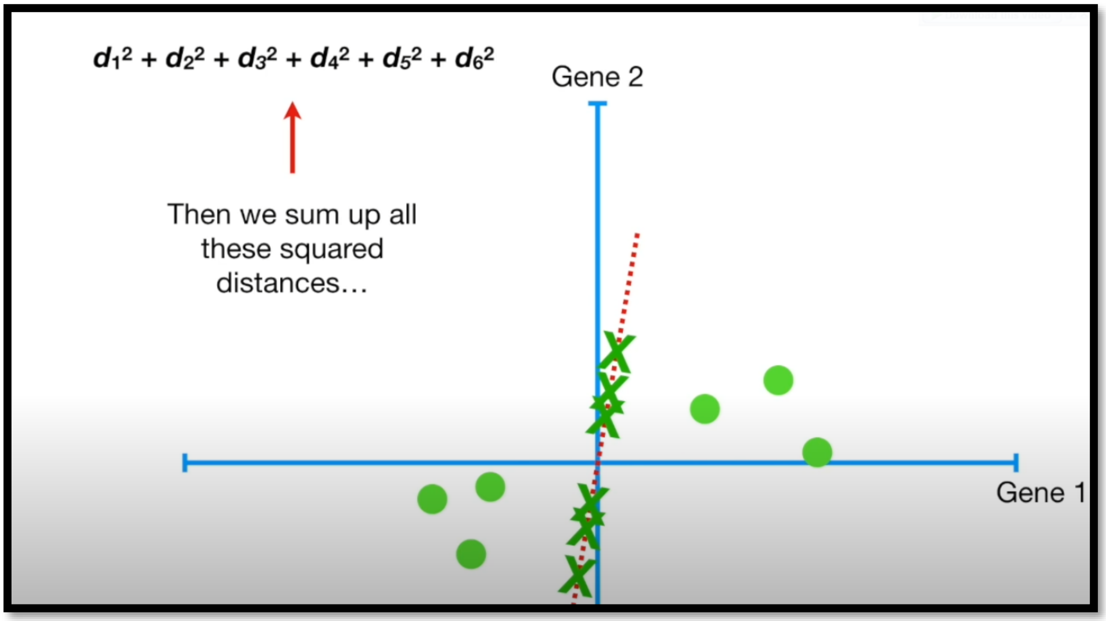
Here are all 6 distances that we measured.



d_1^2 d_2^2 d_3^2 d_4^2 d_5^2 d_6^2

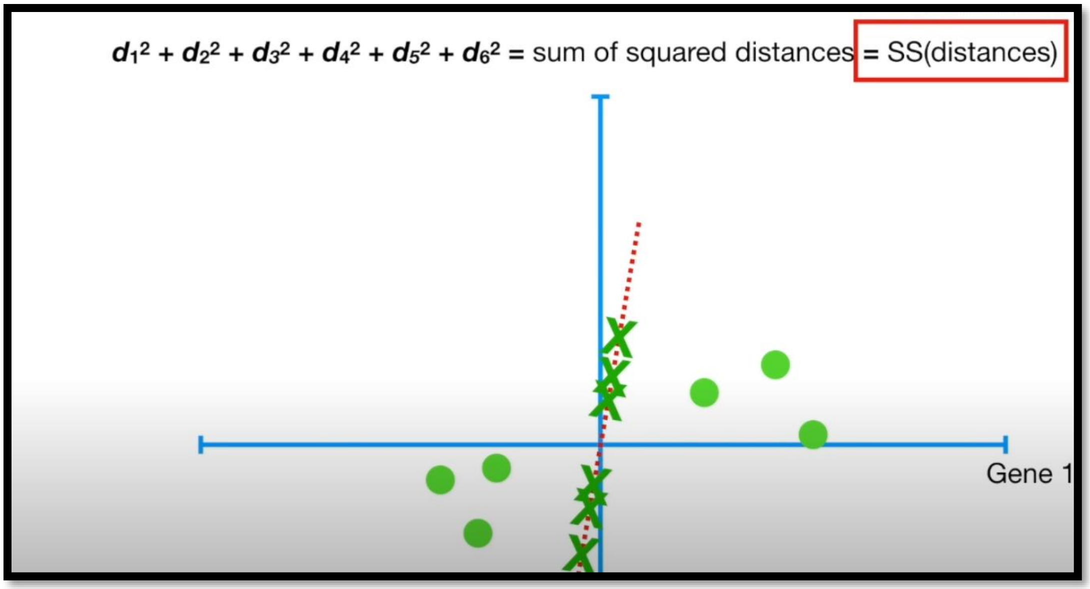
The distances are squared so that negative values…

…don't cancel out positive values.



