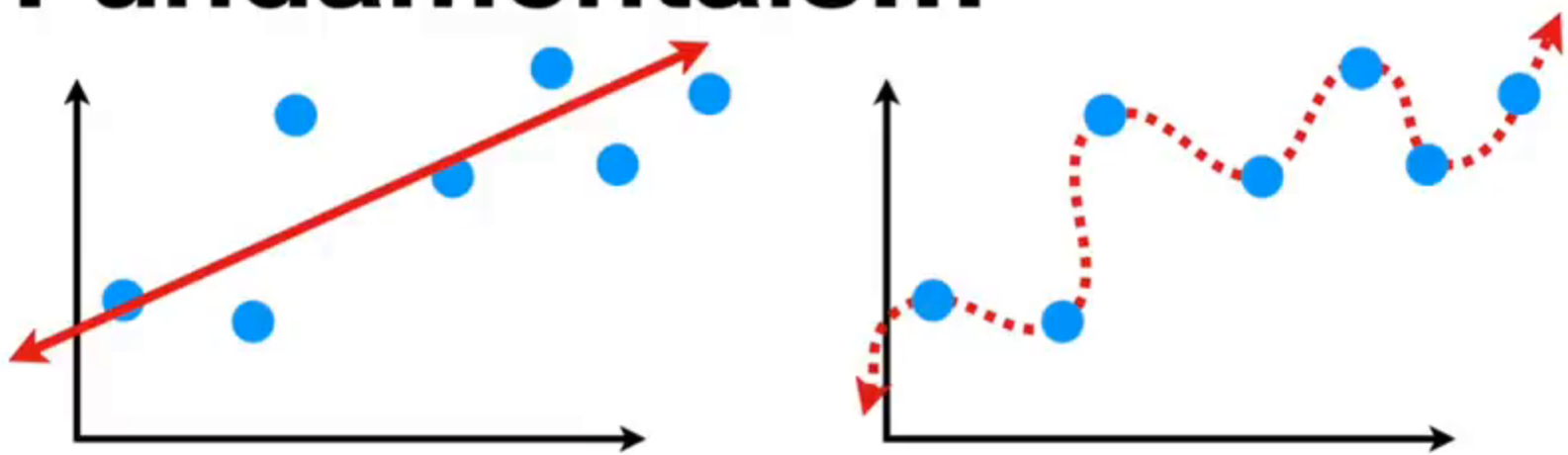
…have a lot of terminology associated with them…

Support Vector Machines (SVM) Clearly Explained!!!

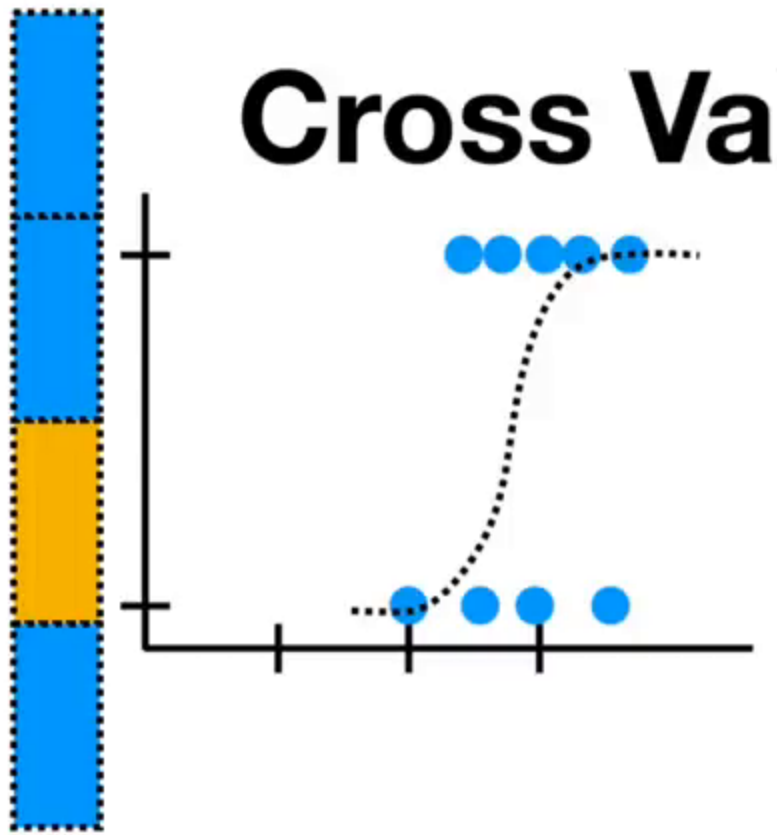
NOTE: This **StatQuest** assumes that you are already familiar with the tradeoff that plagues all of machine learning, the **bias/variance tradeoff**.

Machine Learning Fundamentals...



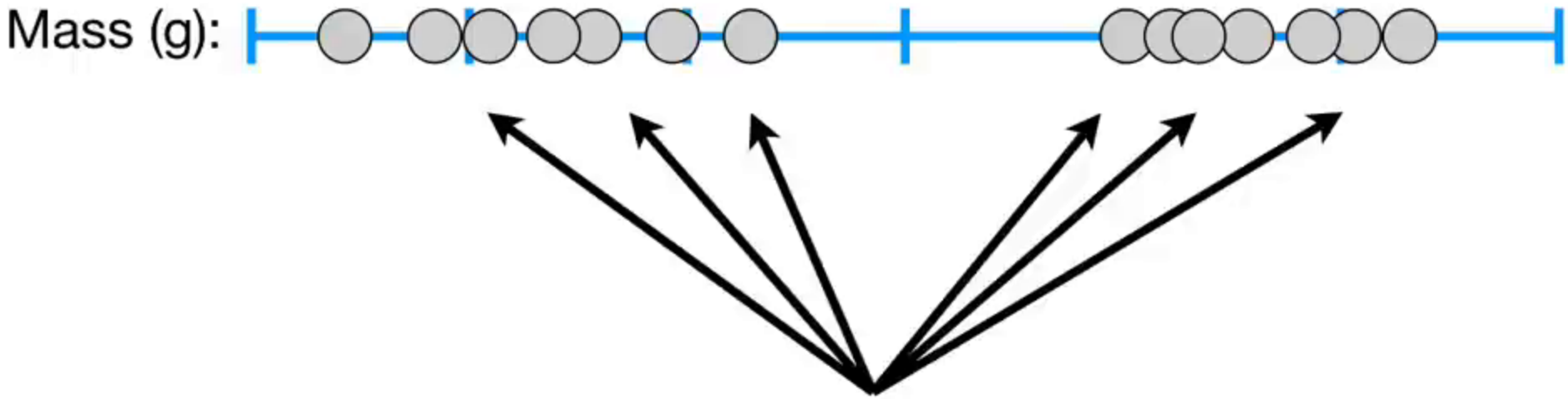
...Bias and Variance!!!

You should also be familiar with **Cross Validation**. If not, check out the **'Quests**. The links are in the description below.

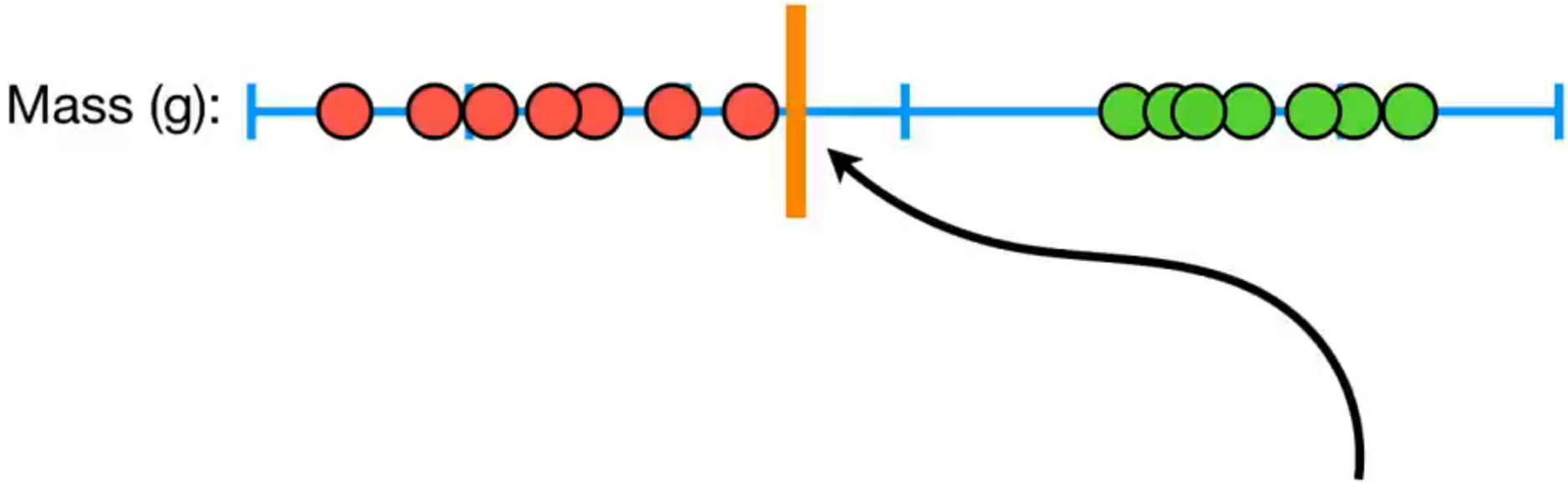


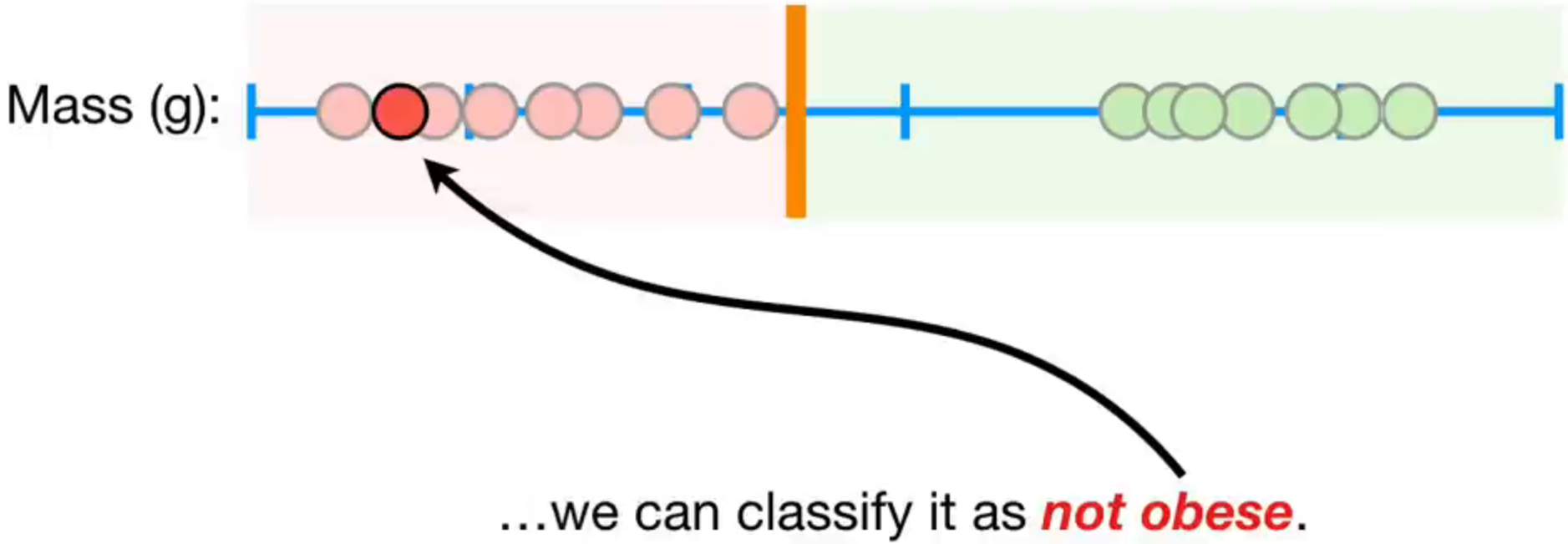
Cross Validation....

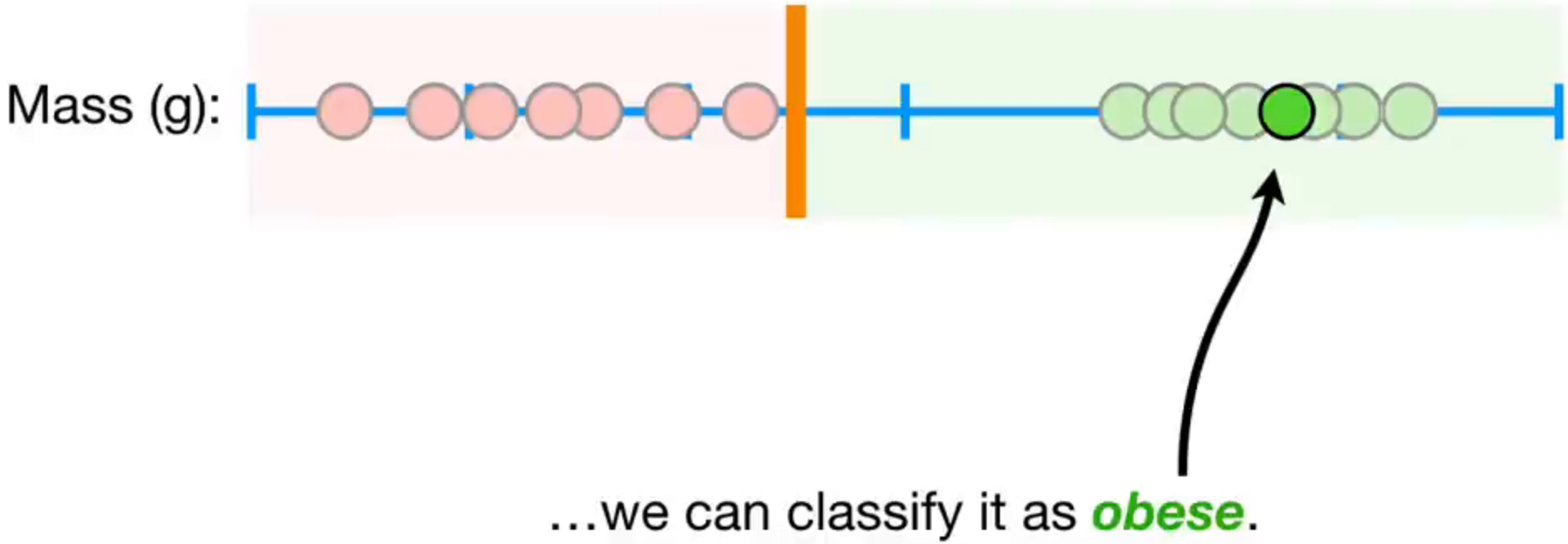
...it's no big deal!!!

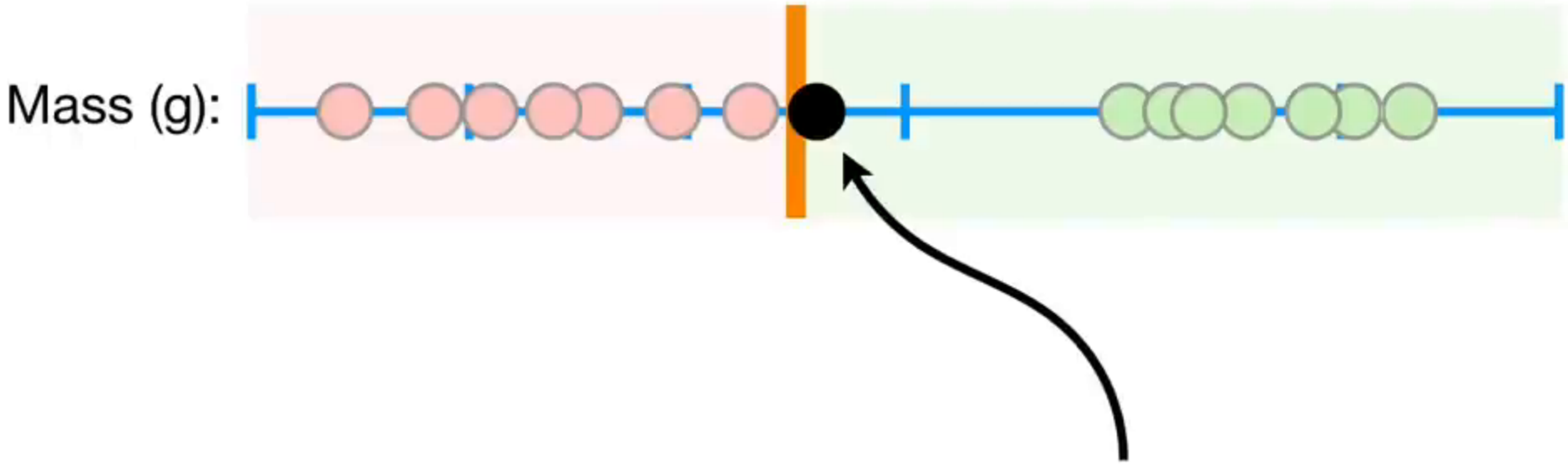


Let’s start by imagining we measured
the mass of a bunch of mice…

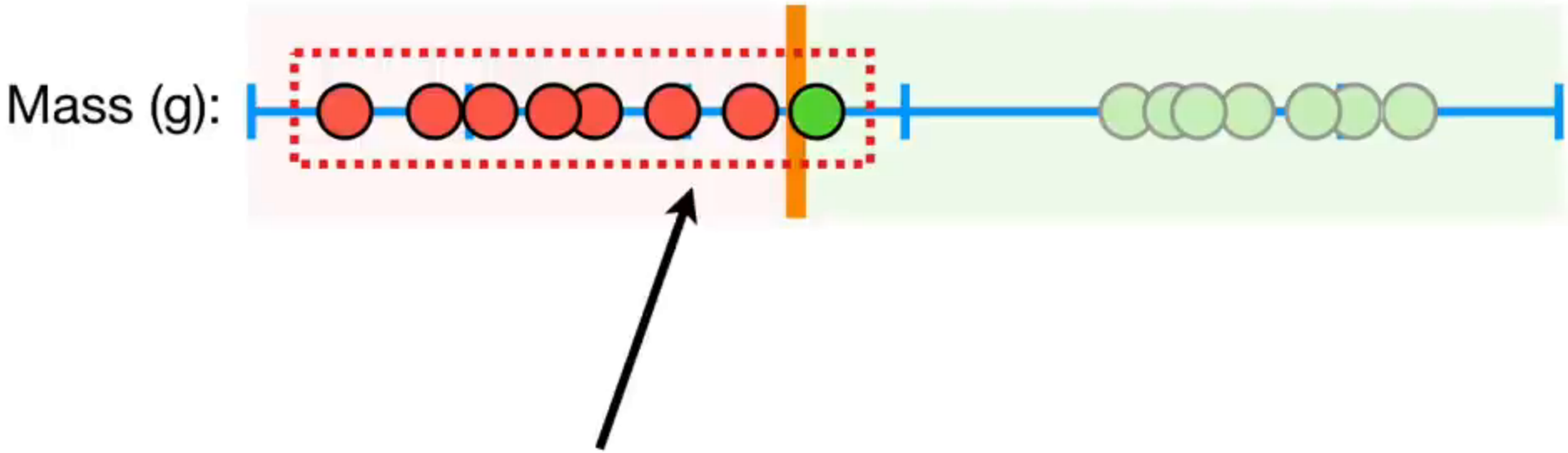

Mass (g):
Based on these observations, we can pick a threshold…


Mass (g):
…we can classify it as *not obese*.

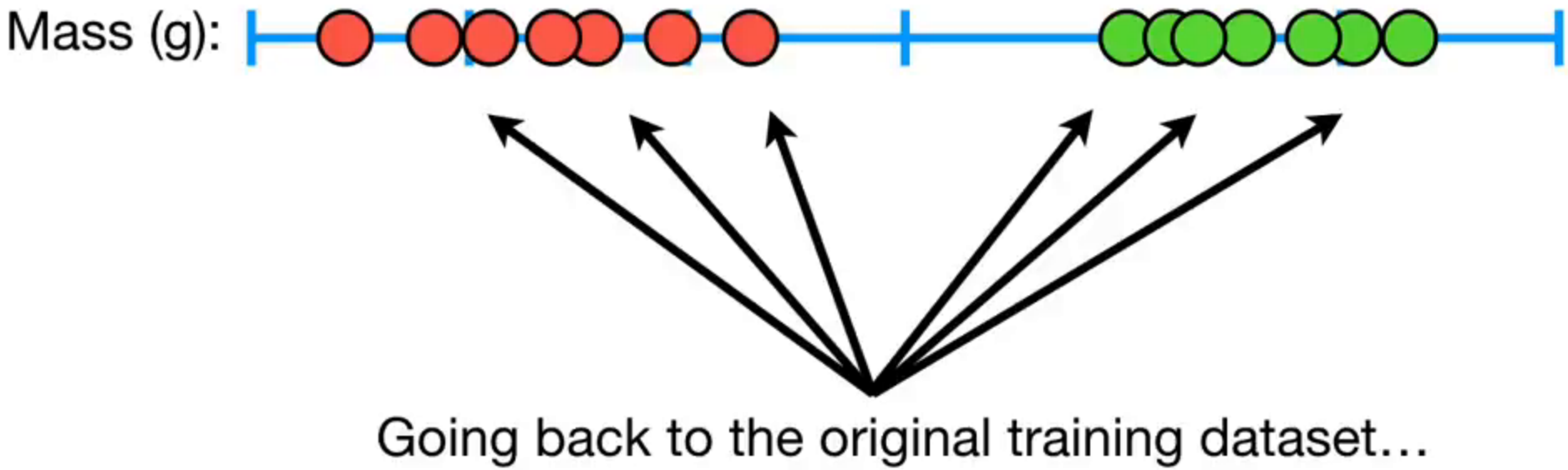

Mass (g):
…we can classify it as *obese*.

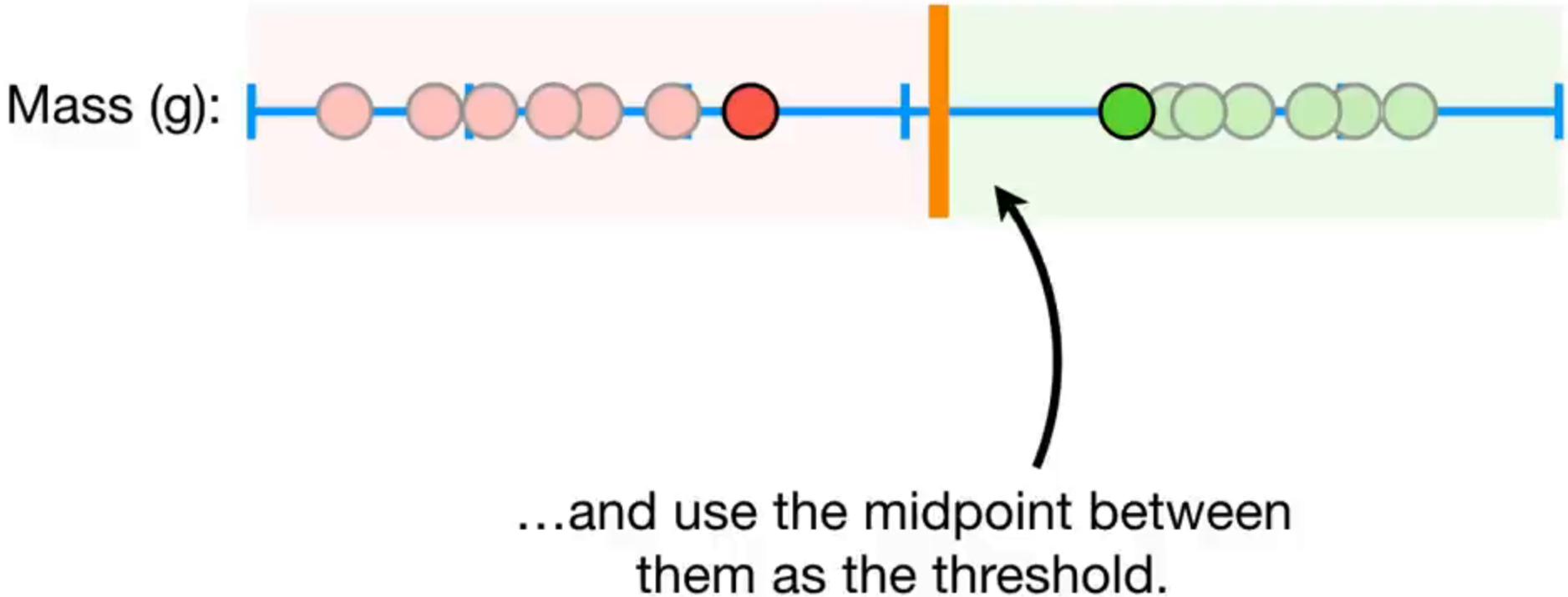


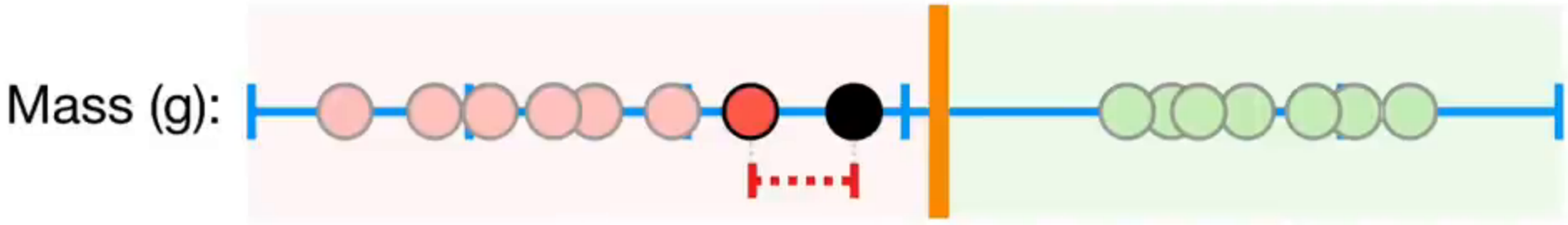
Because this observation has more mass than the threshold, we classify it as ***obese***.



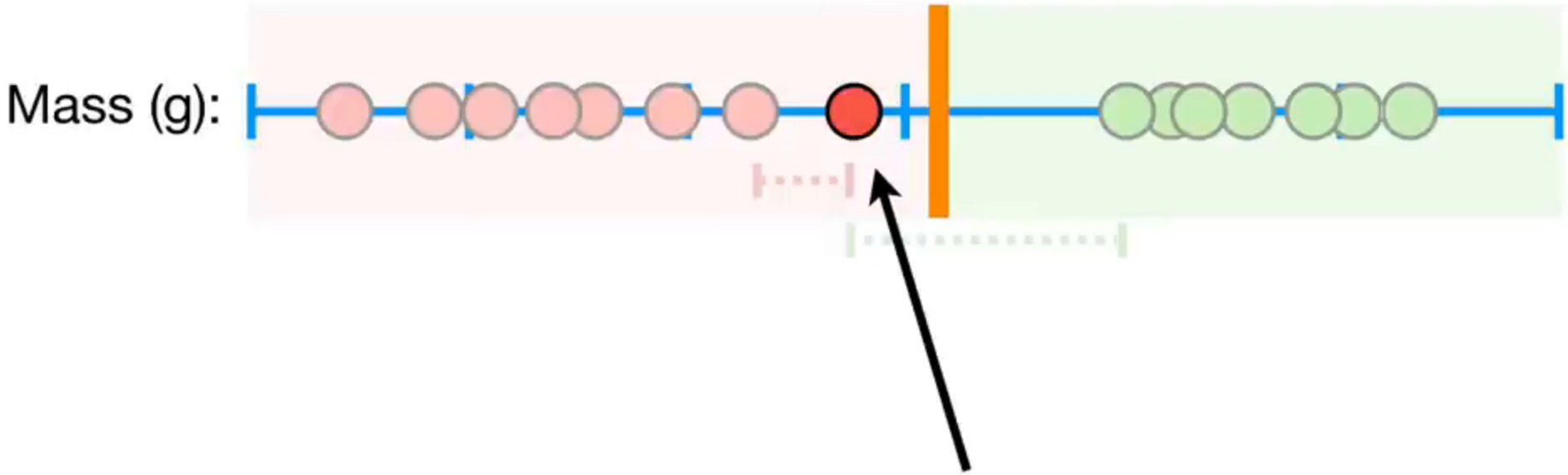
But that doesn't make sense, because it is much closer to the observations that are ***not obese***.

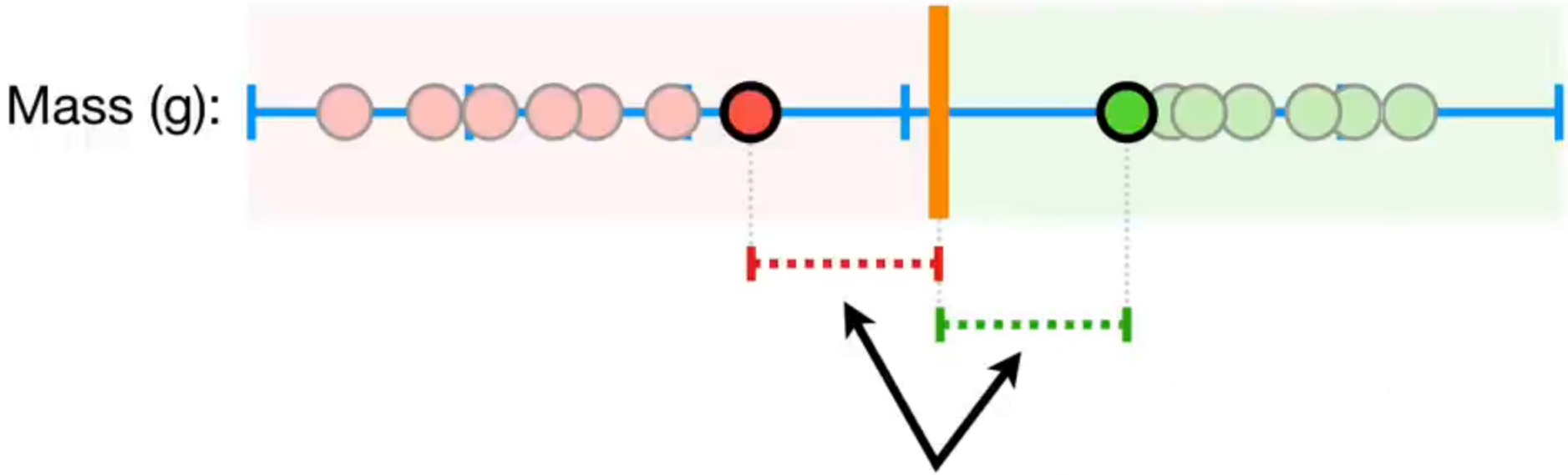

Mass (g):
Going back to the original training dataset…


Mass (g):
…and use the midpoint between them as the threshold.

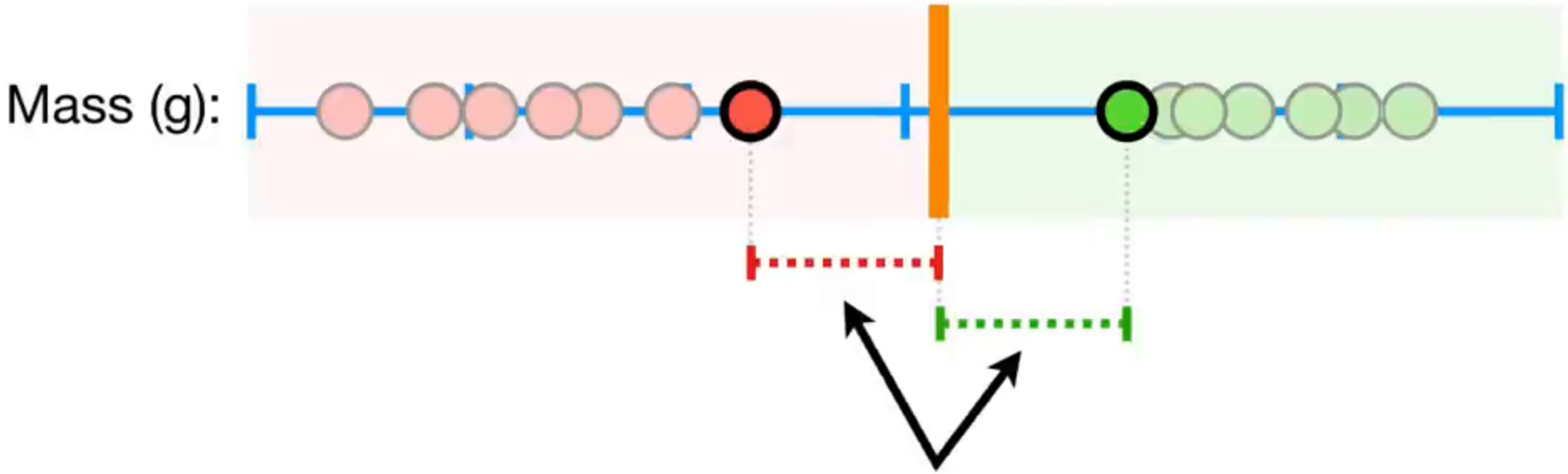


…it will be closer to the observations that are ***not obese***…

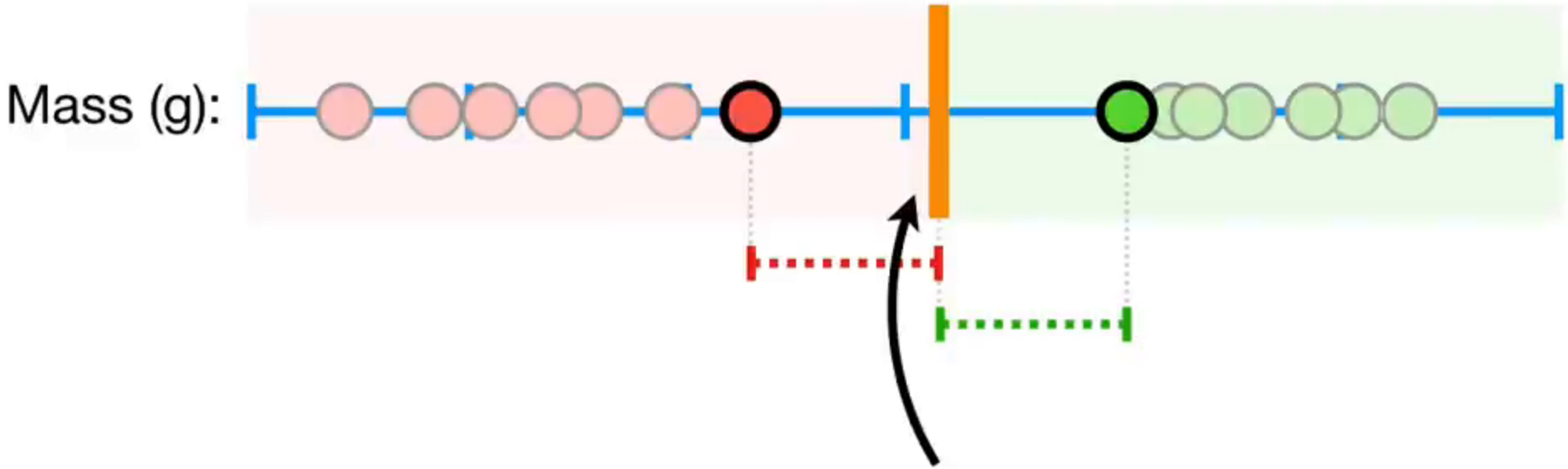

Mass (g):
So it makes sense to classify this new observation as *not obese*.



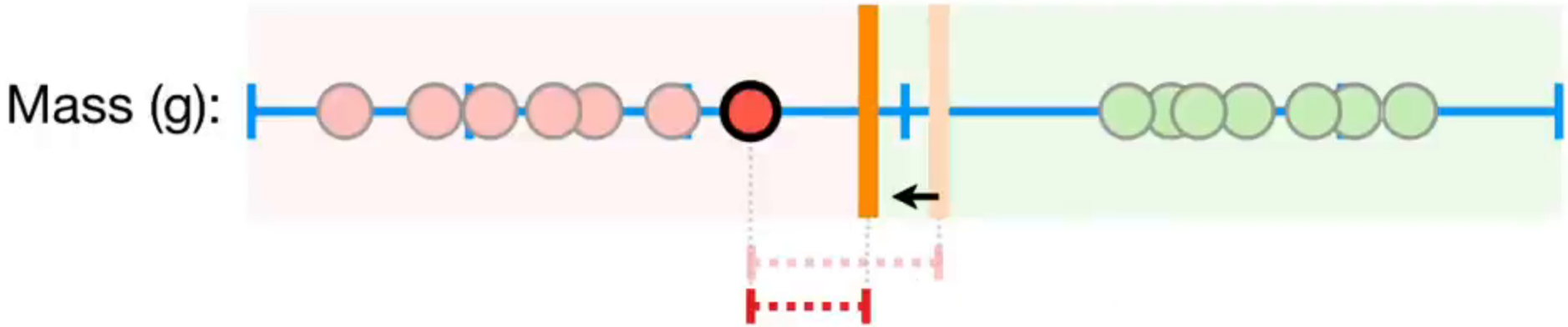
The shortest distance between the observations and the threshold is called the **margin**.



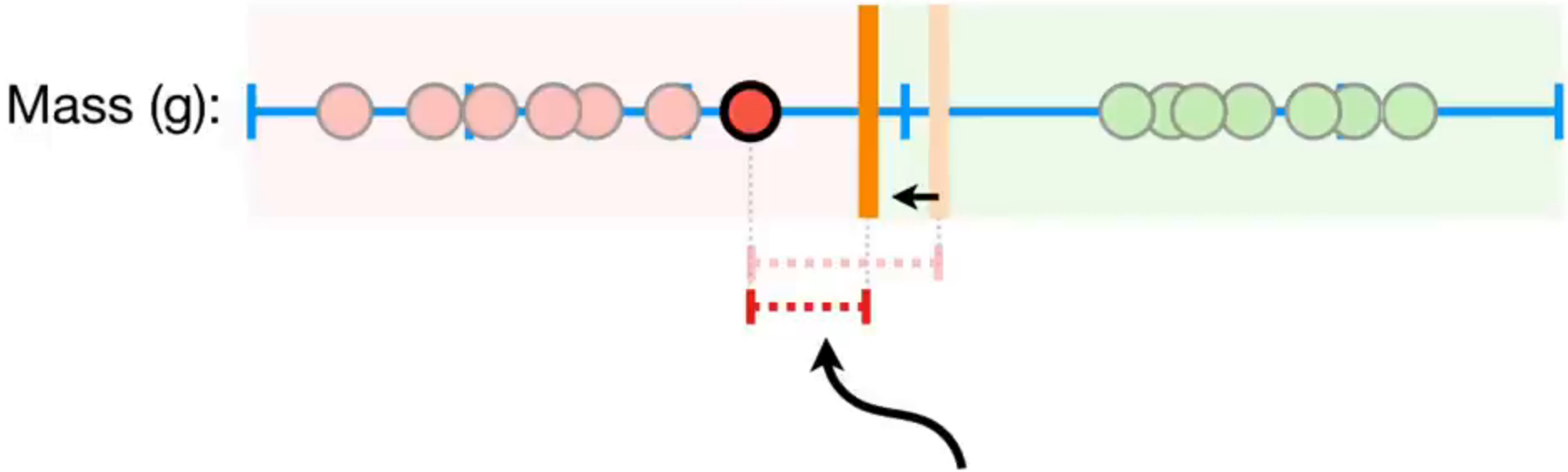
…the distances between the observations and the threshold are the same and both reflect the **margin**.

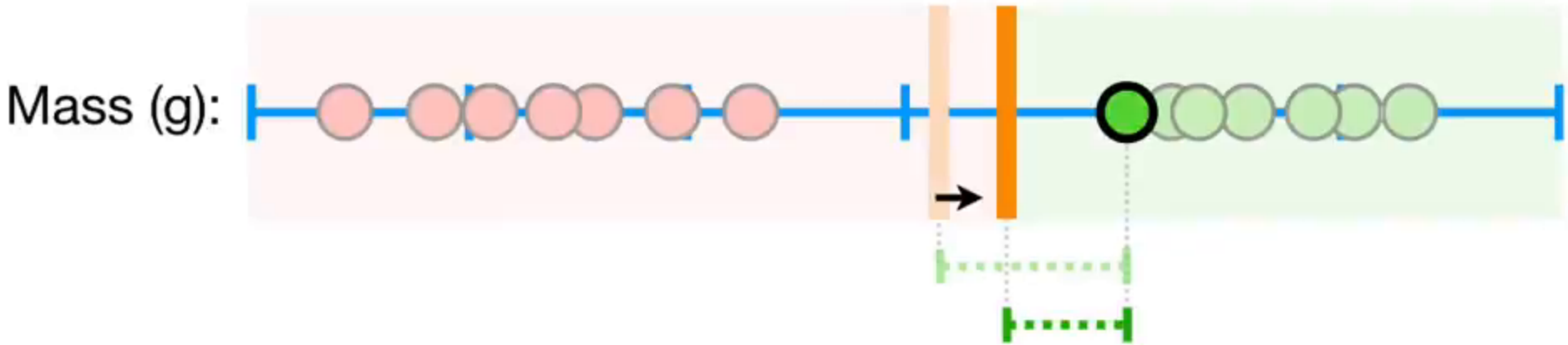


When the threshold is halfway between the two observations, the **margin** is as large as it can be.

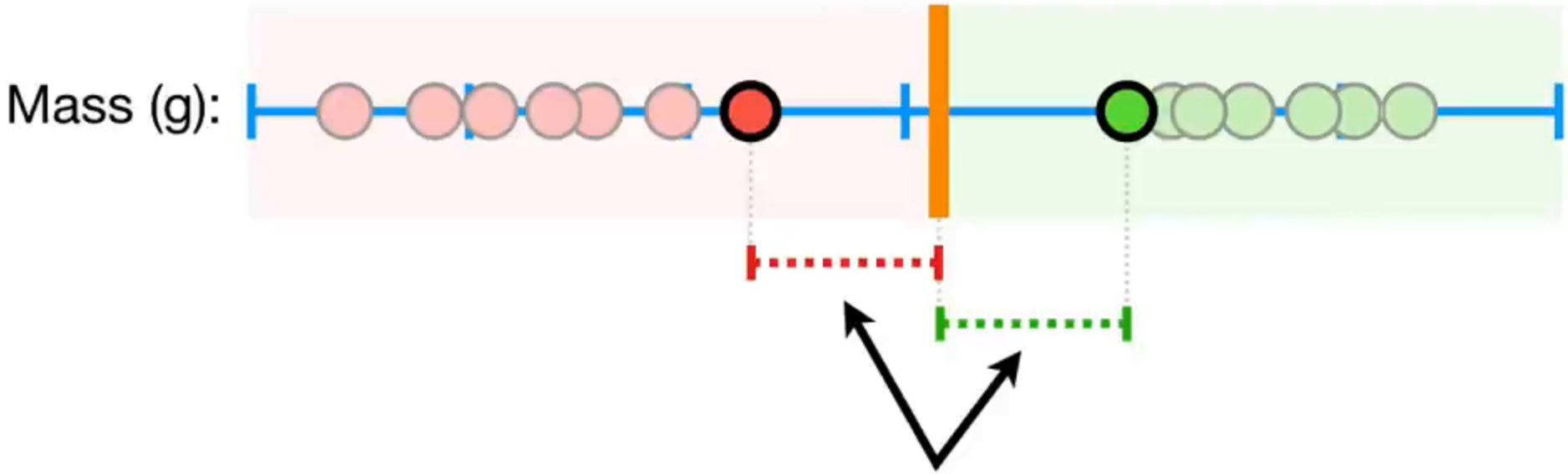


…then the distance between the threshold and the observation that is ***not obese*** would be smaller…

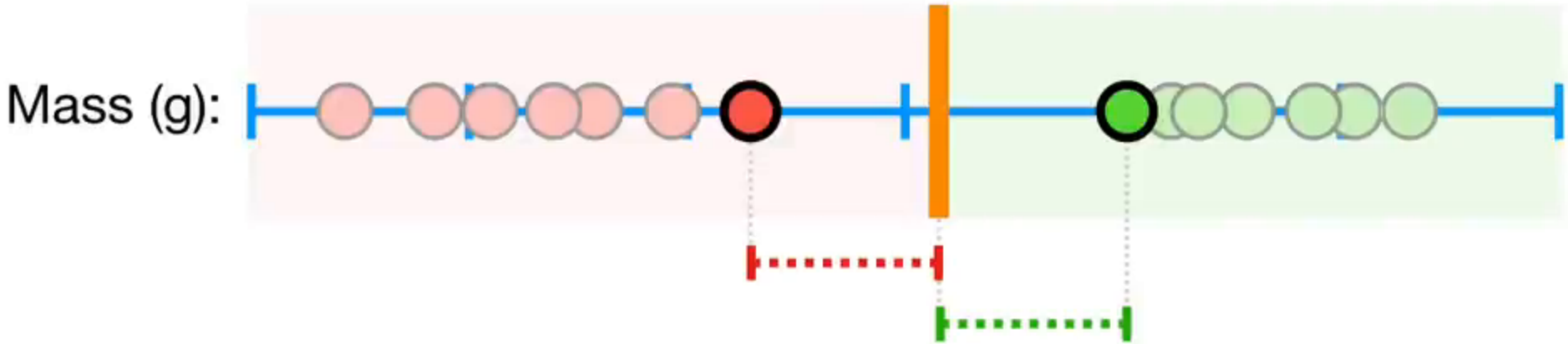

Mass (g):
…and thus, the **margin** would be smaller than it was before.



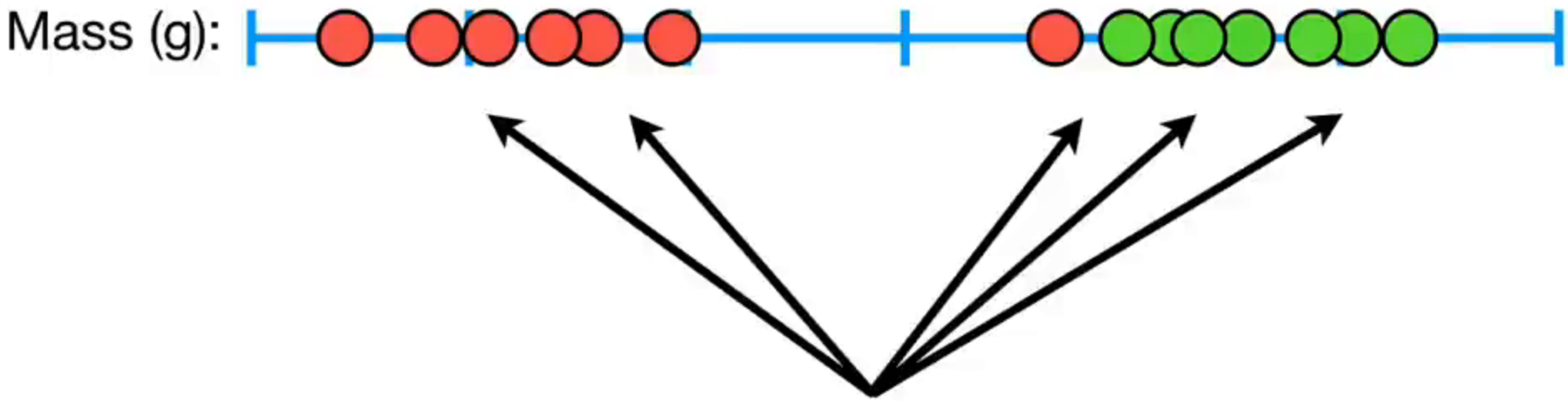
…then the distance between the ***obese*** observation and the threshold would get smaller…



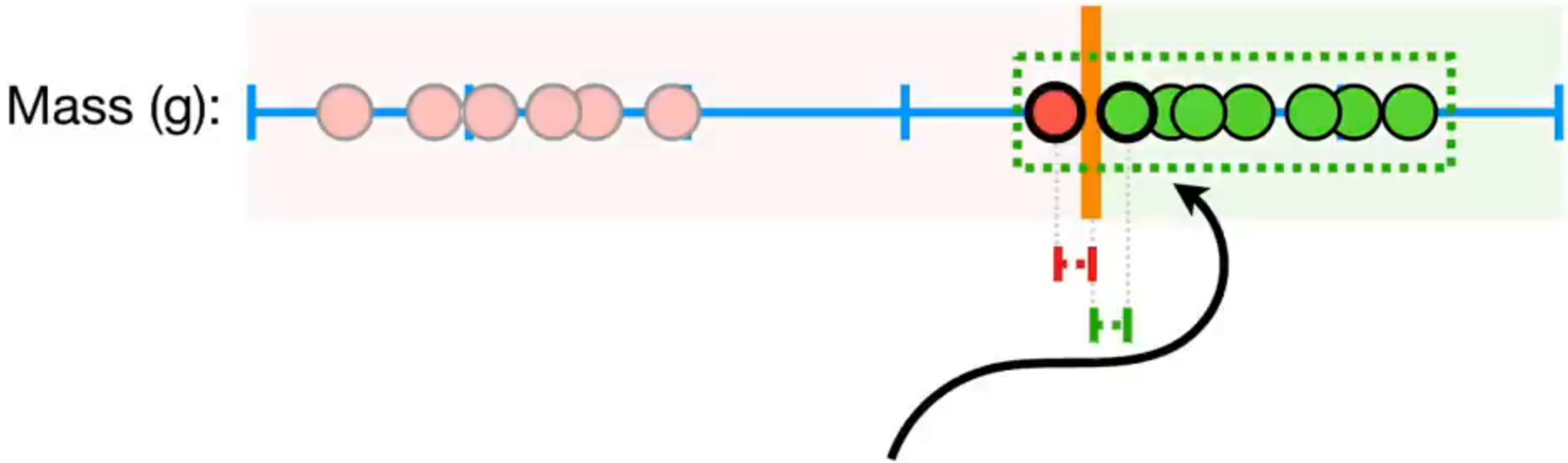
When we use the threshold that gives us the largest **margin** to make classifications…



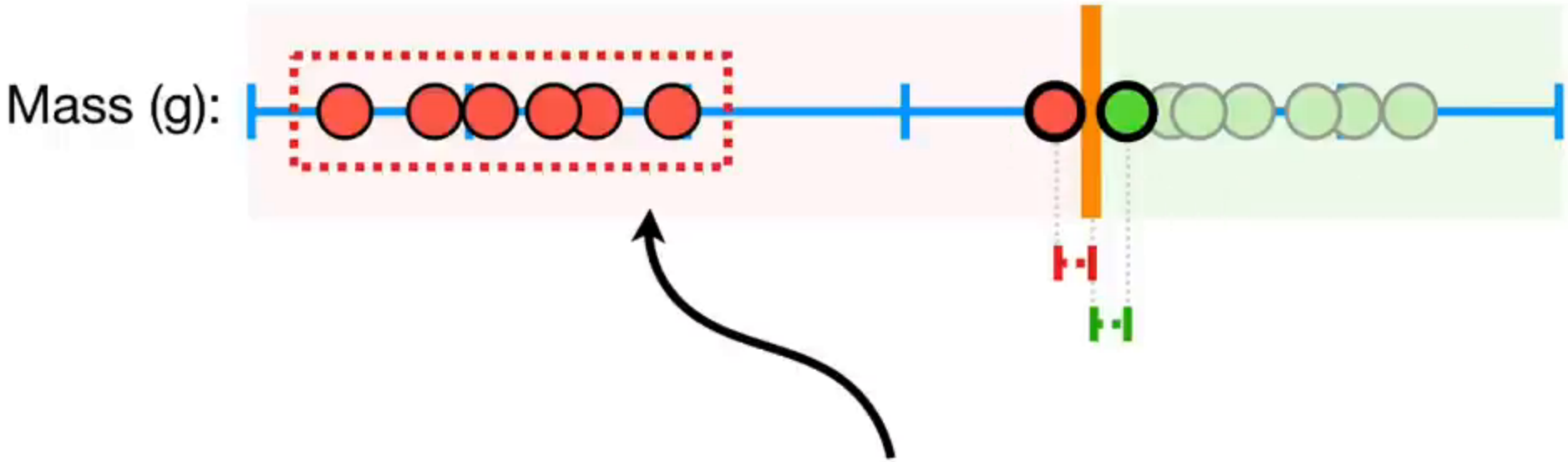
No. No bam.



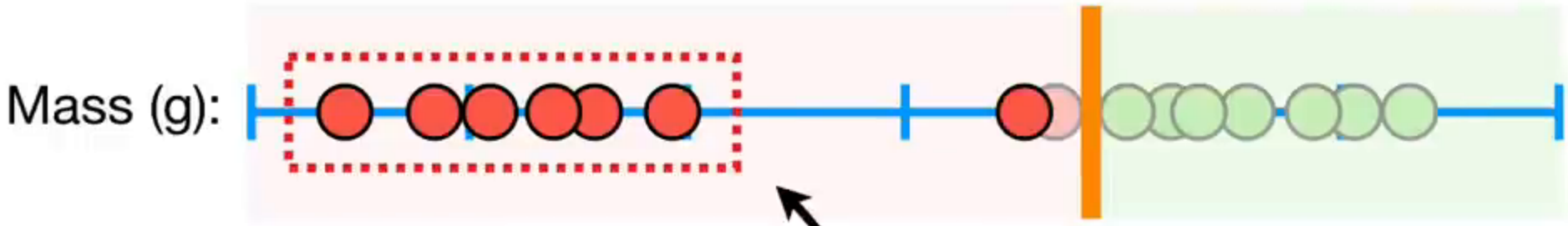
…but what if our training data looked like this….



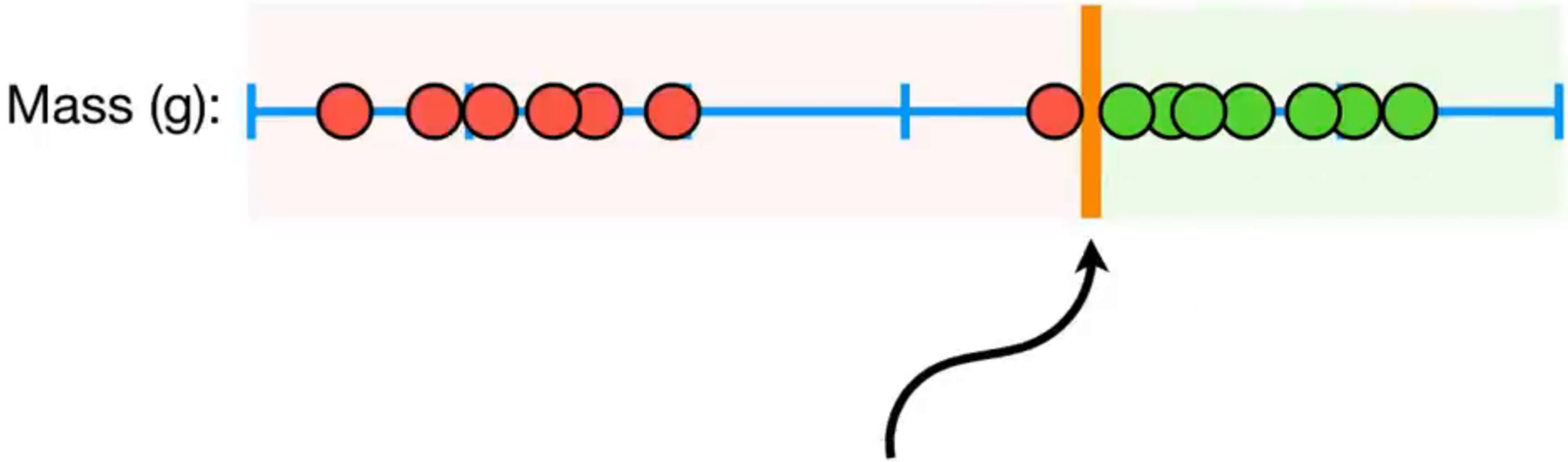
In this case, the **Maximum Margin Classifier** would be super close to the ***obese*** observations…



…and really far from the majority of the observations that are **not obese**.

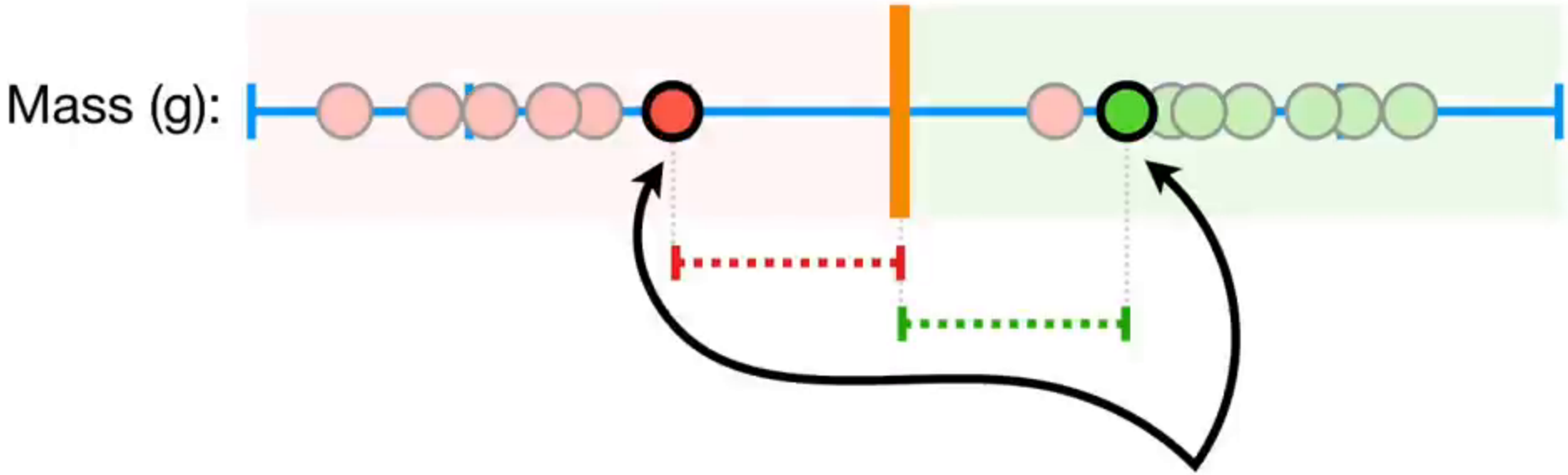


…we would classify it as ***not obese***, even though most of the ***not obese*** observations are much further away than the ***obese*** observations.

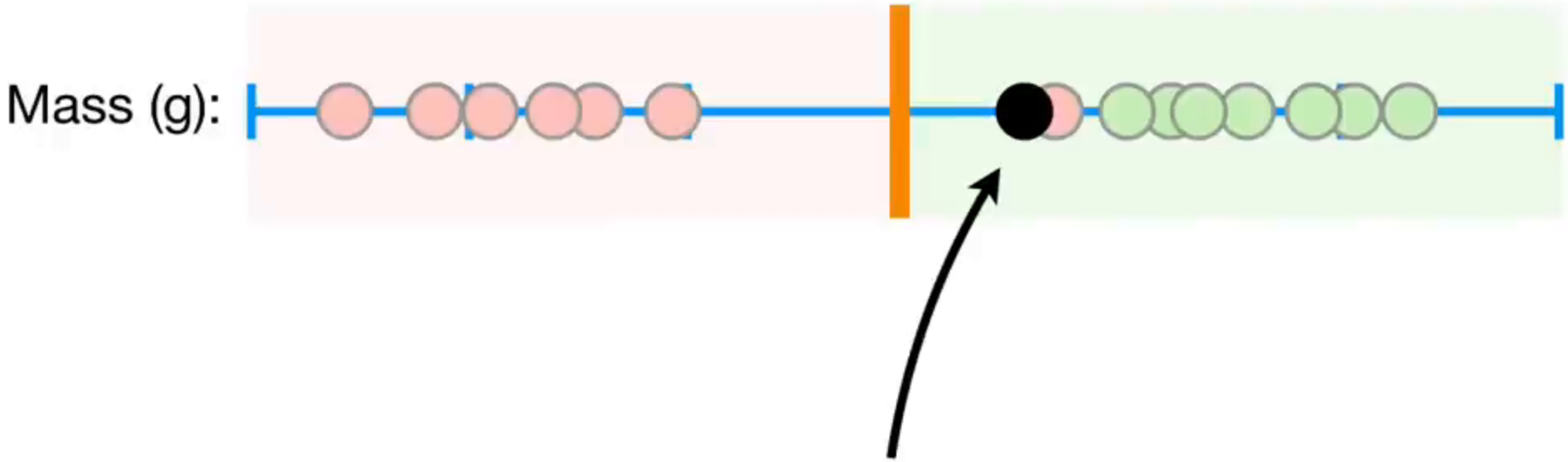


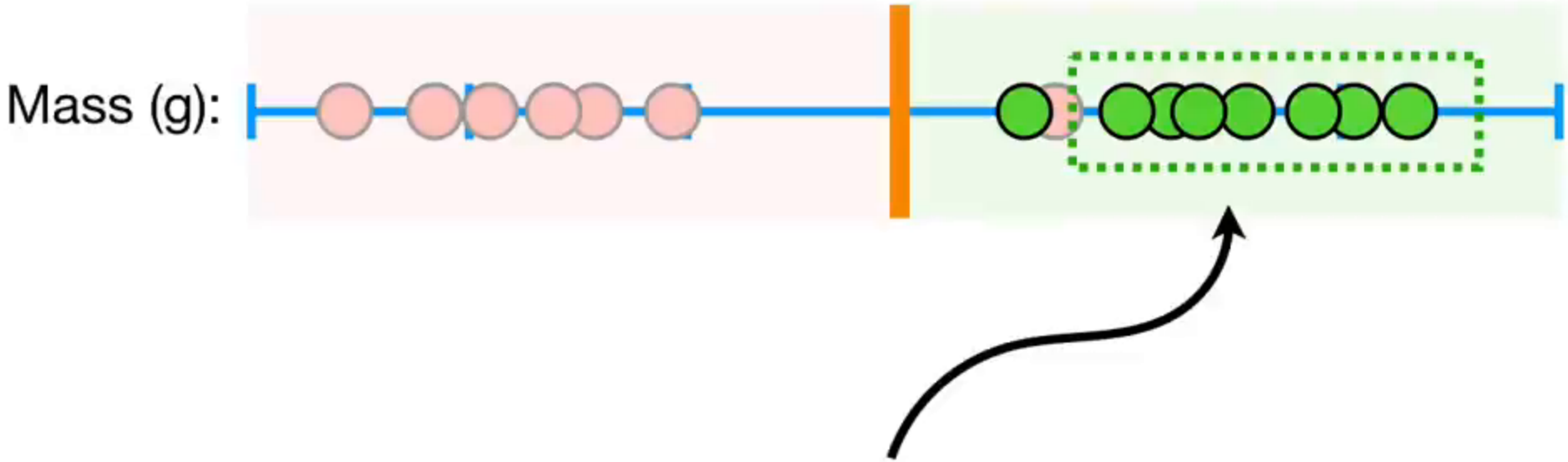
So **Maximal Margin Classifiers** are *super sensitive to outliers* in the training data and that makes them pretty lame.

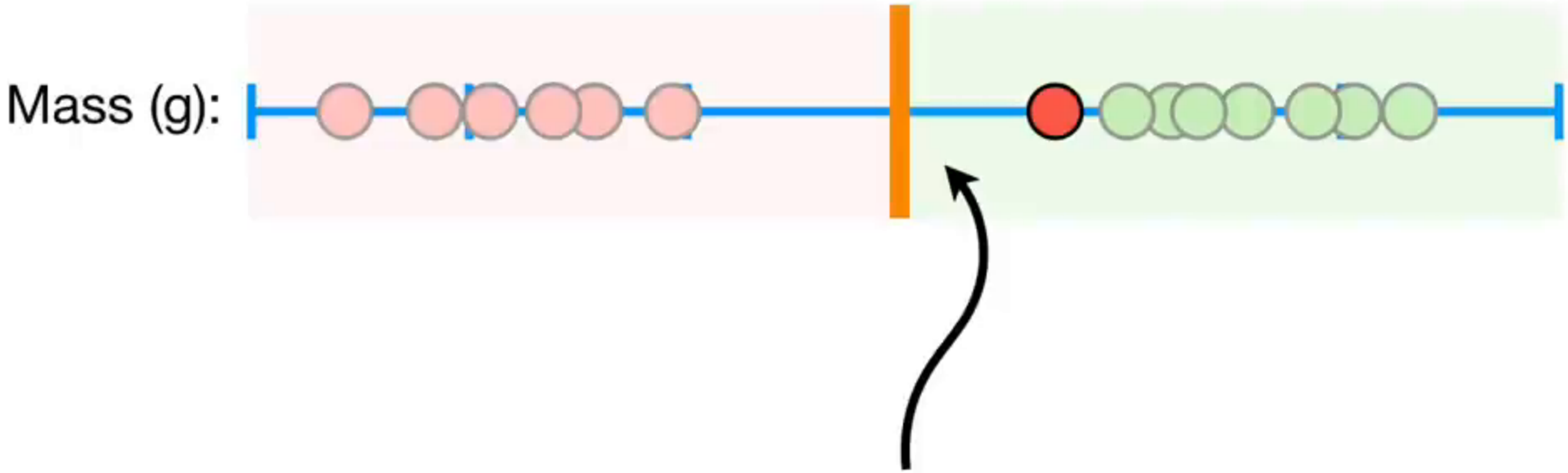
To make a threshold that is not so sensitive to outliers we must **allow misclassifications**.



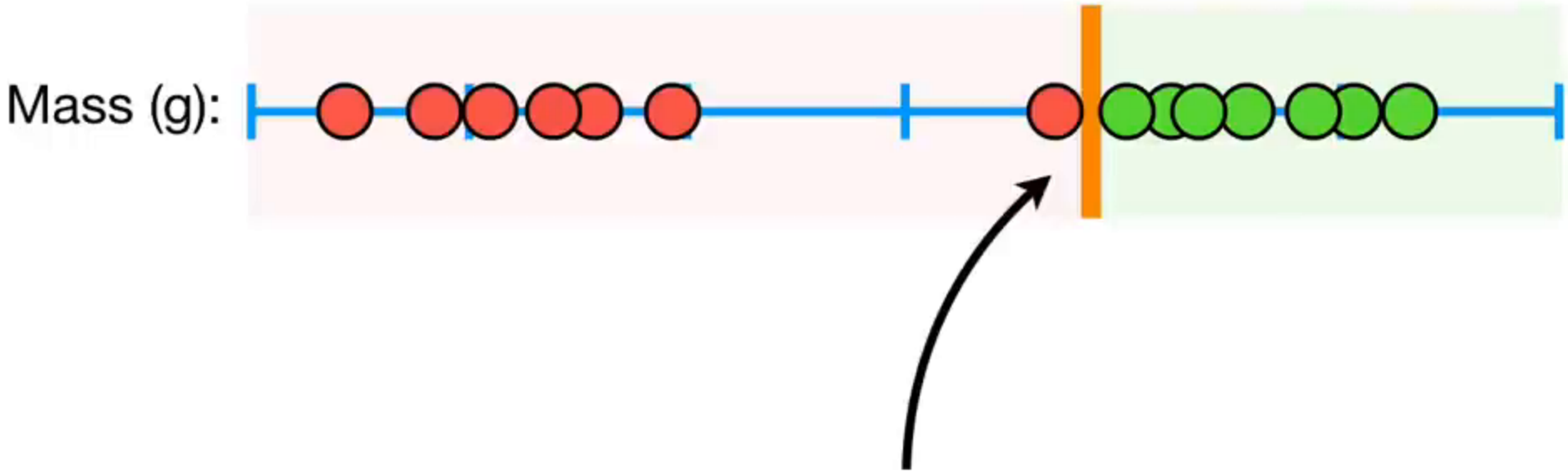
For example, if we put the threshold halfway between these two observations…


Mass (g):
However, now when we get a
new observation here…

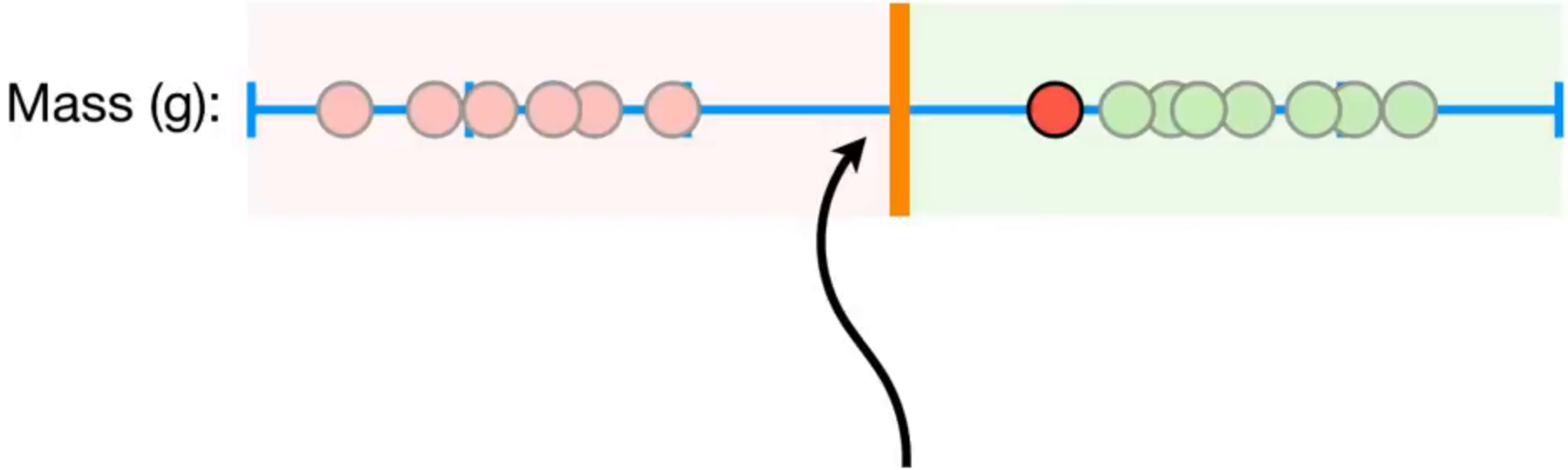

Mass (g):
…and that makes sense
because it is closer to most of
the obese observations.

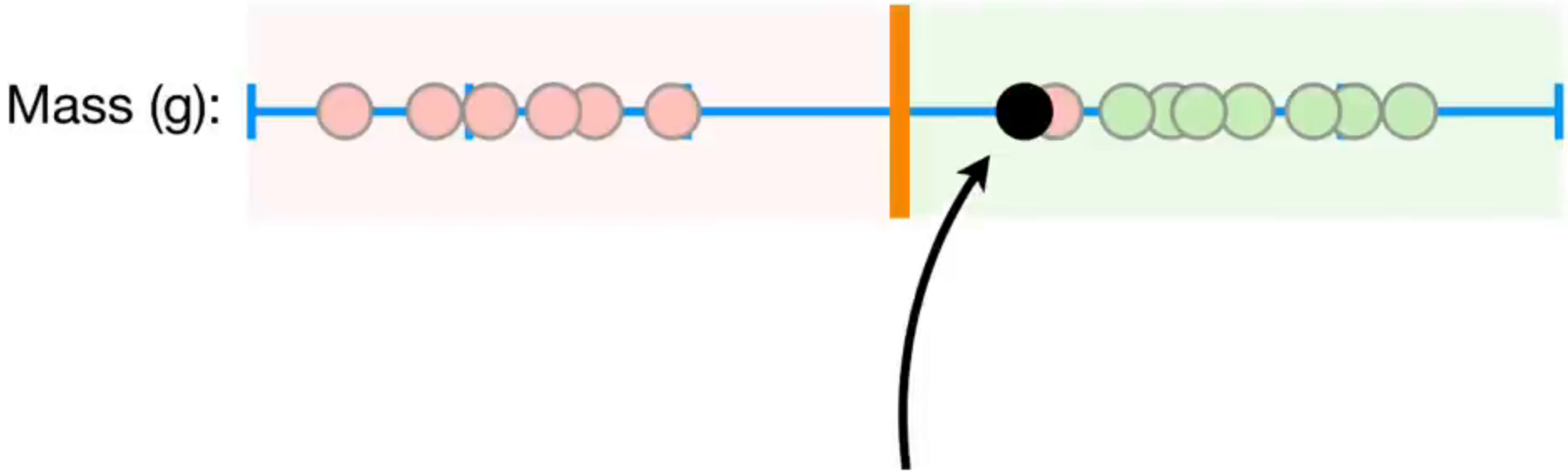


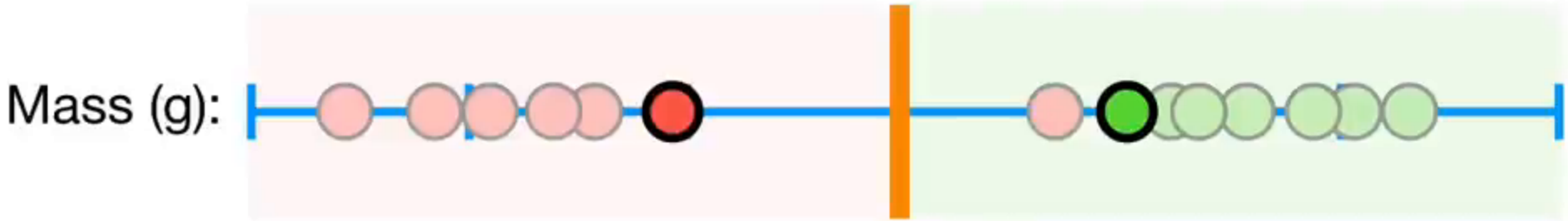
Choosing a threshold that allows misclassifications is an example of the **Bias/Variance Tradeoff** that plagues all of machine learning.



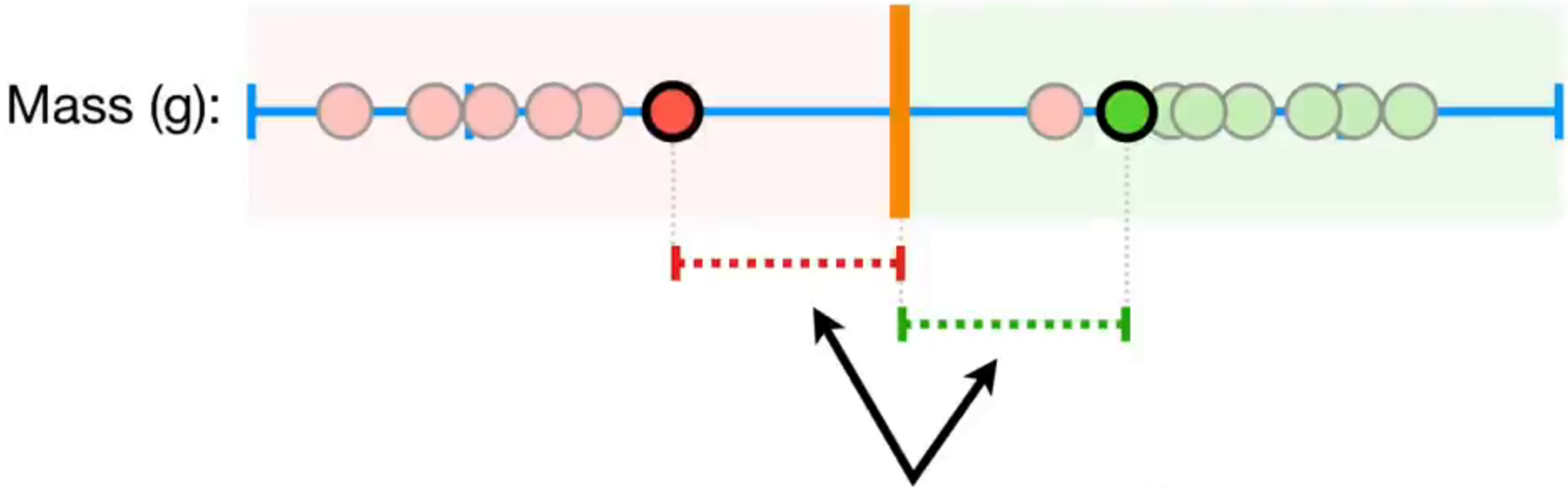
In other words, before we allowed misclassifications, we picked a threshold that was very sensitive to the training data (low bias)…


Mass (g):
In contrast, when we picked a threshold that was less sensitive to the training data and allowed misclassifications (higher bias)…

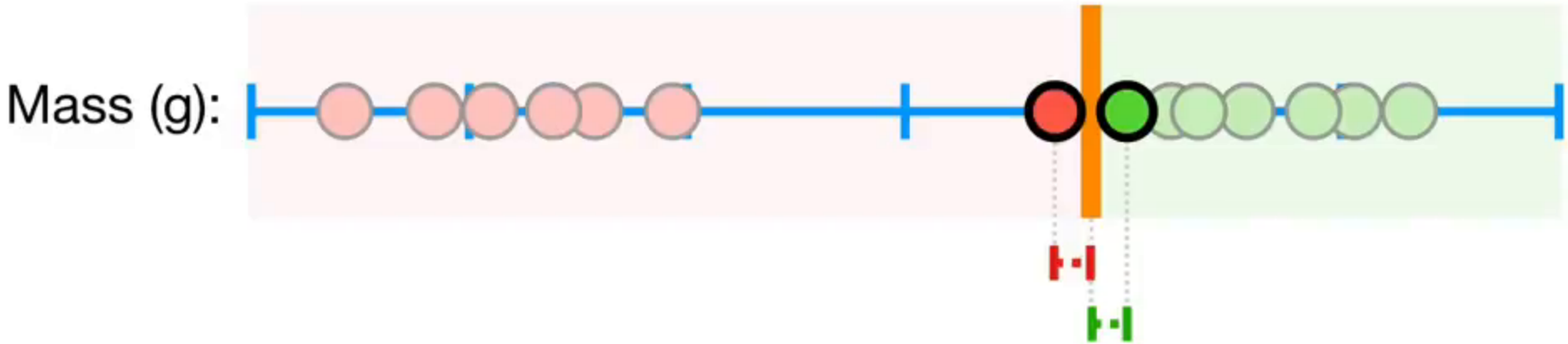

Mass (g):
…it performed better when we got
new data (low variance).



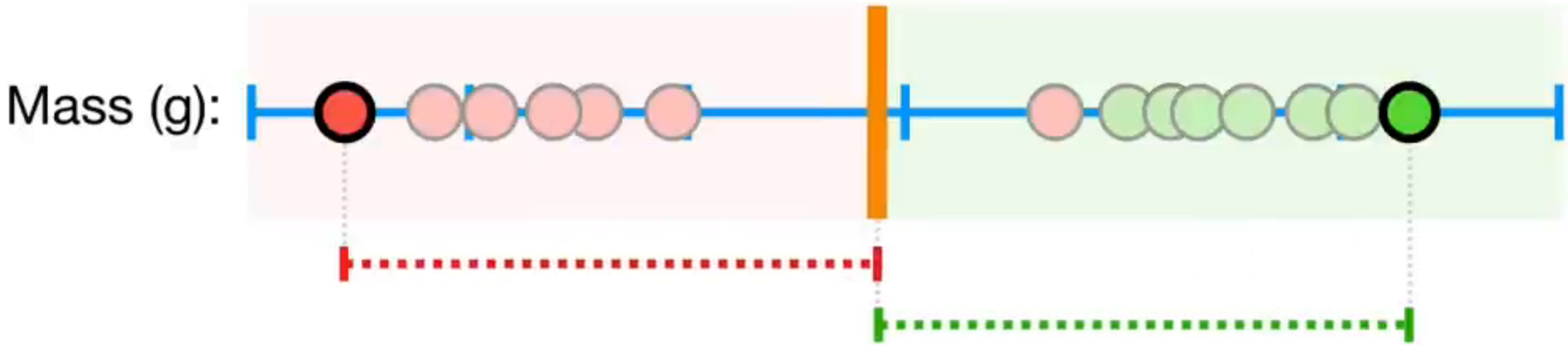
Terminology Alert!!!



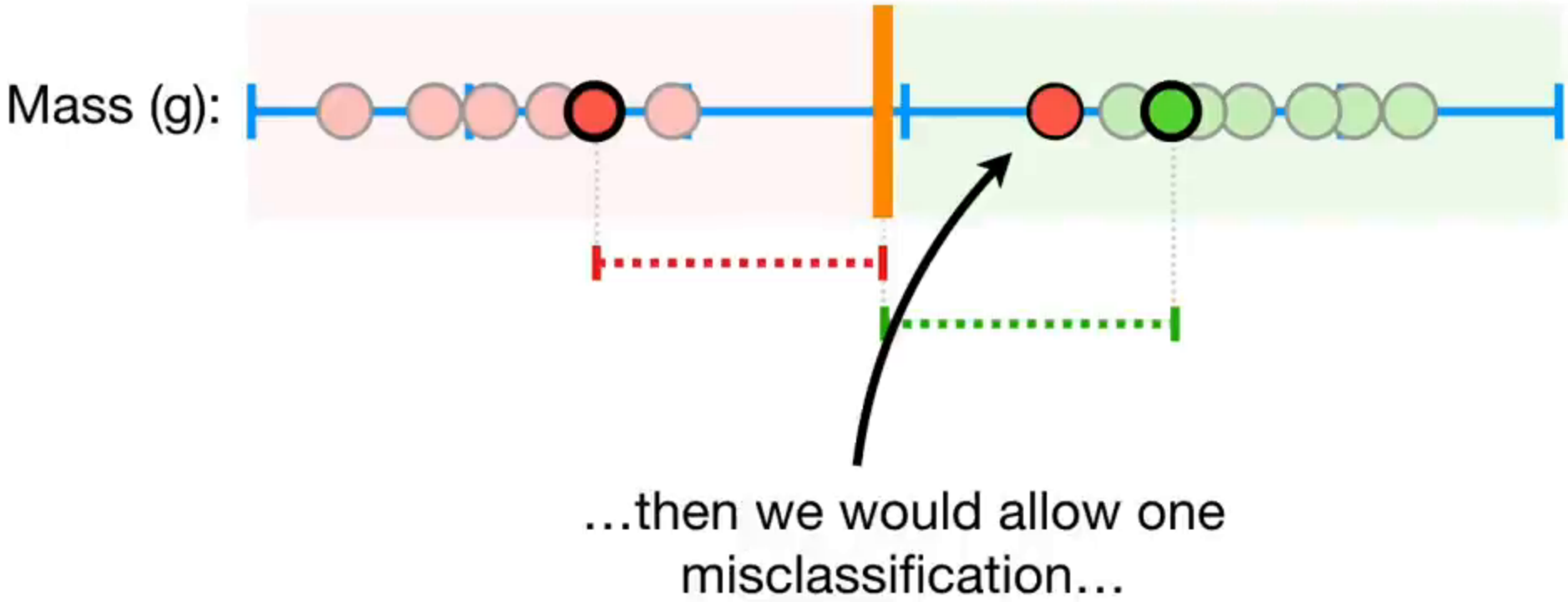
When we allow misclassifications, the distance between the observations and the threshold is called a **Soft Margin**.

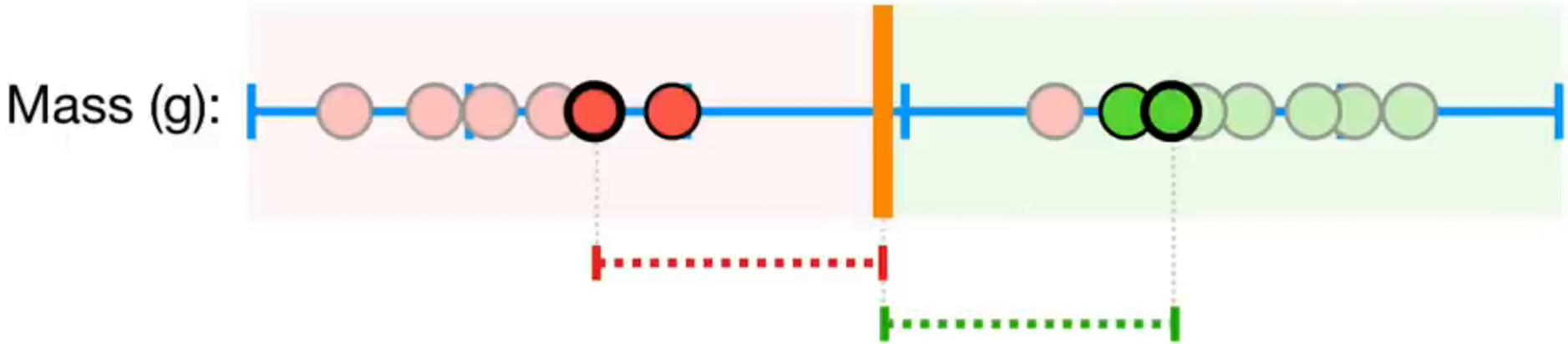


The answer is simple: We use **Cross Validation** to determine how many misclassifications and observations to allow inside of the **Soft Margin** to get the best classification.

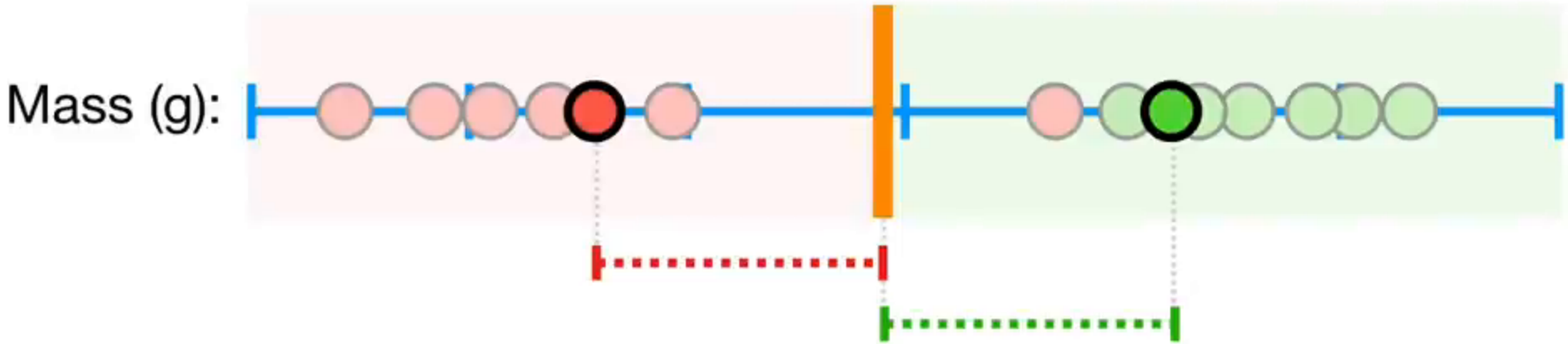


The answer is simple: We use **Cross Validation** to determine how many misclassifications and observations to allow inside of the **Soft Margin** to get the best classification.


Mass (g):
…then we would allow one misclassification…

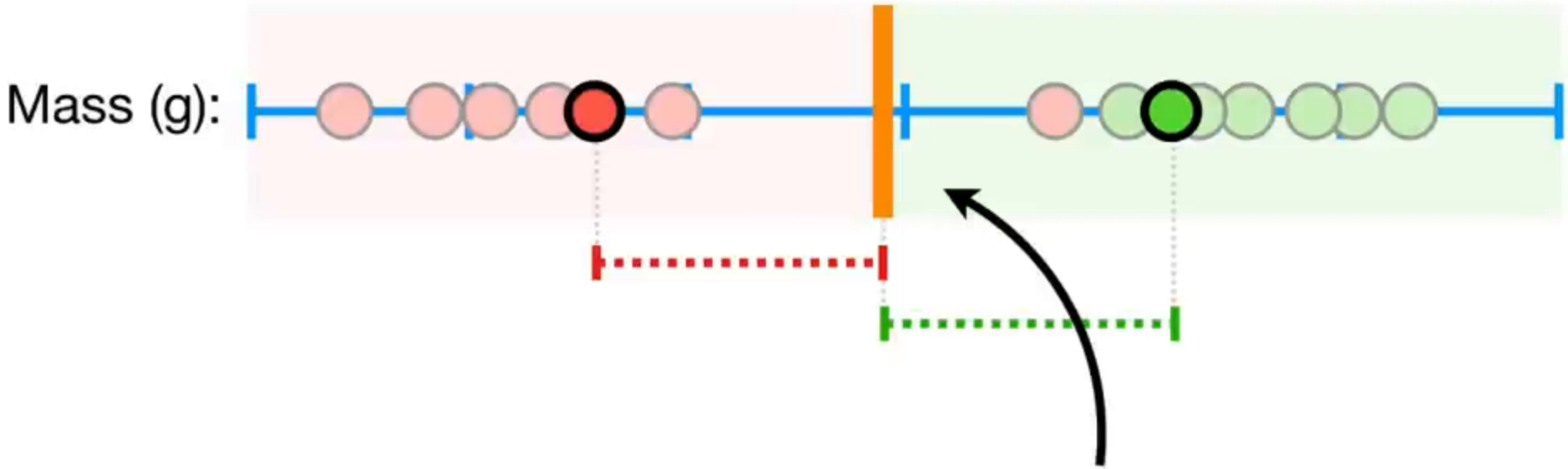


BAM!

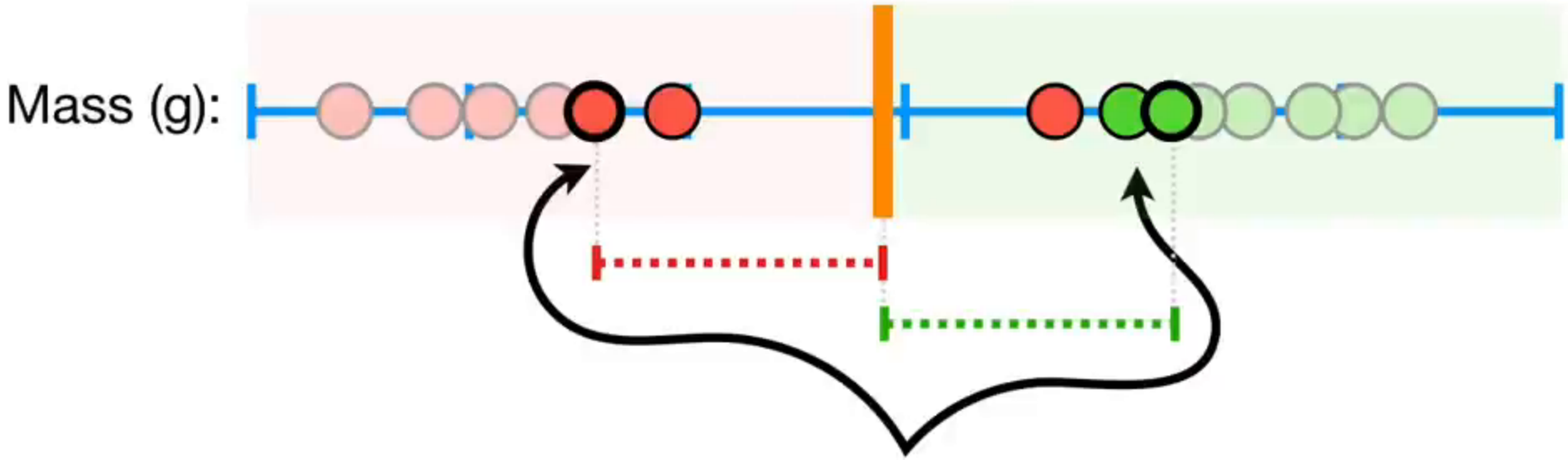


When we use a **Soft Margin** to determine the location of a threshold…

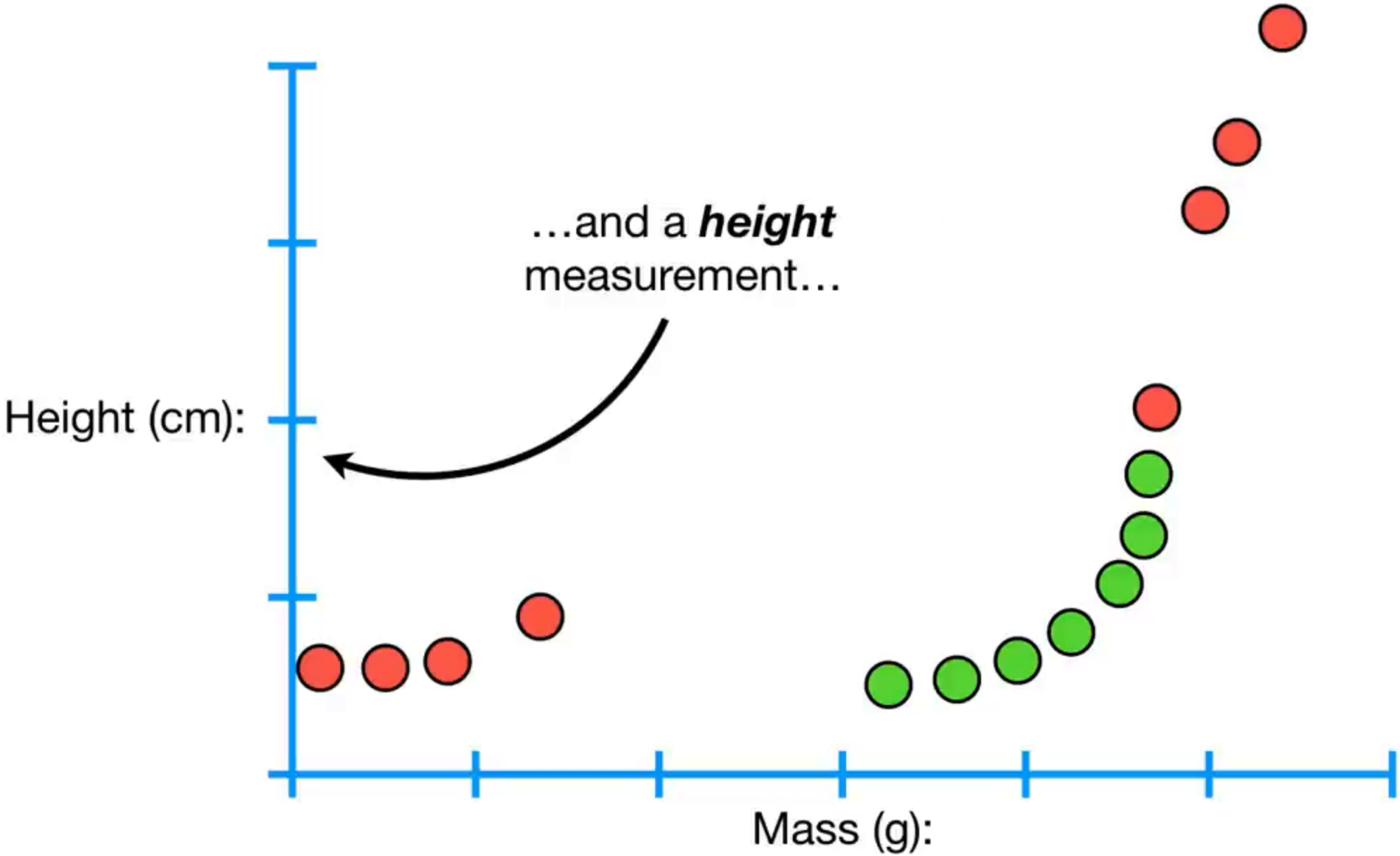
(Brace yourself!!!! Terminology alert!)

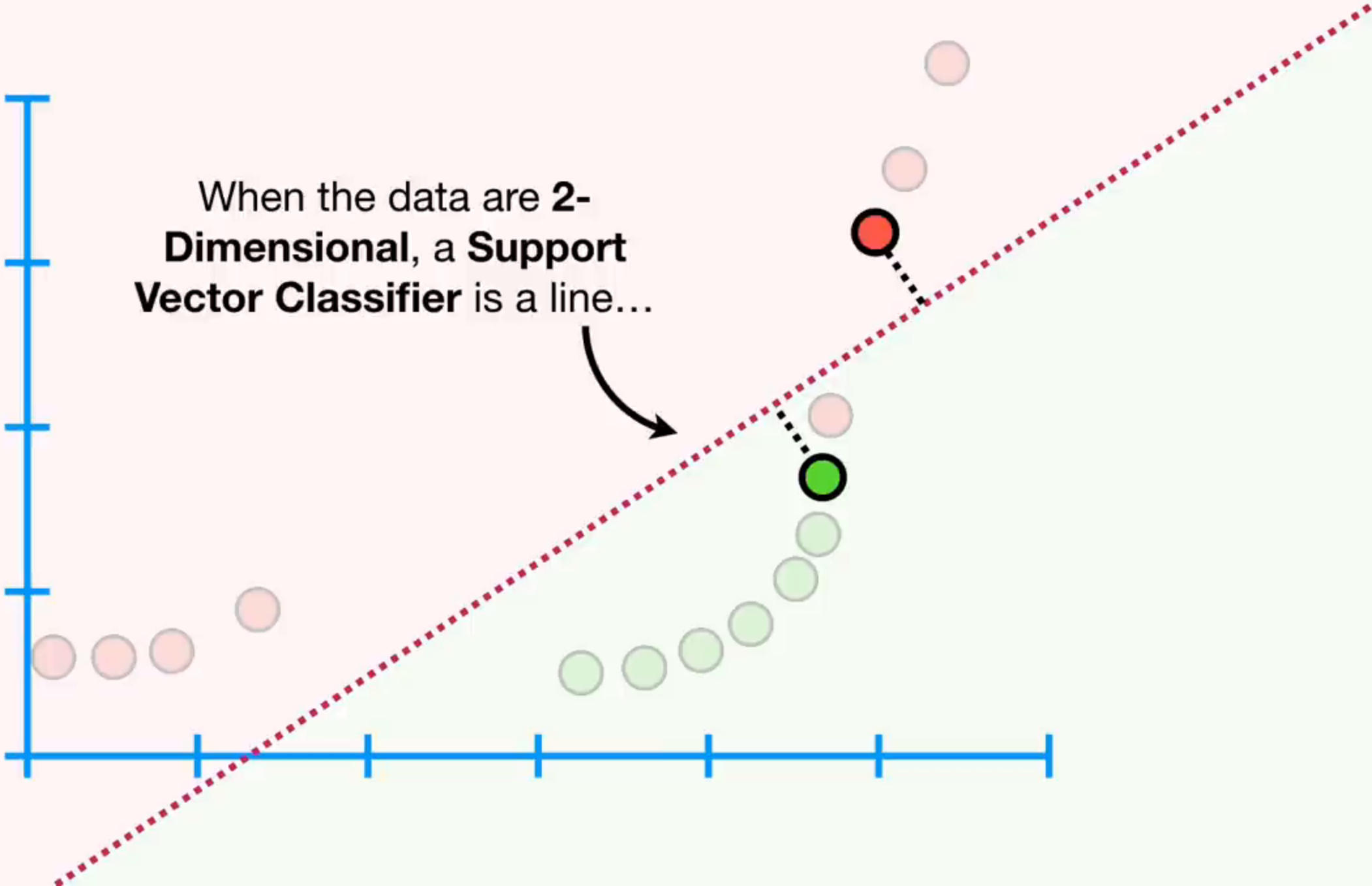


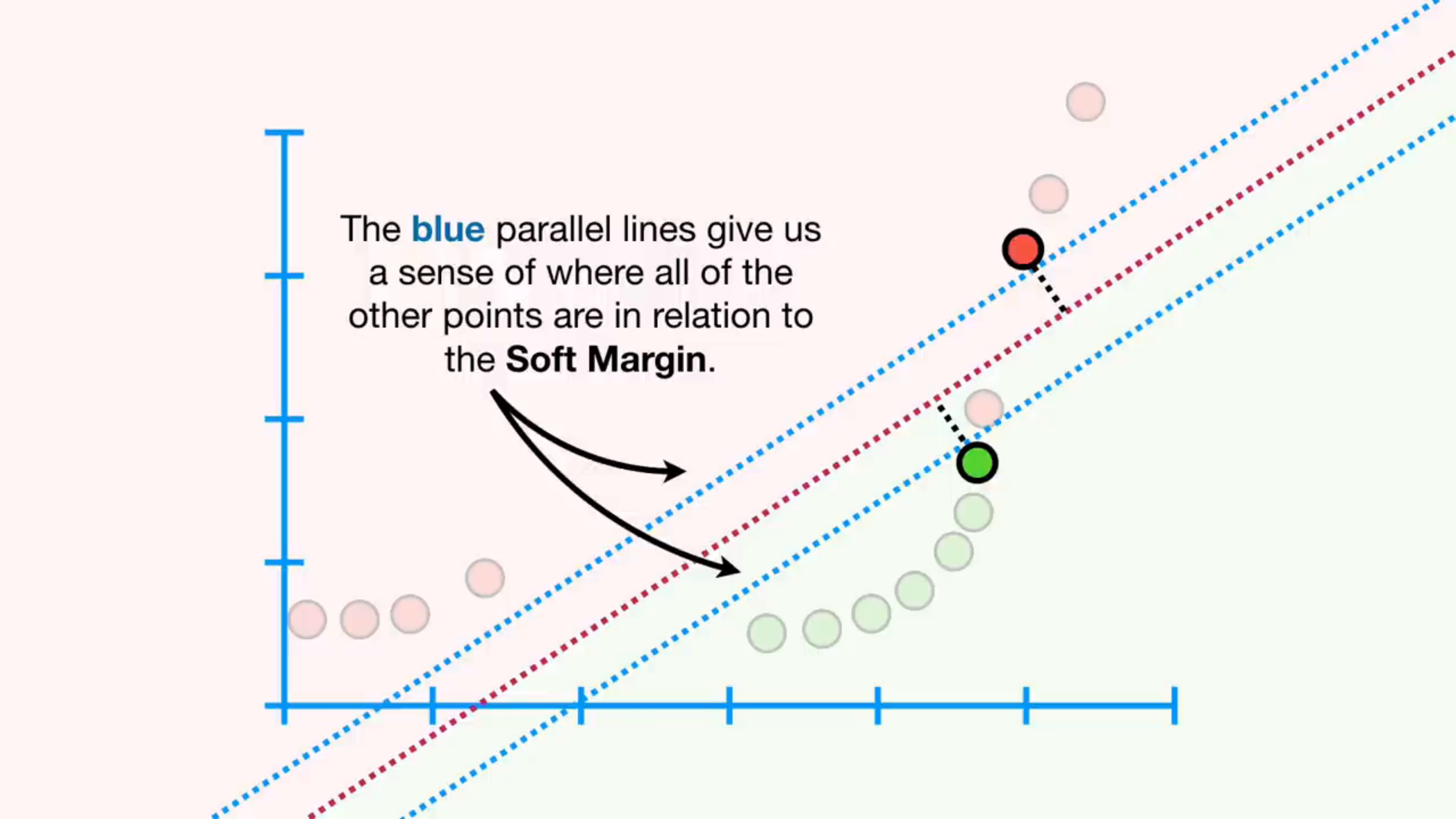
…then we are using a **Soft Margin Classifier** aka a **Support Vector Classifier** to classify observations.