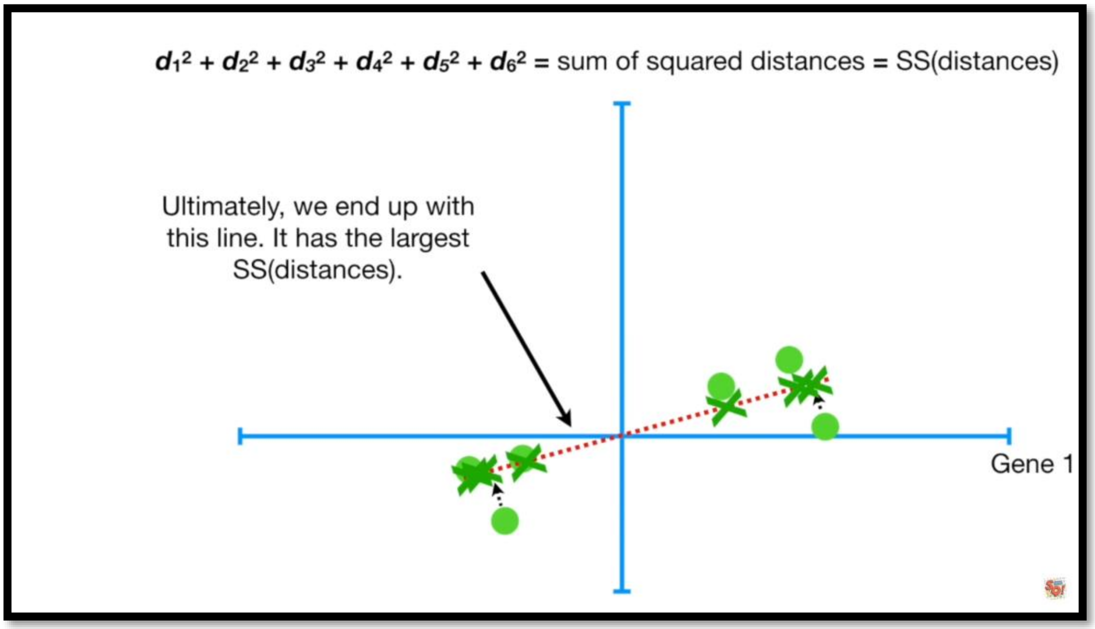
$d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2 + d_6^2$

Then we sum up all these squared distances…



$d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2 + d_6^2$ = sum of squared distances = SS(distances)

Now we rotate the line and repeat until we end up with the line with the largest sum of squared distances between the projected points and the origin. Ultimately, we end up with this line.



$d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2 + d_6^2$ = sum of squared distances = SS(distances)

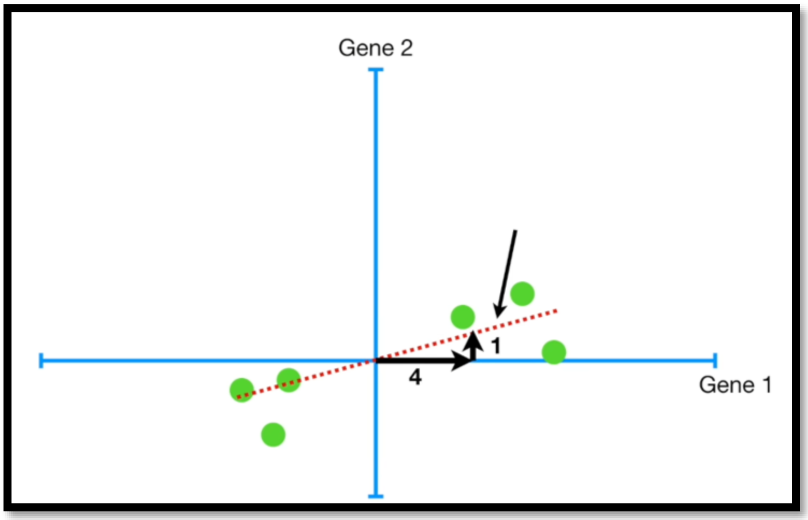
Ultimately, we end up with this line. It has the largest SS(distances).

This line is called **Principal Component 1 (PC1).** And the sum of squared distances of PC1 is called the **Eigenvalue** for PC1.

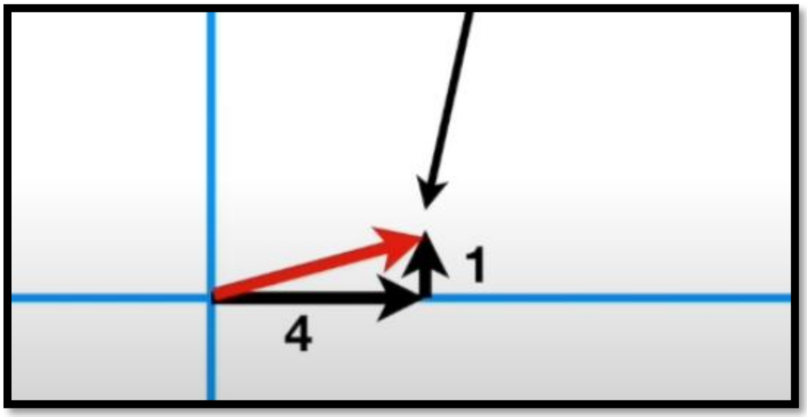
The square root of Eigenvalue of PC1 is called the **Singular Value** for PC1.

Eigenvalue of PC1 divided by (n-1) is called **Variation** of PC1.

PC1 has a slope of 0.25, in other words, for every 4 units that we go out along Gene1 axis, we go up 1 unit along the Gene2 axis.



That means that the data are mostly spread out along Gene1 axis, and only a little bit spread out along the Gene2 axis. The ratio Gene1:Gene2 4:1 tells us that Gene1 is more important when it comes to describing how the data are spread out. This is called a **linear combination** of Genes 1 and 2. Going over 4 and up 1 gets us to this point:



We can solve for the red line using Pythagorean theorem.

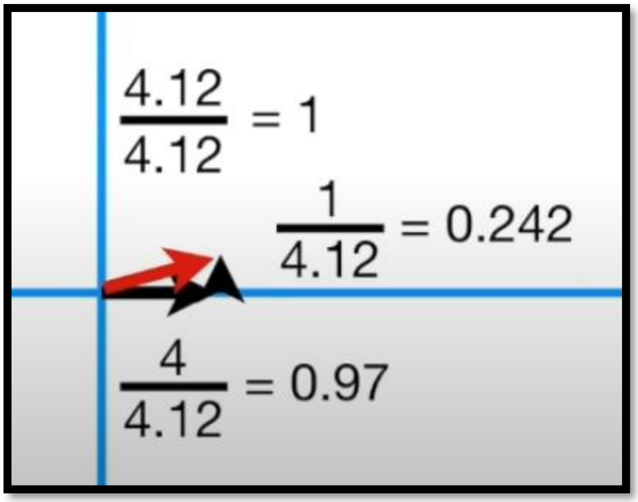
$a^2 = b^2 + c^2$

$a^2 = 4^2 + 1^2 = 17$

$a = \sqrt{17} = 4.12$

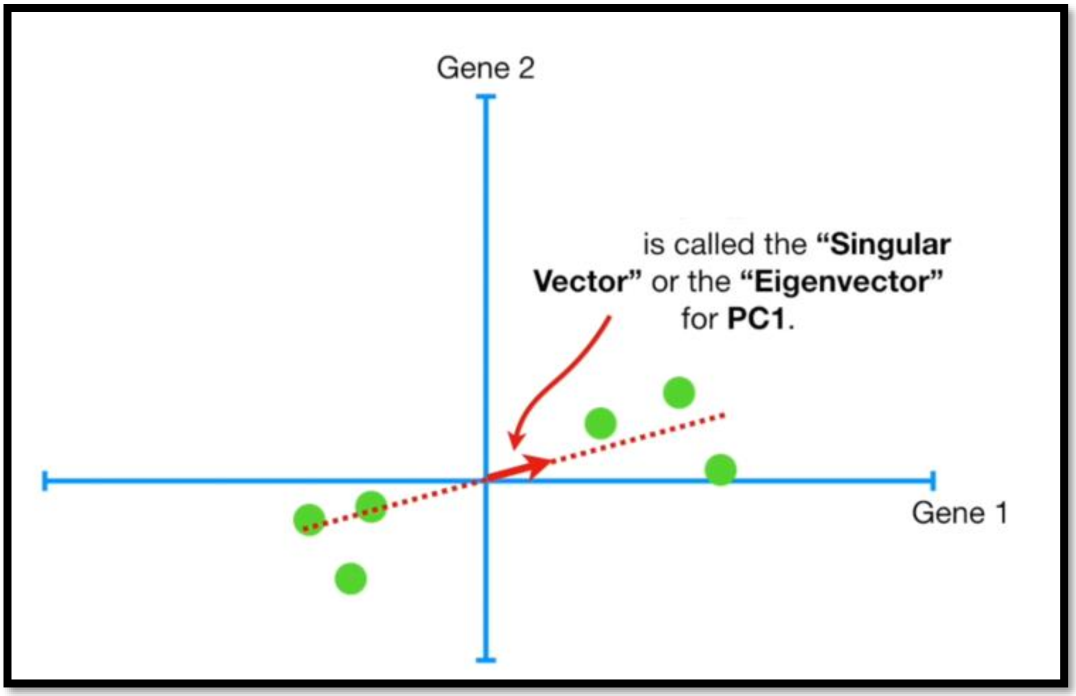
So, the length of the red line is 4.12.

When we do PCA with Singular Value Decomposition (SVD), the length of the red line has to be 1, so we have to divide each side by 4.12.



Now, in order to make PC1, we need 0.97 of Gene 1 and 0.242 of Gene 2. The ratio of course still the same (4:1).

This red line is the one-unit long vector that consists of 0.97 of Gene1 and 0.242 of Gene2. This one-unit vector is called the **Singular Vector** or the **Eigenvector** for PC1. And the proportions for all genes are called **Loading Scores.**