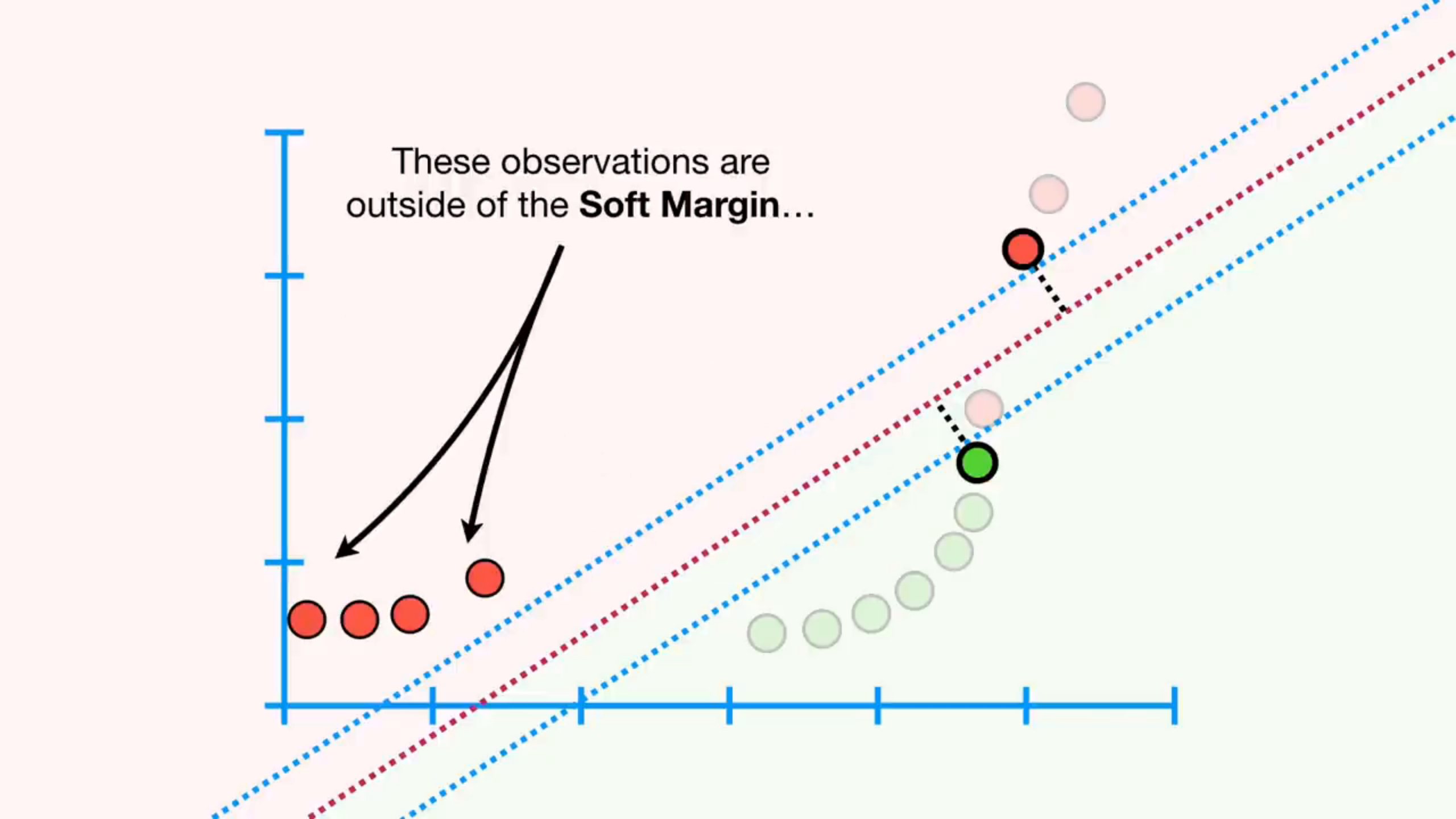


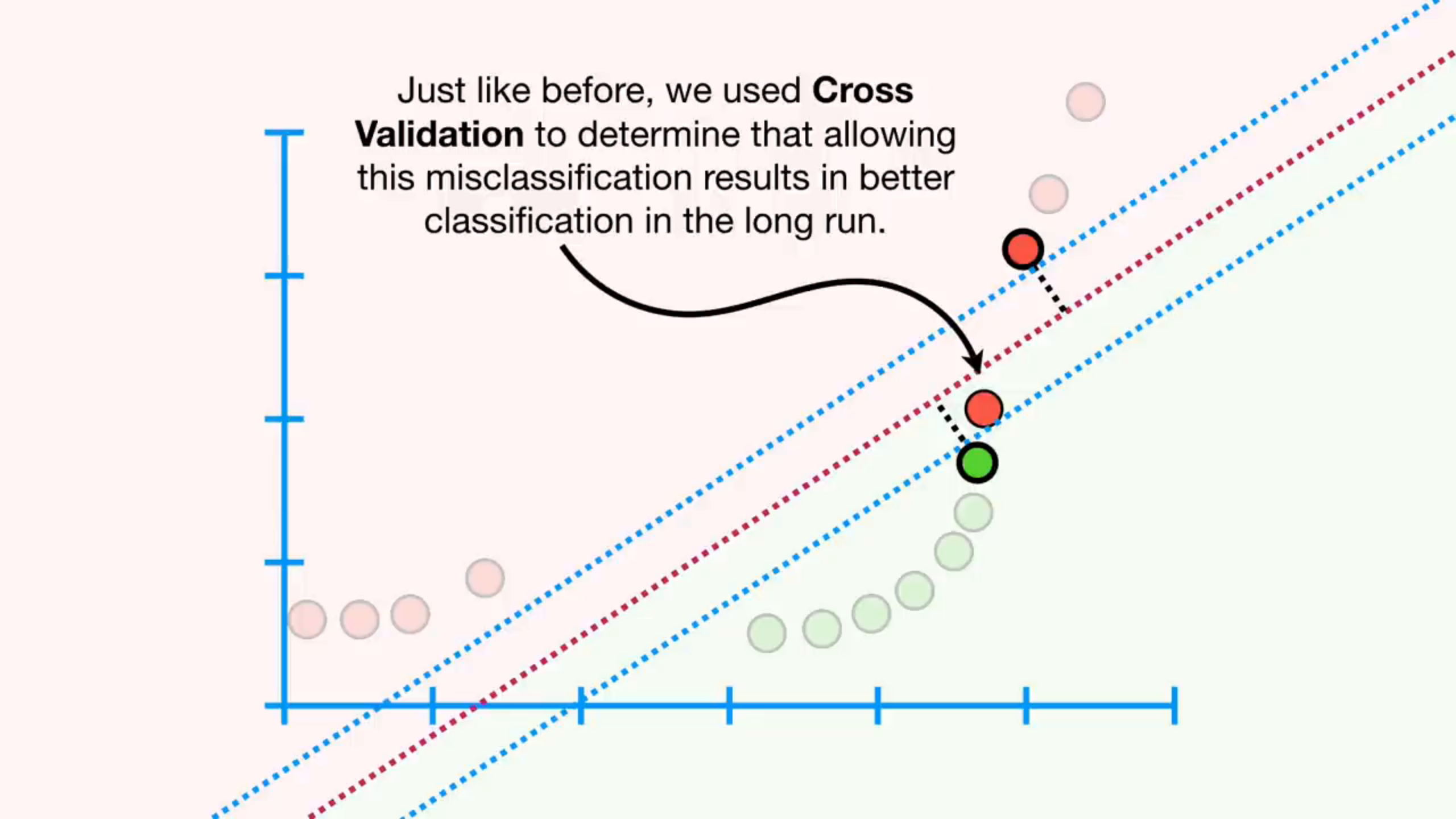
The name **Support Vector Classifier** comes from the fact that the observations on the edge *and within* the **Soft Margin** are called **Support Vectors**.

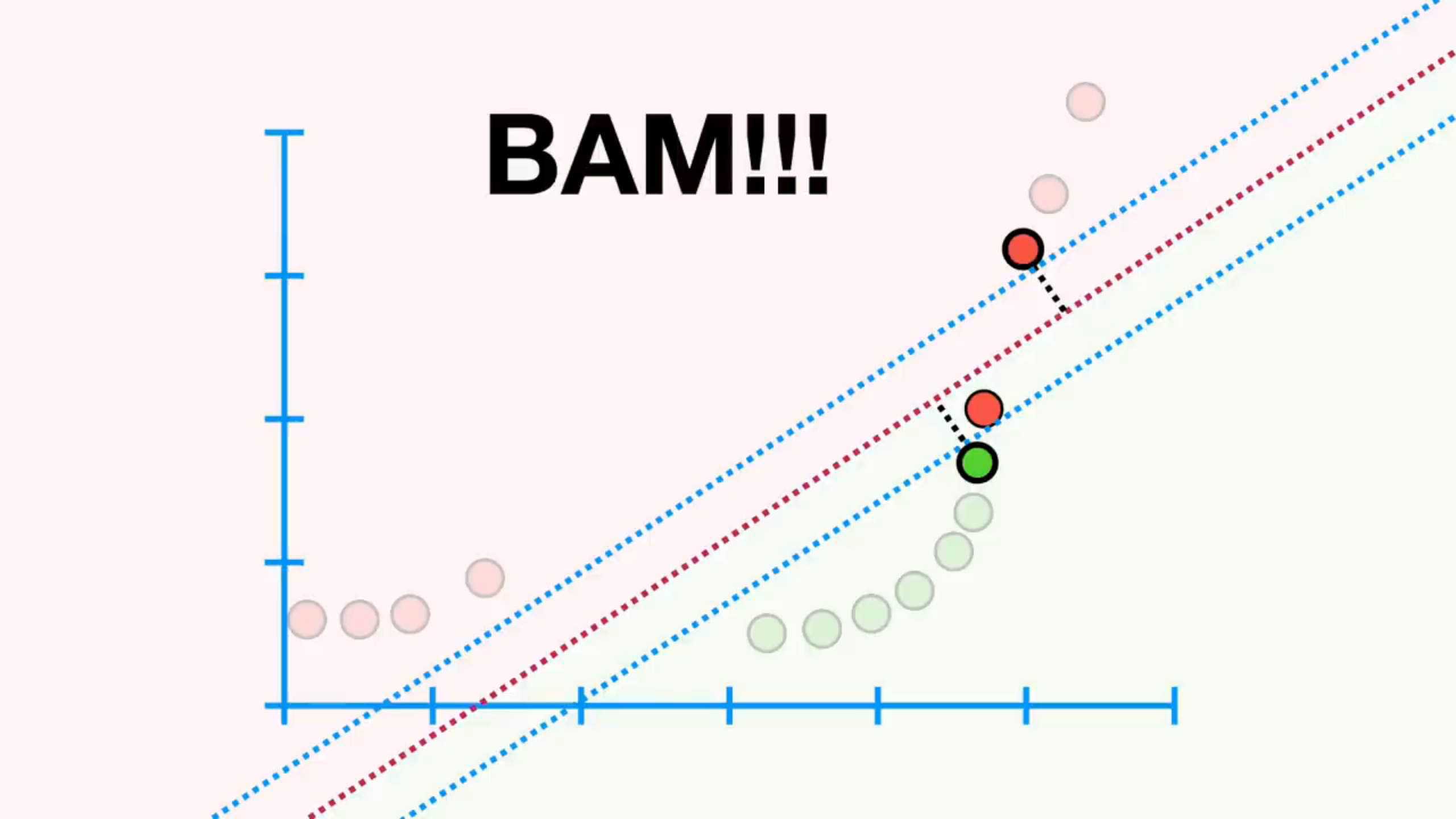

…and a *height* measurement…
Height (cm):
Mass (g):

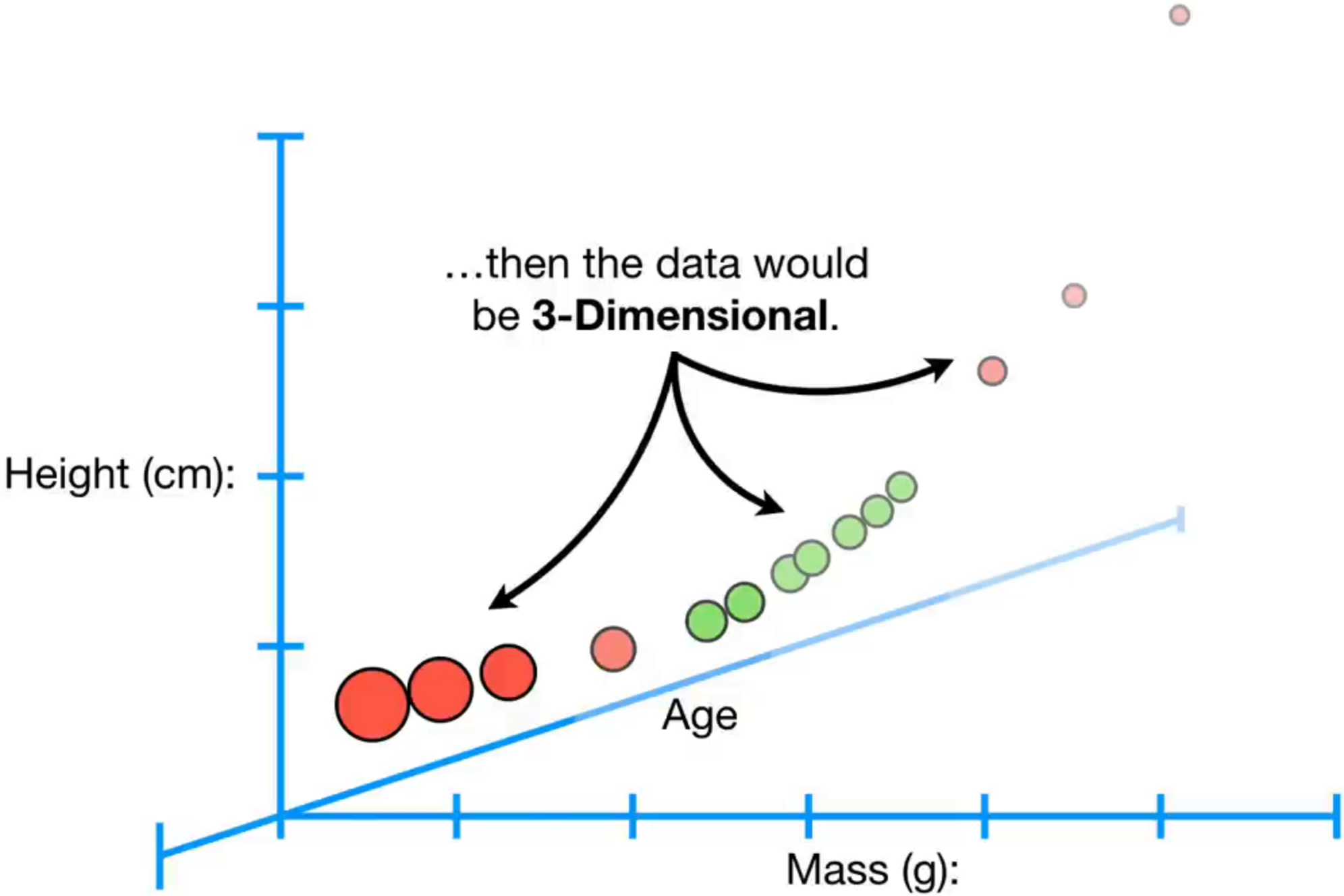

When the data are **2-Dimensional**, a **Support Vector Classifier** is a line…

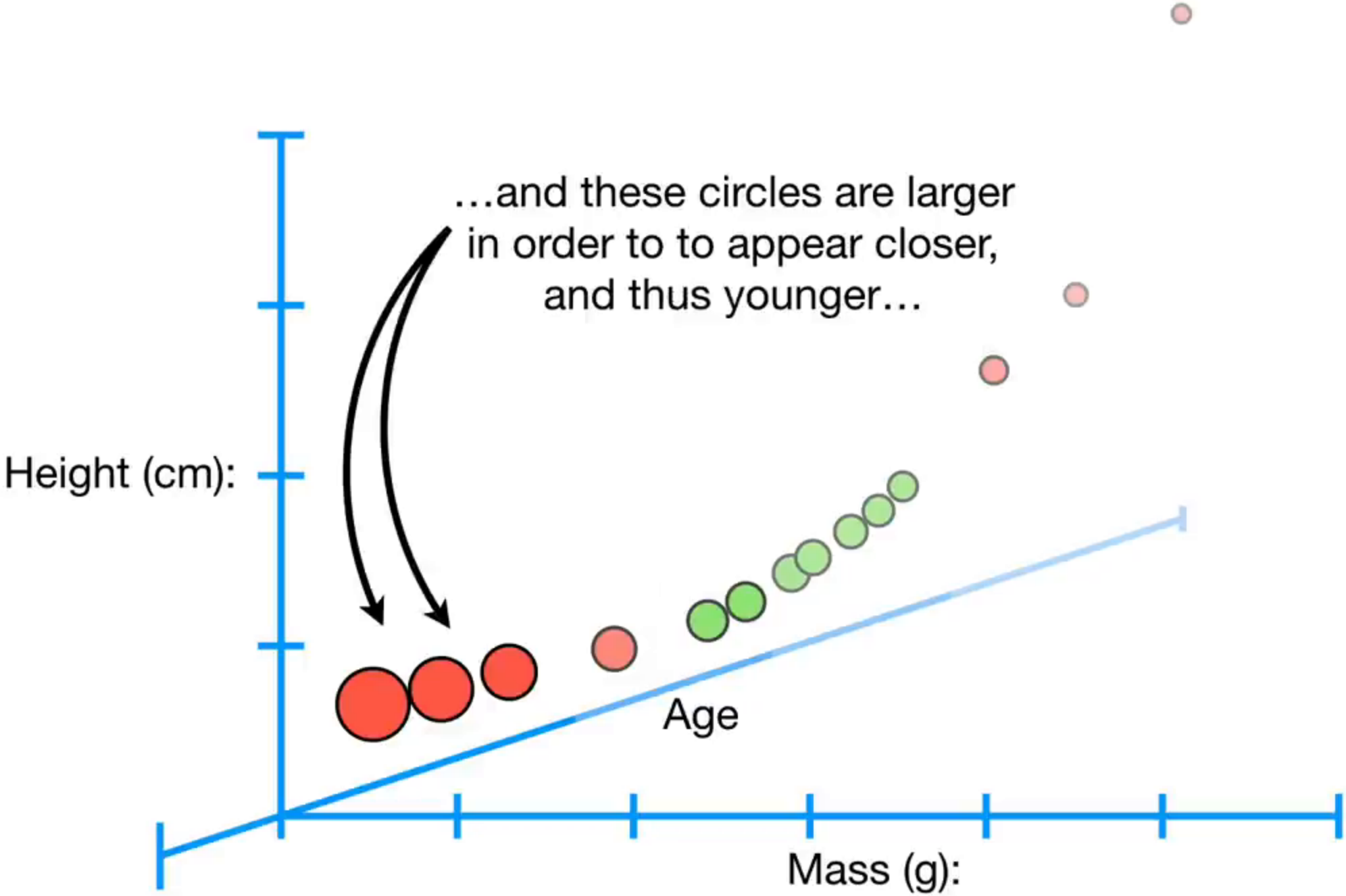

The blue parallel lines give us a sense of where all of the other points are in relation to the Soft Margin.

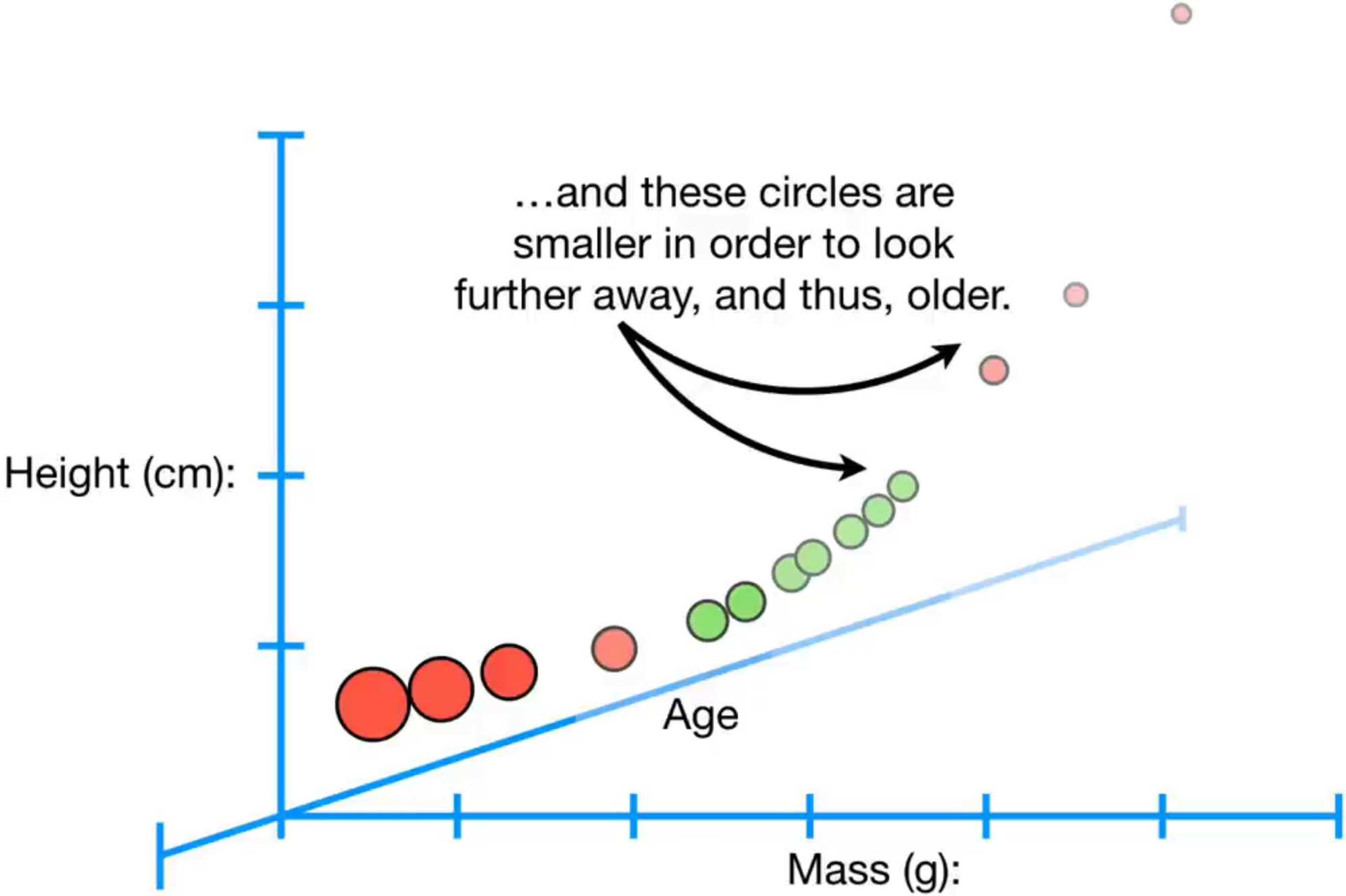

These observations are
outside of the **Soft Margin**…

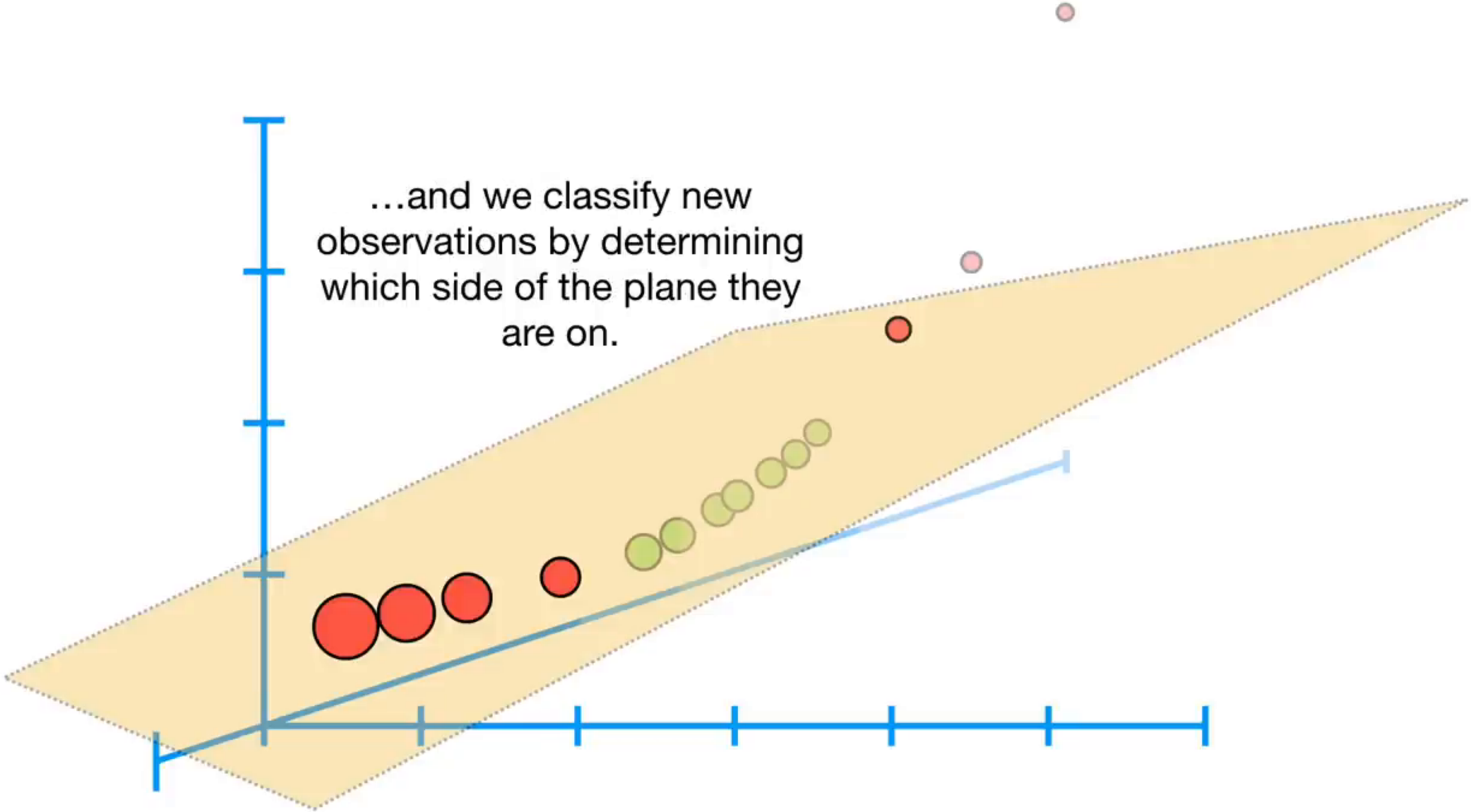

Just like before, we used **Cross Validation** to determine that allowing this misclassification results in better classification in the long run.

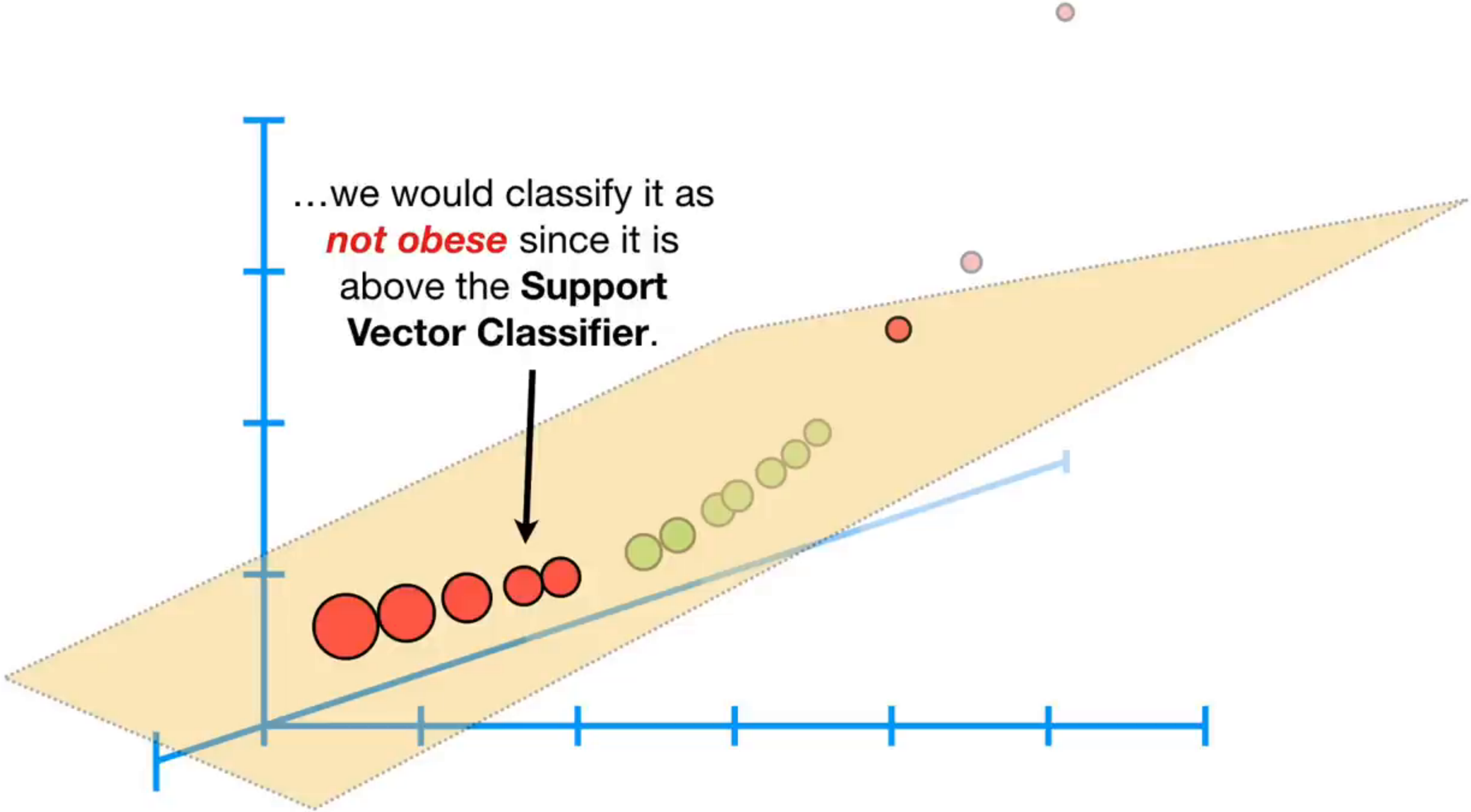

BAM!!!


…then the data would
be **3-Dimensional**.
Height (cm):
Age
Mass (g):


…and these circles are larger
in order to to appear closer,
and thus younger…
Height (cm):
Age
Mass (g):


…and these circles are
smaller in order to look
further away, and thus, older.
Height (cm):
Age
Mass (g):


…and we classify new observations by determining which side of the plane they are on.

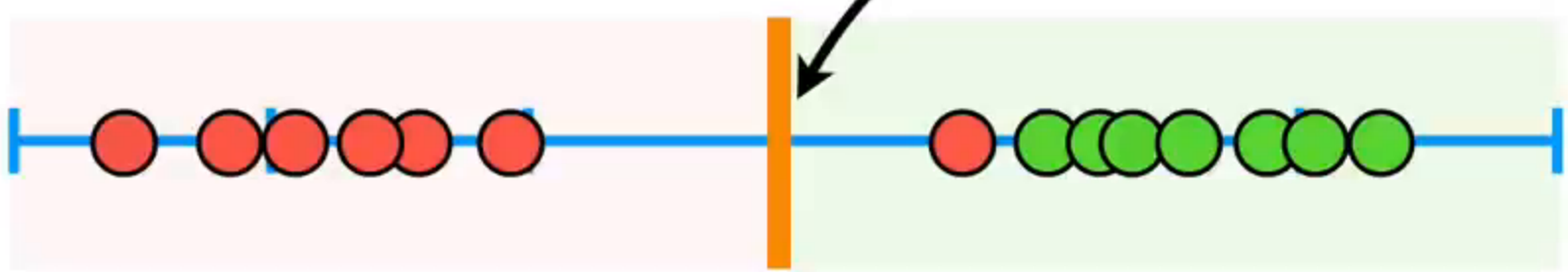

…we would classify it as
not obese since it is
above the Support
Vector Classifier.

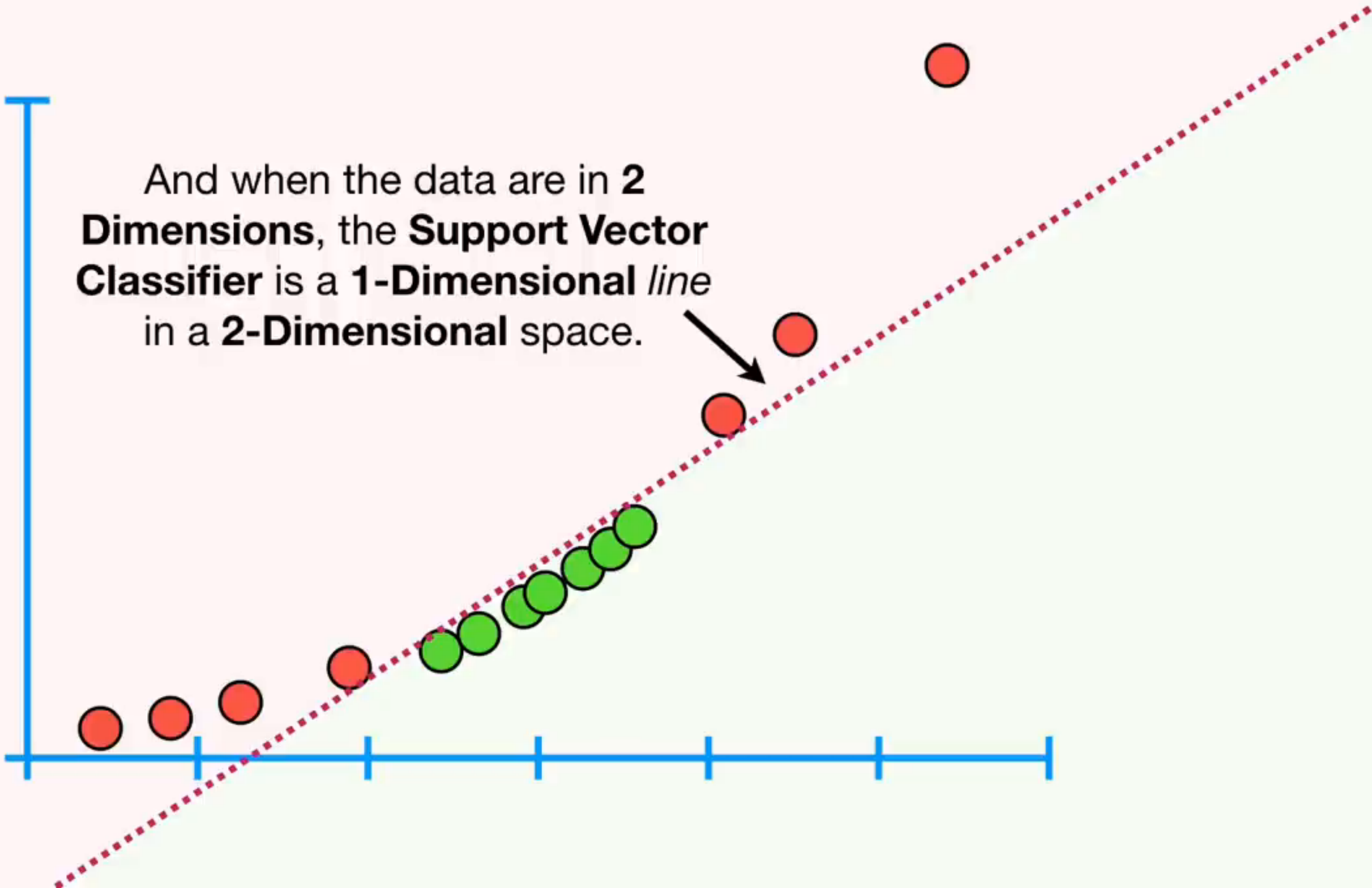
NOTE: If we measured ***mass***, ***height***, ***age*** and ***blood pressure***, then the data would be in **4 Dimensions**…

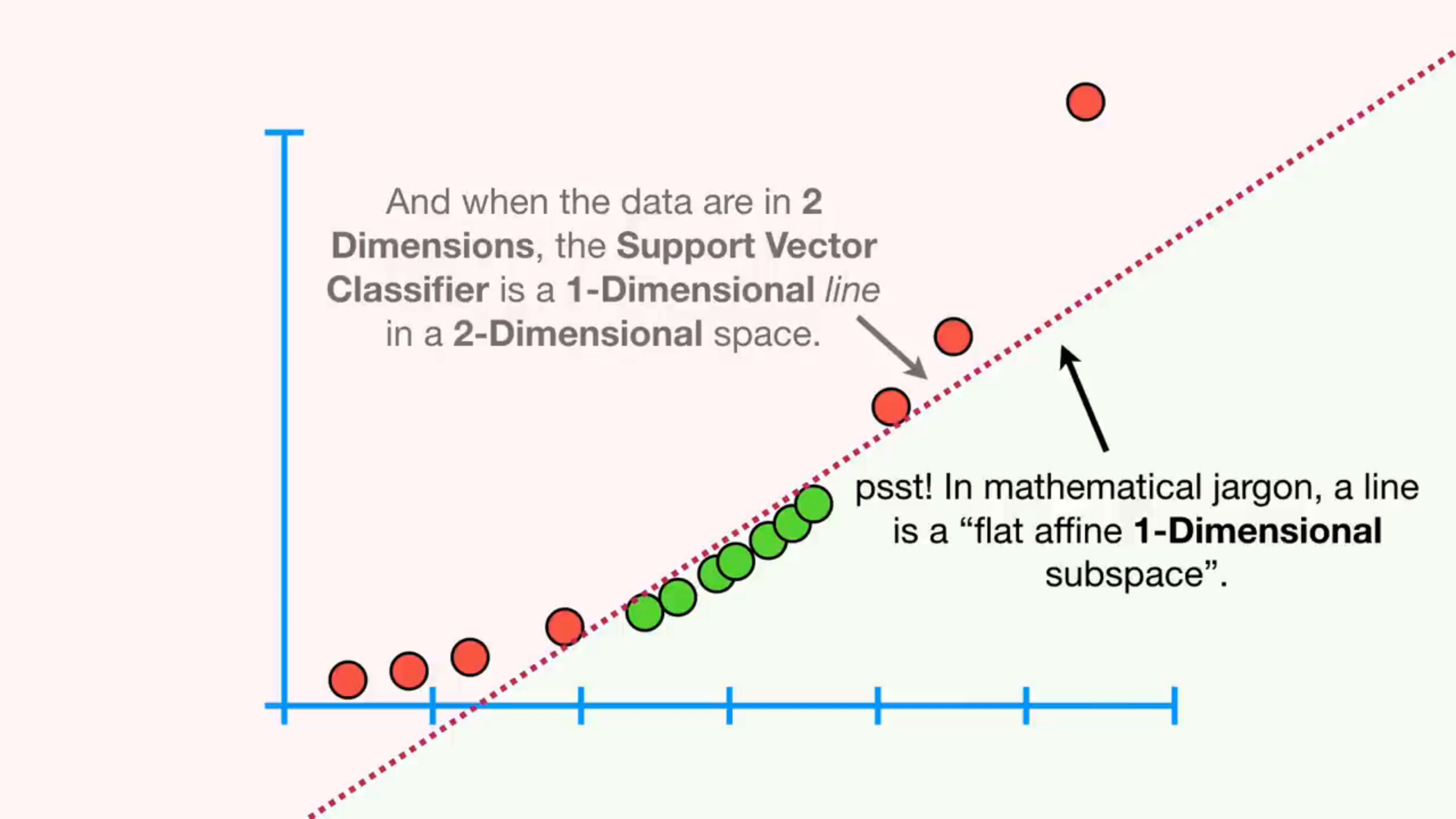
NOTE: If we measured ***mass***, ***height***, ***age*** and ***blood pressure***, then the data would be in **4 Dimensions**…

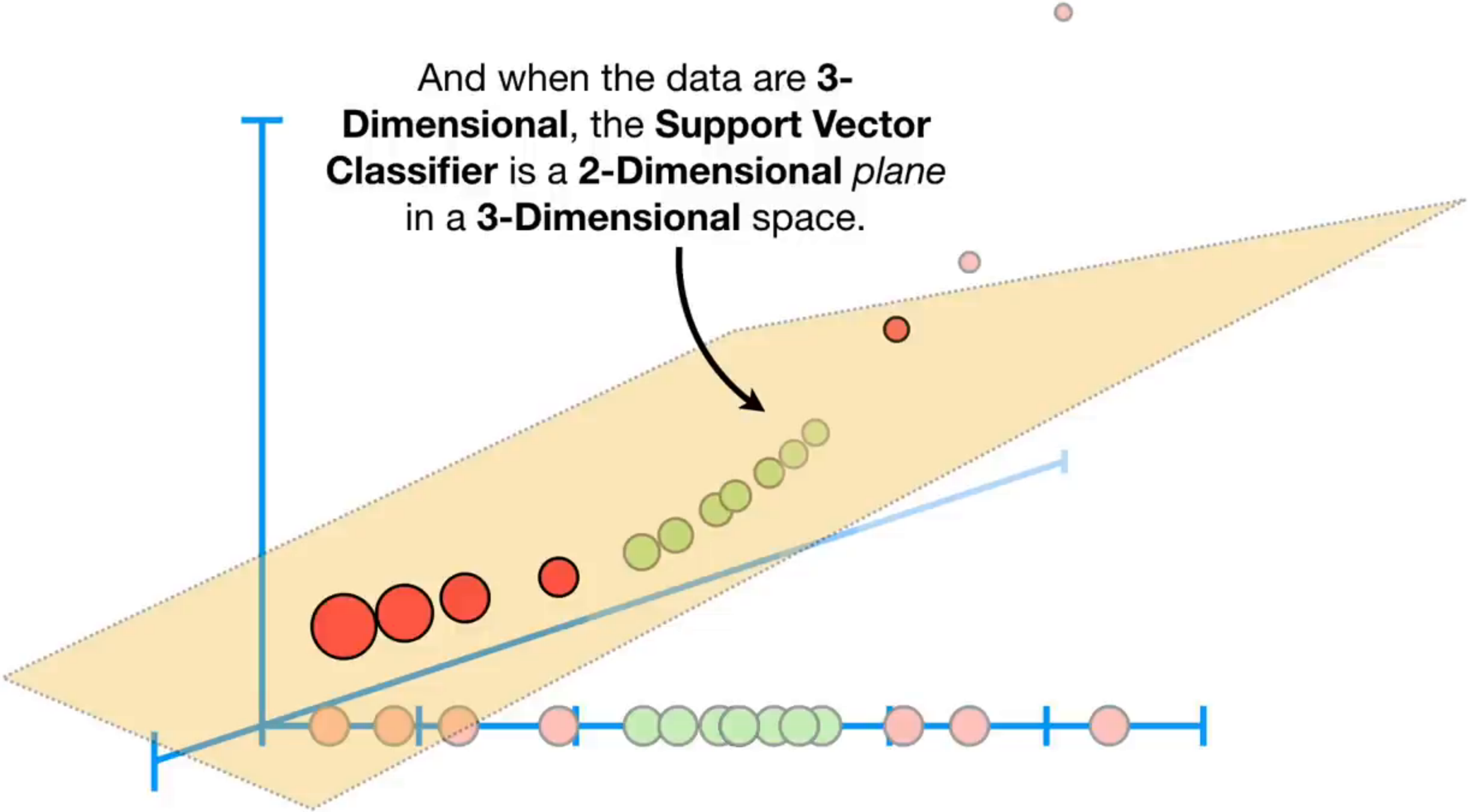
…and I don't know how to draw a **4-Dimensional** graph…

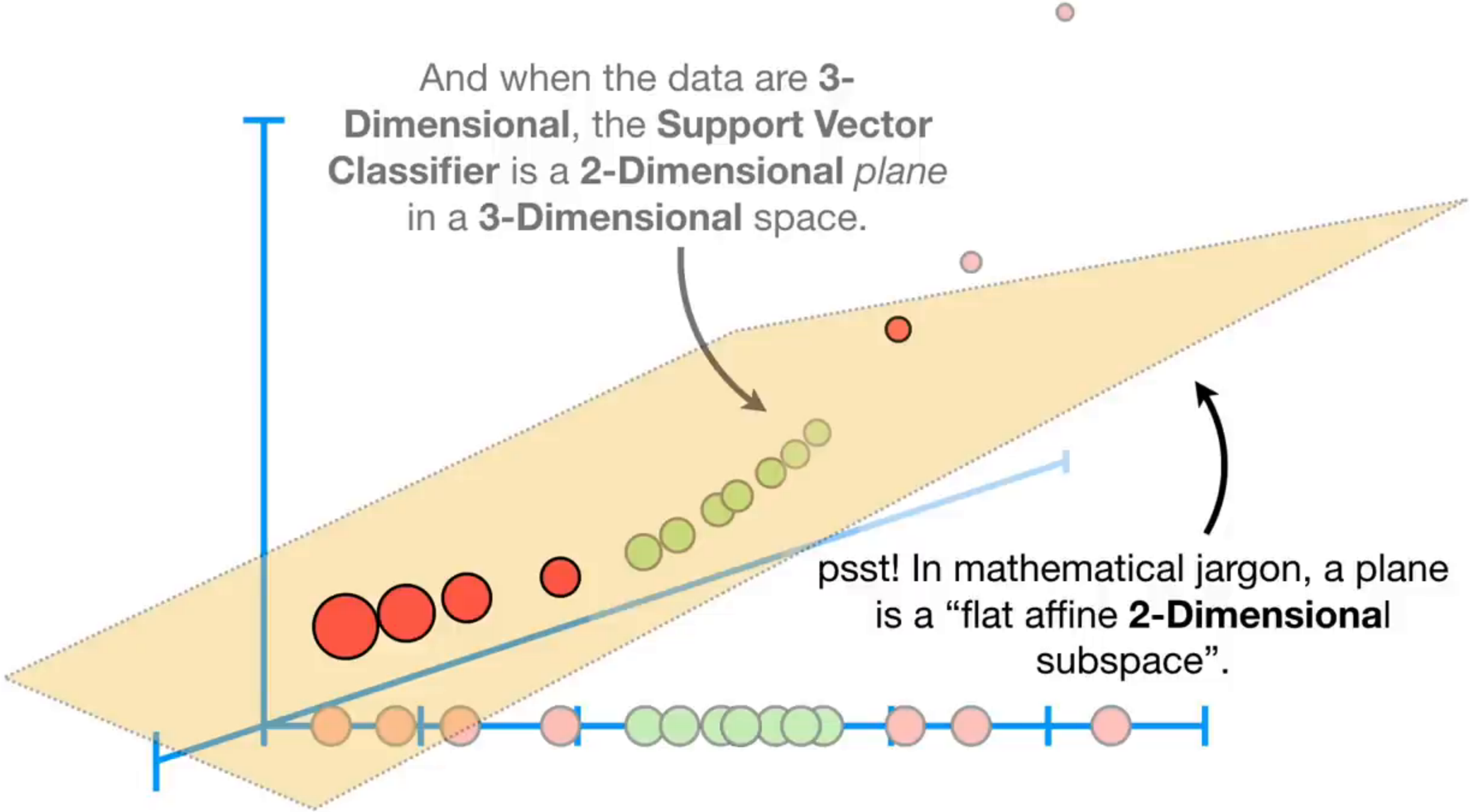
But we know that when the data are **1-Dimensional**, the **Support Vector Classifier** is a single *point* on a **1-Dimensional** number line.




And when the data are in **2 Dimensions**, the **Support Vector Classifier** is a **1-Dimensional** *line* in a **2-Dimensional** space.


And when the data are in **2 Dimensions**, the **Support Vector Classifier** is a **1-Dimensional** *line* in a **2-Dimensional** space.
psst! In mathematical jargon, a line is a “flat affine **1-Dimensional** subspace”.


And when the data are **3-Dimensional**, the **Support Vector Classifier** is a **2-Dimensional** *plane* in a **3-Dimensional** space.

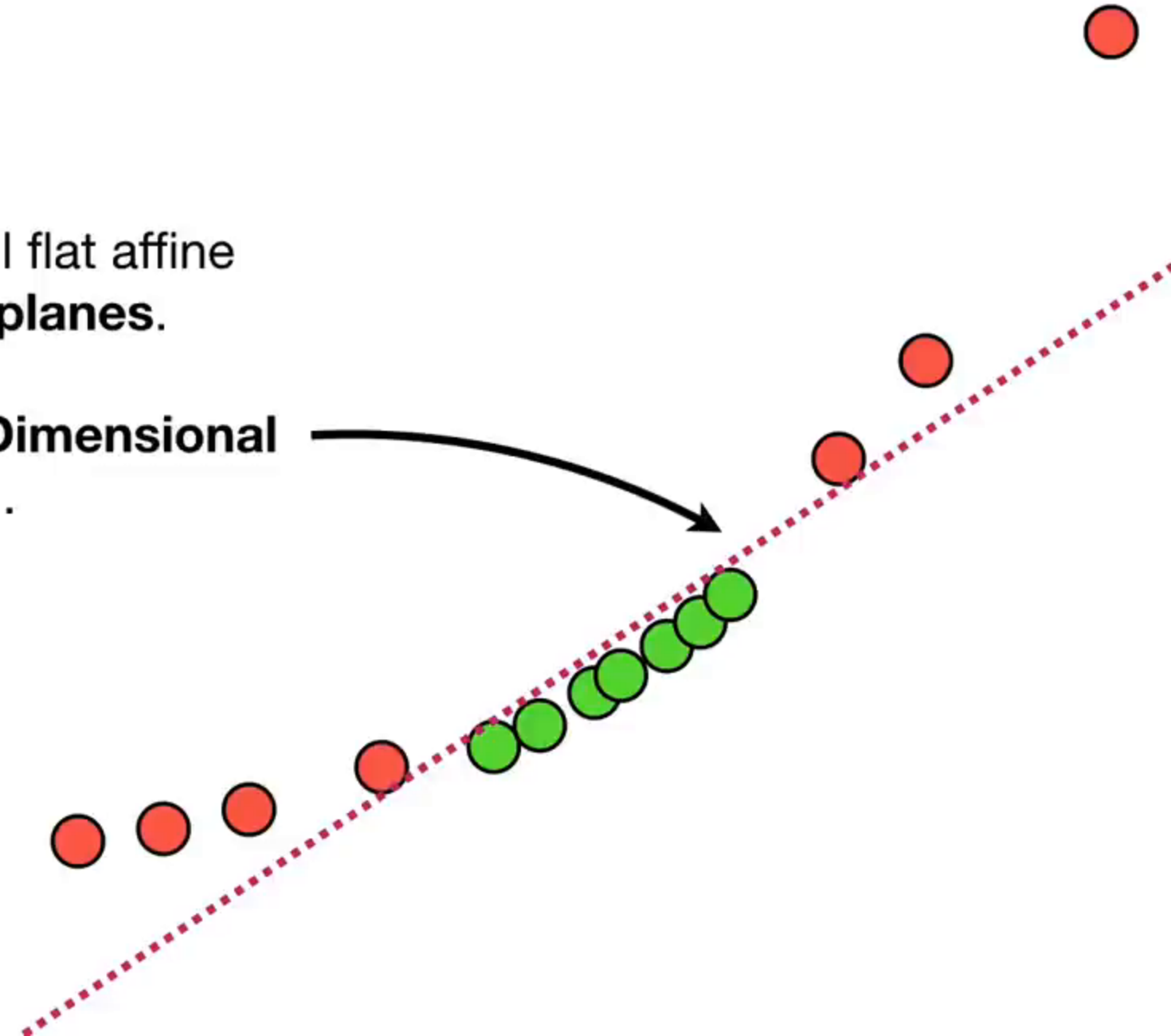

And when the data are **3-Dimensional**, the **Support Vector Classifier** is a **2-Dimensional** *plane* in a **3-Dimensional** space.
psst! In mathematical jargon, a plane is a “flat affine **2-Dimensional** subspace”.

And when the data are in **4 or more Dimensions**, the **Support Vector Classifier** is a *hyperplane*.

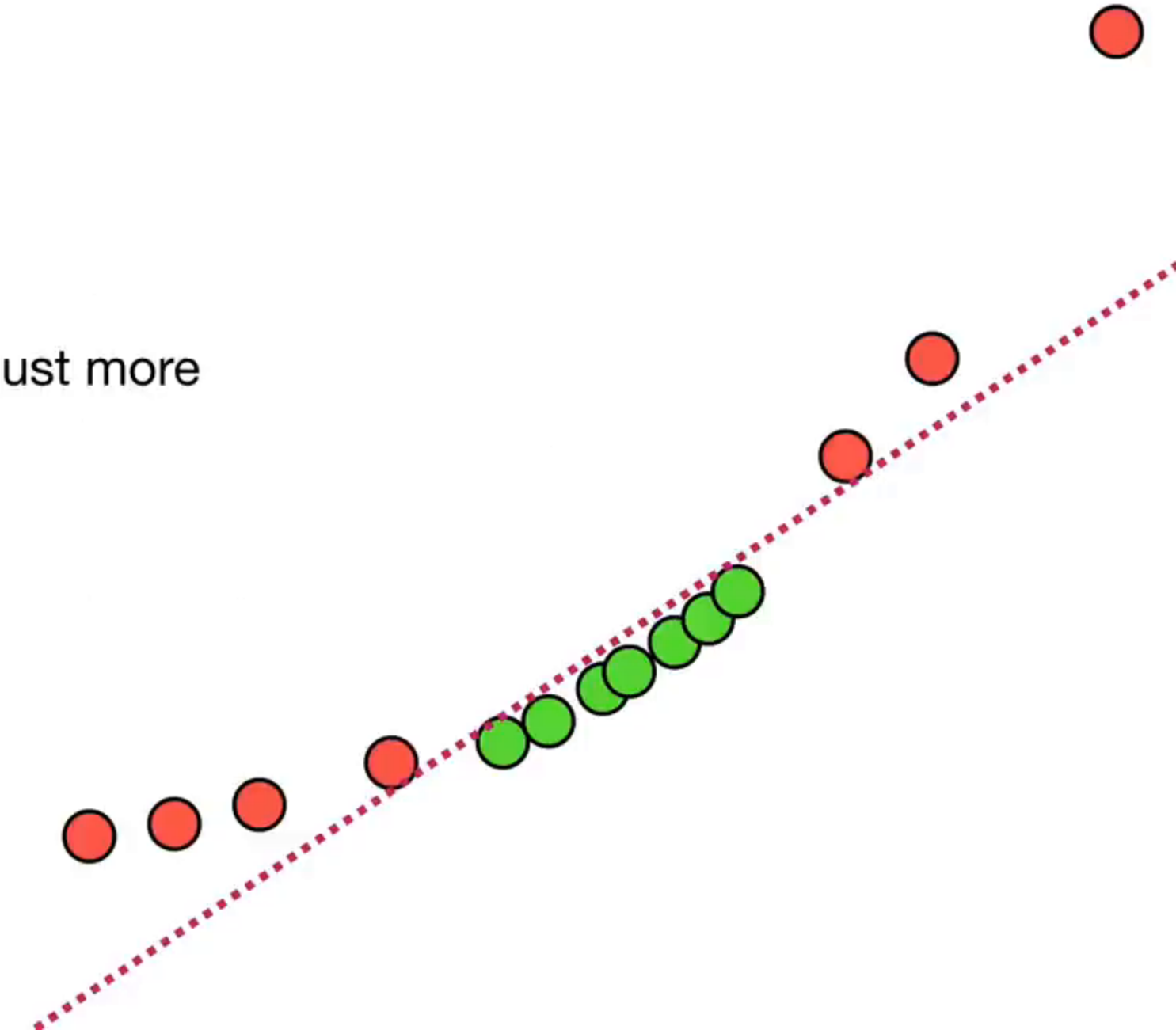
NOTE: Technically speaking, all flat affine subspaces are called **hyperplanes**.

NOTE: Technically speaking, all flat affine subspaces are called **hyperplanes**.

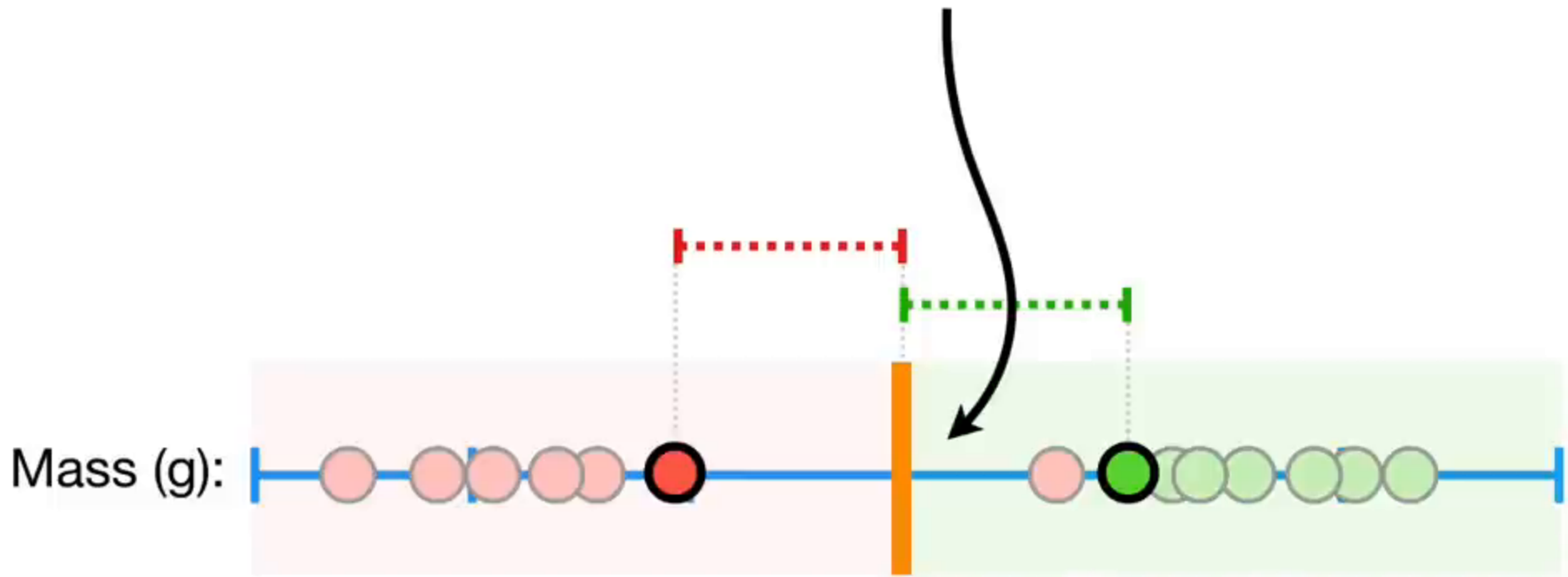
So, technically speaking, this **1-Dimensional** *line* is a **hyperplane**…



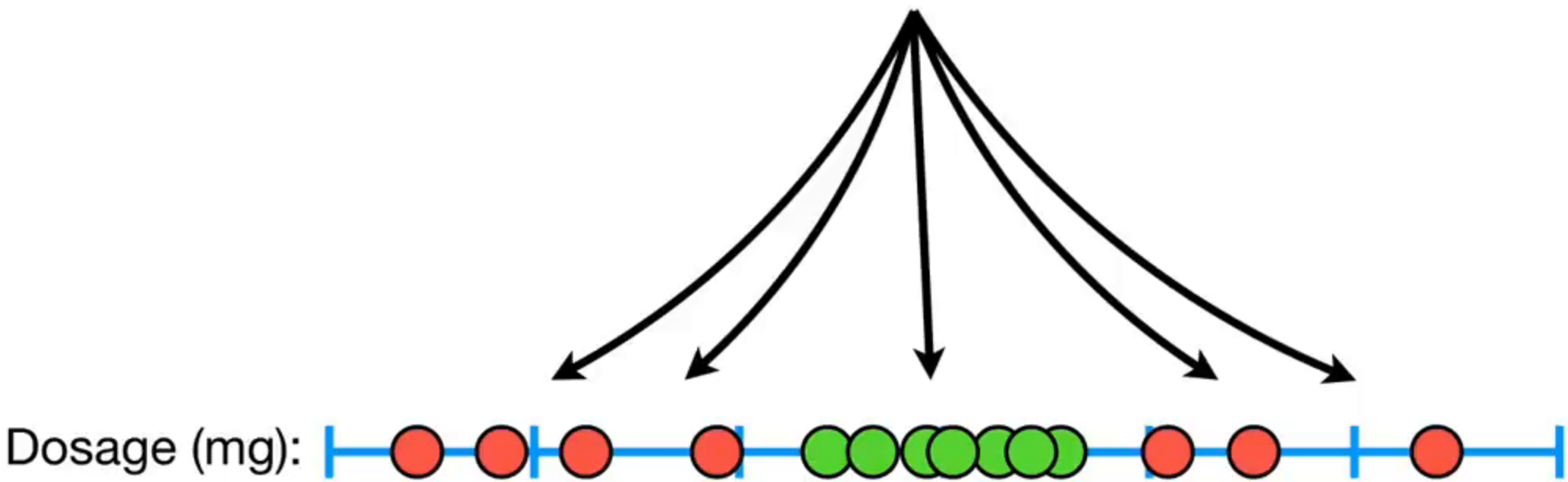
Small bam, because this is just more terminology.



Support Vector Classifiers seem pretty cool because they can handle…

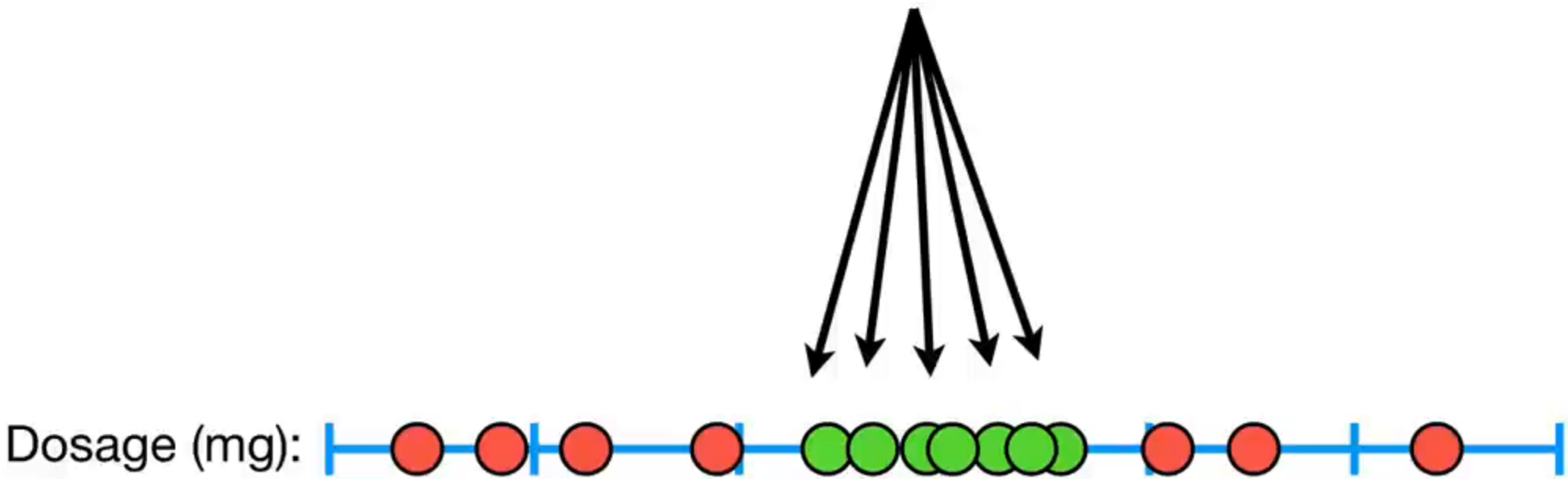


…but what if this was our training data and we had tons of overlap?

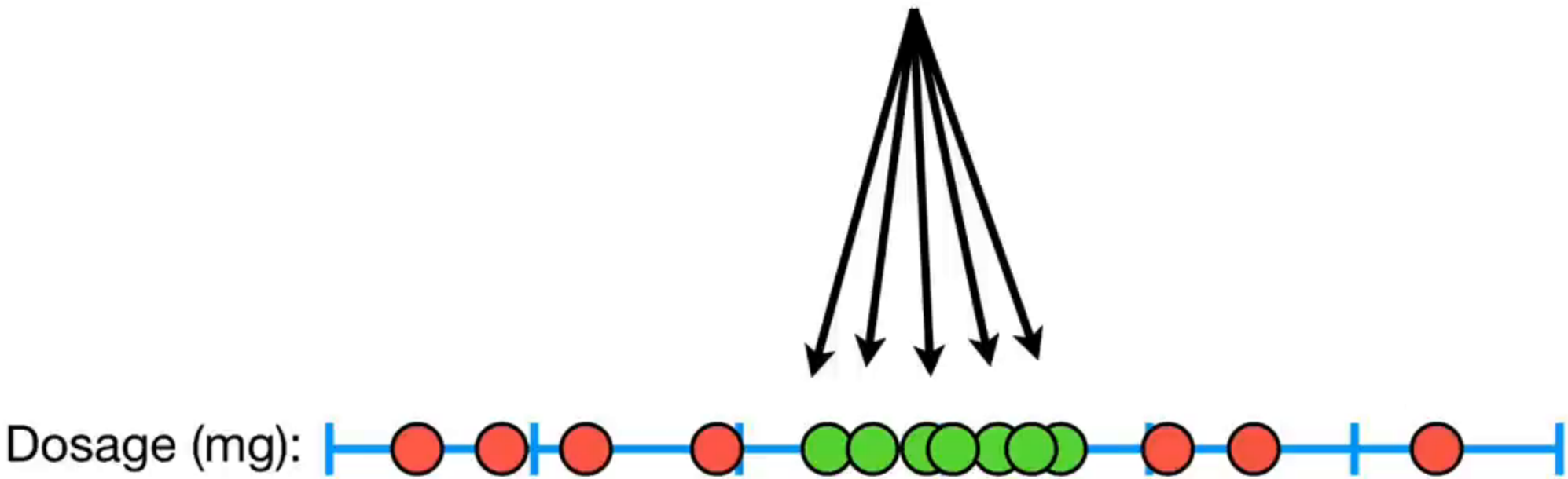


In this new example, with tons of overlap, we are now looking at **Drug Dosages**…

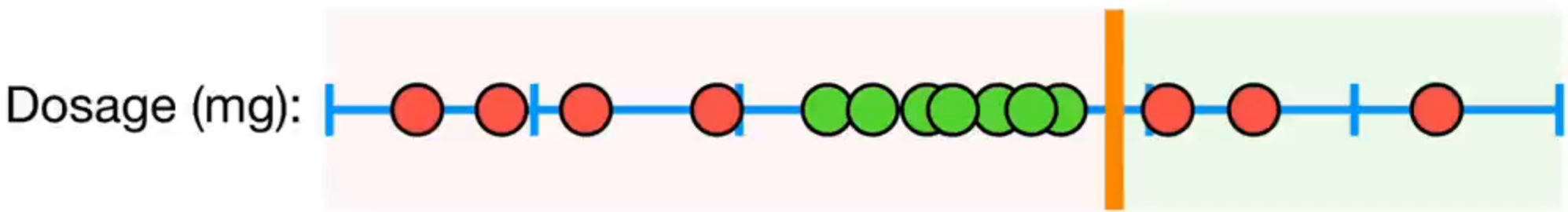
…and the **green dots** represent
patients that were ***cured***.



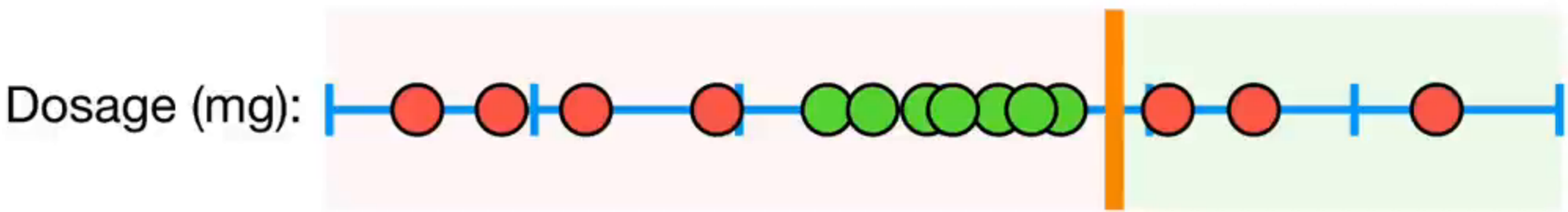
It only works when the
dosage is just right.



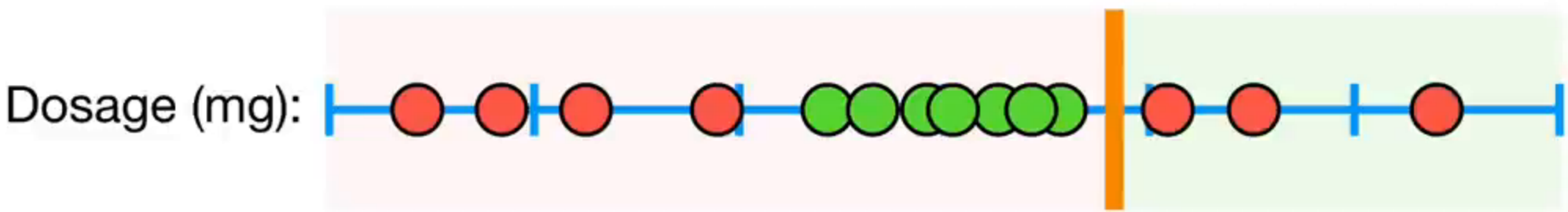
So **Support Vector Classifiers** are are only semi-cool, since they don't perform well with this type of data.



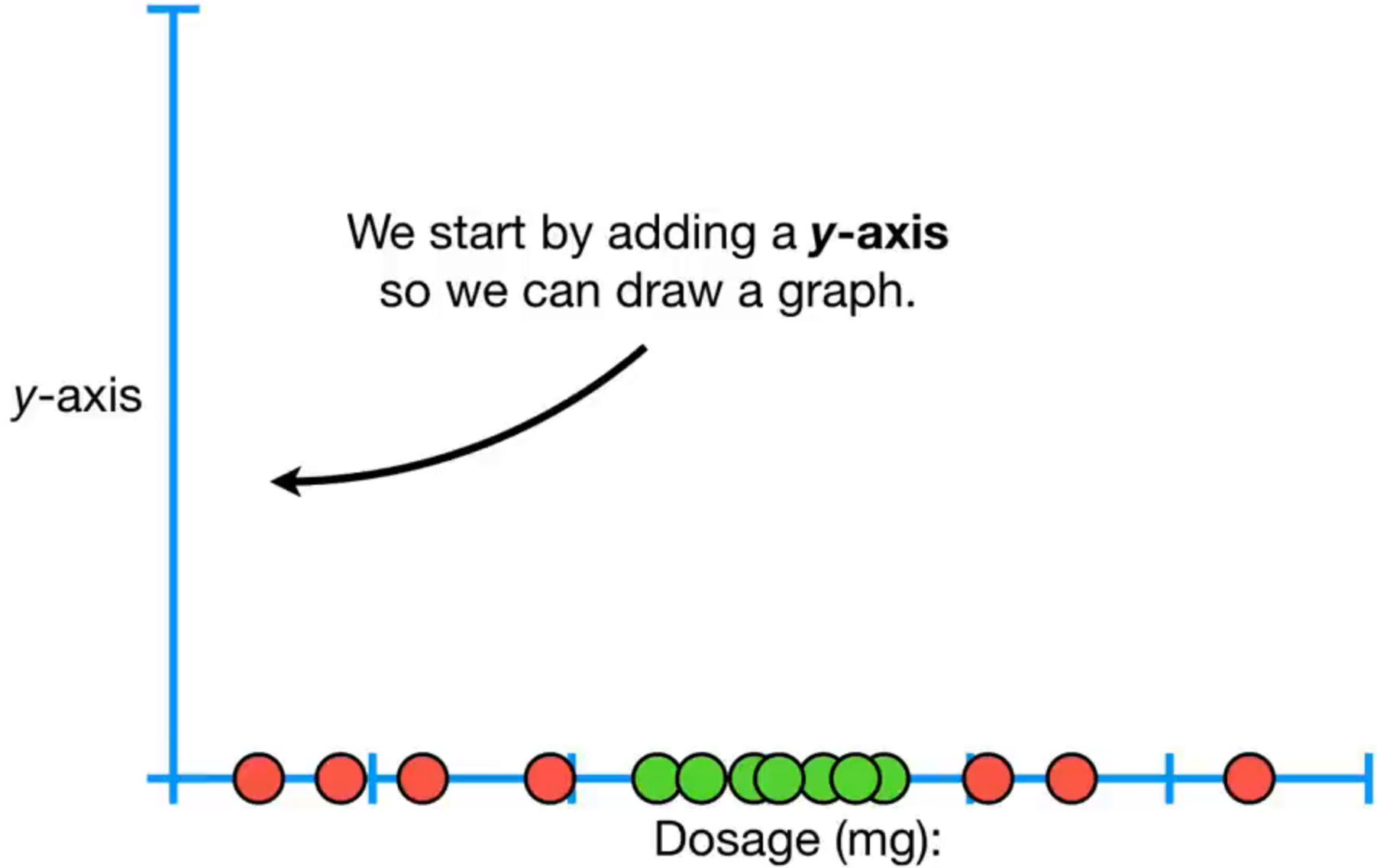
Can we do better than **Maximal Margin Classifiers** and **Support Vector Classifiers**?

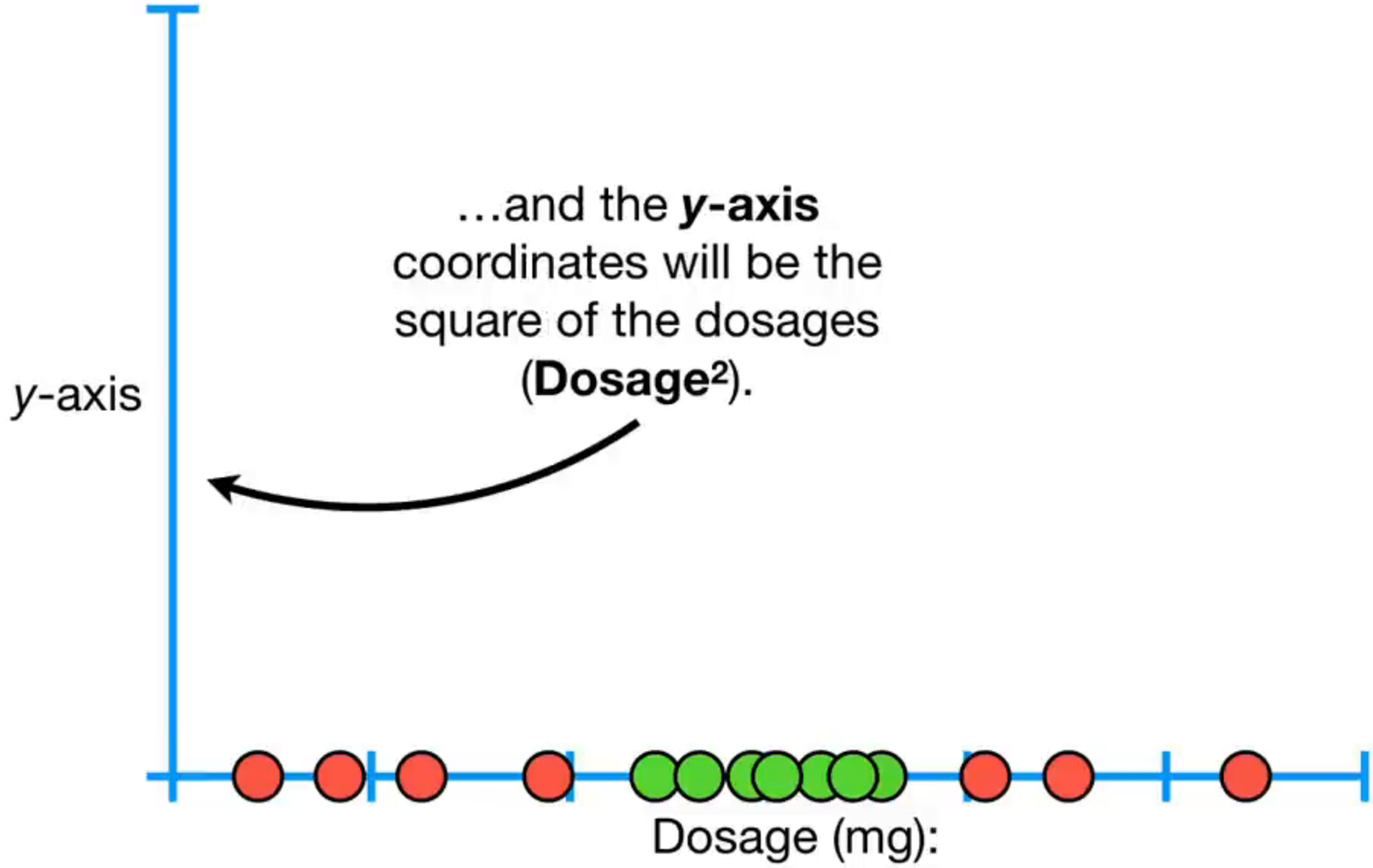


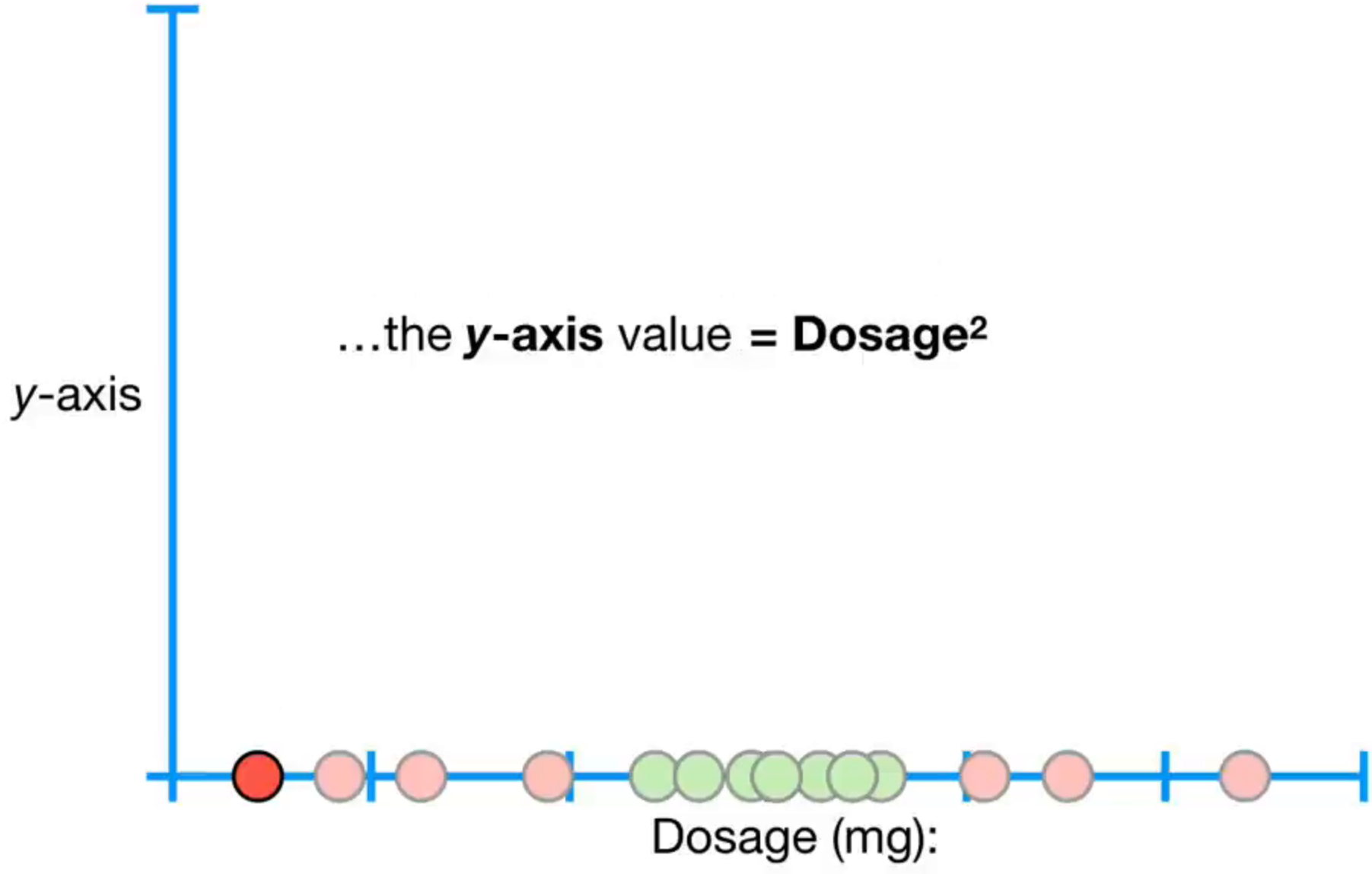
Since **Maximal Margin Classifiers** and **Support Vector Classifiers** can't handle this data, it's high time we talked about…

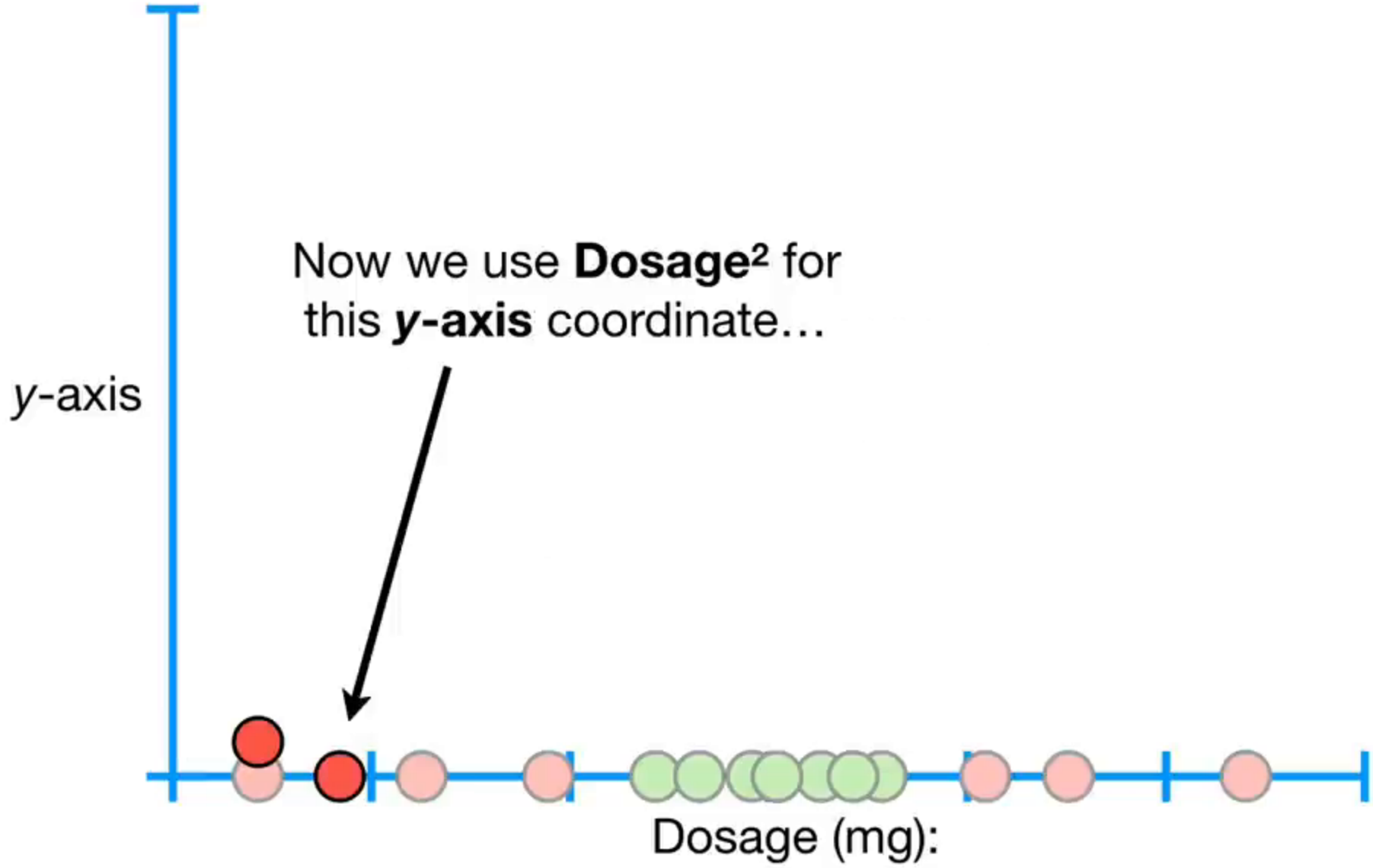


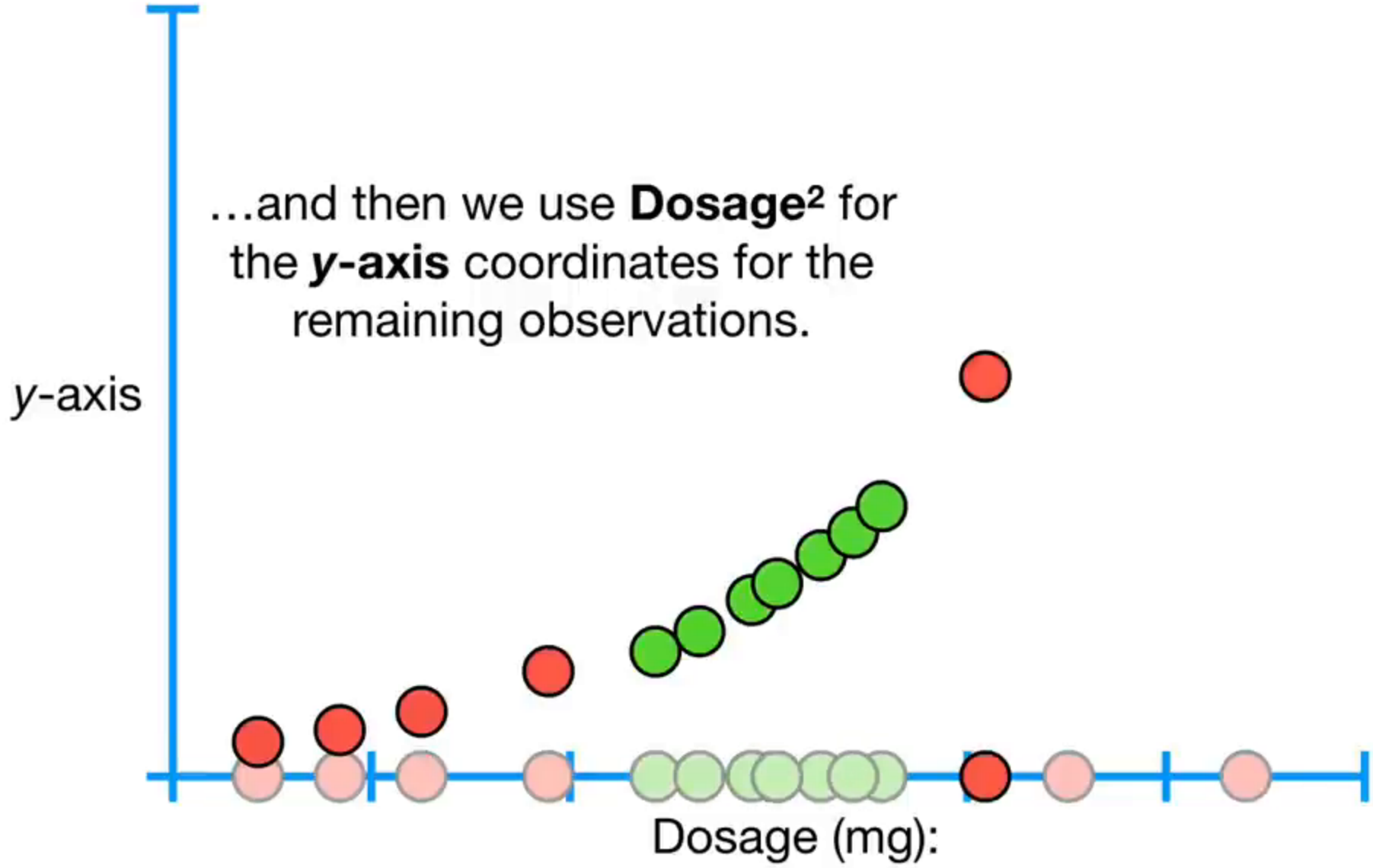
So let's start by getting get an intuitive sense of the main ideas behind **Support Vector Machines**.

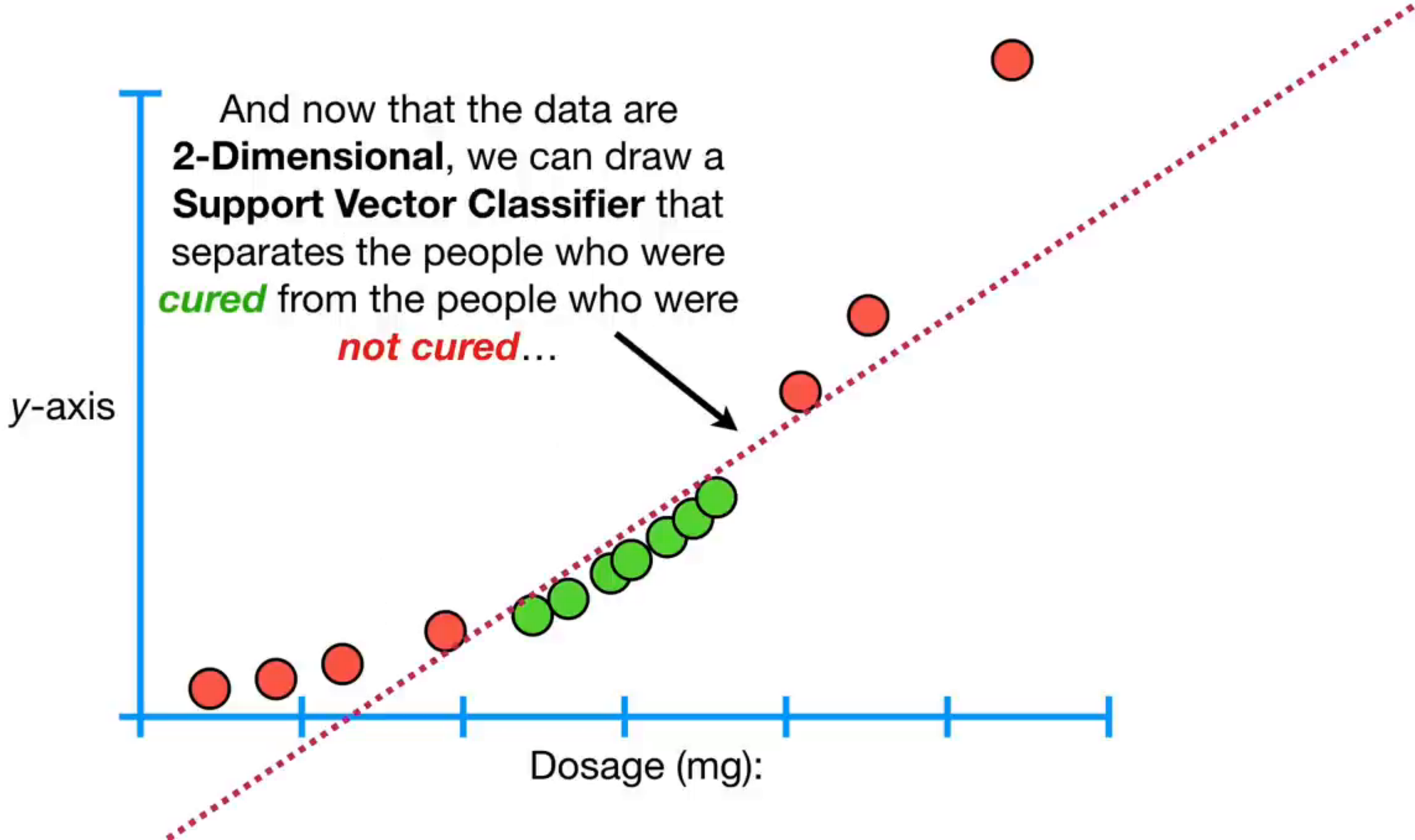

We start by adding a **y-axis**
so we can draw a graph.
y-axis
Dosage (mg):

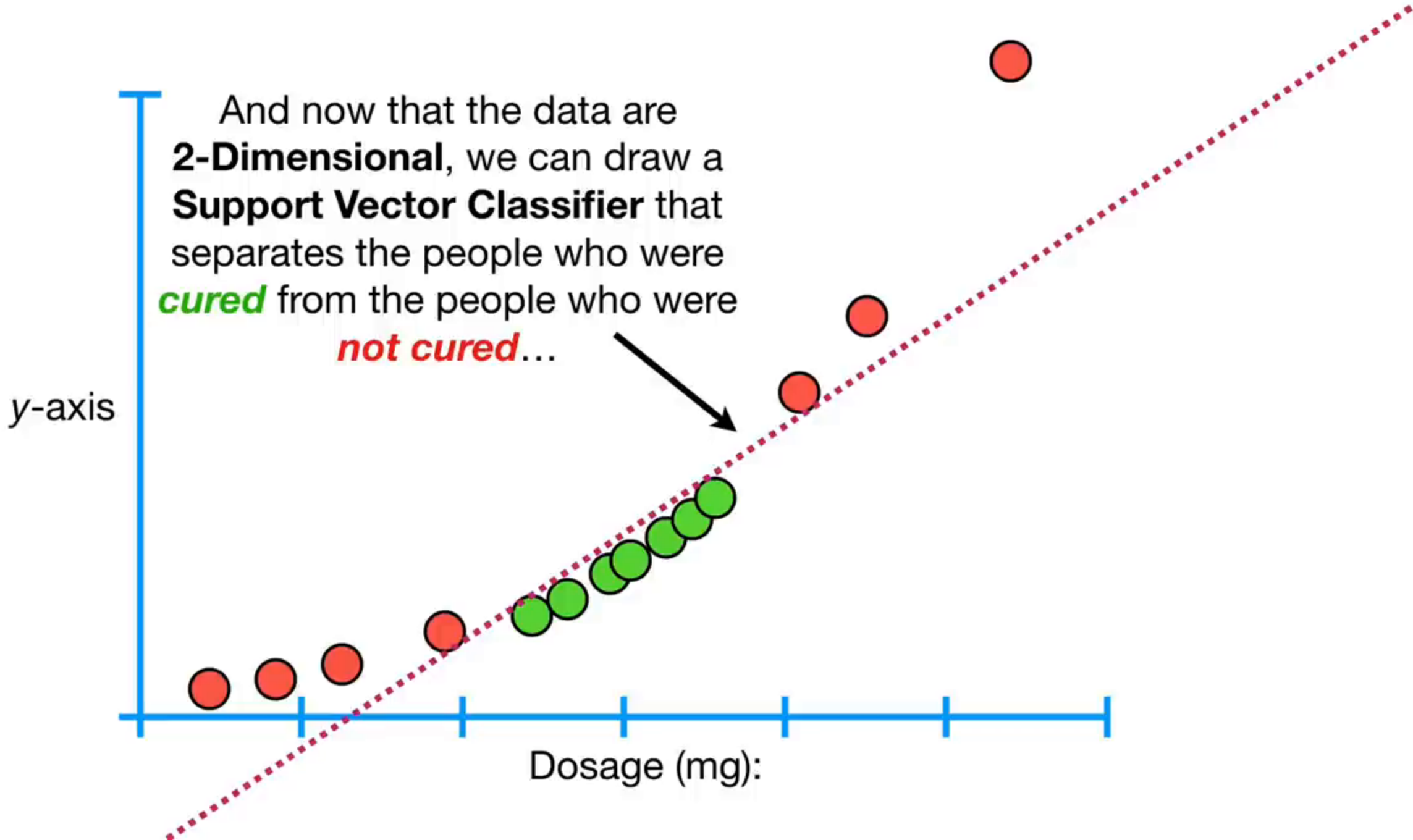

…and the **y-axis** coordinates will be the square of the dosages (**Dosage²**).
y-axis
Dosage (mg):

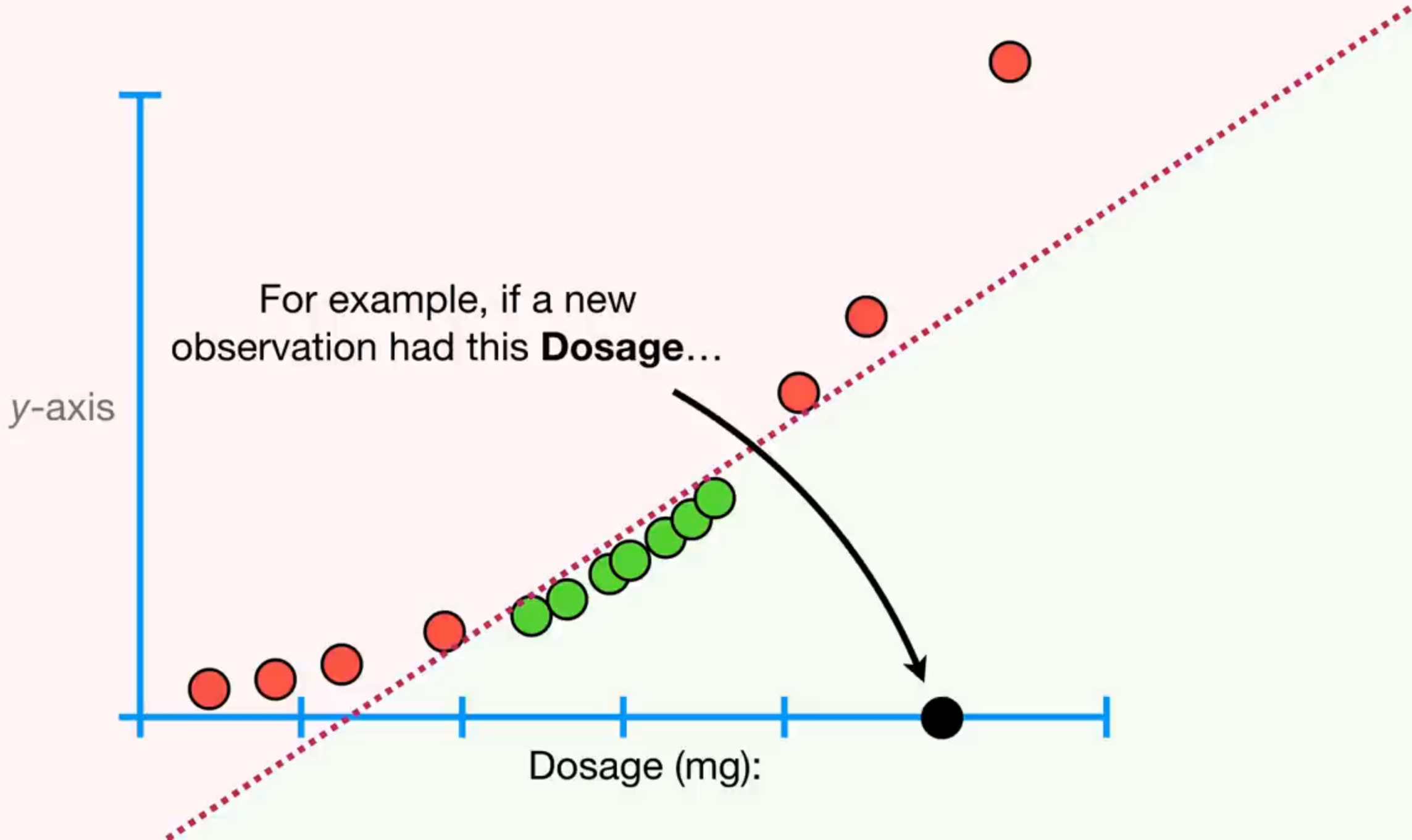

…the y-axis value = Dosage²
y-axis
Dosage (mg):