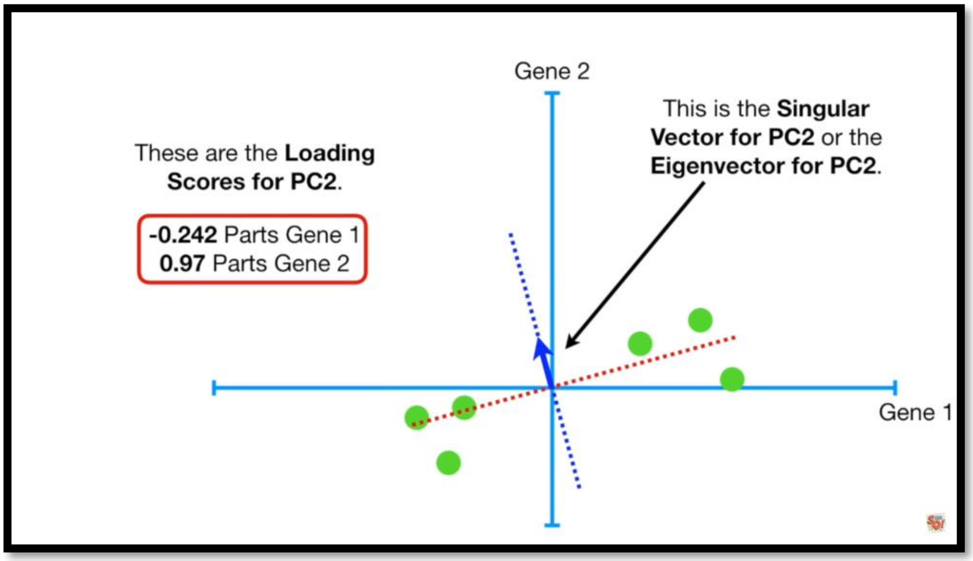


is called the **"Singular Vector"** or the **"Eigenvector"** for **PC1**.

Let’s Find PC2

Because this is only a 2D graph, PC2 is simply the line through the origin that is perpendicular to PC1 without any further optimization. This means that PC2 should consist of **-1** part of Gene1 and **4** parts of Gene 2.

If we scale everything then PC2 consists of -0.242 parts of Gene1 and 0.97 parts of Gene2.

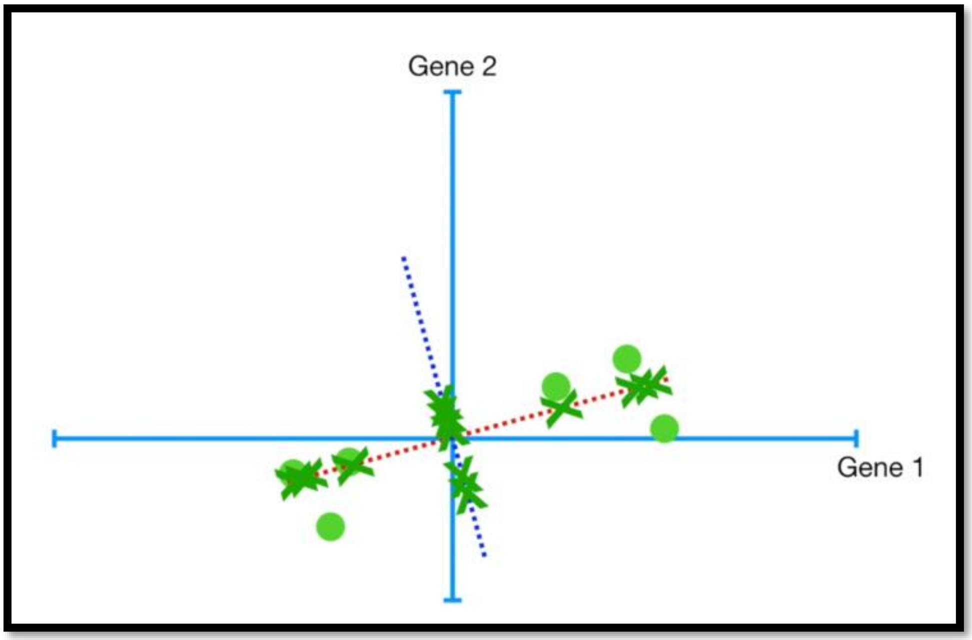


These are the **Loading Scores for PC2**.

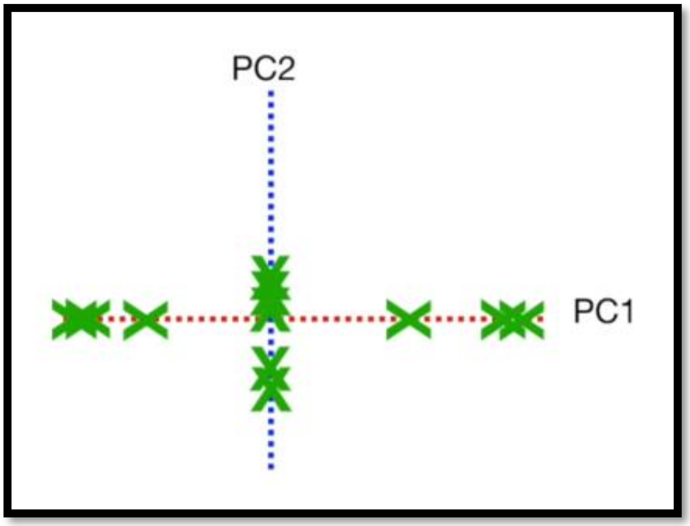
-0.242 Parts Gene 1
0.97 Parts Gene 2

This is the **Singular Vector for PC2** or the **Eigenvector for PC2**.

We have worked out PC1 and PC2!

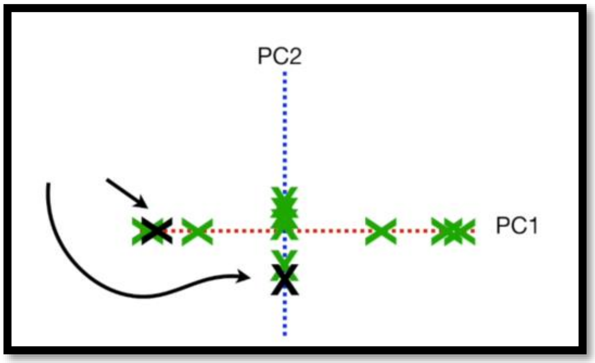


To draw the final PCA plot, we simply rotate everything so PC1 is horizontal.