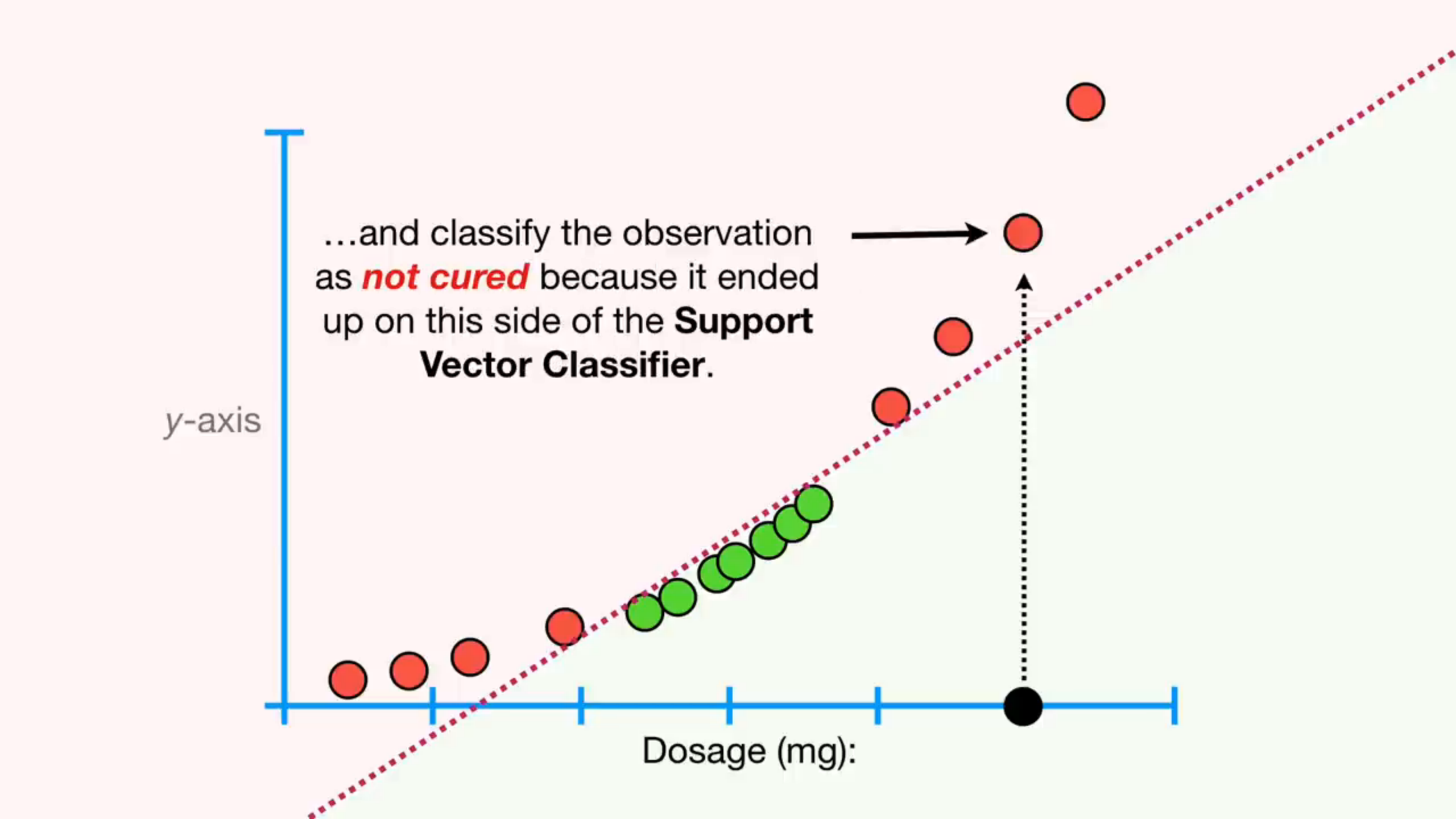

Now we use **Dosage²** for this ***y*-axis** coordinate…
y-axis
Dosage (mg):

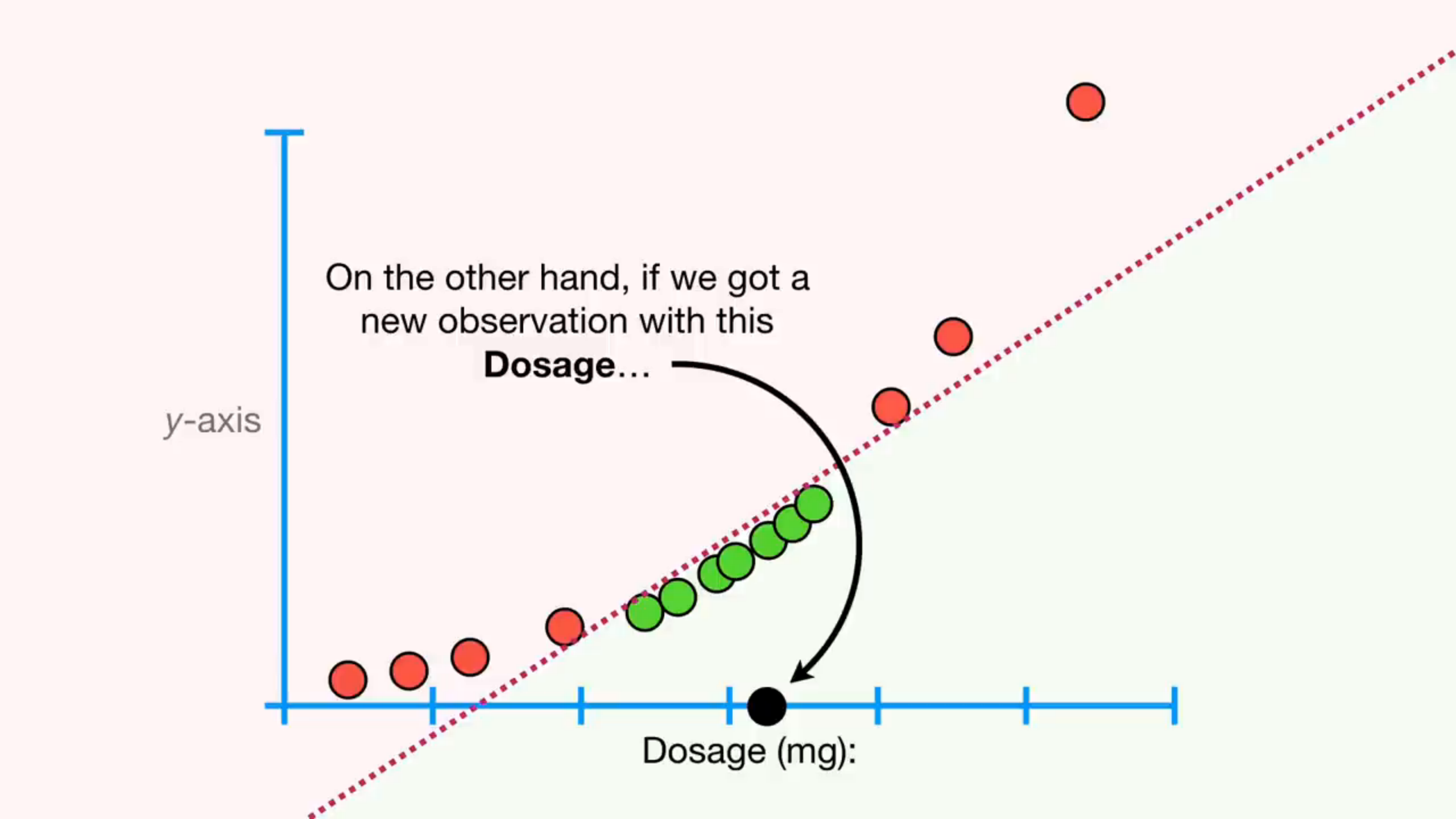

…and then we use **Dosage²** for the ***y*-axis** coordinates for the remaining observations.
y-axis
Dosage (mg):

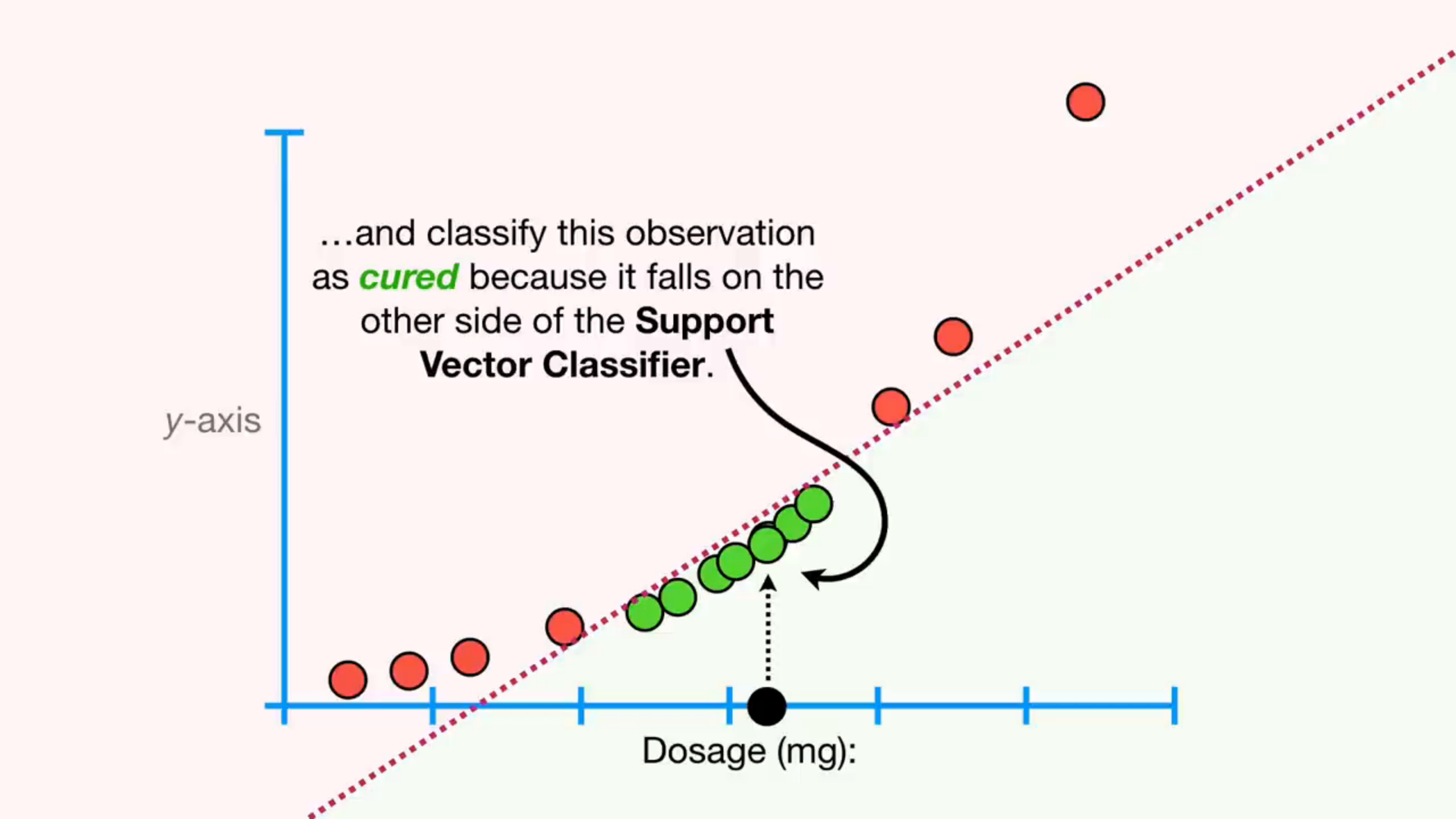

And now that the data are
2-Dimensional, we can draw a
Support Vector Classifier that
separates the people who were
cured from the people who were
not cured…
y-axis
Dosage (mg):

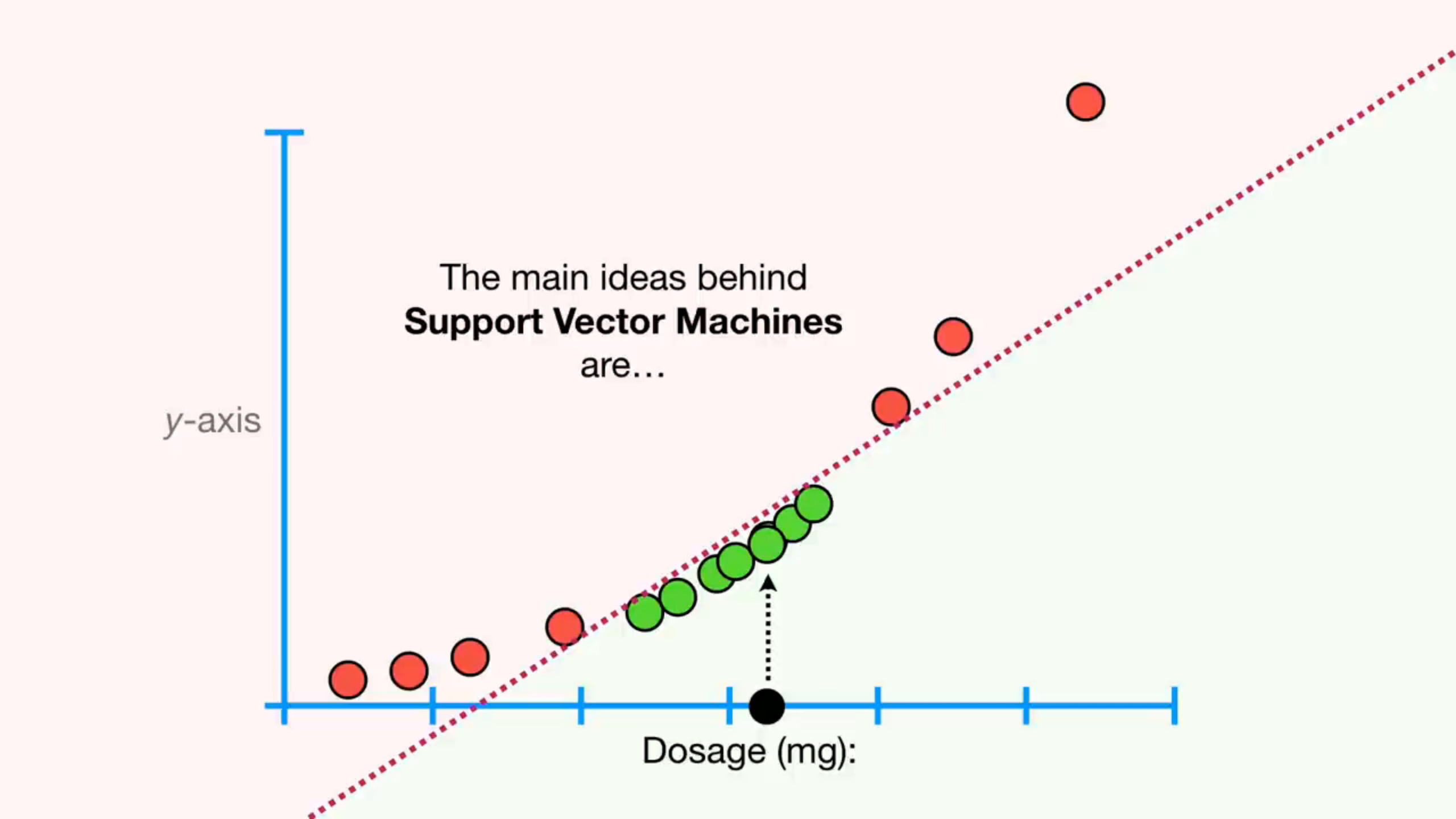

And now that the data are
2-Dimensional, we can draw a
Support Vector Classifier that
separates the people who were
cured from the people who were
not cured…
y-axis
Dosage (mg):

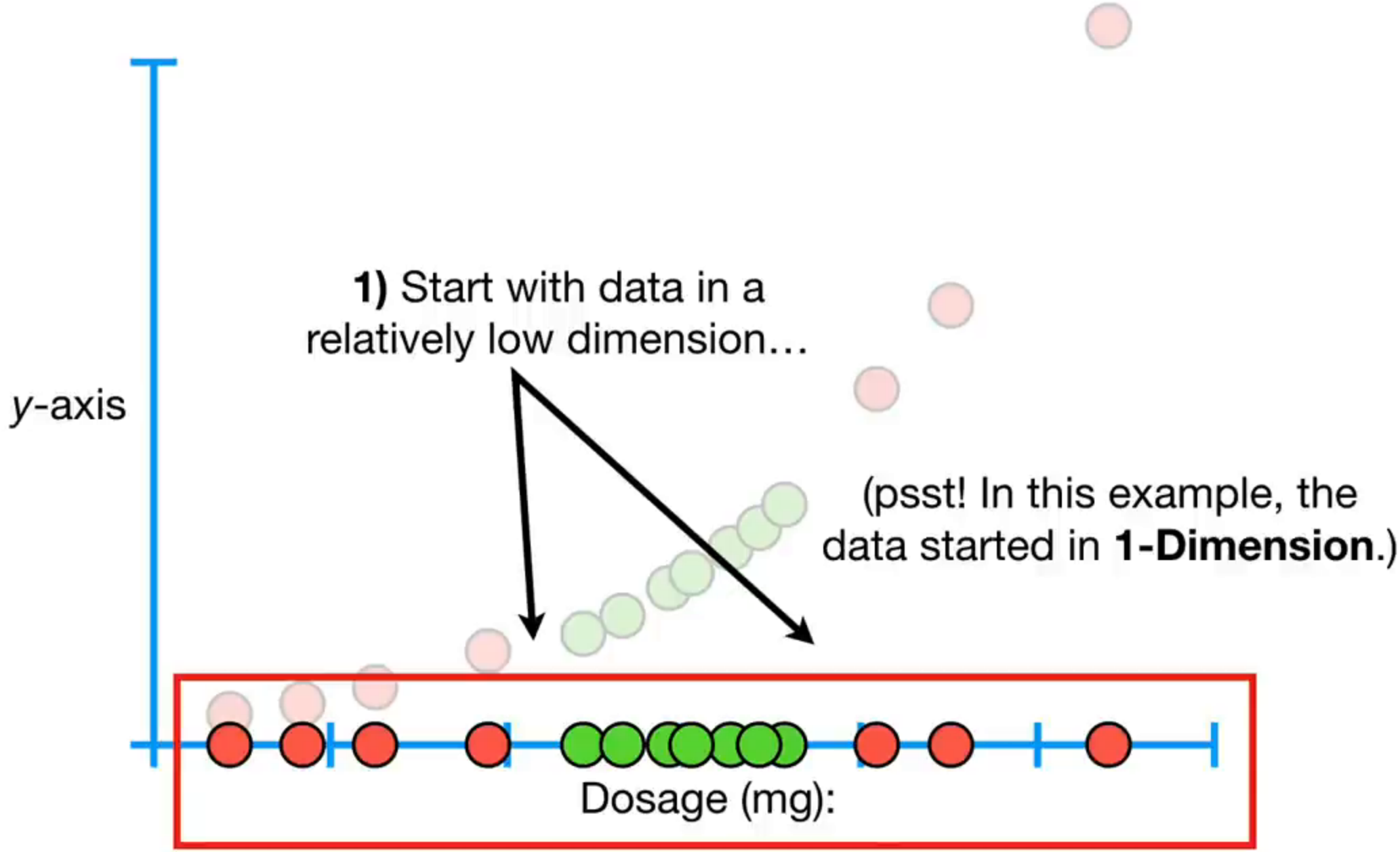

For example, if a new observation had this **Dosage**…
y-axis
Dosage (mg):

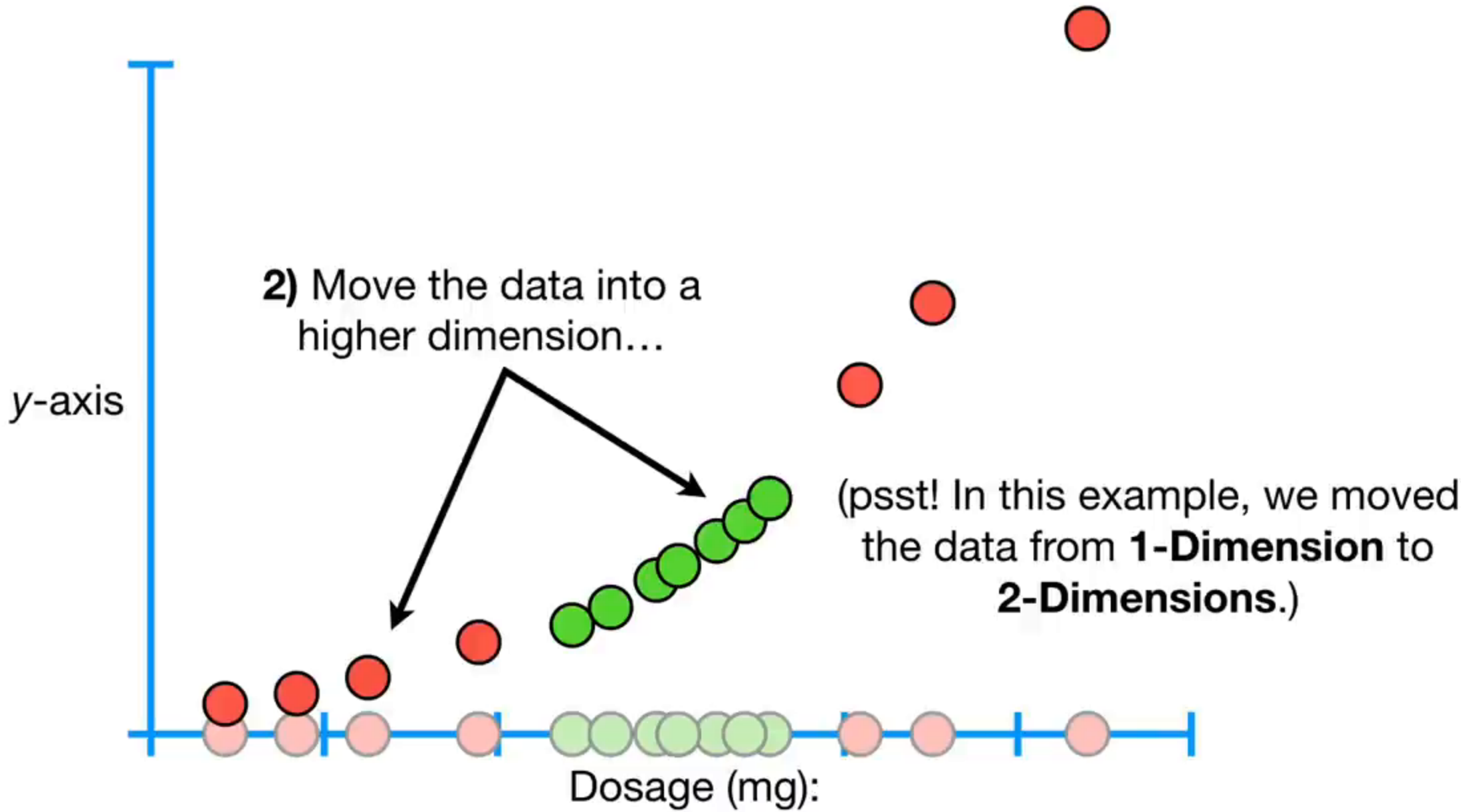

…and classify the observation as *not cured* because it ended up on this side of the **Support Vector Classifier**.
y-axis
Dosage (mg):

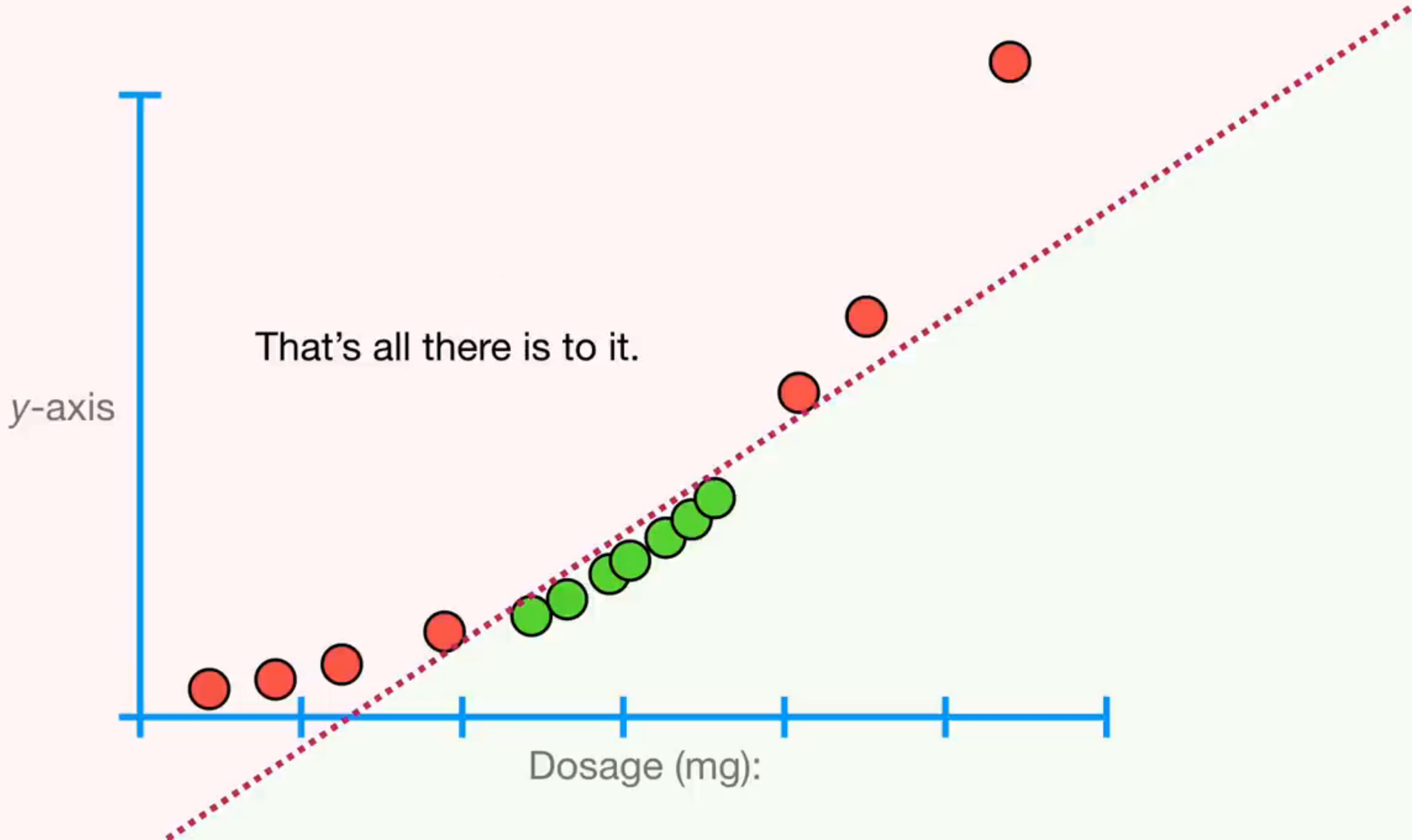

On the other hand, if we got a new observation with this **Dosage**...
y-axis
Dosage (mg):

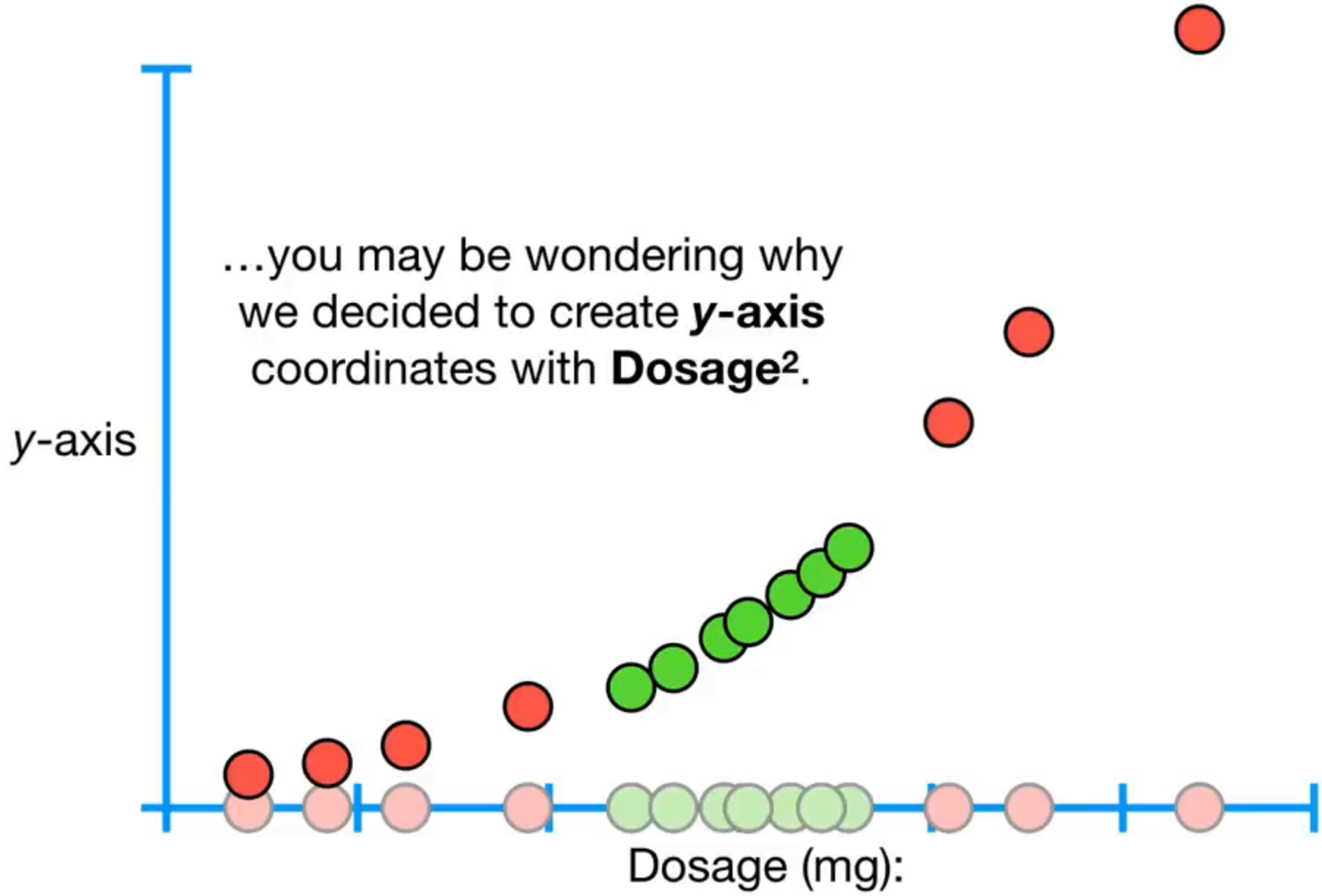

…and classify this observation as *cured* because it falls on the other side of the **Support Vector Classifier**.
y-axis
Dosage (mg):

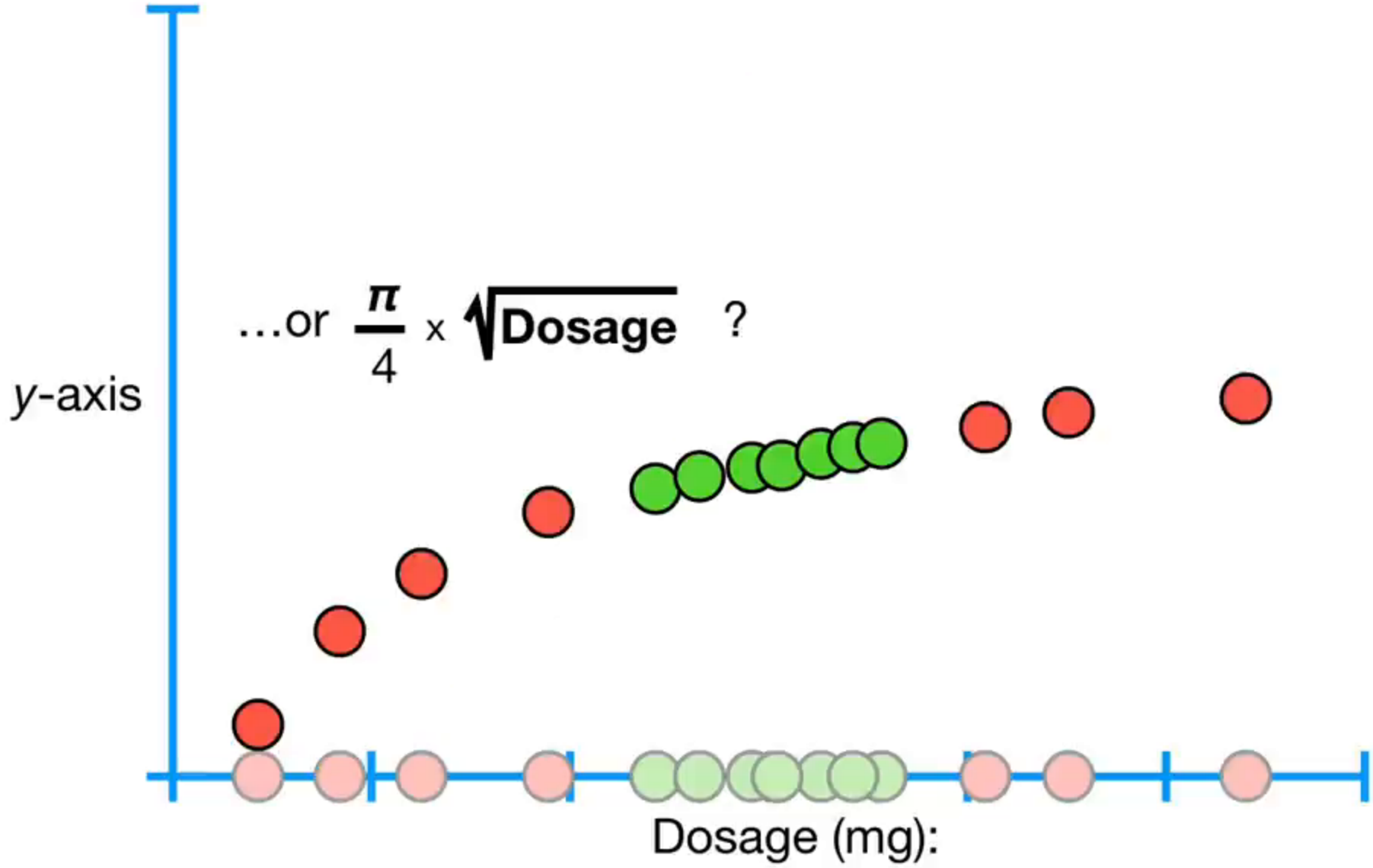

The main ideas behind
Support Vector Machines
are…
y-axis
Dosage (mg):

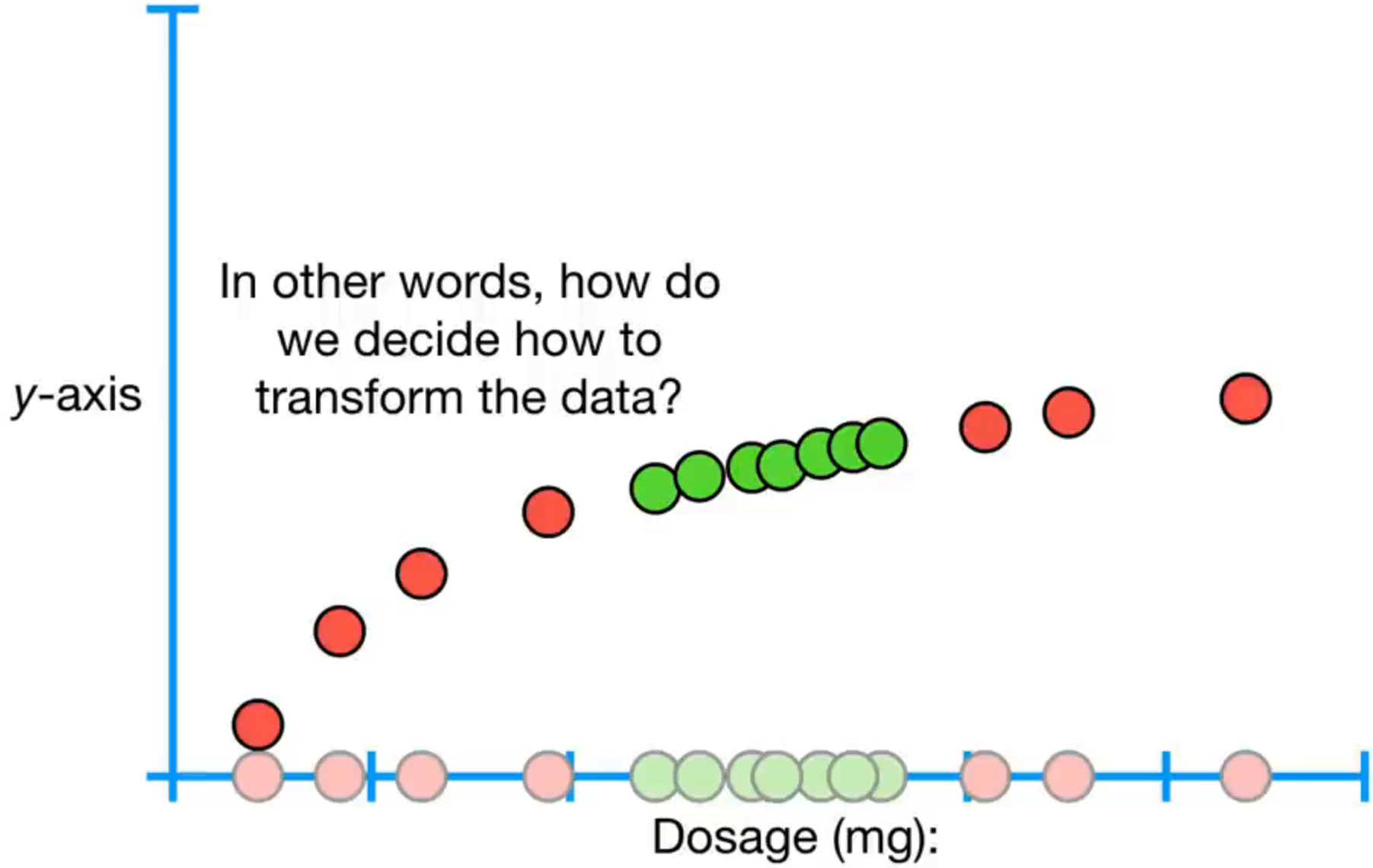

y-axis
1) Start with data in a relatively low dimension…
(psst! In this example, the data started in 1-Dimension.)
Dosage (mg):

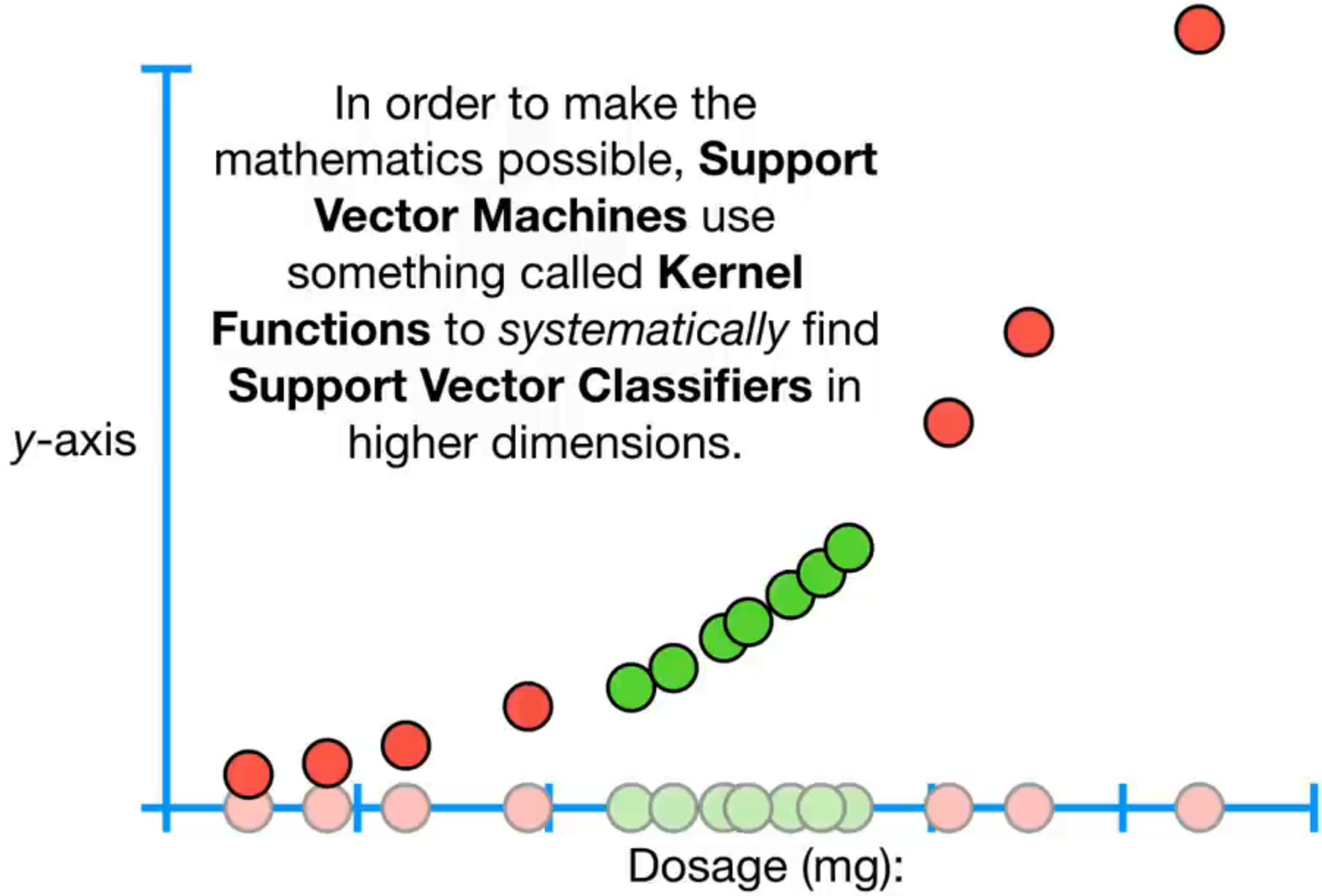

2) Move the data into a higher dimension…
y-axis
(psst! In this example, we moved the data from 1-Dimension to 2-Dimensions.)
Dosage (mg):

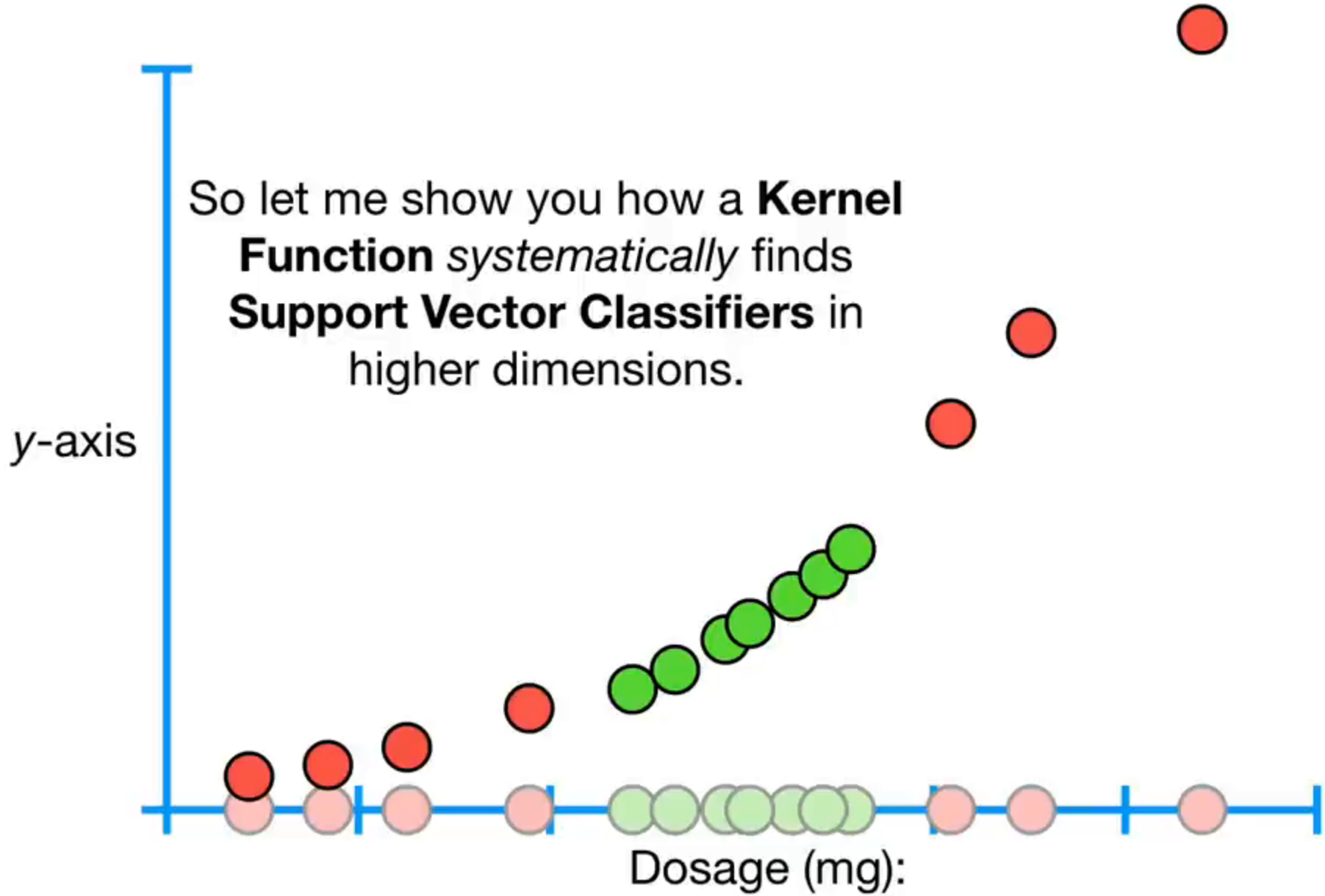

That’s all there is to it.
y-axis
Dosage (mg):

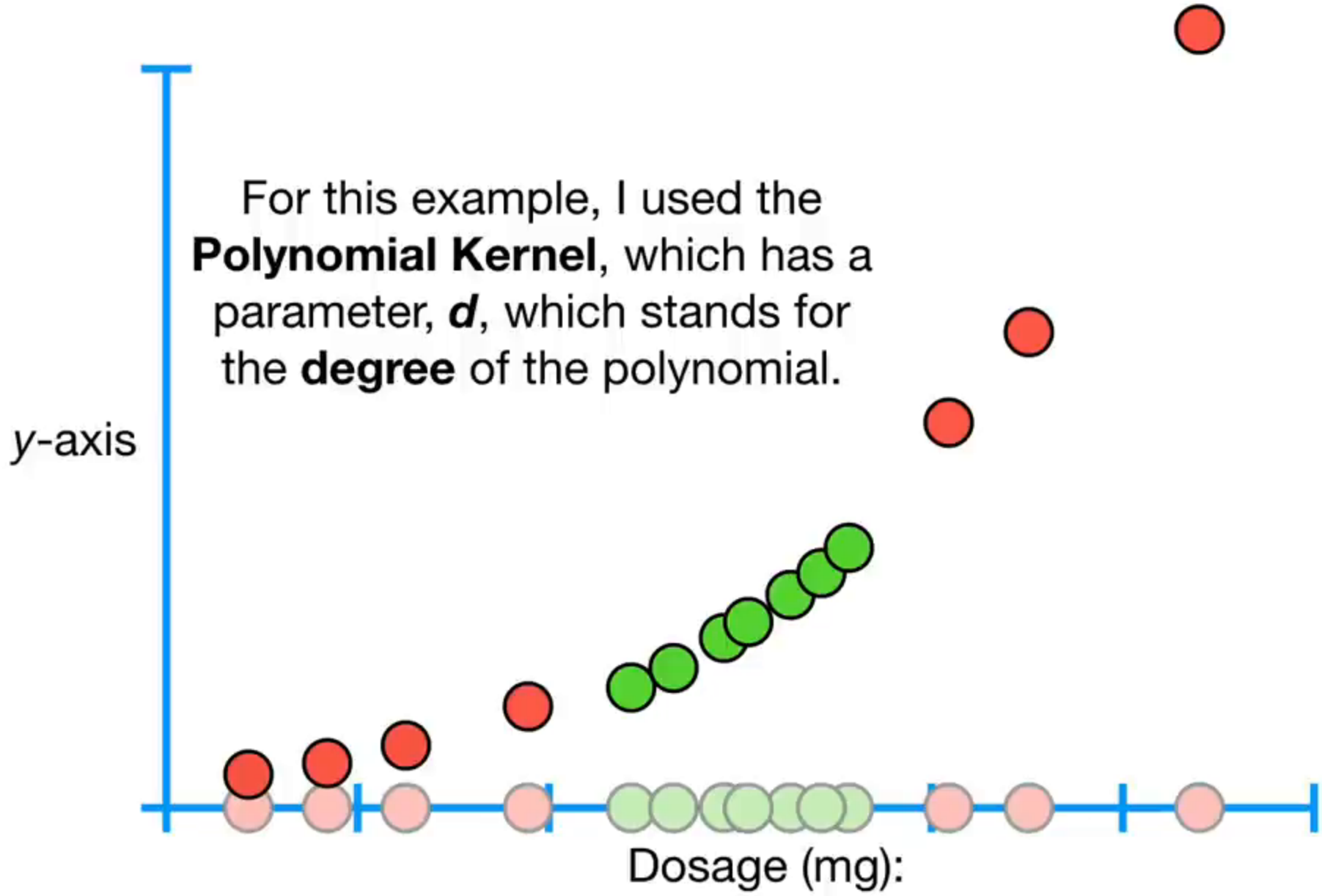

…you may be wondering why we decided to create **y-axis** coordinates with **Dosage²**.
y-axis
Dosage (mg):


…or $\frac{\pi}{4} \times \sqrt{\text{Dosage}}$?
y-axis
Dosage (mg):

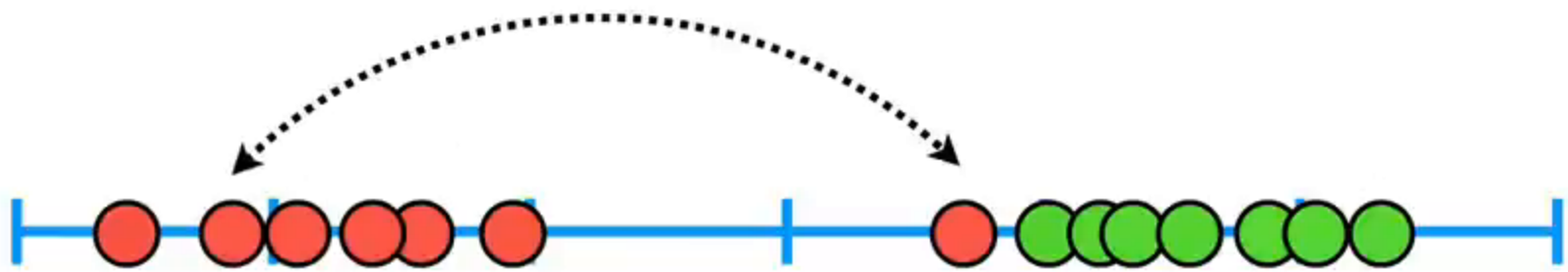

In other words, how do we decide how to transform the data?
y-axis
Dosage (mg):