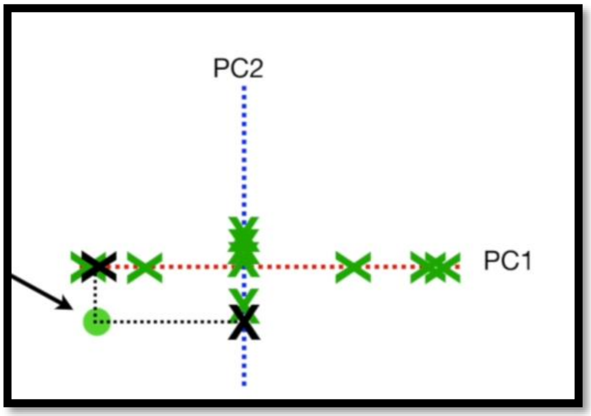


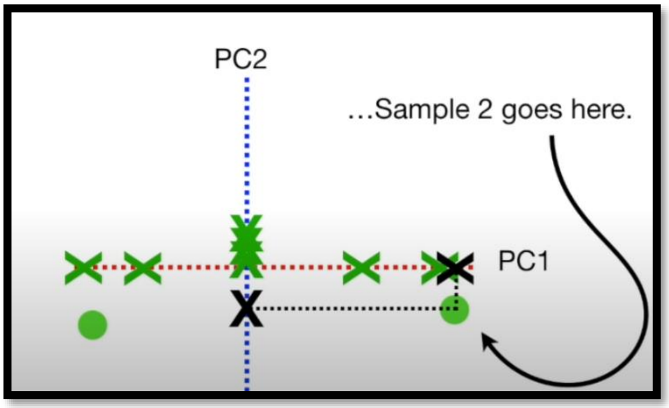
Then we use the projected points to find where the samples go in the PCA plot.

For example, these projected points correspond to sample 6.

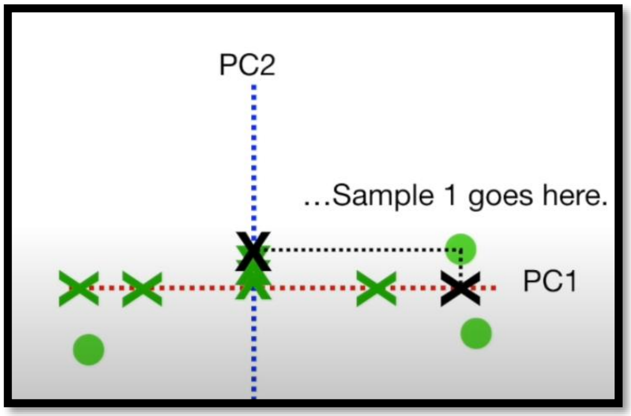


So, sample 6 goes here:



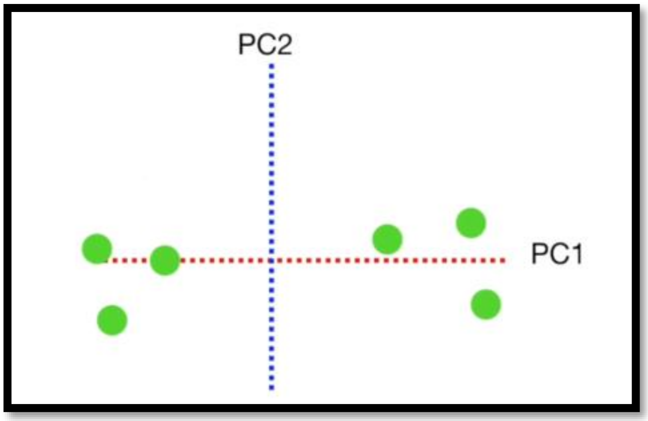


…Sample 2 goes here.



…Sample 1 goes here.

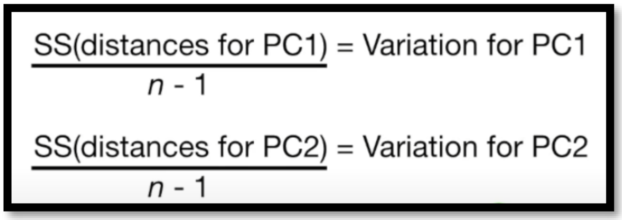
And so on, the final graph looks like this:



That’s how PCA is done using SVD.

What About Variations?

We can convert the eigenvalues to variation by dividing by the sample size minus 1.