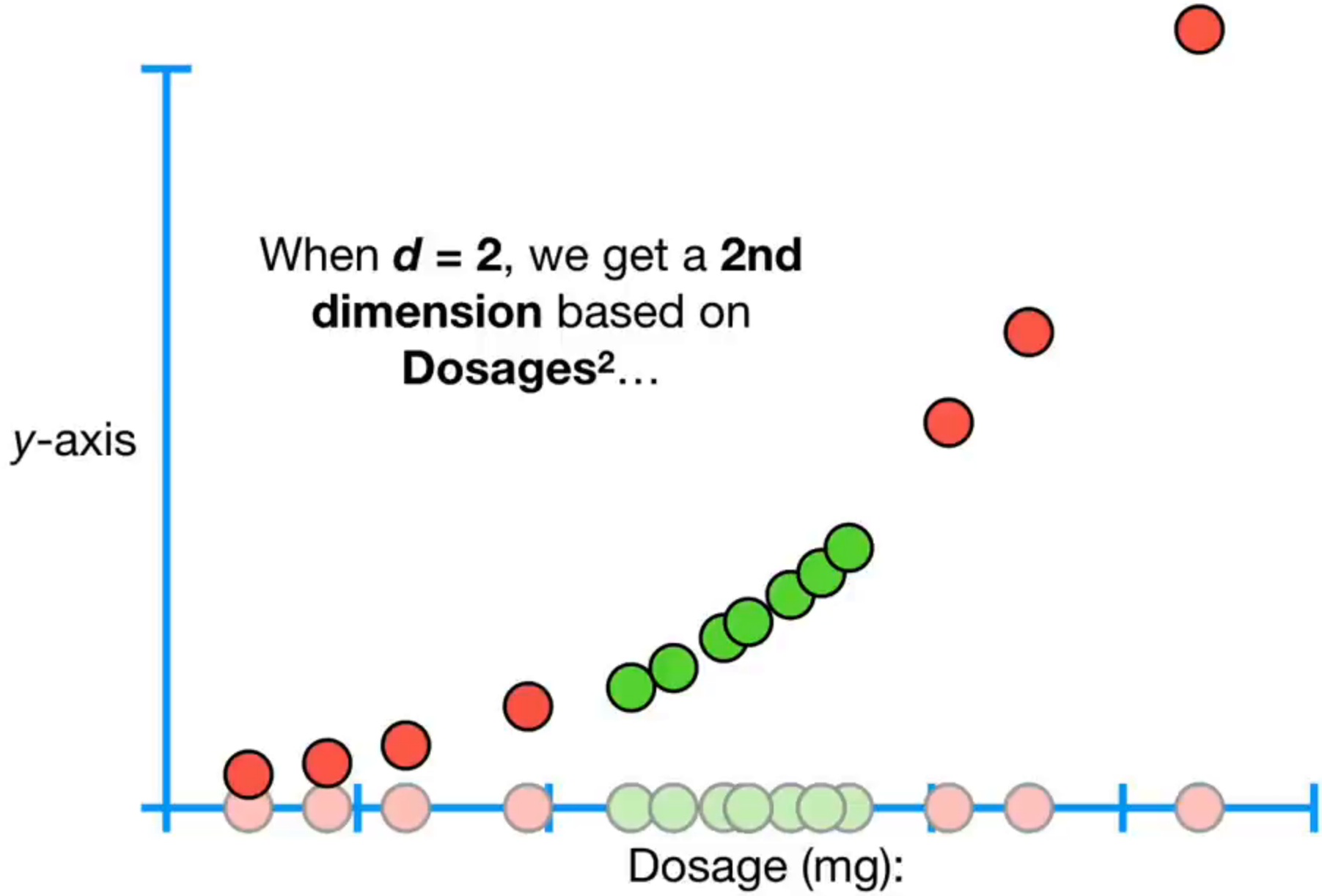

In order to make the mathematics possible, **Support Vector Machines** use something called **Kernel Functions** to *systematically* find **Support Vector Classifiers** in higher dimensions.
y-axis
Dosage (mg):

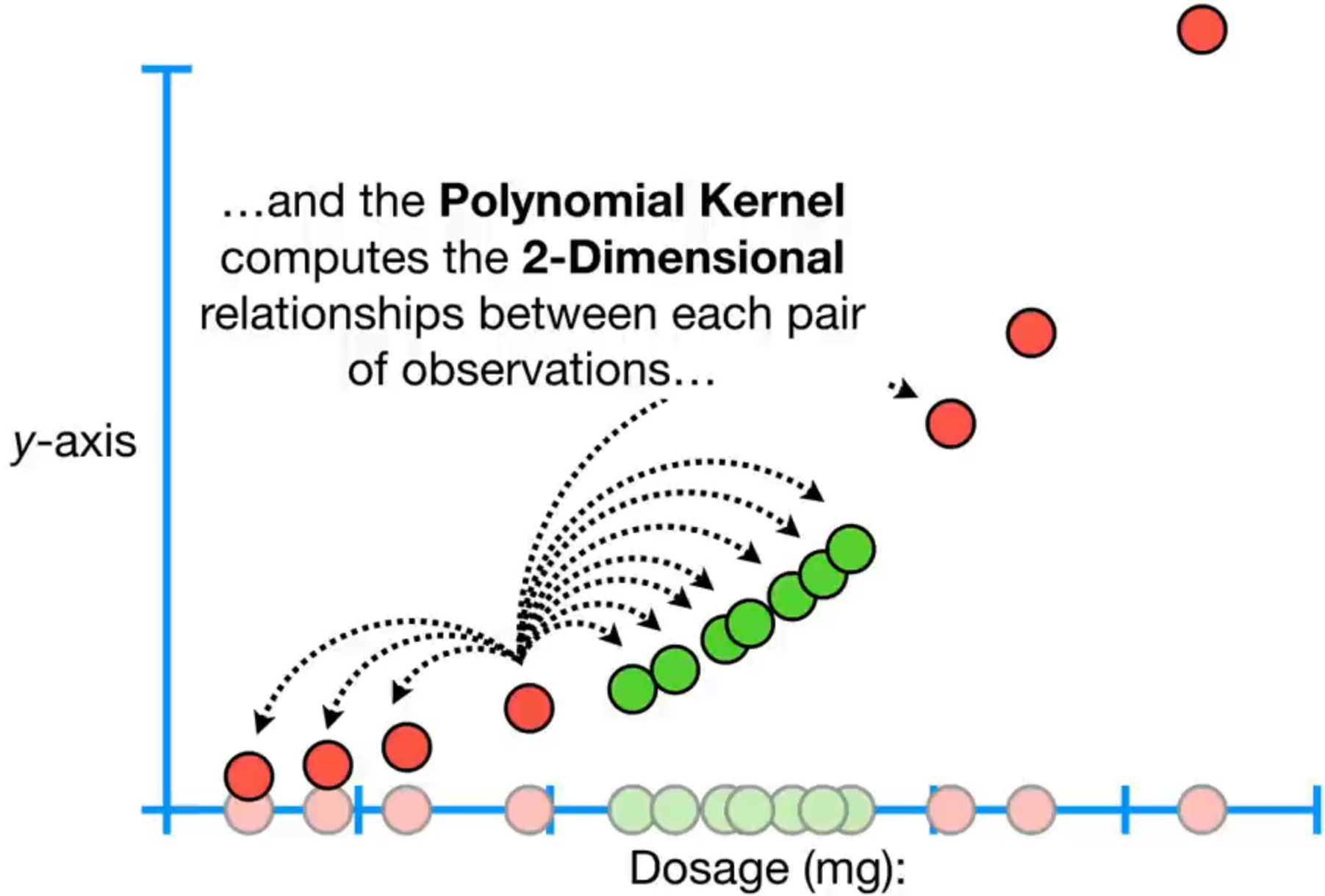

So let me show you how a **Kernel Function** *systematically* finds **Support Vector Classifiers** in higher dimensions.
y-axis
Dosage (mg):

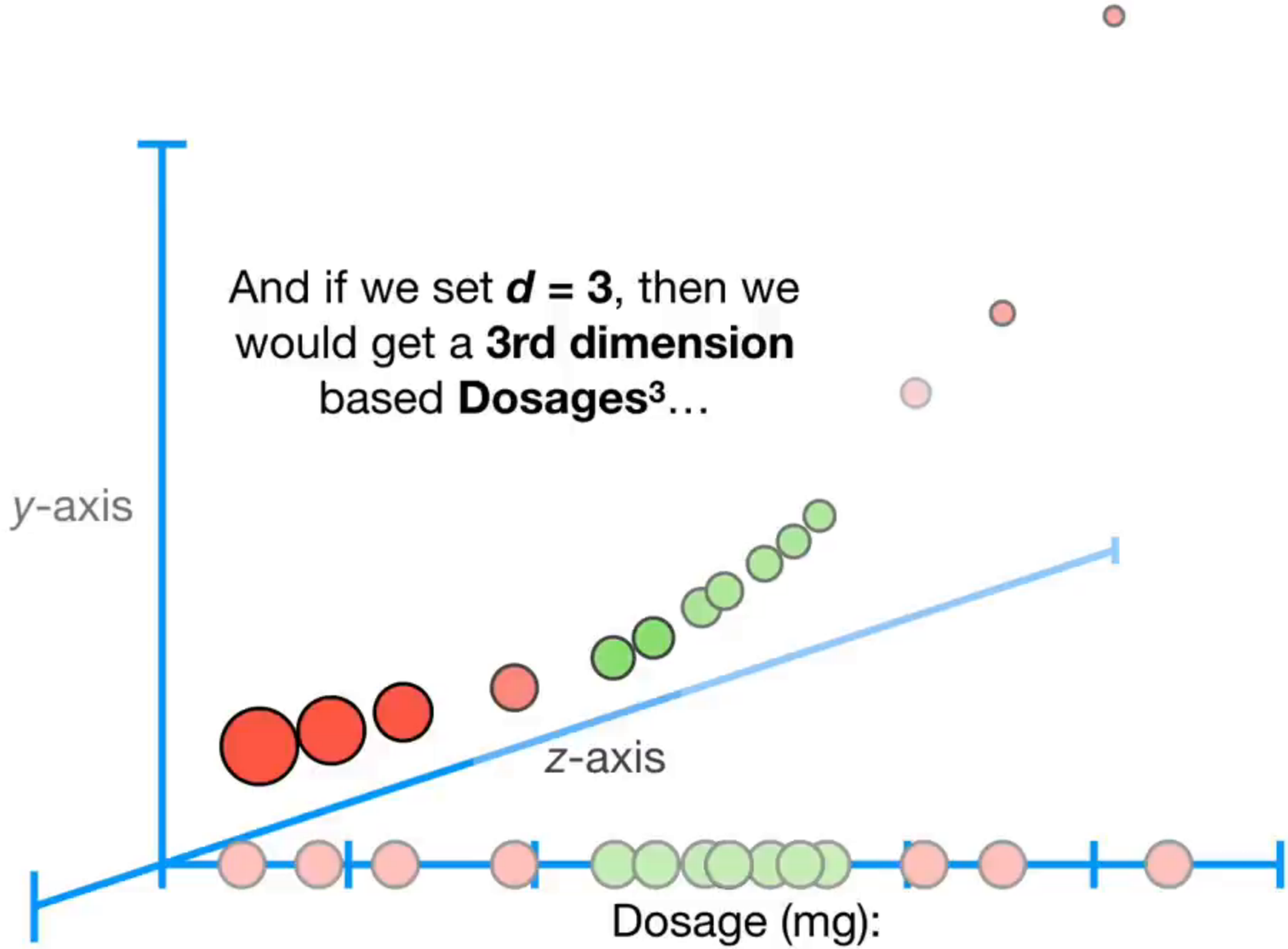

For this example, I used the
Polynomial Kernel, which has a
parameter, ***d***, which stands for
the **degree** of the polynomial.
y-axis
Dosage (mg):

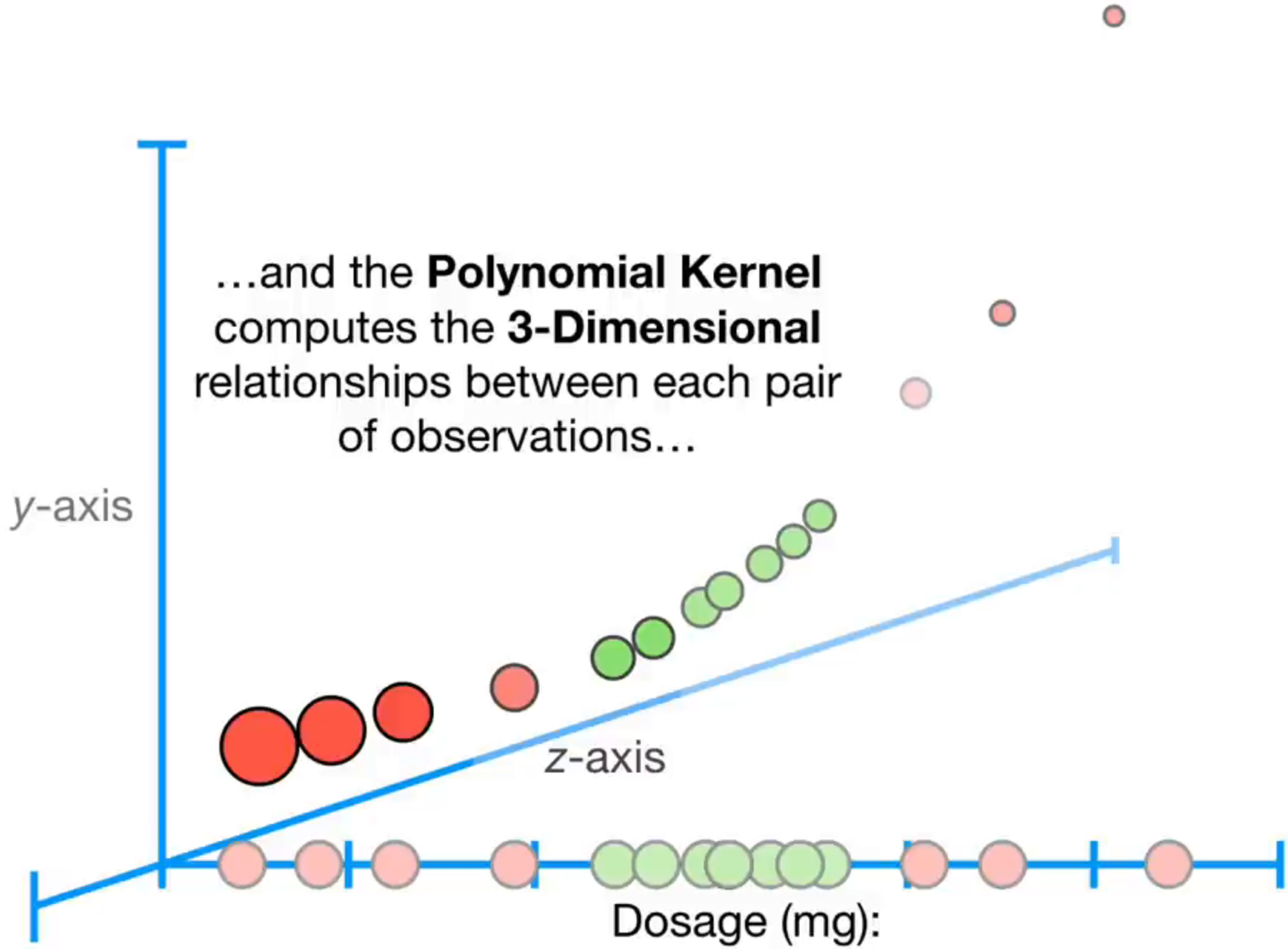
When ***d* = 1**, the **Polynomial Kernel** computes the relationships between each pair of observations in **1-Dimension**…




When **d = 2**, we get a **2nd dimension** based on **Dosages²**…
y-axis
Dosage (mg):


…and the **Polynomial Kernel** computes the **2-Dimensional** relationships between each pair of observations…
y-axis
Dosage (mg):

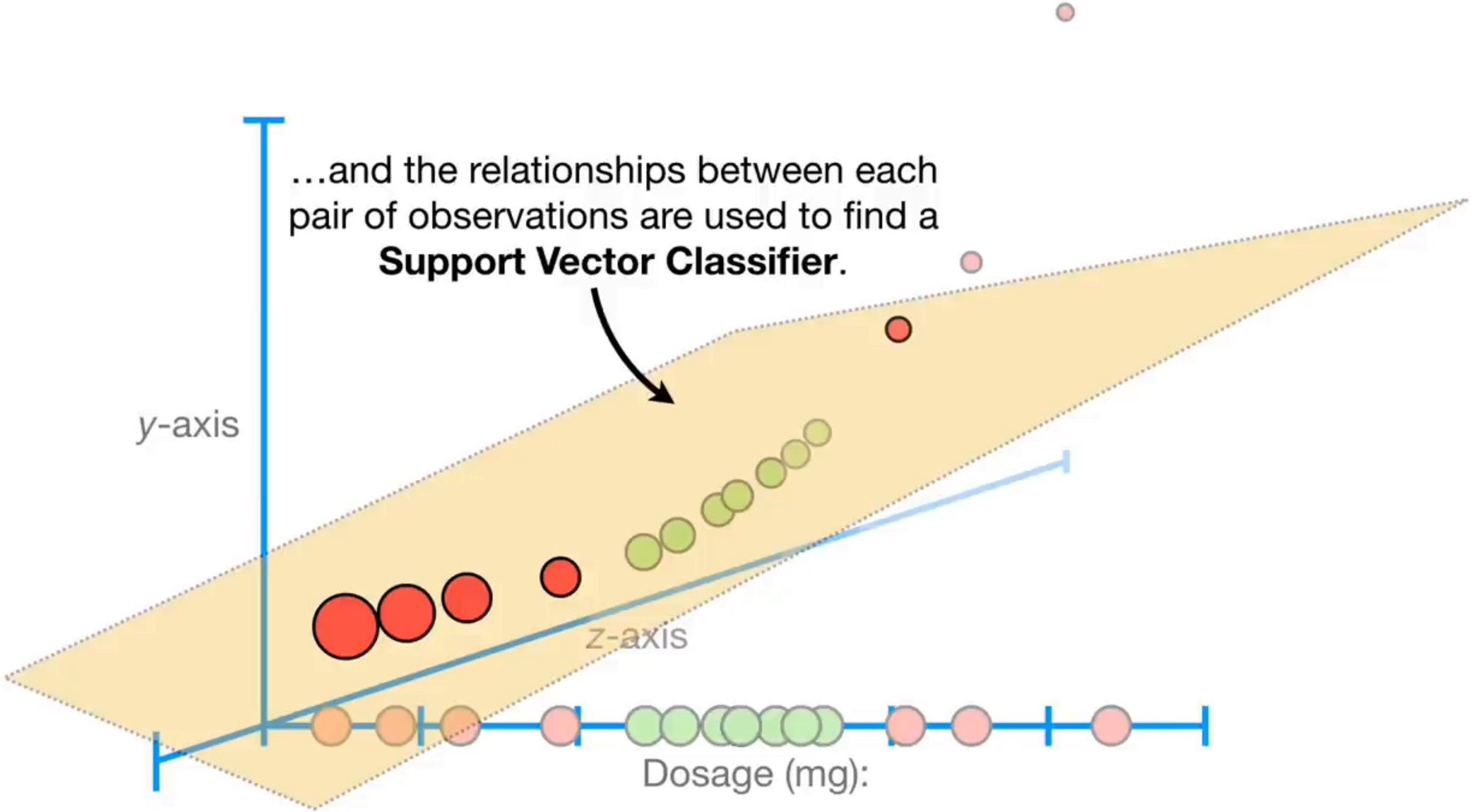

And if we set **d = 3**, then we would get a **3rd dimension** based **Dosages³**…
y-axis
z-axis
Dosage (mg):

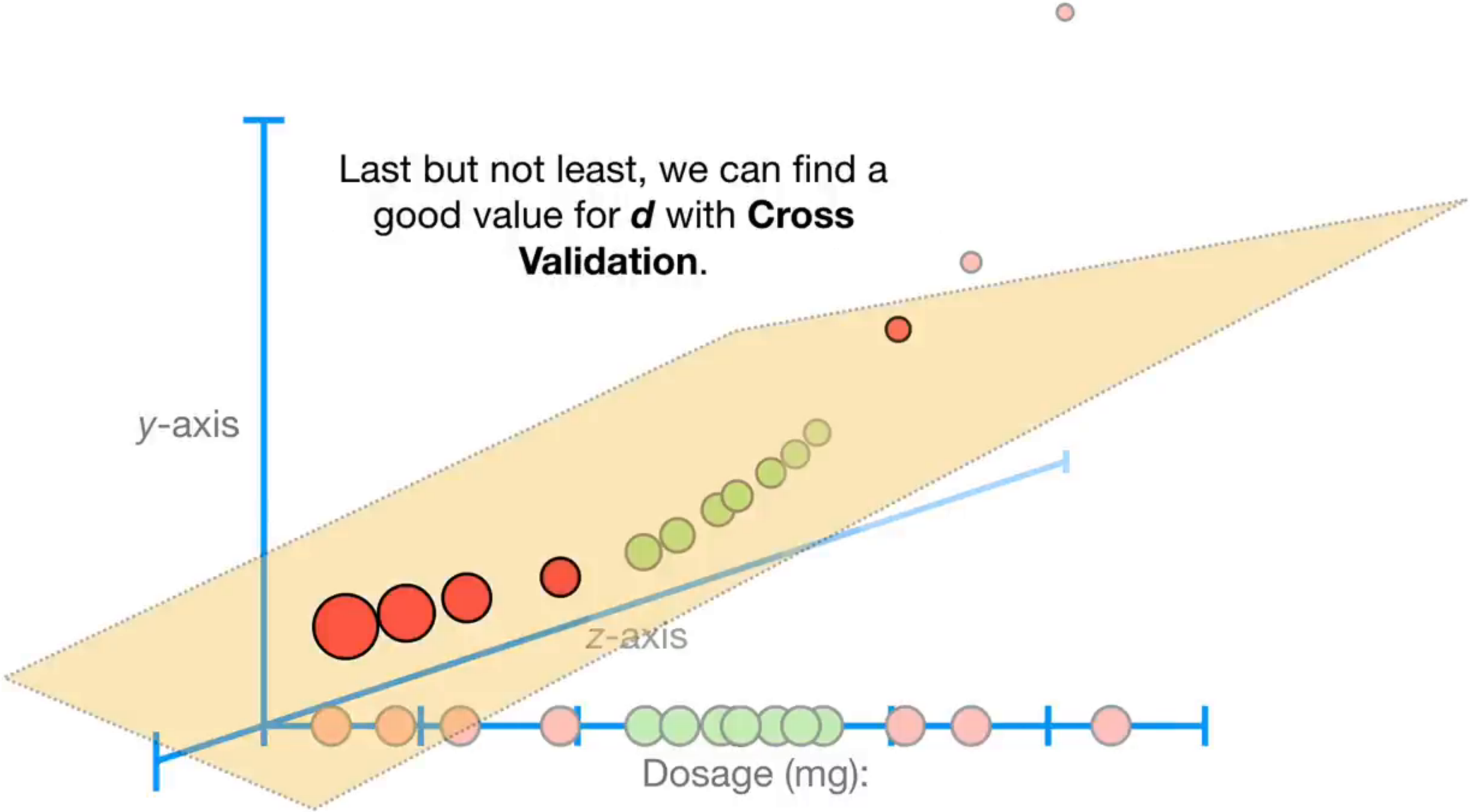

…and the **Polynomial Kernel** computes the **3-Dimensional** relationships between each pair of observations…
y-axis
z-axis
Dosage (mg):

And when ***d* = 4 or more**, then we get even more dimensions to find a **Support Vector Classifier**.

In summary, the **Polynomial Kernel** systematically increases dimensions by setting ***d***, the degree of the polynomial…

***d* = 1**


…and the relationships between each pair of observations are used to find a **Support Vector Classifier**.
y-axis
z-axis
Dosage (mg):


Last but not least, we can find a good value for *d* with **Cross Validation**.
y-axis
z-axis
Dosage (mg):

Another very commonly used **Kernel** is the **Radial Kernel**, also known as the **Radial Basis Function (RBF) Kernel**.

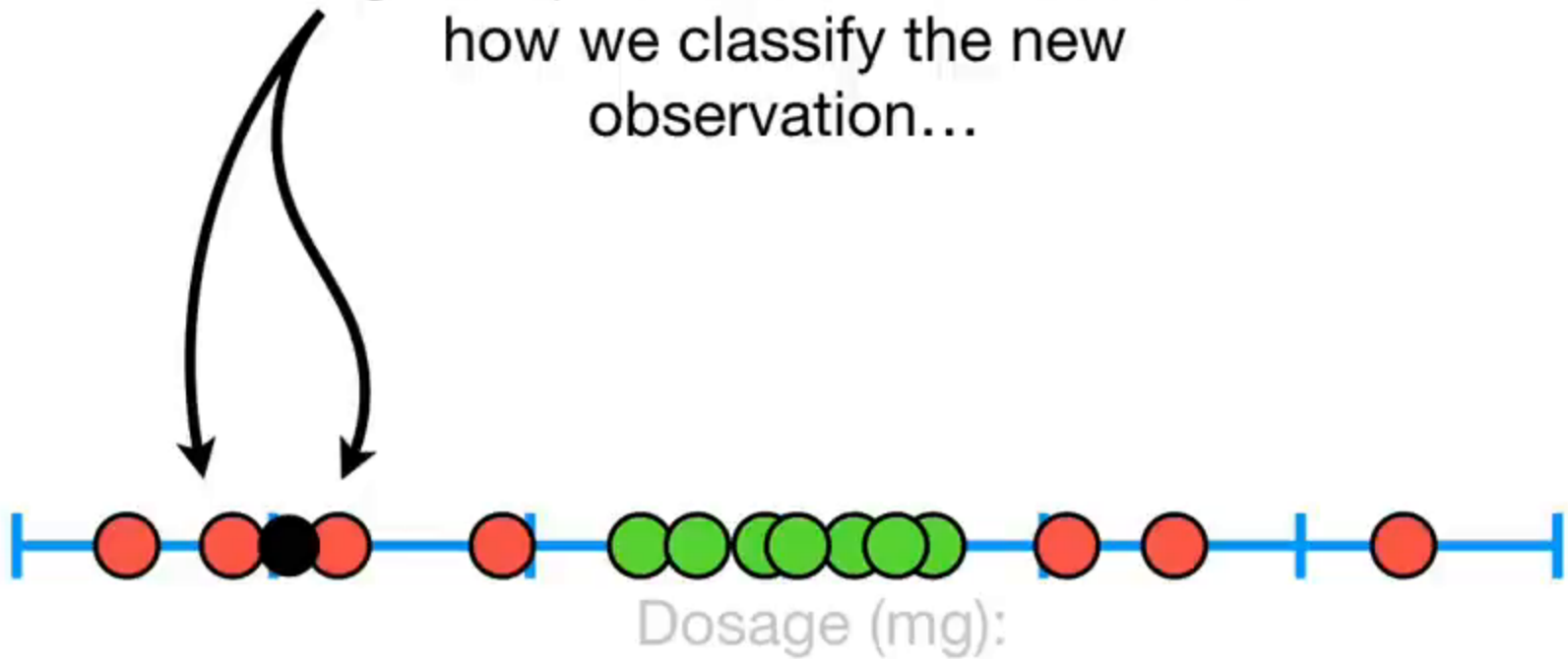
Unfortunately, the **Radial Kernel** finds **Support Vector Classifiers** in *infinite dimensions*, so I can't give you an example of what it does exactly.

Dosage (mg):

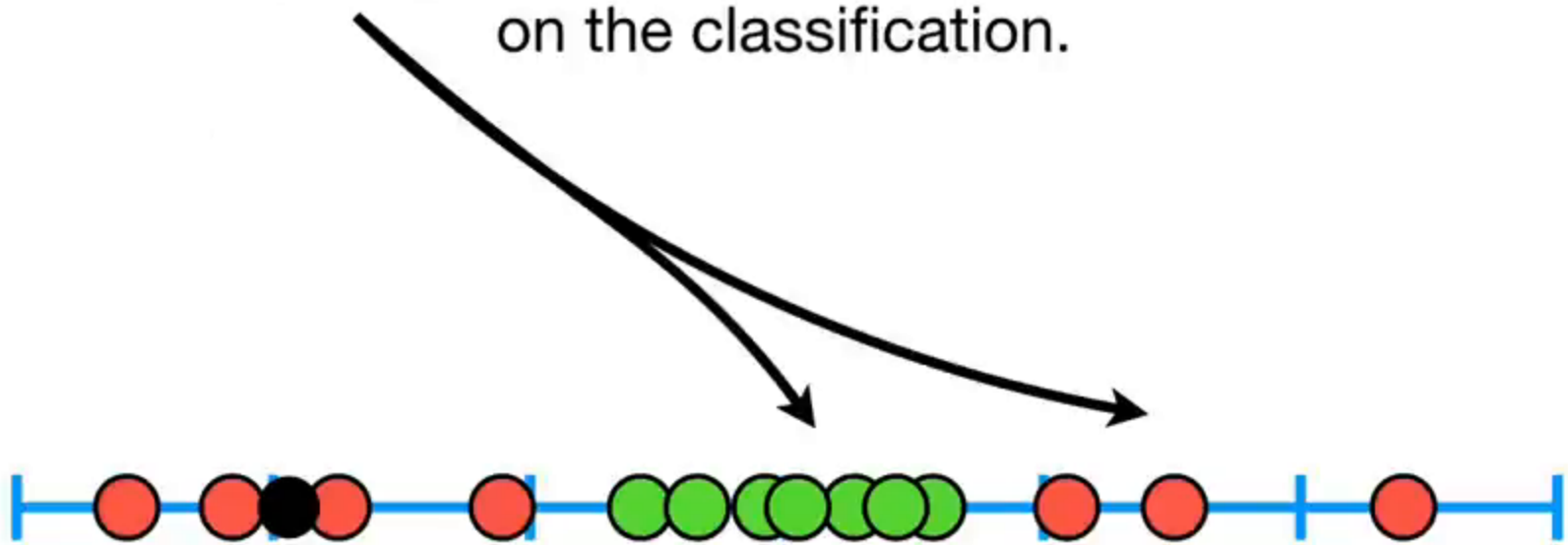
…the **Radial Kernel** behaves like a
Weighted Nearest Neighbor model.

Dosage (mg):

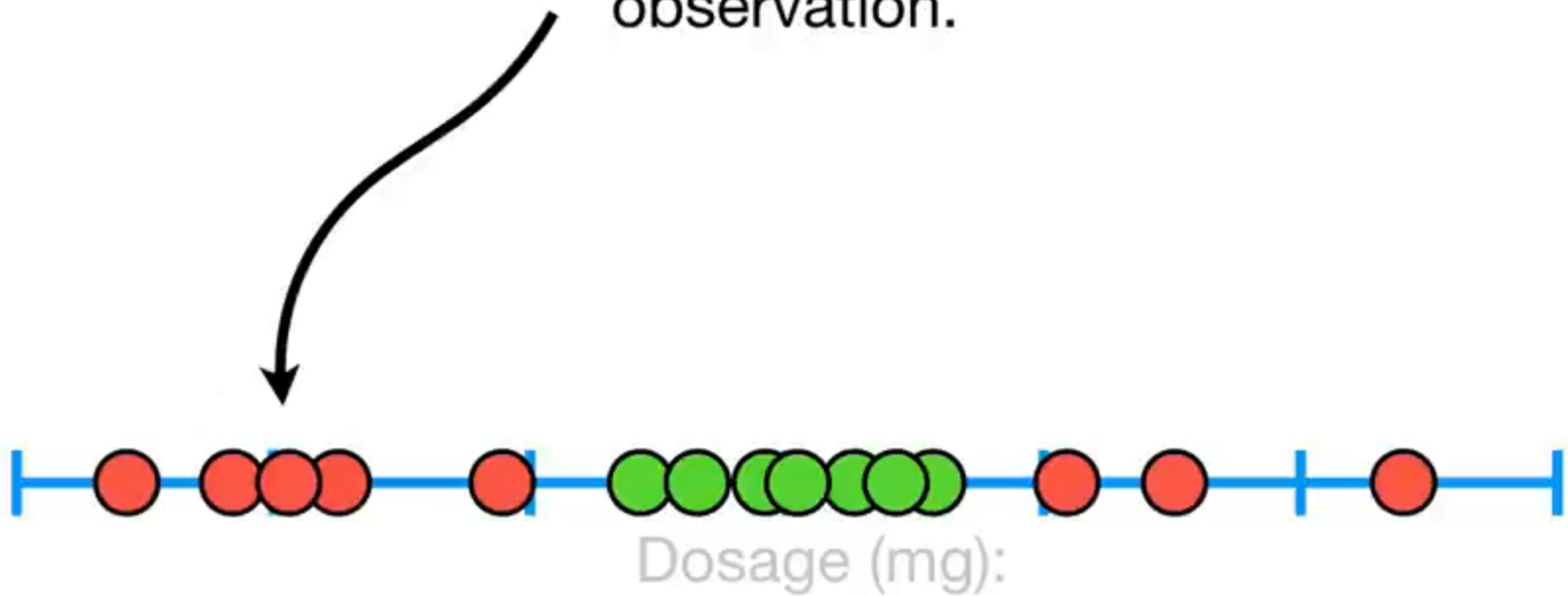
In other words, the closest observations (aka the nearest neighbors) have a lot of influence on how we classify the new observation…



…and observations that are further away have relatively little influence on the classification.

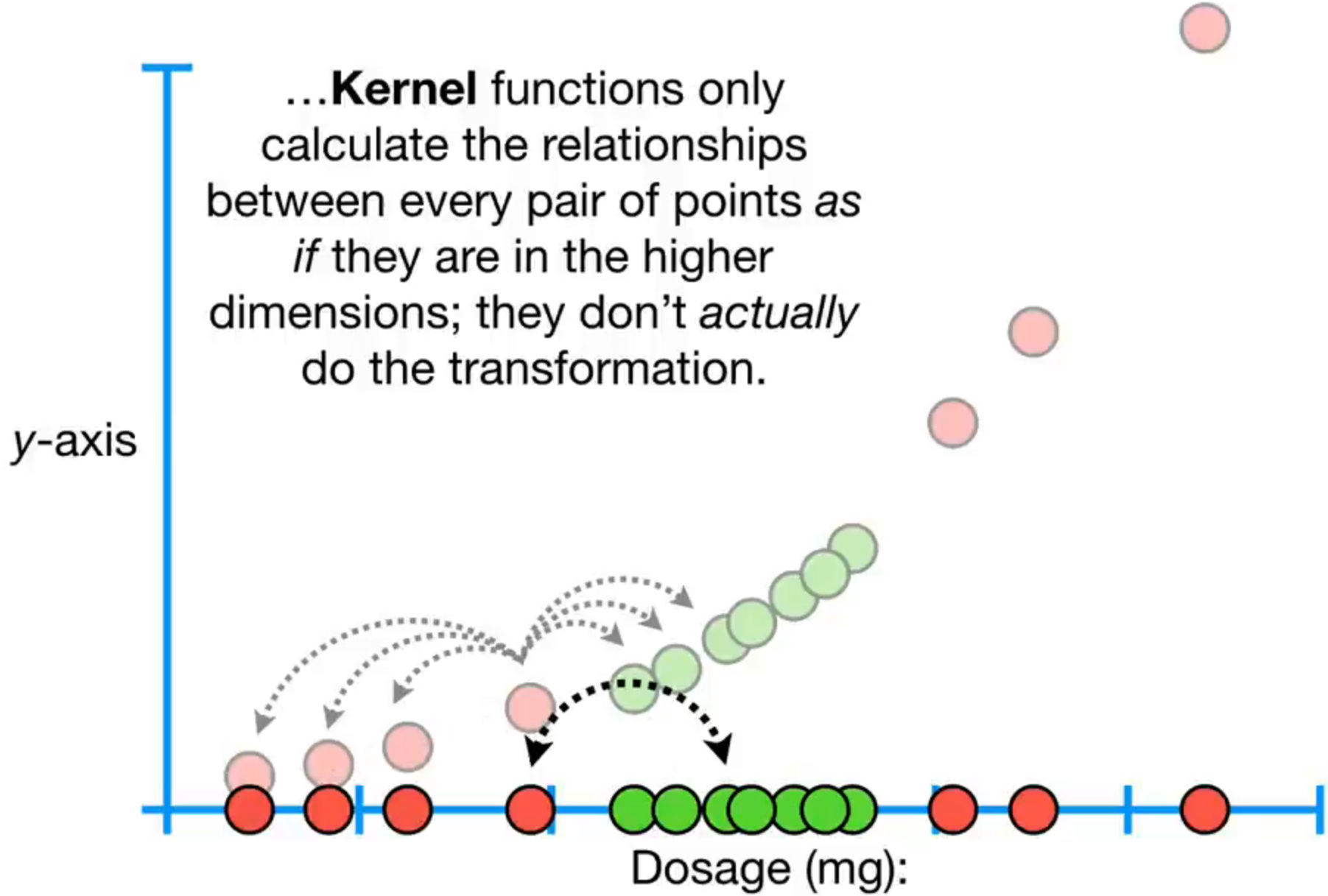


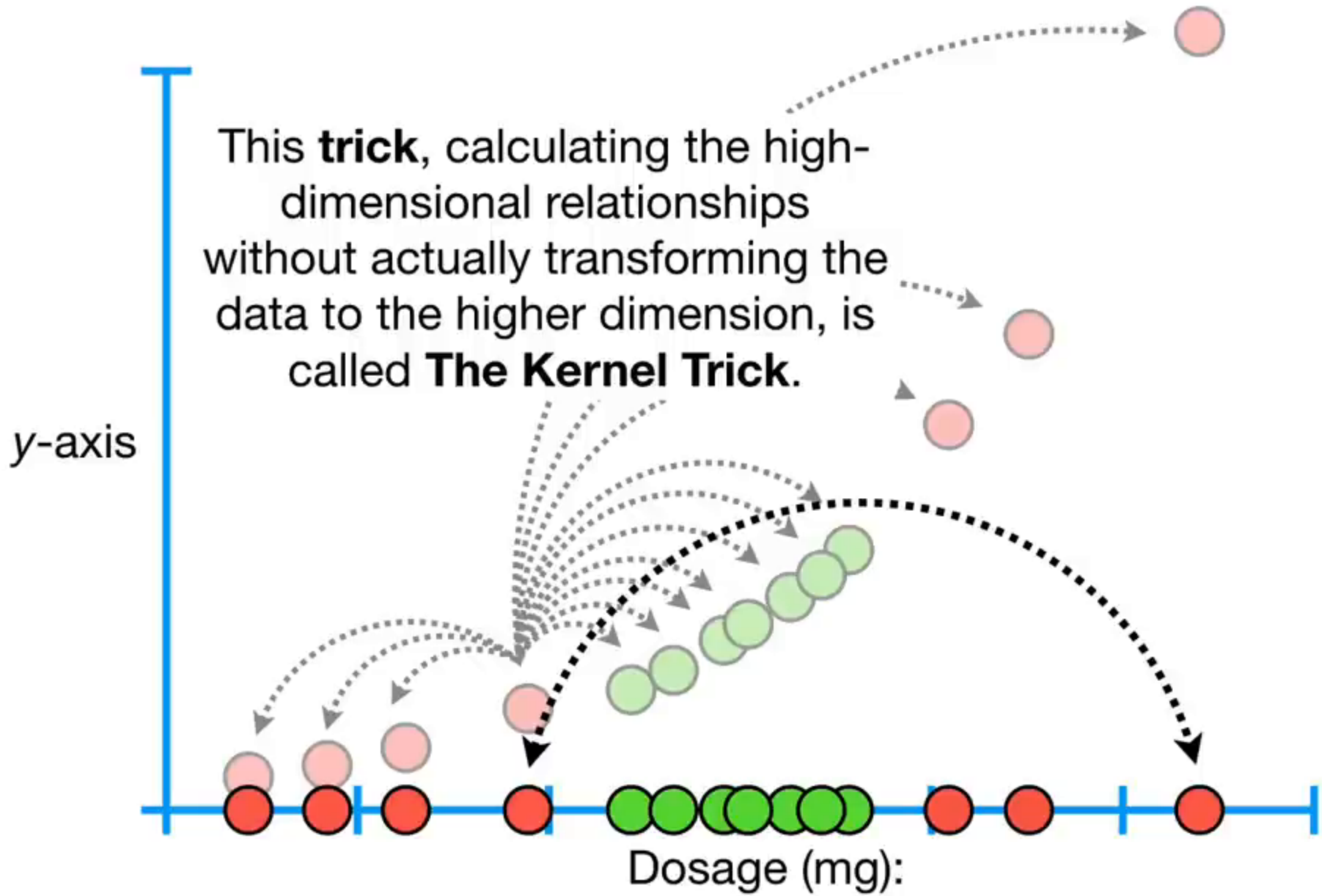
…the **Radial Kernel** uses their classification for the new observation.

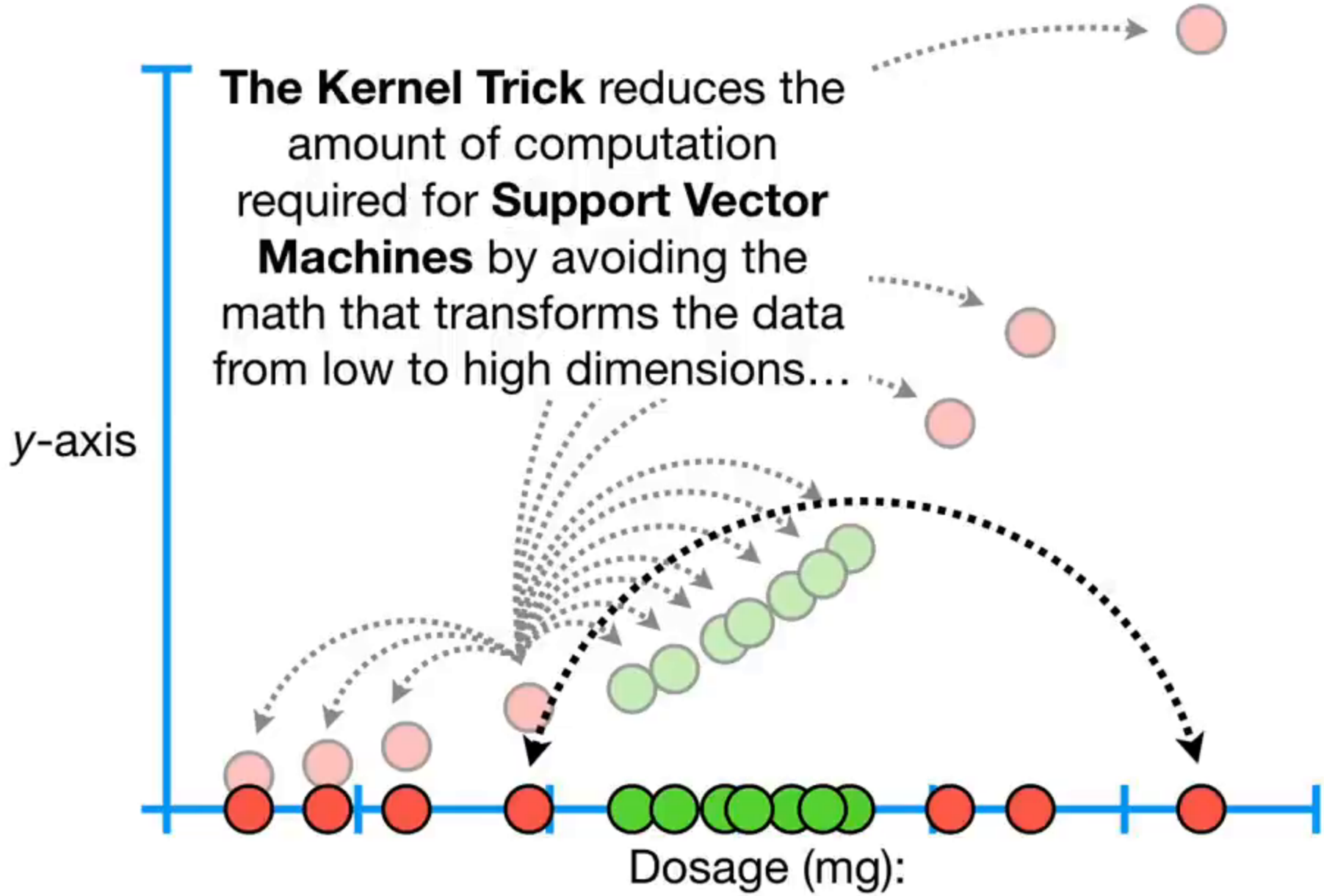


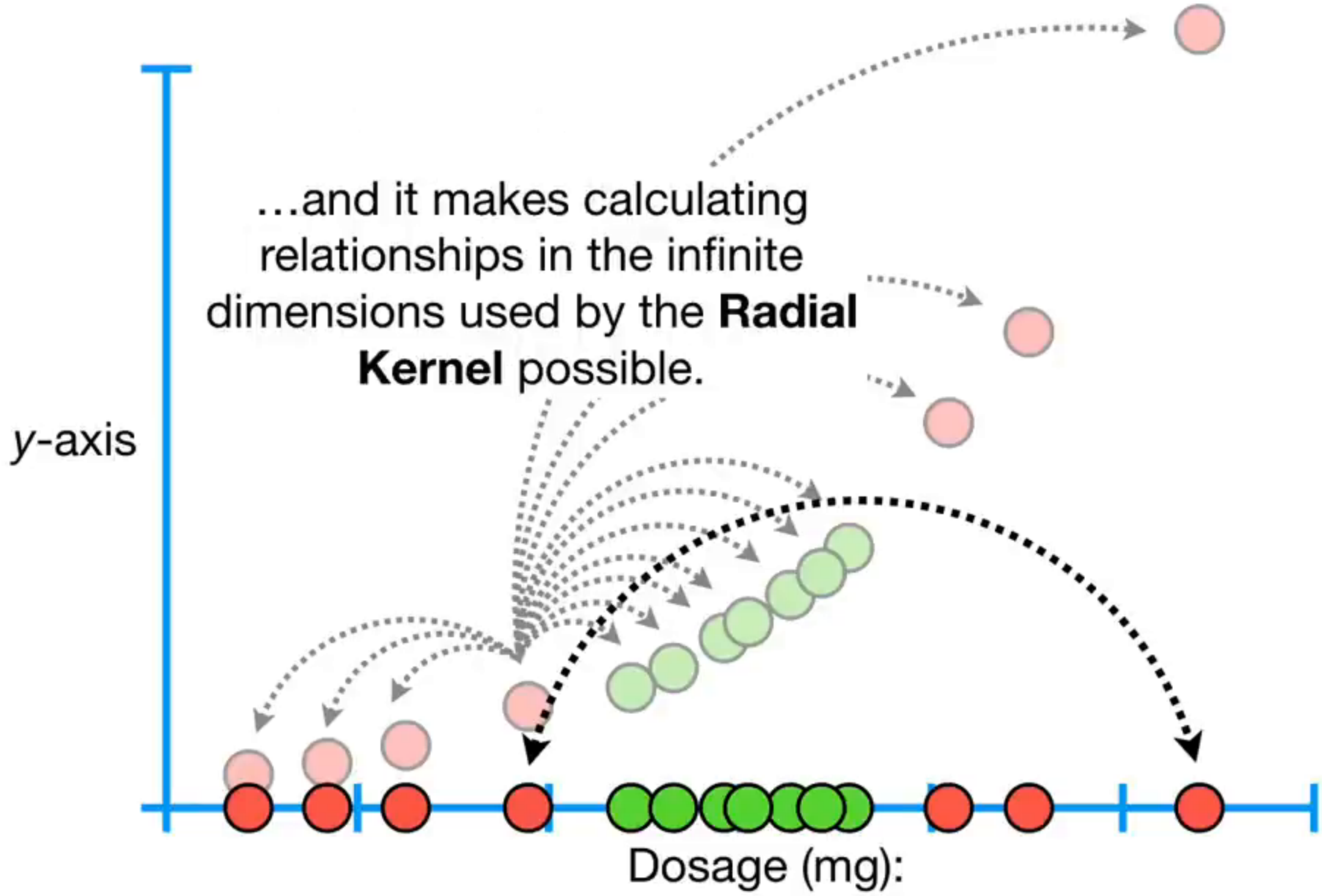
Now, for the sake of completeness, let me mention one last detail about **Kernels**.

Although the examples I have given show the data being transformed from a relatively low dimension…


…**Kernel** functions only calculate the relationships between every pair of points *as if* they are in the higher dimensions; they don't *actually* do the transformation.
y-axis
Dosage (mg):


This **trick**, calculating the high-dimensional relationships without actually transforming the data to the higher dimension, is called **The Kernel Trick**.
y-axis
Dosage (mg):

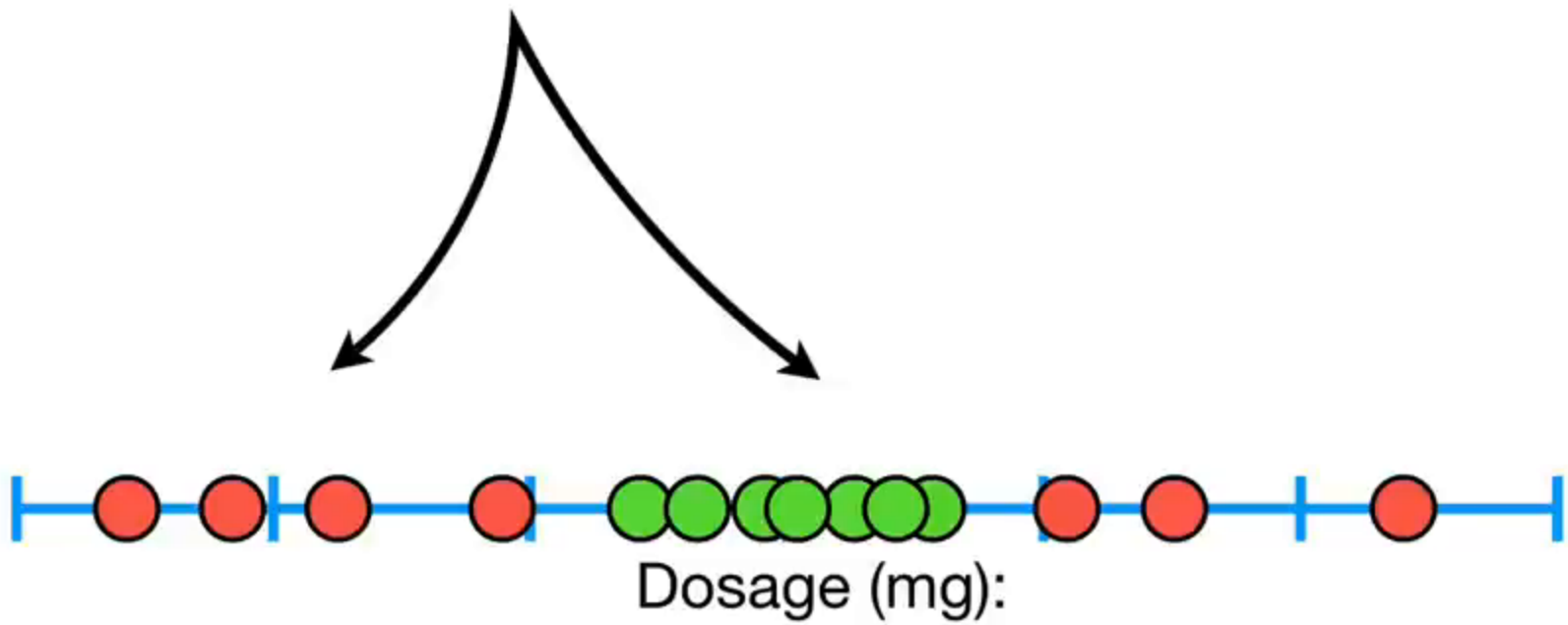

The Kernel Trick reduces the amount of computation required for **Support Vector Machines** by avoiding the math that transforms the data from low to high dimensions…
y-axis
Dosage (mg):

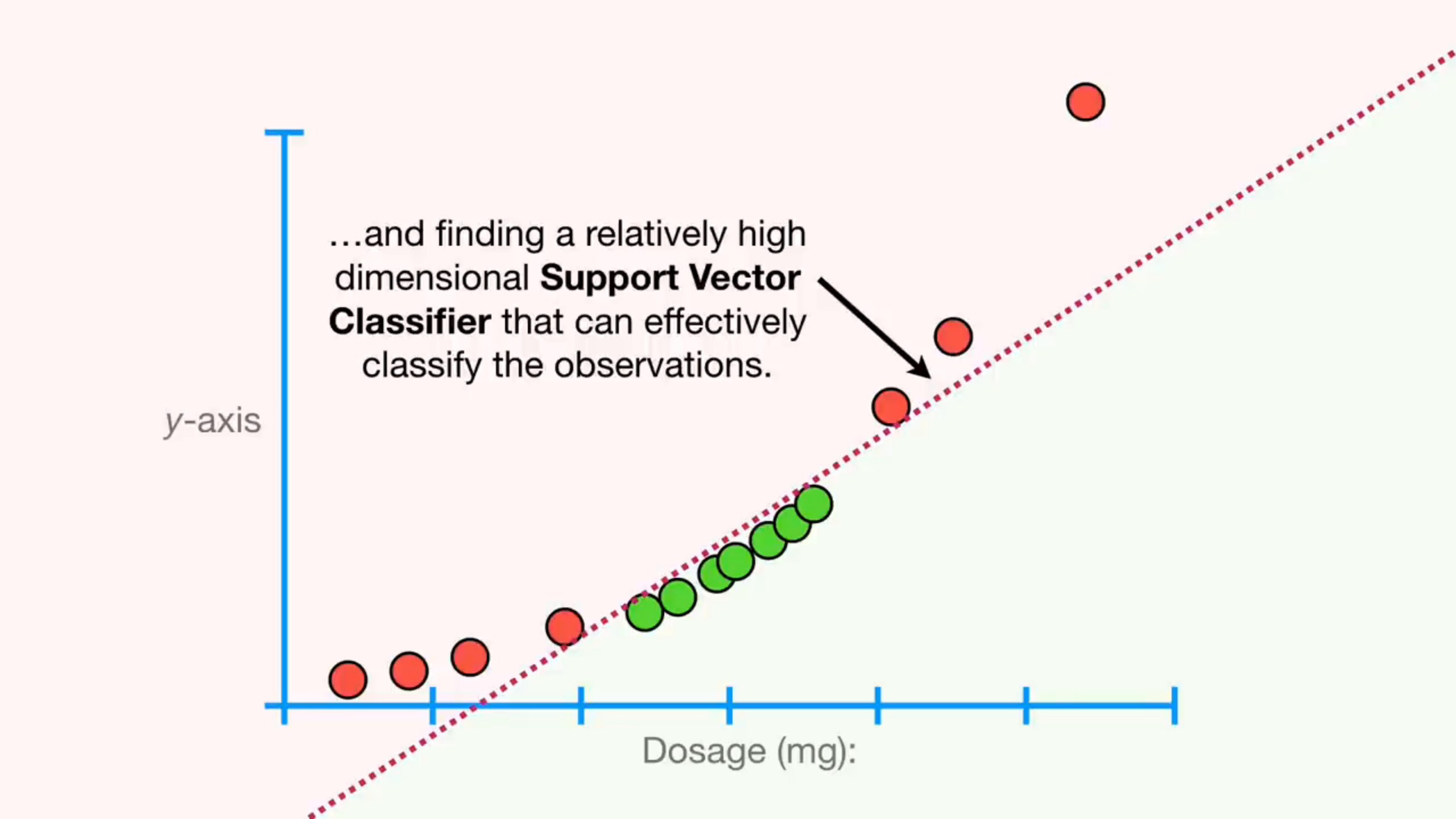

…and it makes calculating relationships in the infinite dimensions used by the **Radial Kernel** possible.
y-axis
Dosage (mg):

However, regardless of how the relationships are calculated, the concepts are the same.