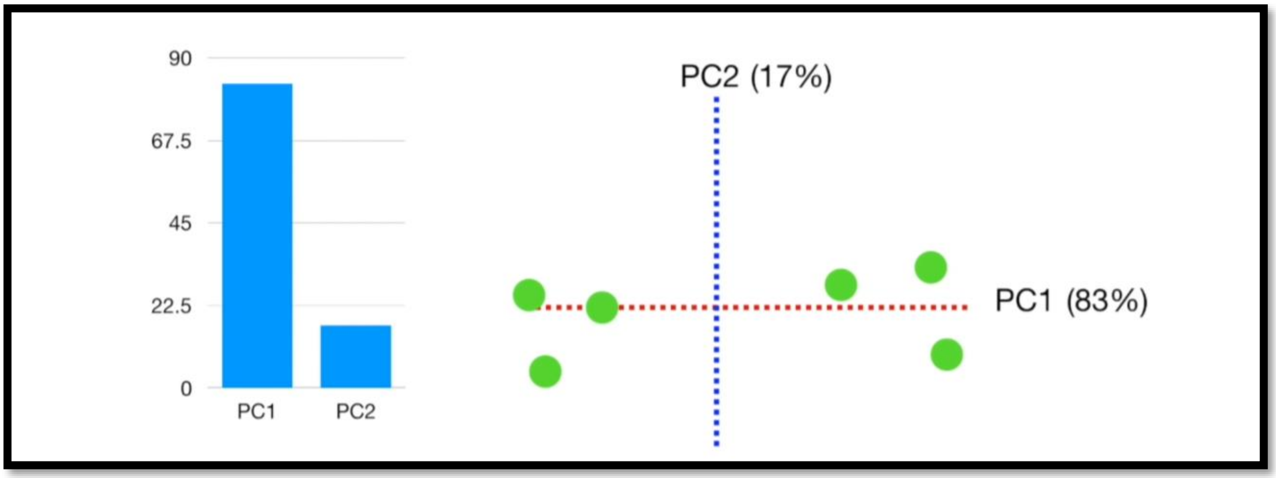


$$\frac{\text{SS(distances for PC1)}}{n - 1} = \text{Variation for PC1}$$

$$\frac{\text{SS(distances for PC2)}}{n - 1} = \text{Variation for PC2}$$

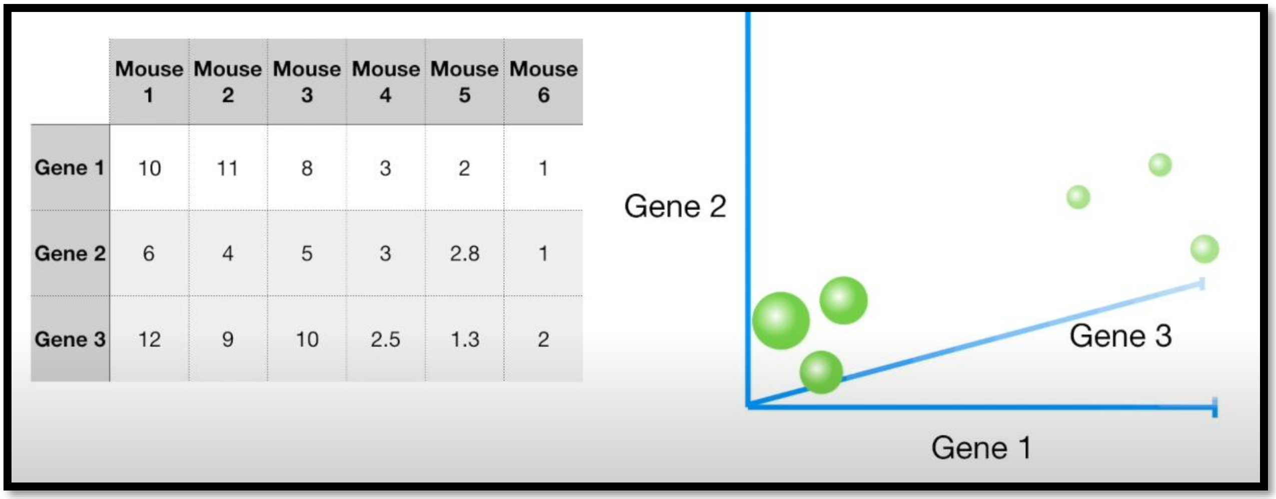
For example, imagine that the variation for PC1 is 15, and PC2 is 3. That means that the total variation around both PCs is 15+3 = 18. And that means PC1 accounts for 15/18 = 0.83 = 83% of the total variation around the PCs. PC2 accounts for 3/18 = 0.17 = 17% of the total variation around the PCs.

A **Scree Plot** is a graphical representation of the percentages of variations that each PC accounts for.



PCA with Three Variables

Pretty much the same as 2 variables.



	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1
Gene 3	12	9	10	2.5	1.3	2

We will center the data and then find the best fitting line that goes through the origin (PC1). PC1 has 3 parts, for example, 0.62 Gene1, 0.15 Gene2, and 0.77 Gene3. In this case, Gene3 is the most important feature for PC1.

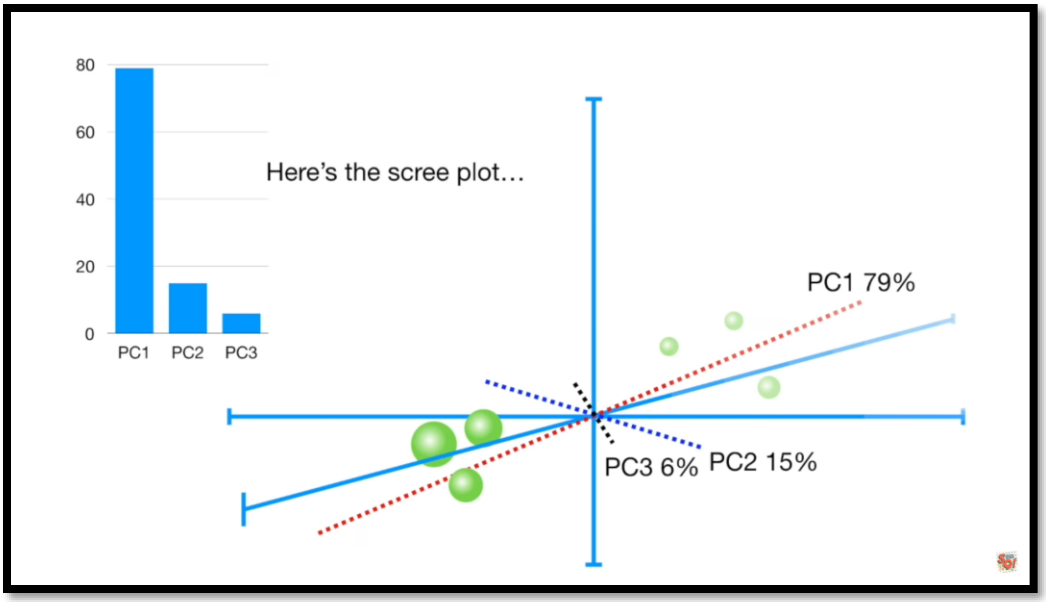
We then find PC2 (the next best fitting line given that it goes through the origin and it is perpendicular to PC1). PC2 has 0.77 parts of Gene1, 0.62 of Gene2, and 0.15 of Gene3. In this case, Gene1 is the most important feature for PC3.

Lastly, we find PC3, the best fitting line that goes through the origin and is perpendicular to PC1 and PC2.

If we had more genes, we'd just keep on finding more and more principal components by adding perpendicular lines and rotating them.

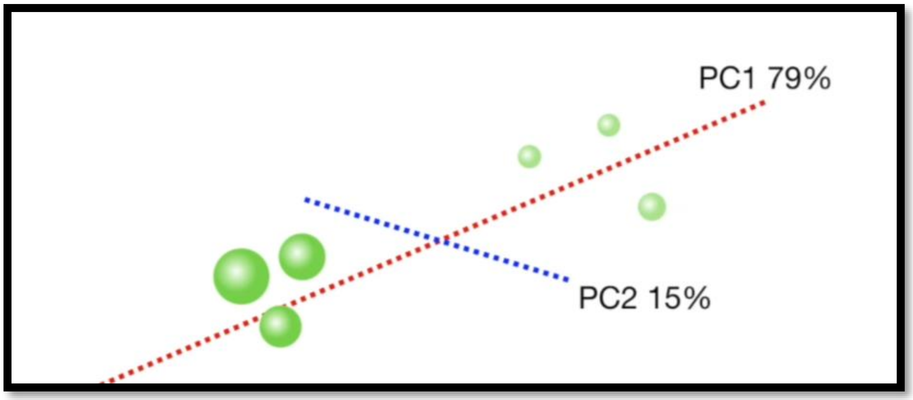
In theory, there is one component per feature, but in practice, the number of PCs is either the number of features or the number of samples, whichever is smaller.

Once we have all of the PCs, we can use the eigenvalues to determine the proportion of variation that each PC accounts for and plot the scree plot.

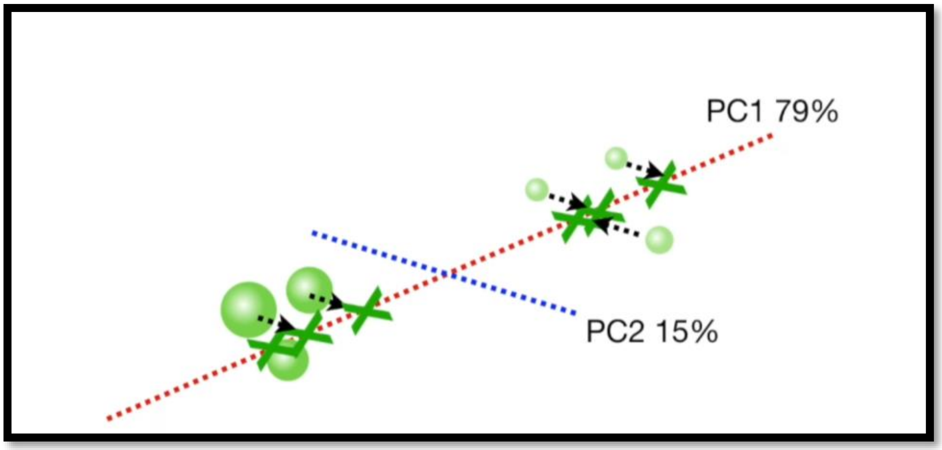


It is obvious that PC1 and PC2 account for the vast majority of the variation. That means that a 2D graph, using just PC1 and PC2, would be a good approximation of the previous 3D graph since it would account for 94% of the variation in the data.