Dosage (mg):

When we have **2** categories, but no obvious linear classifier that separates them in a nice way…




…and finding a relatively high dimensional **Support Vector Classifier** that can effectively classify the observations.
y-axis
Dosage (mg):

The End!!!

Subscribe!!!

Support StatQuest!!! 

Subscribe!!!

Support StatQuest!!! 

…this StatQuest isn't about that kernel!

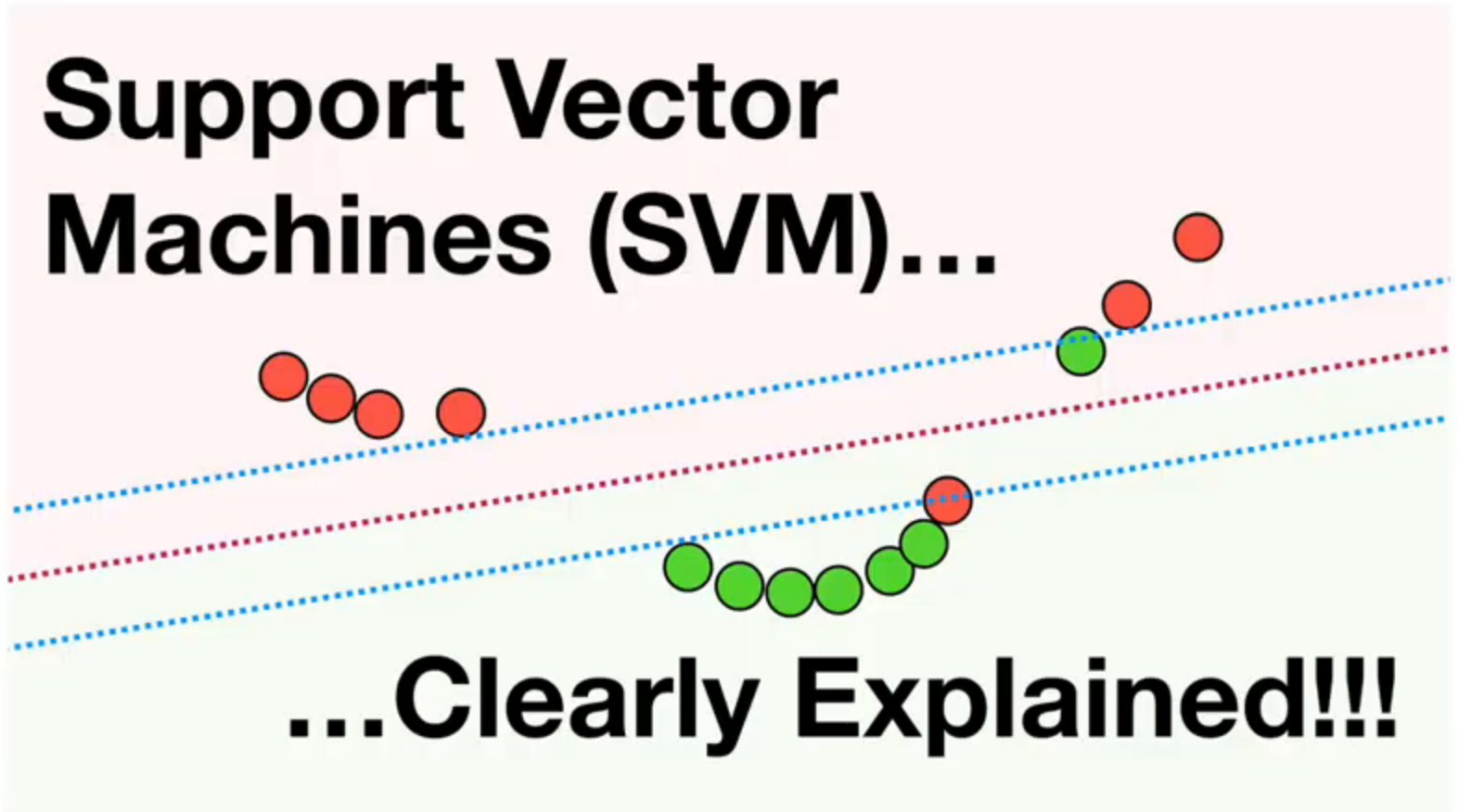
Support Vector Machines
Part 2:
The Polynomial Kernel

Specifically, we're going to talk about the **Polynomial Kernel's** parameters…

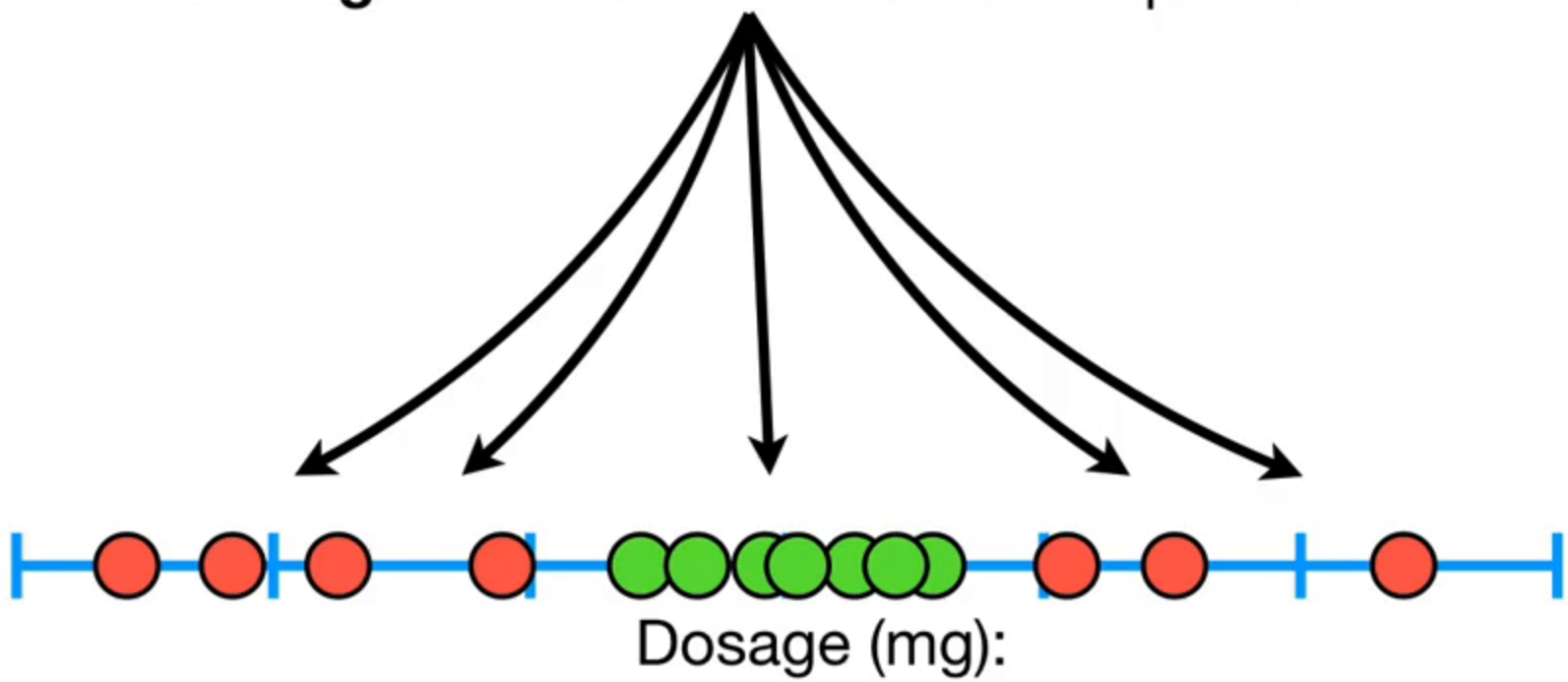
Polynomial Kernel

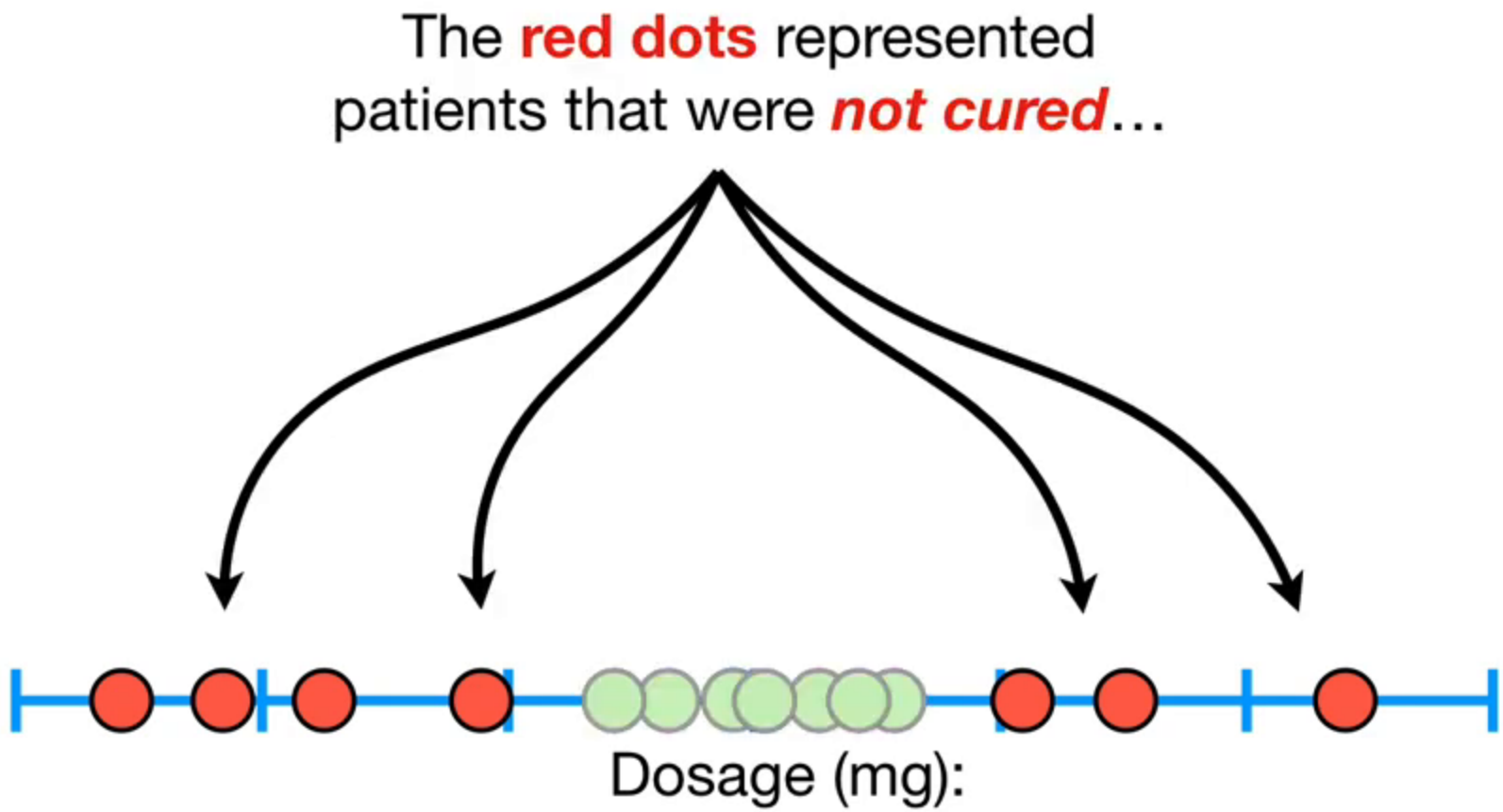
$$(a \times b + r)^d$$

NOTE: This **StatQuest** assumes that you are already familiar with **Support Vector Machines**. If not, check out the **'Quest**. The link is in the description below.

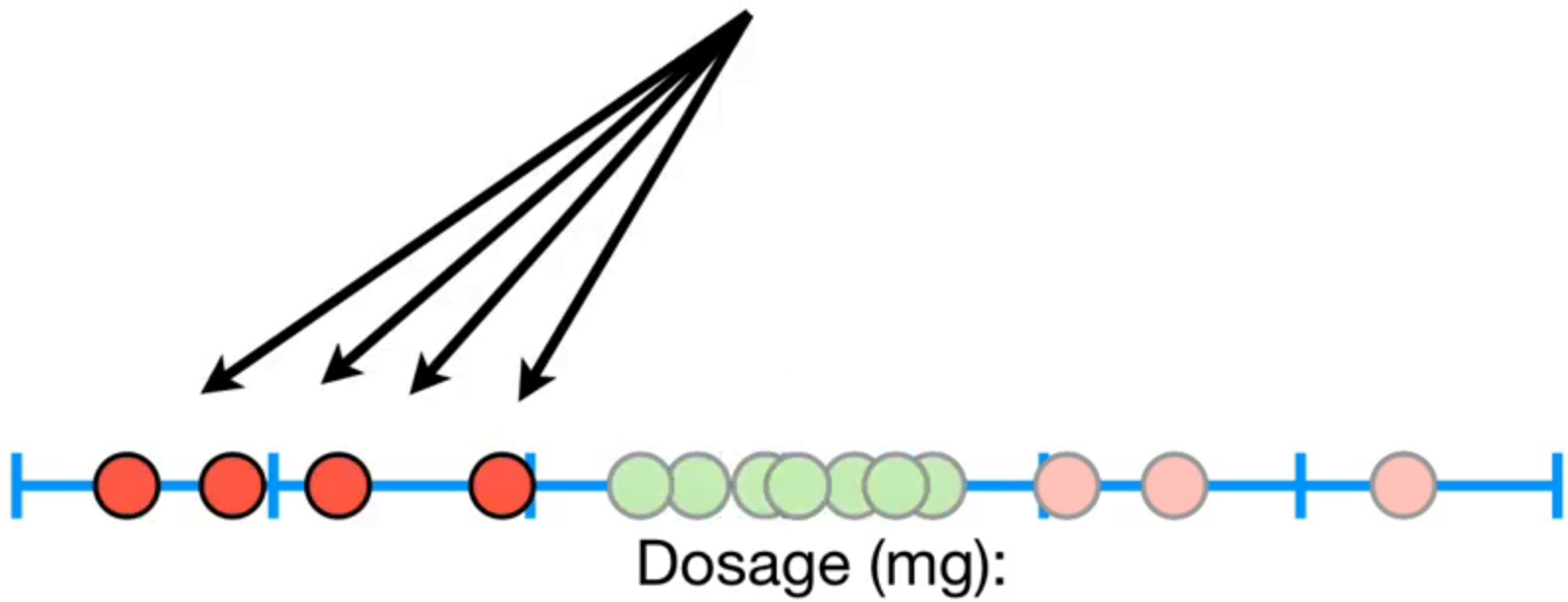


In the **StatQuest** on **Support Vector Machines**, we had a **Training Dataset** based on **Drug Dosages** measured in a bunch of patients.

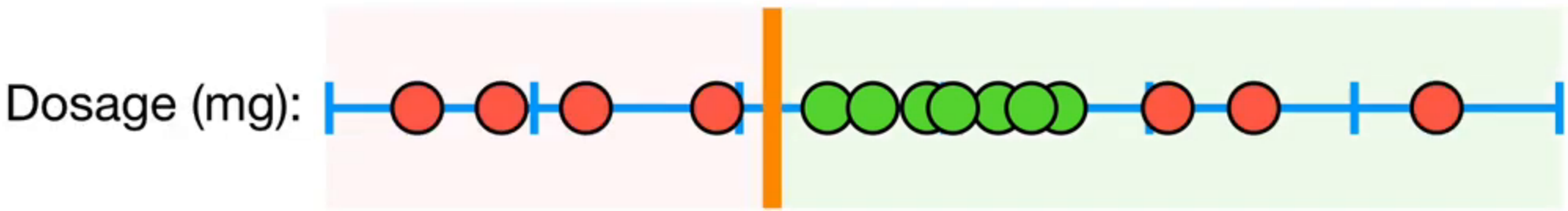



The red dots represented
patients that were not cured…
Dosage (mg):

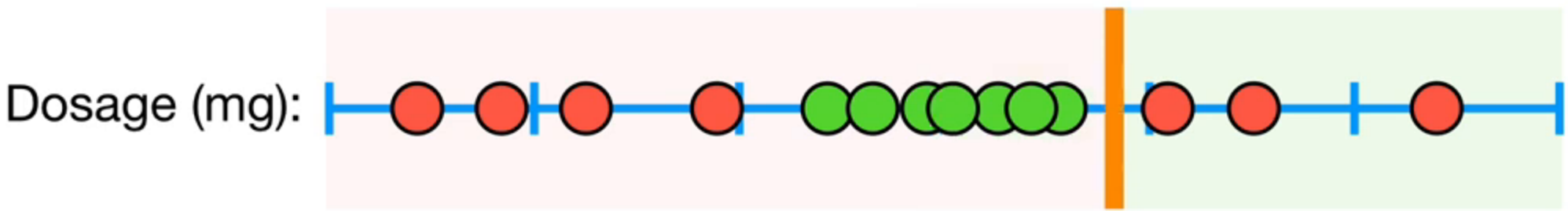
In other words, the drug doesn't work if the dosage is too small…

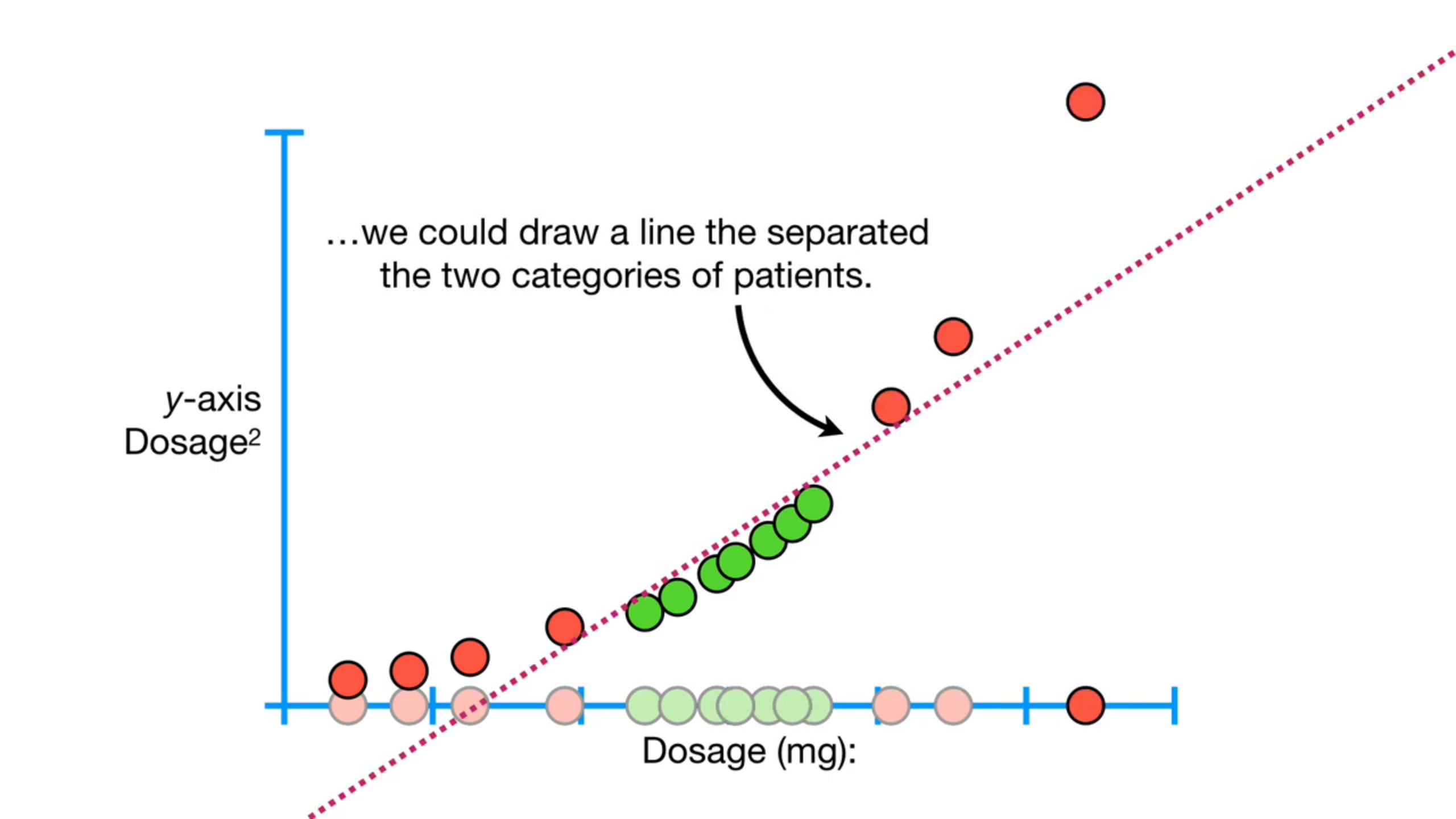


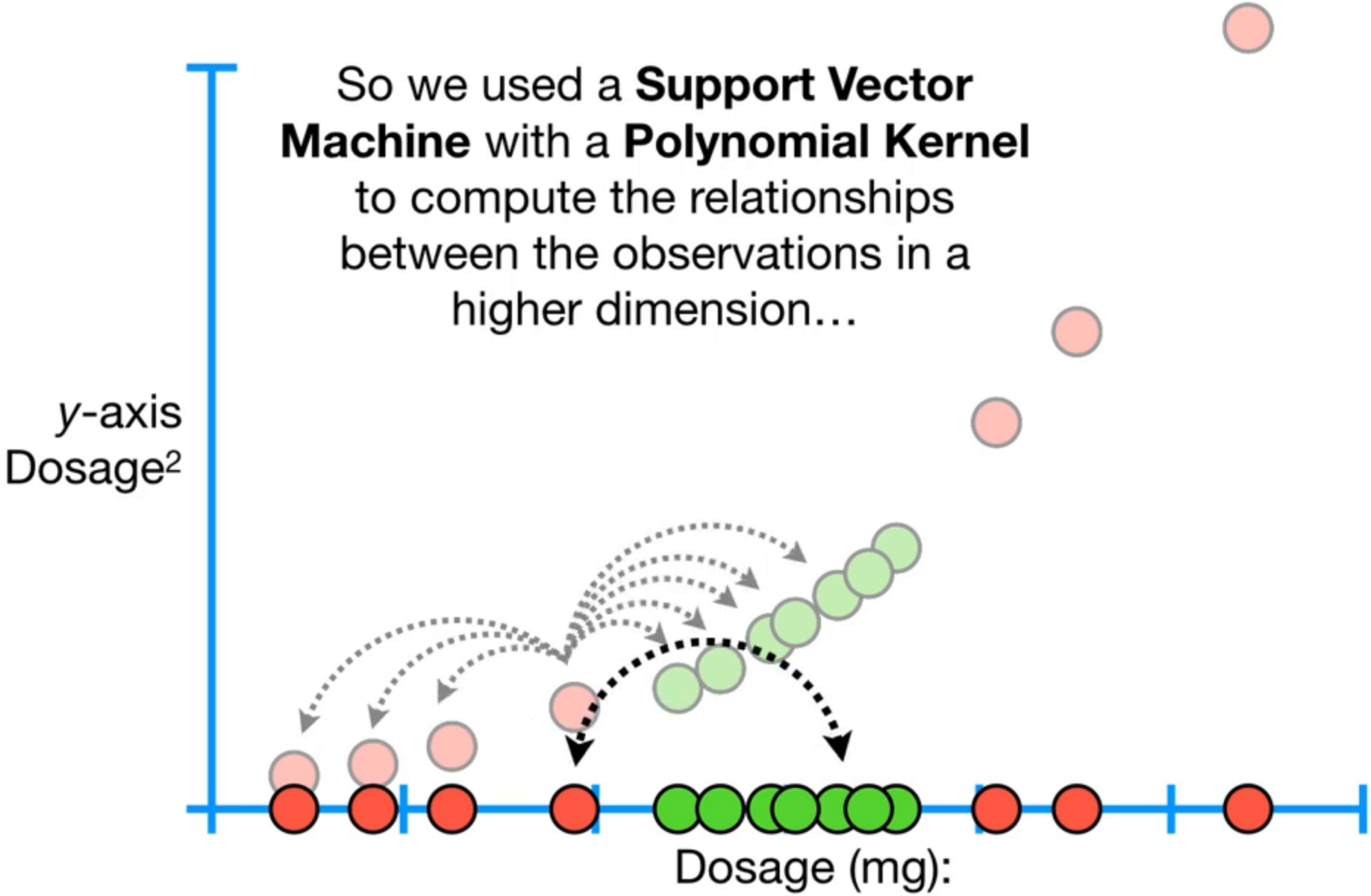
Because this **Training Dataset** had so much overlap, we were unable to find a satisfying **Support Vector Classifier** to separate the patients that were cured from the patients that were not cured.

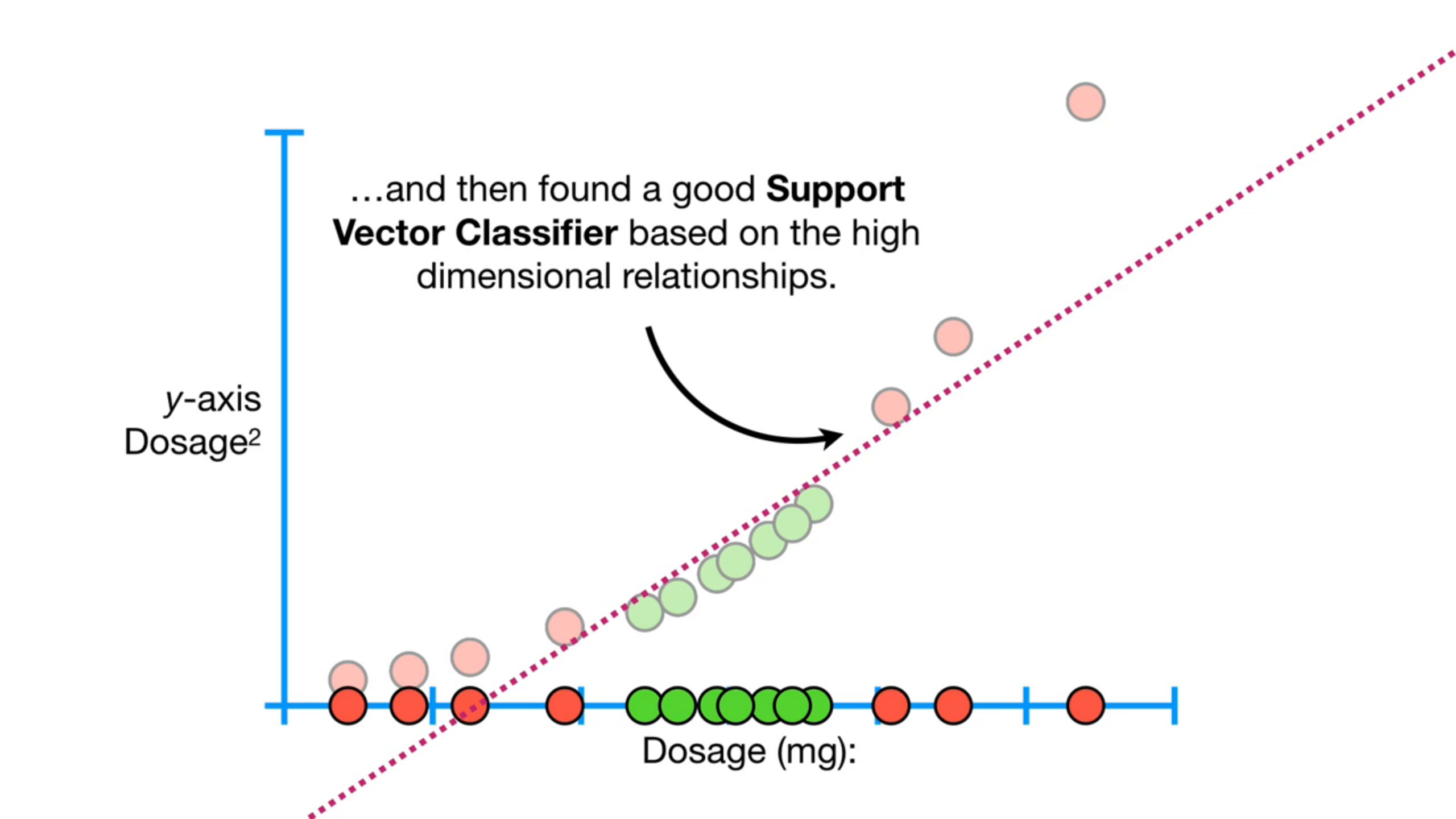


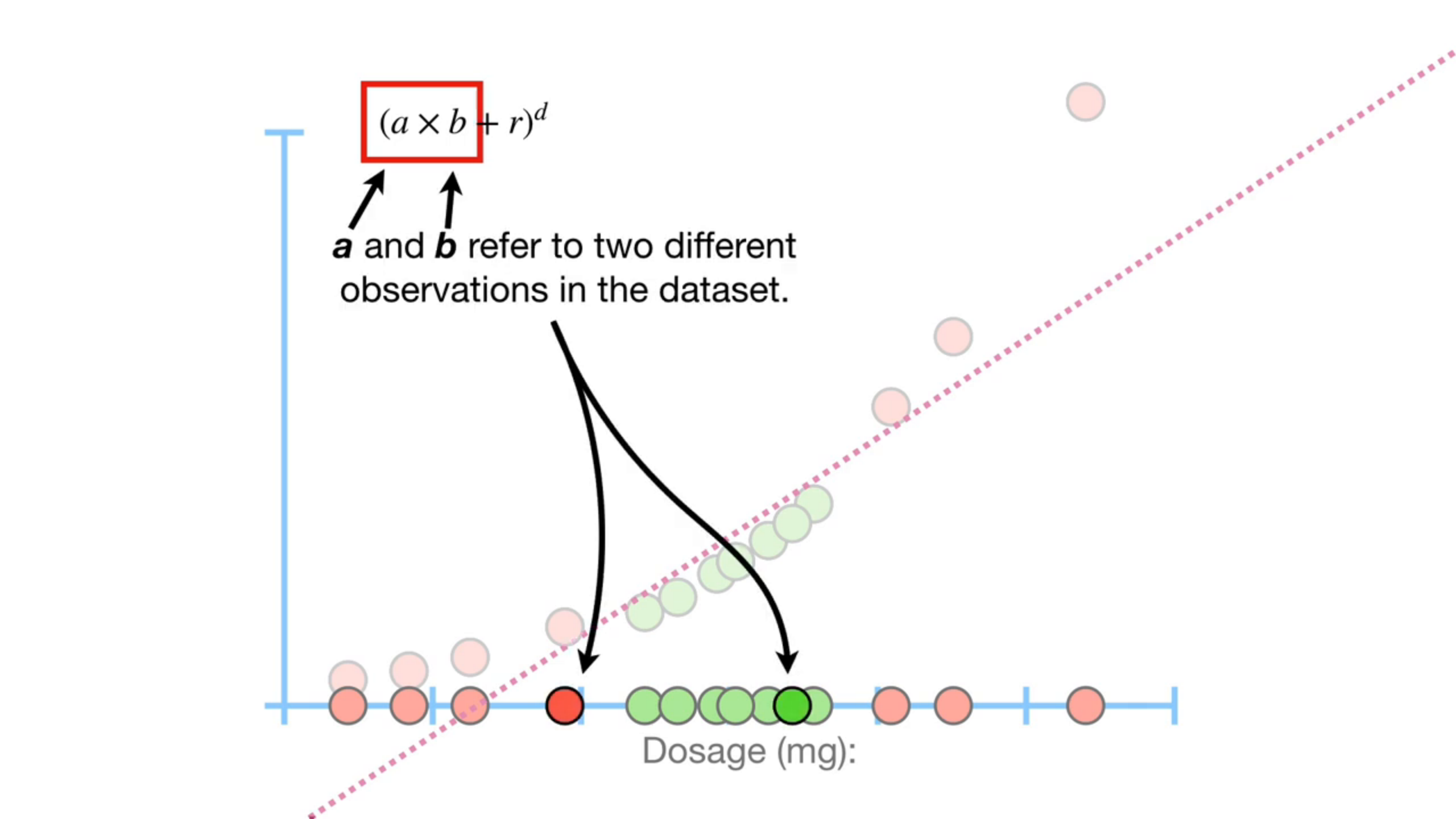
Because this **Training Dataset** had so much overlap, we were unable to find a satisfying **Support Vector Classifier** to separate the patients that were cured from the patients that were not cured.

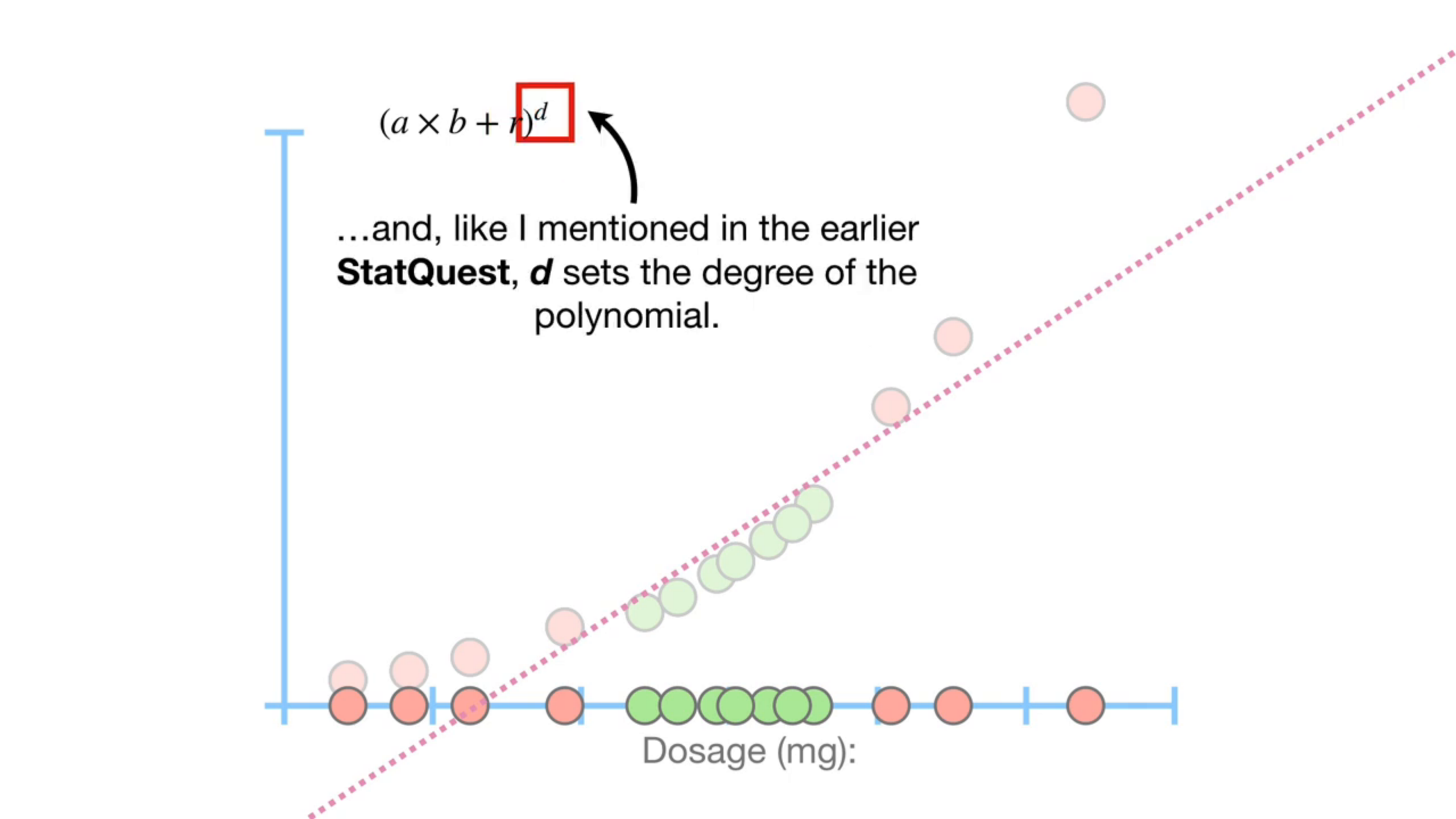


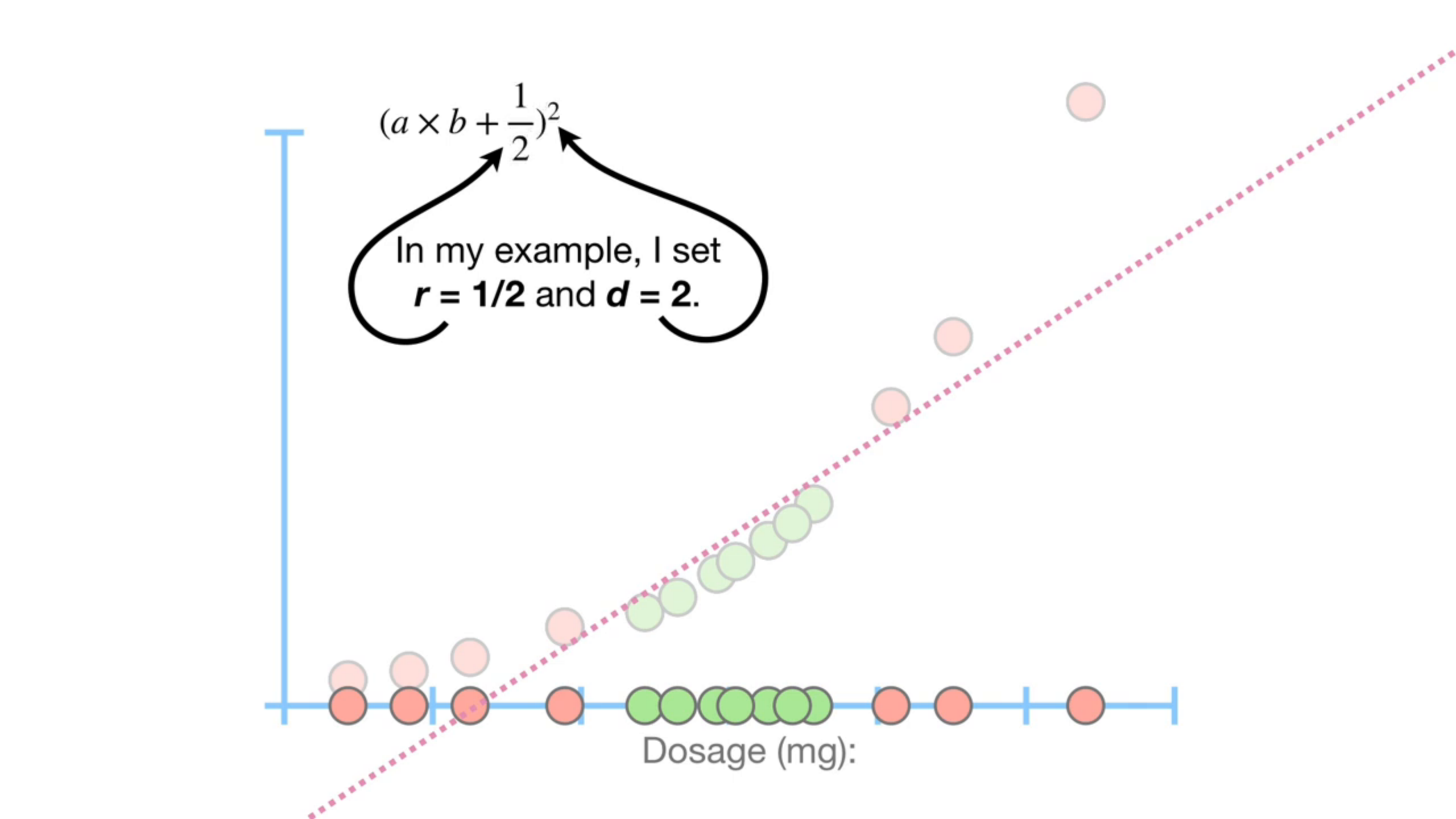

…we could draw a line the separated
the two categories of patients.
y-axis
Dosage²
Dosage (mg):


So we used a **Support Vector Machine** with a **Polynomial Kernel** to compute the relationships between the observations in a higher dimension…
y-axis
Dosage²
Dosage (mg):


…and then found a good **Support Vector Classifier** based on the high dimensional relationships.
y-axis
Dosage²
Dosage (mg):


$(a \times b + r)^d$
a and **b** refer to two different observations in the dataset.
Dosage (mg):

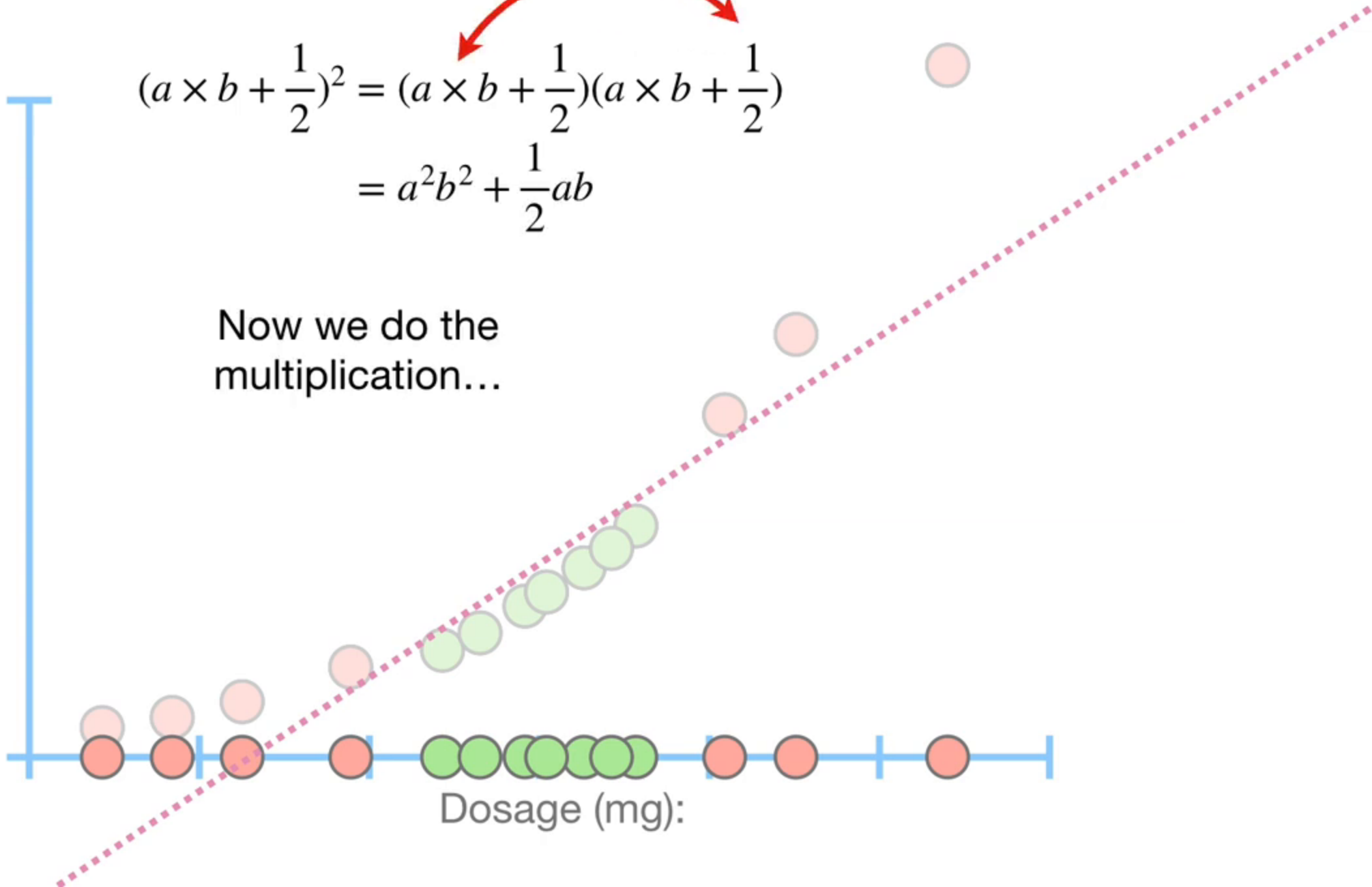

$(a \times b + r)^d$
…and, like I mentioned in the earlier **StatQuest**, ***d*** sets the degree of the polynomial.
Dosage (mg):


$(a \times b + \frac{1}{2})^2$
In my example, I set
r = 1/2 and **d = 2**.
Dosage (mg):

$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= a^2b^2 + \frac{1}{2}ab$$

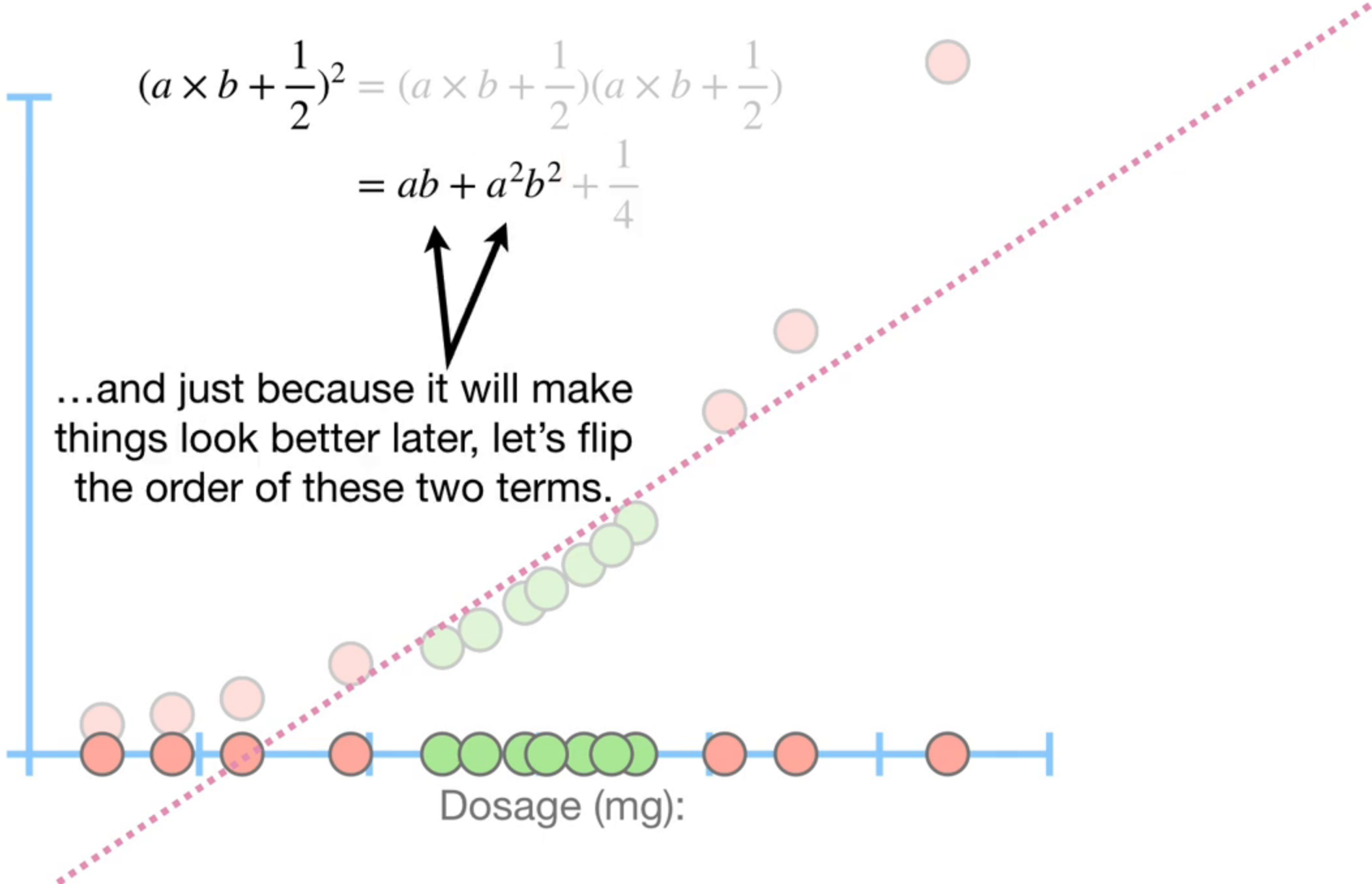
Now we do the multiplication…



$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

…and just because it will make things look better later, let's flip the order of these two terms.