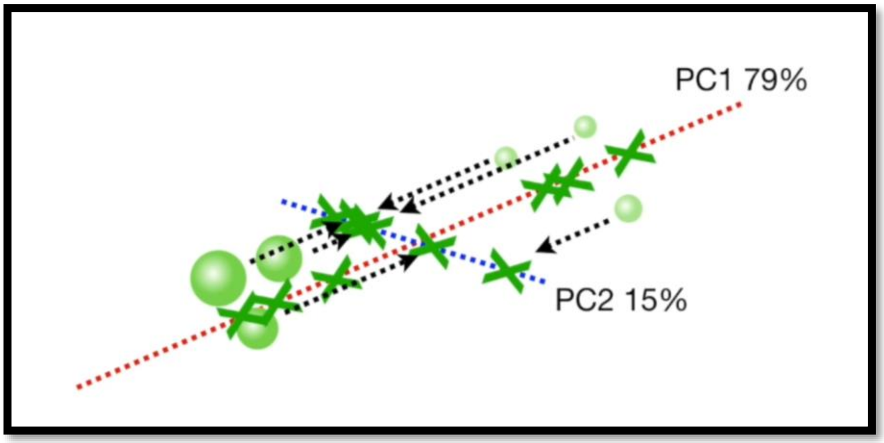
To convert the 3D graph into a 2D graph, we just strip away everything but the data points and PC1 and PC2.



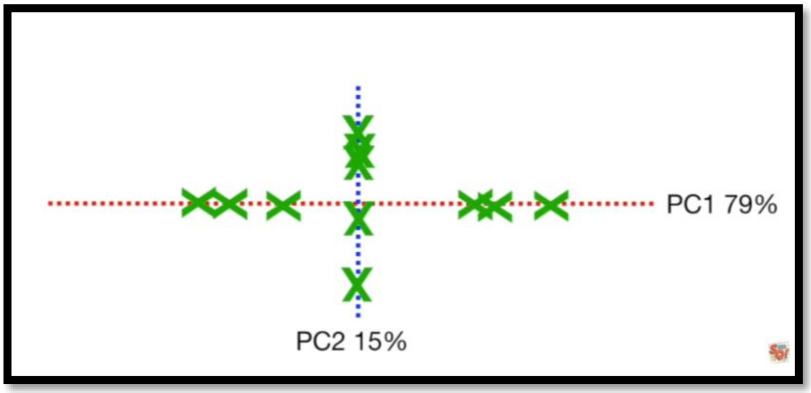
The we project the samples onto PC1:



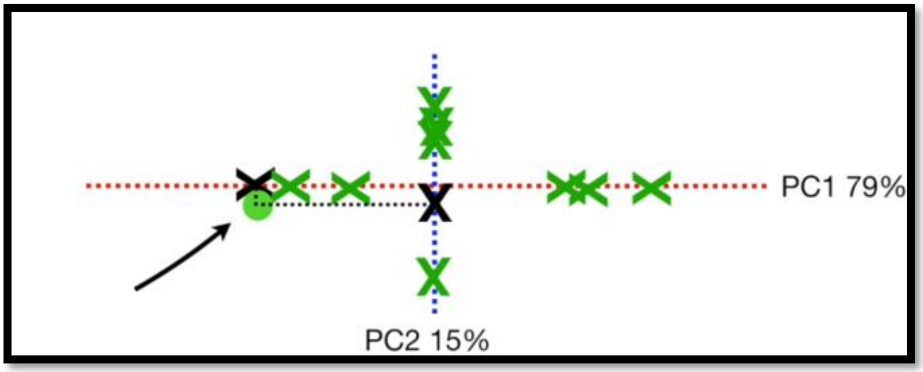
And PC2:



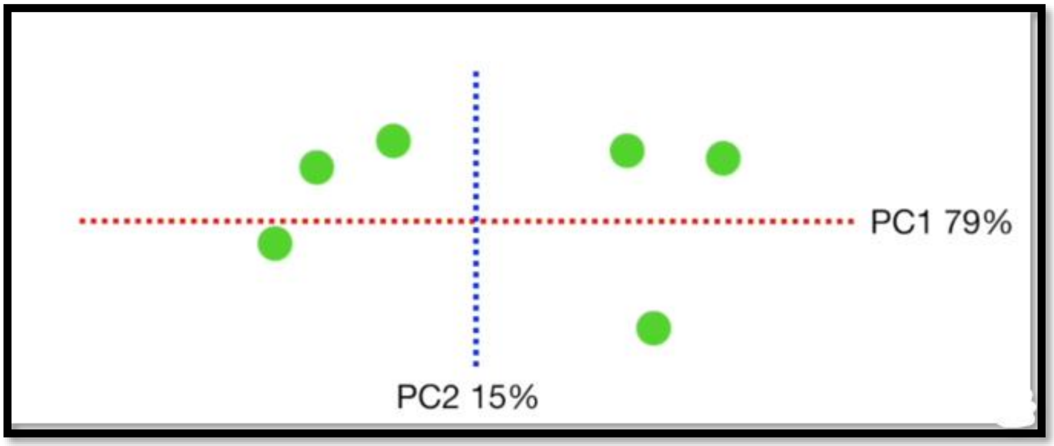
Then we rotate so that PC1 is horizontal.



This is sample 4:



And so on, this is the final PCA plot:



Prepared By:

Ibrahim M. Nasser

References:

StatQuest: Principal Component Analysis (PCA), Step-by-Step