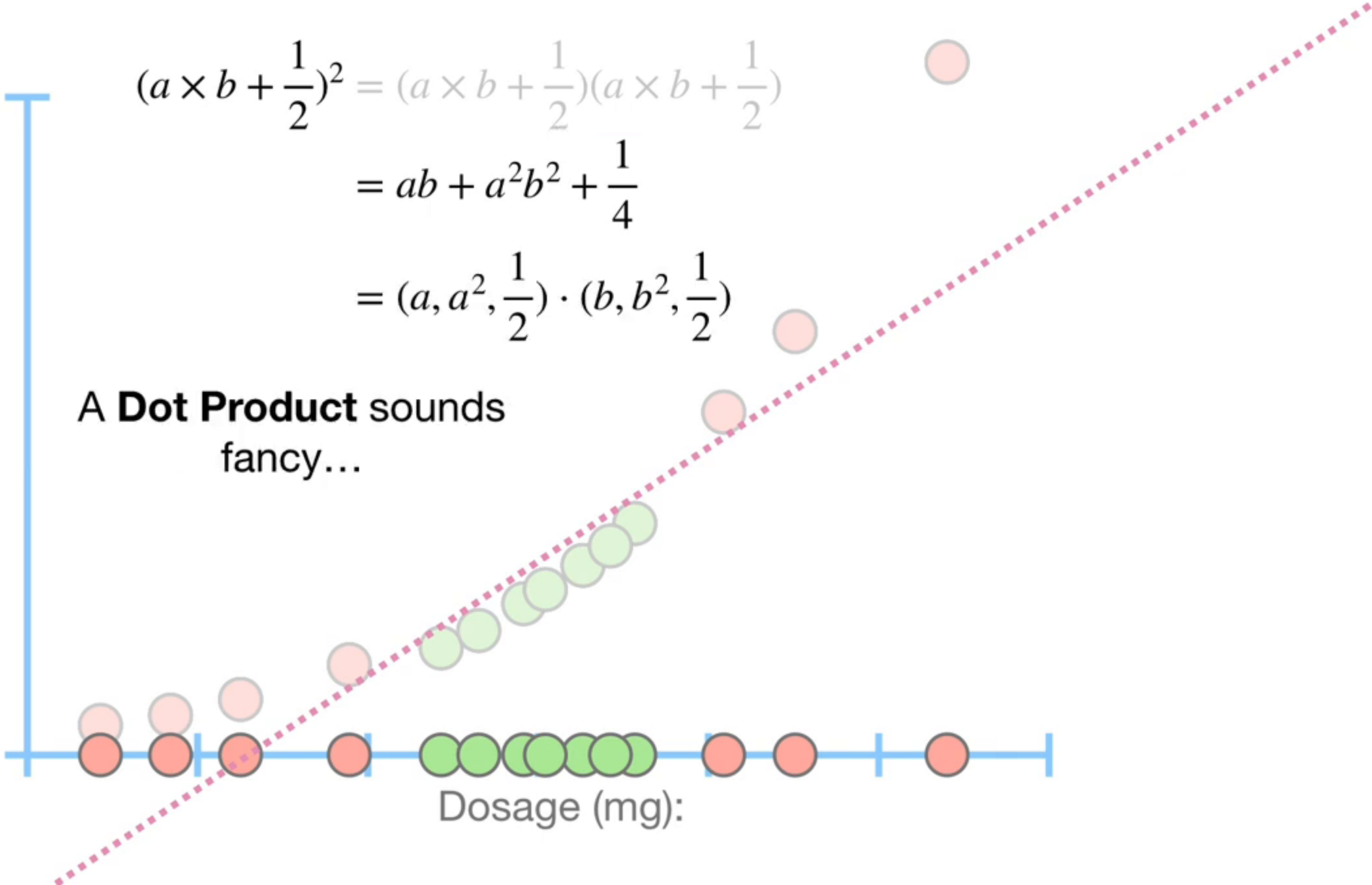


$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

$$= (a, a^2, \frac{1}{2}) \cdot (b, b^2, \frac{1}{2})$$

A **Dot Product** sounds fancy…

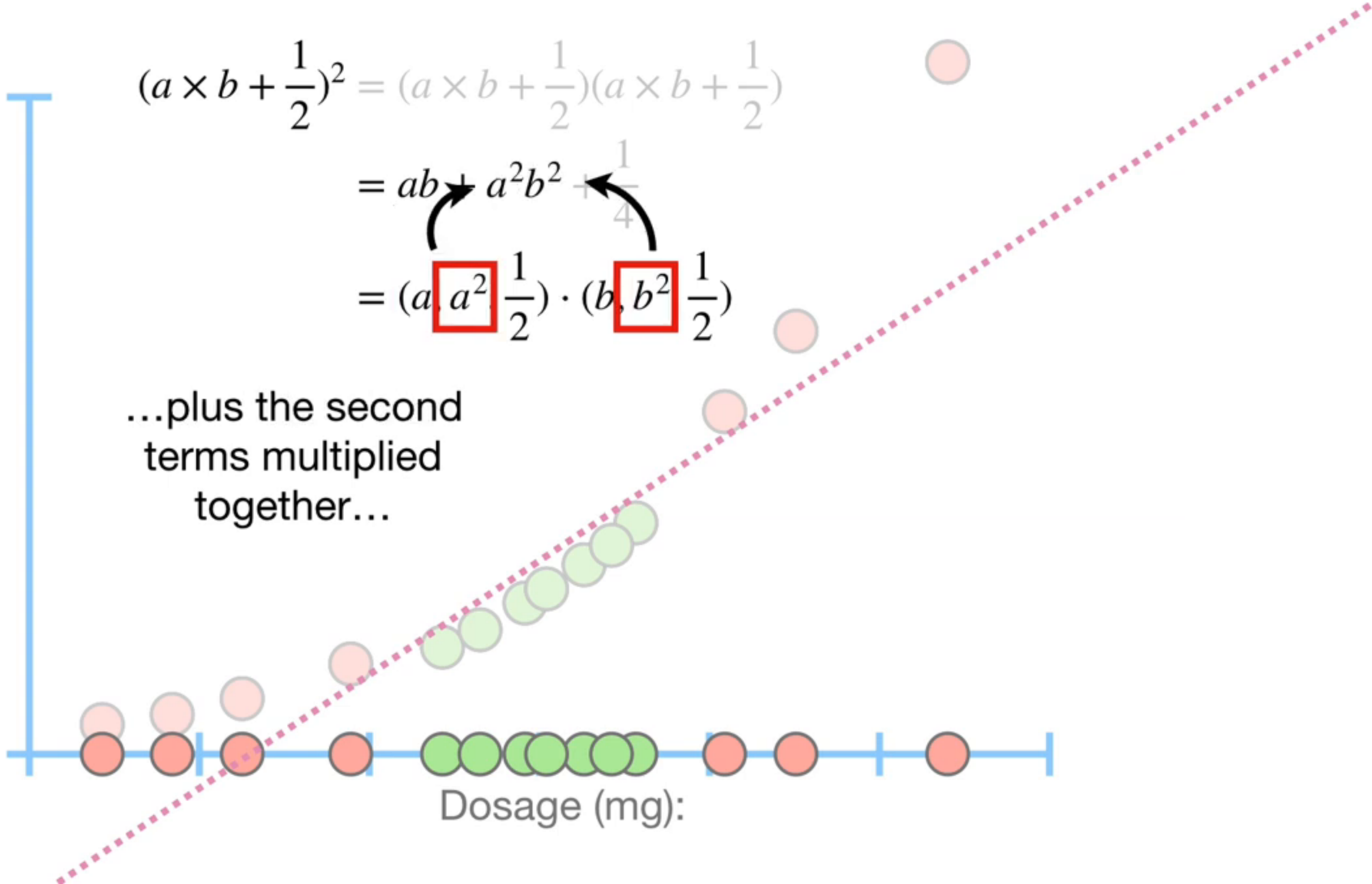


$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

$$= (a, a^2, \frac{1}{2}) \cdot (b, b^2, \frac{1}{2})$$

…plus the second terms multiplied together…

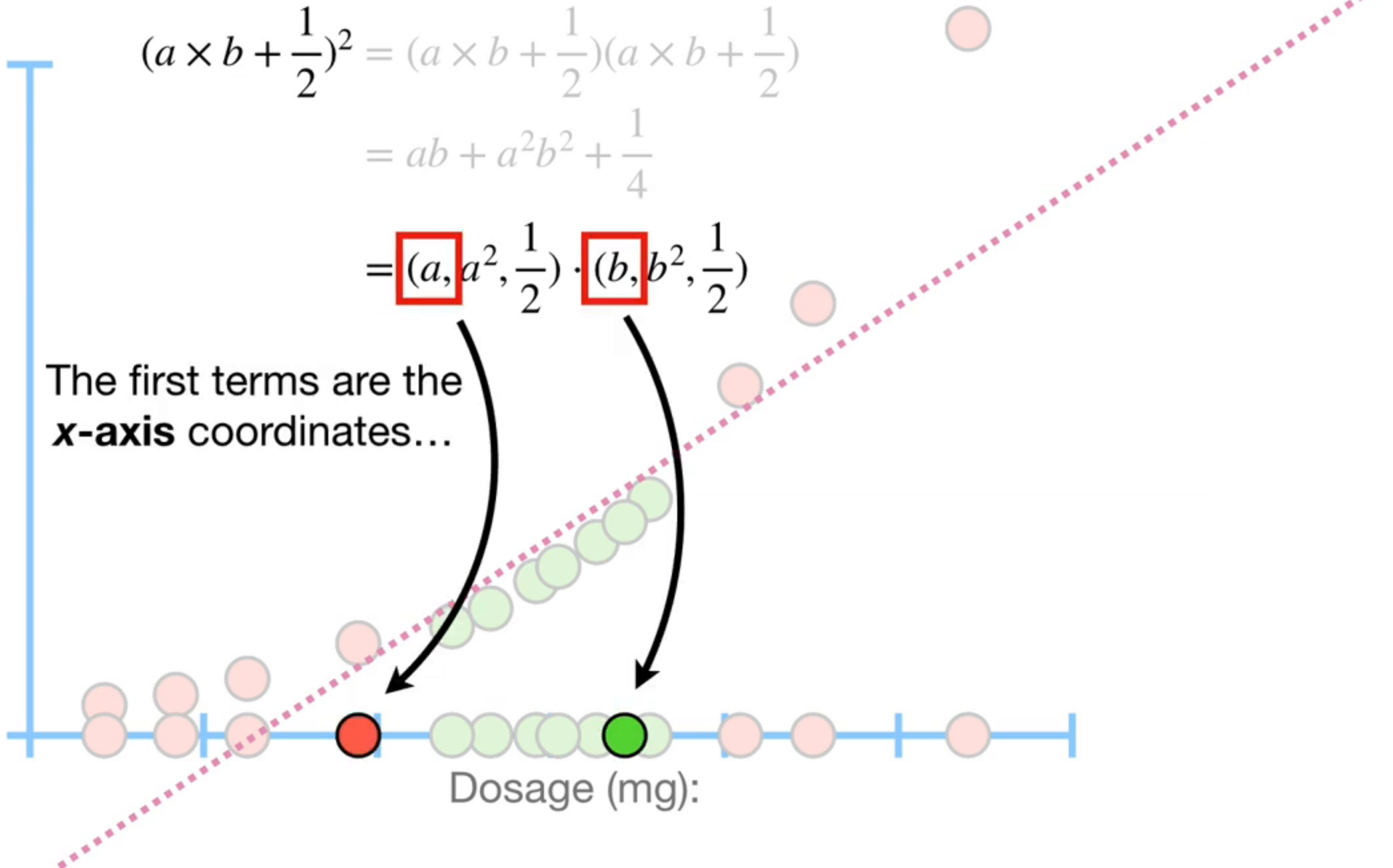


$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

$$= (a, a^2, \frac{1}{2}) \cdot (b, b^2, \frac{1}{2})$$

The first terms are the ***x*-axis** coordinates…

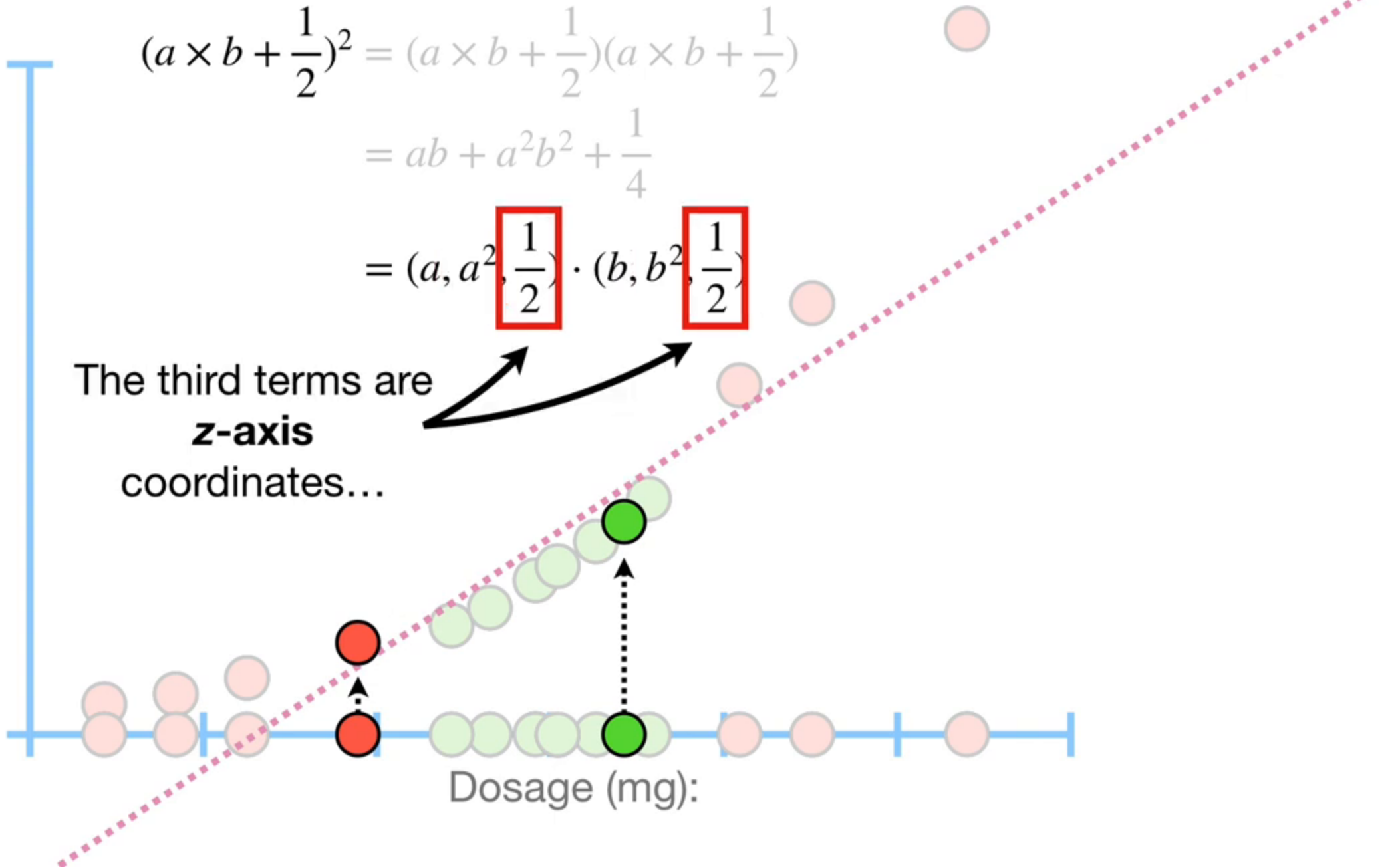


$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

$$= (a, a^2, \frac{1}{2}) \cdot (b, b^2, \frac{1}{2})$$

The third terms are ***z*-axis** coordinates…

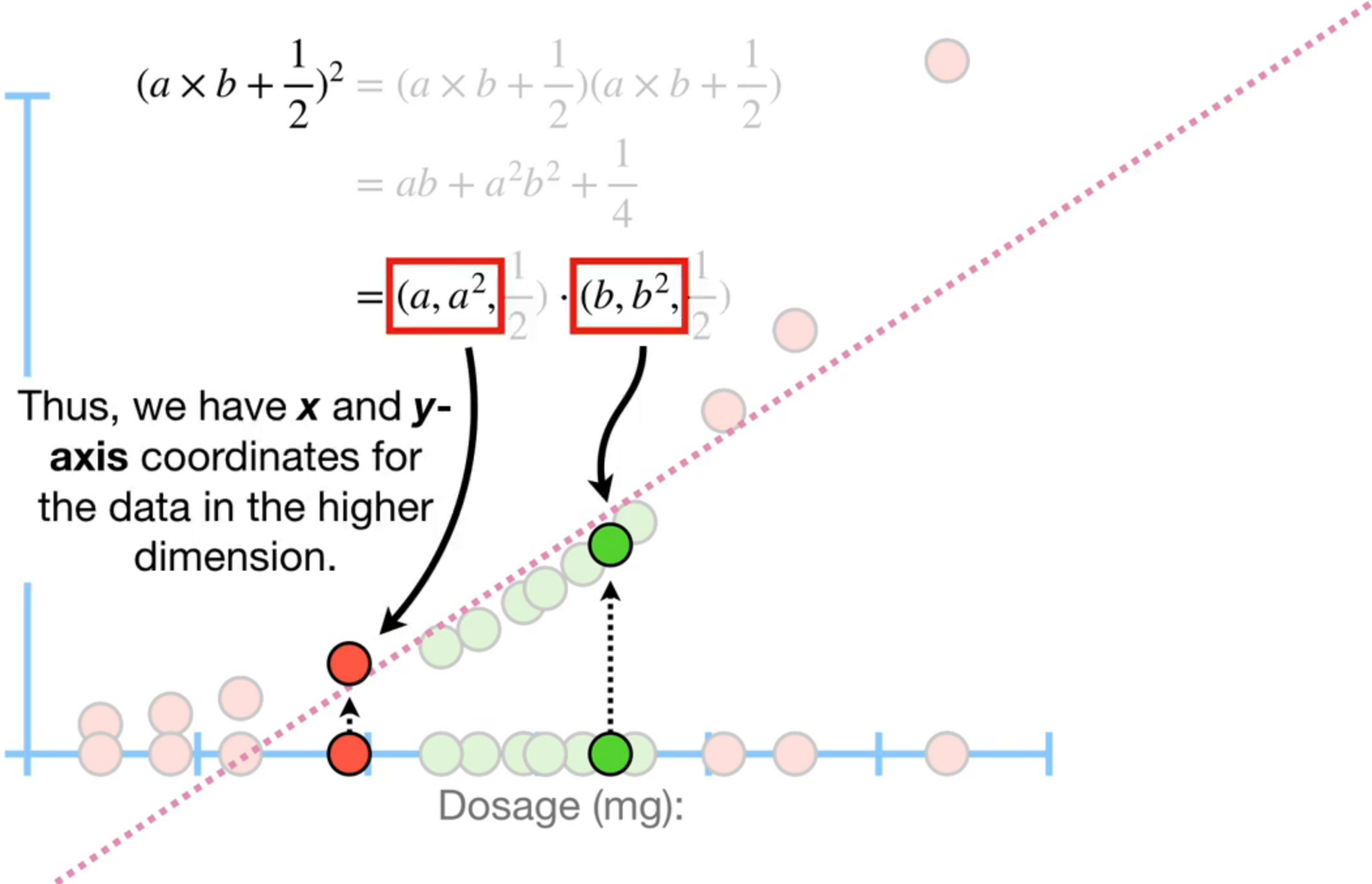


$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

$$= (a, a^2, \frac{1}{2}) \cdot (b, b^2, \frac{1}{2})$$

Thus, we have ***x*** and ***y*-axis** coordinates for the data in the higher dimension.



$$(a \times b + 1)^2$$

Alternatively, we could have set ***r*** **= 1** and ***d*** **= 2.**

Dosage (mg):

$$(a \times b + 1)^2 = (a \times b + 1)(a \times b + 1)$$

$$= 2ab + a^2b^2 + 1$$

…and this **Dot Product**.

$$= (\sqrt{2}a, a^2, 1) \cdot (\sqrt{2}b, b^2, 1)$$

Dosage (mg):

$$(a \times b + 1)^2 = (a \times b + 1)(a \times b + 1)$$

$$= 2ab + a^2b^2 + 1$$

…and the result should equal to the polynomial.

$$= (\sqrt{2}a, a^2, 1) \cdot (\sqrt{2}b, b^2, 1)$$

$$2ab + a^2b^2 + 1$$

Dosage (mg):

$$(a \times b + 1)^2 = (a \times b + 1)(a \times b + 1)$$

$$= 2ab + a^2b^2 + 1$$

$$= (\sqrt{2}a, a^2, 1) \cdot (\sqrt{2}b, b^2, 1)$$

…the new ***x*-axis** coordinates are the square root of **2** times the original **Dosage** values.

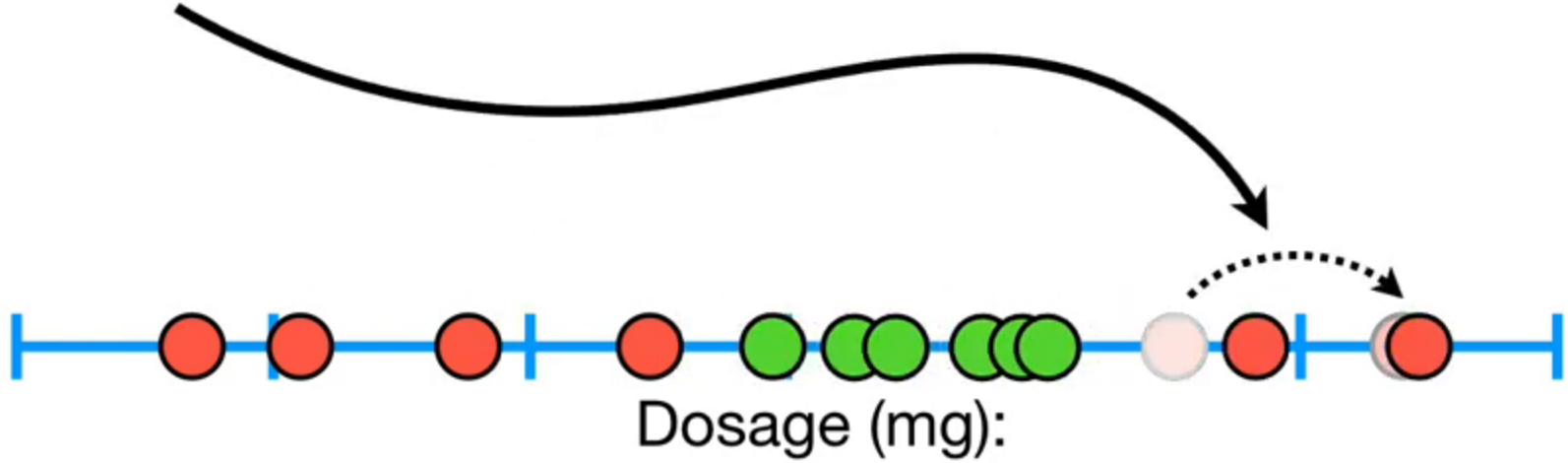
Dosage (mg):

$$(a \times b + 1)^2 = (a \times b + 1)(a \times b + 1)$$

$$= 2ab + a^2b^2 + 1$$

$$= (\sqrt{2}a, a^2, 1) \cdot (\sqrt{2}b, b^2, 1)$$

So we move the points on the ***x*-axis** over by a factor of the square root of **2**.

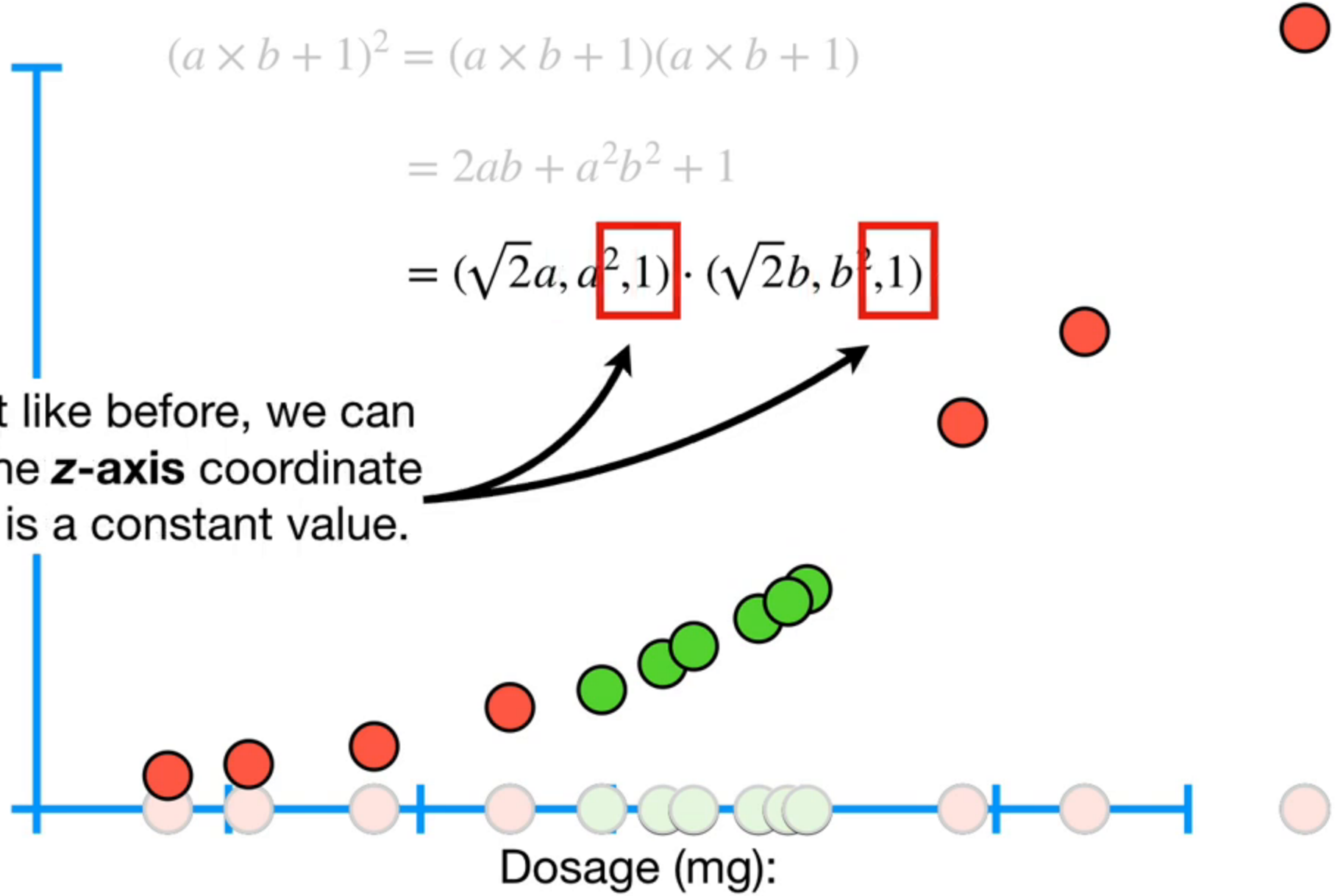


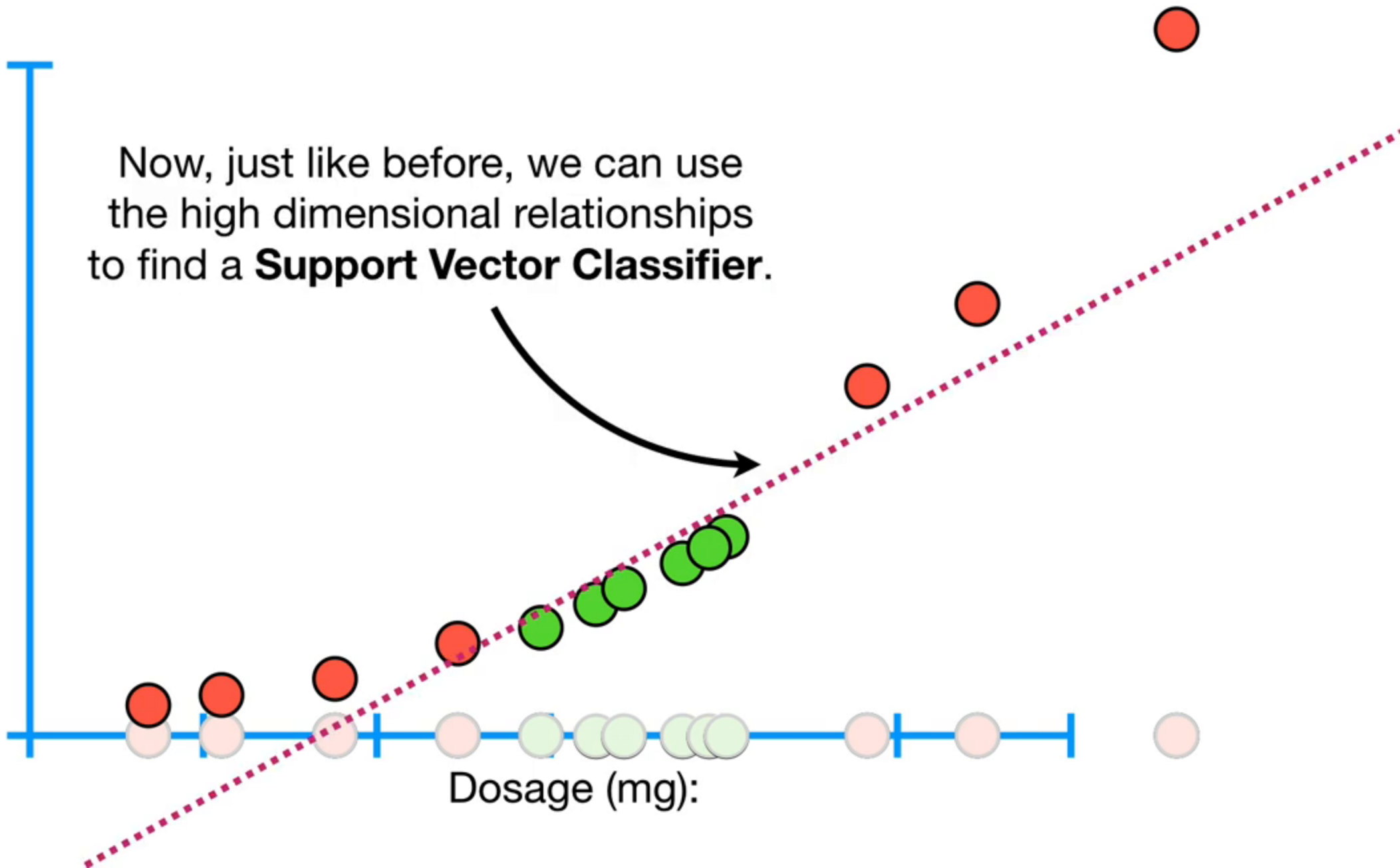
$$(a \times b + 1)^2 = (a \times b + 1)(a \times b + 1)$$

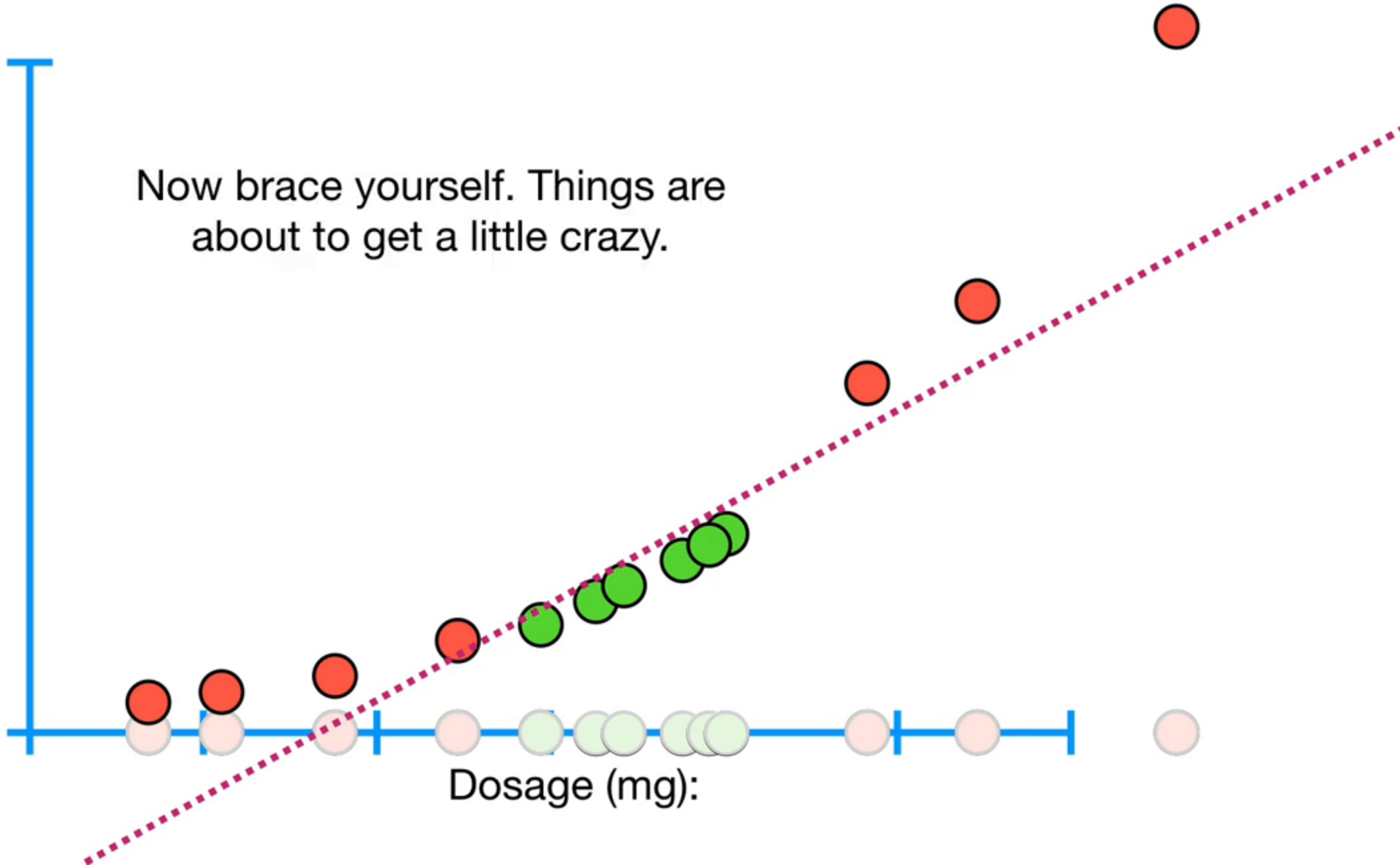
$$= 2ab + a^2b^2 + 1$$

$$= (\sqrt{2}a, a^2, 1) \cdot (\sqrt{2}b, b^2, 1)$$

And just like before, we can ignore the ***z*-axis** coordinate since it is a constant value.




Now, just like before, we can use the high dimensional relationships to find a **Support Vector Classifier**.
Dosage (mg):

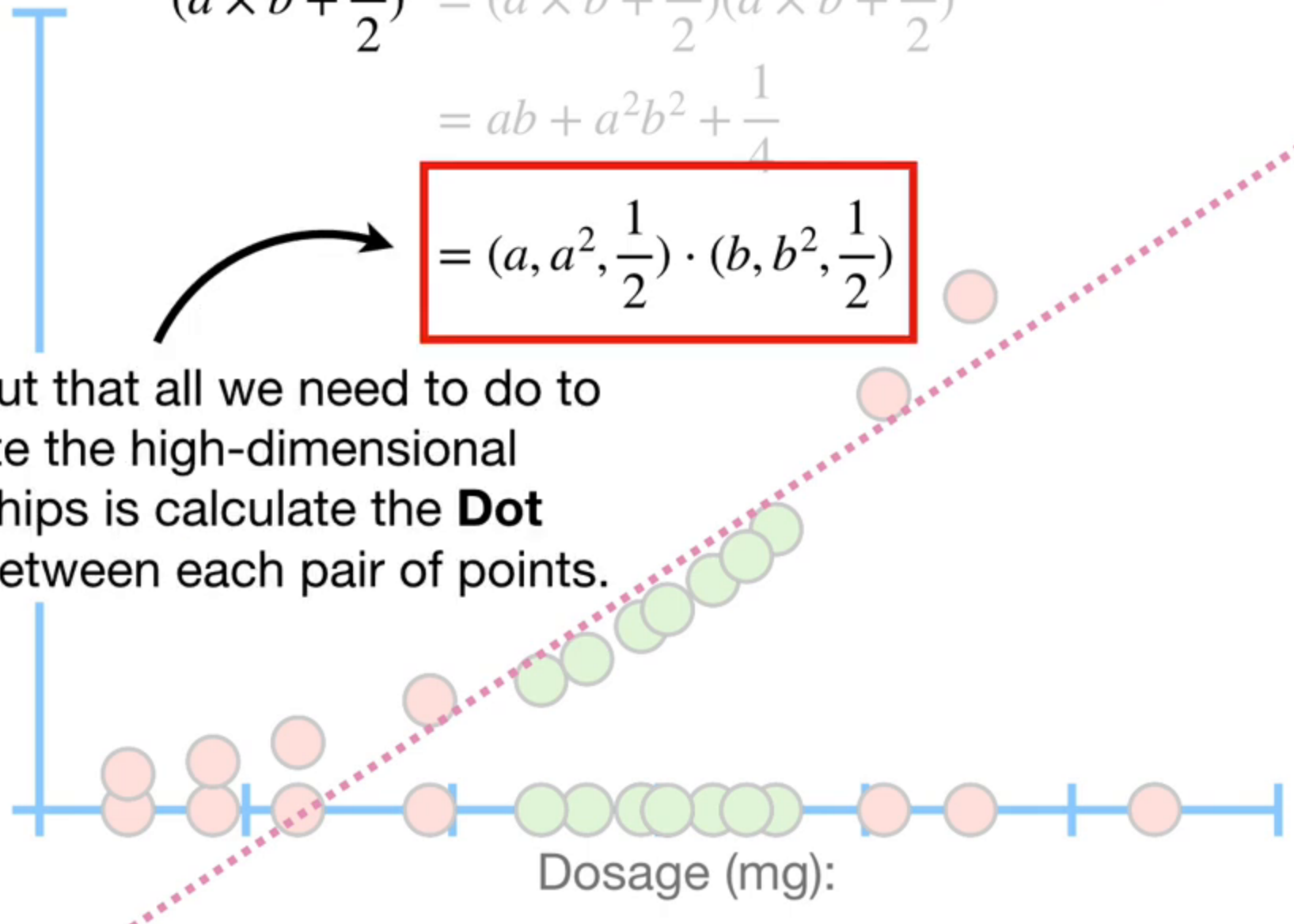

Now brace yourself. Things are about to get a little crazy.
Dosage (mg):

$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

$$= (a, a^2, \frac{1}{2}) \cdot (b, b^2, \frac{1}{2})$$

…it turns out that all we need to do to calculate the high-dimensional relationships is calculate the **Dot Products** between each pair of points.

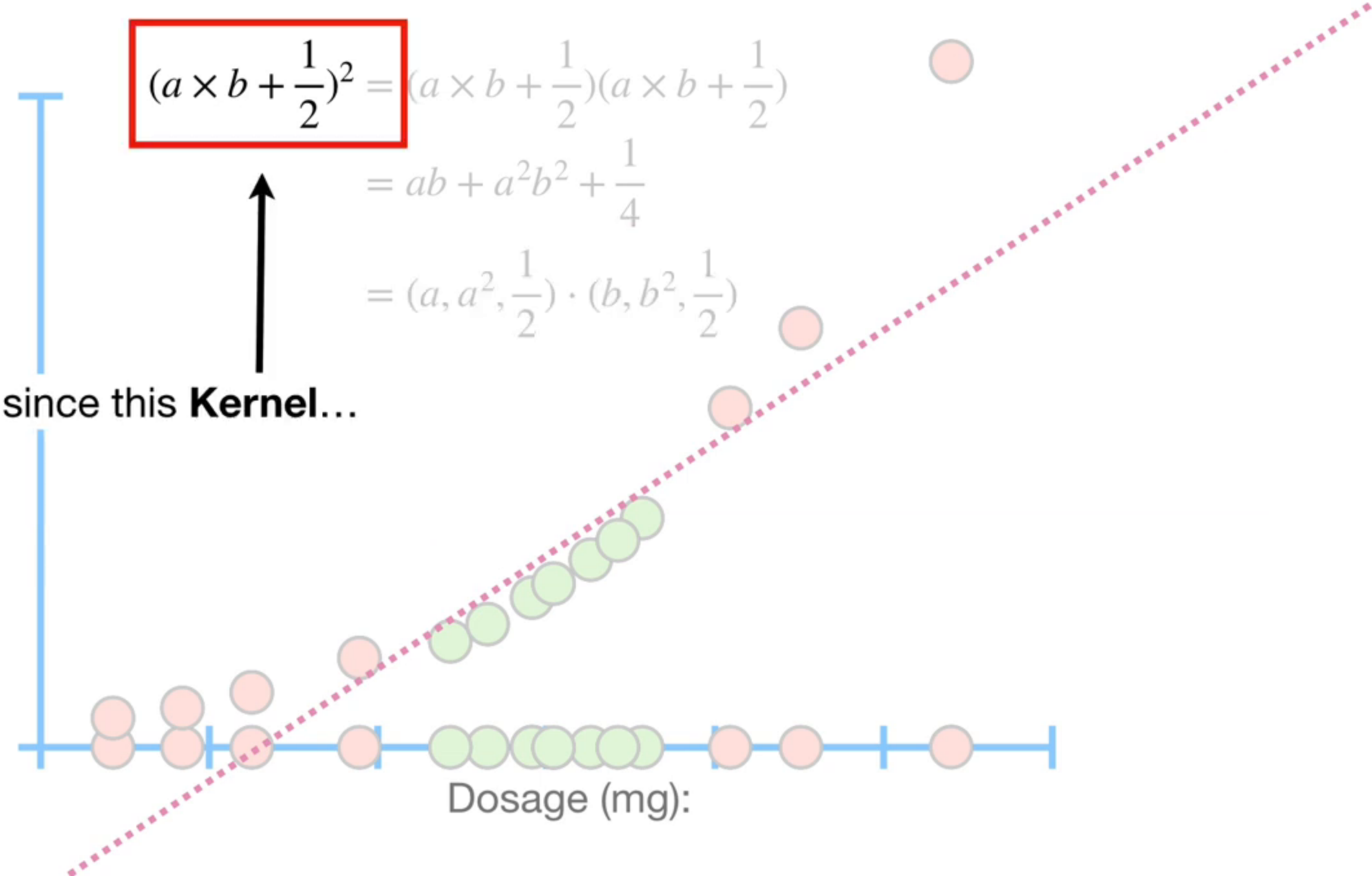


$$(a \times b + \frac{1}{2})^2 = (a \times b + \frac{1}{2})(a \times b + \frac{1}{2})$$

$$= ab + a^2b^2 + \frac{1}{4}$$

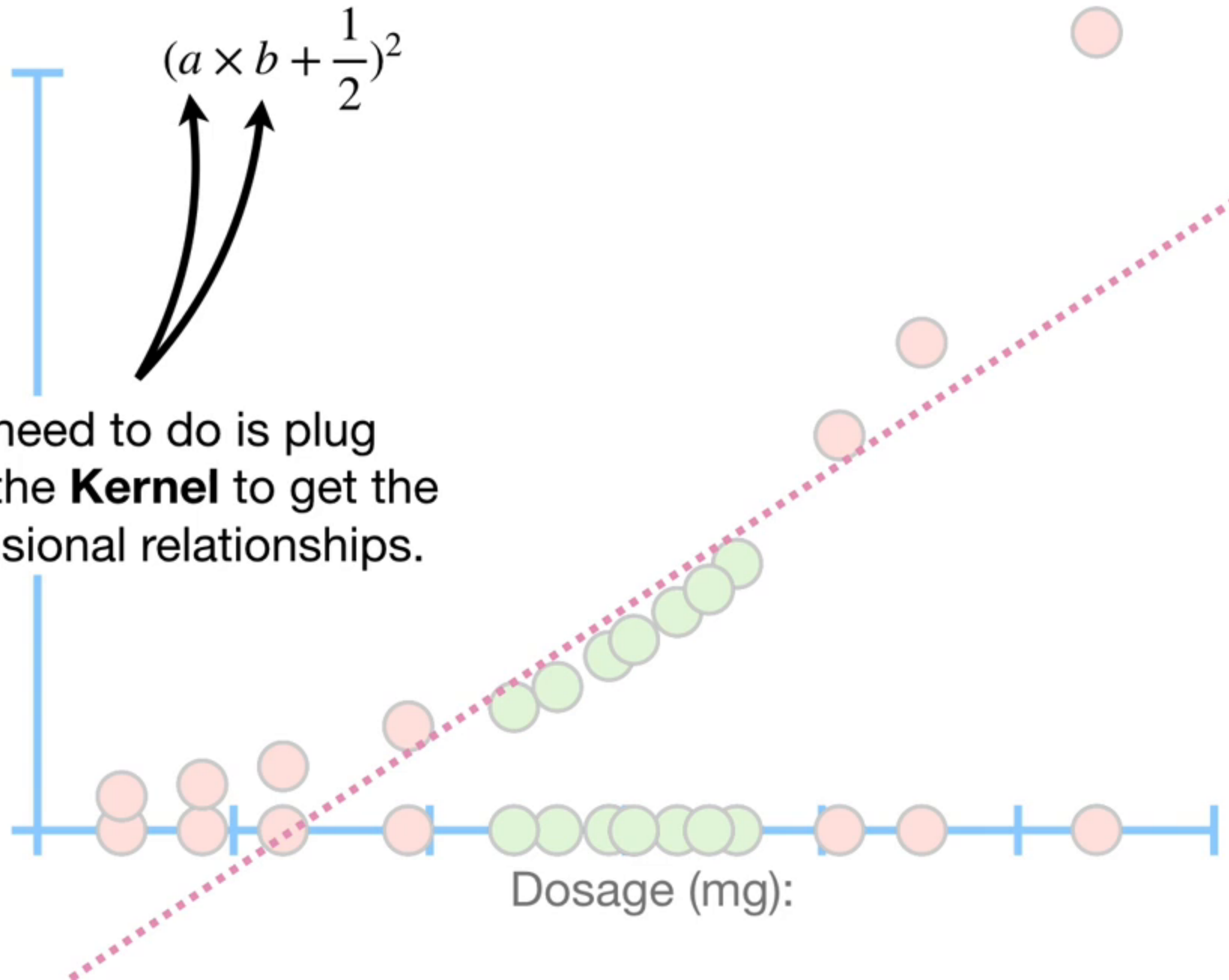
$$= (a, a^2, \frac{1}{2}) \cdot (b, b^2, \frac{1}{2})$$

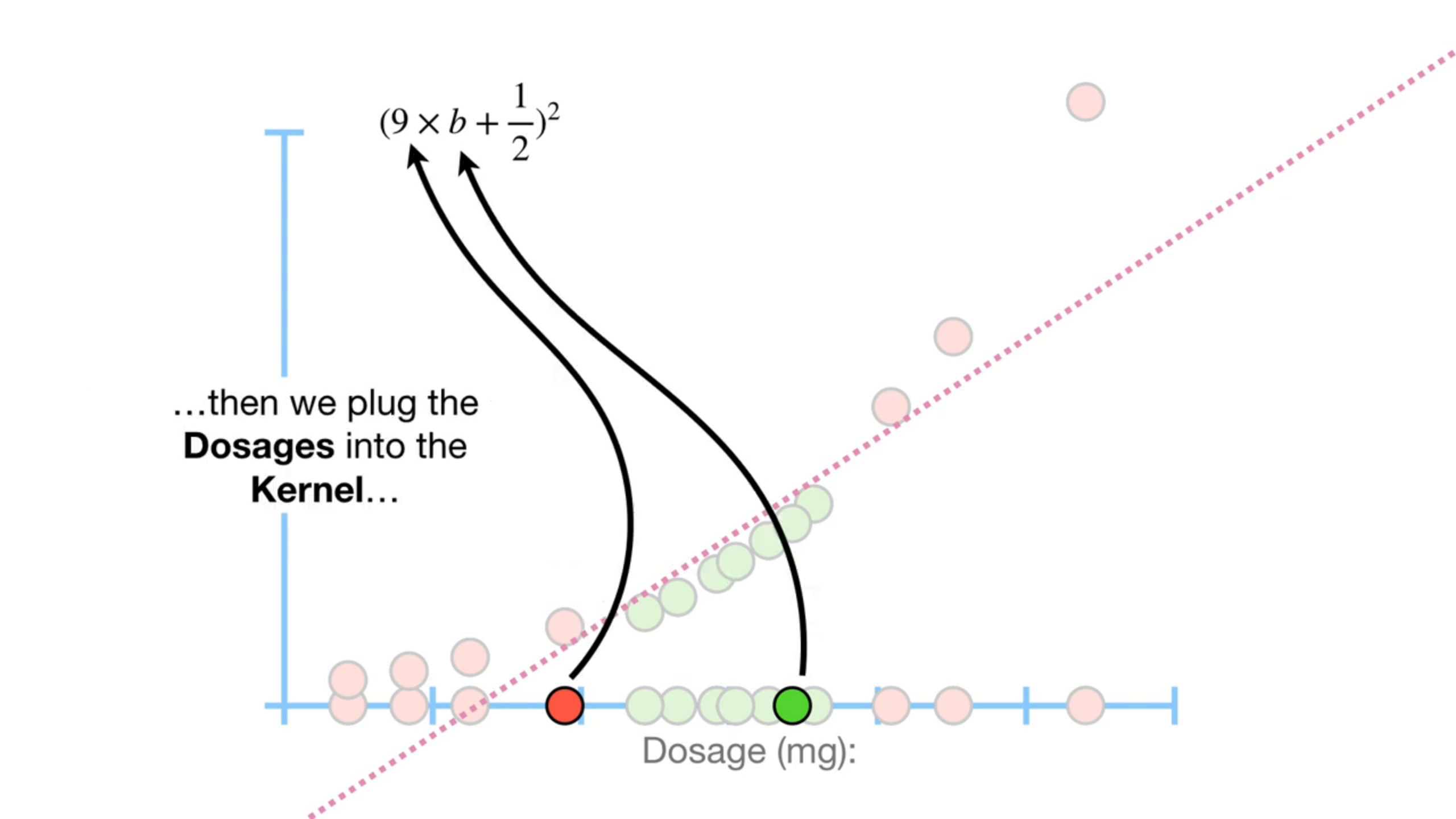
And since this **Kernel**…



$$(a \times b + \frac{1}{2})^2$$

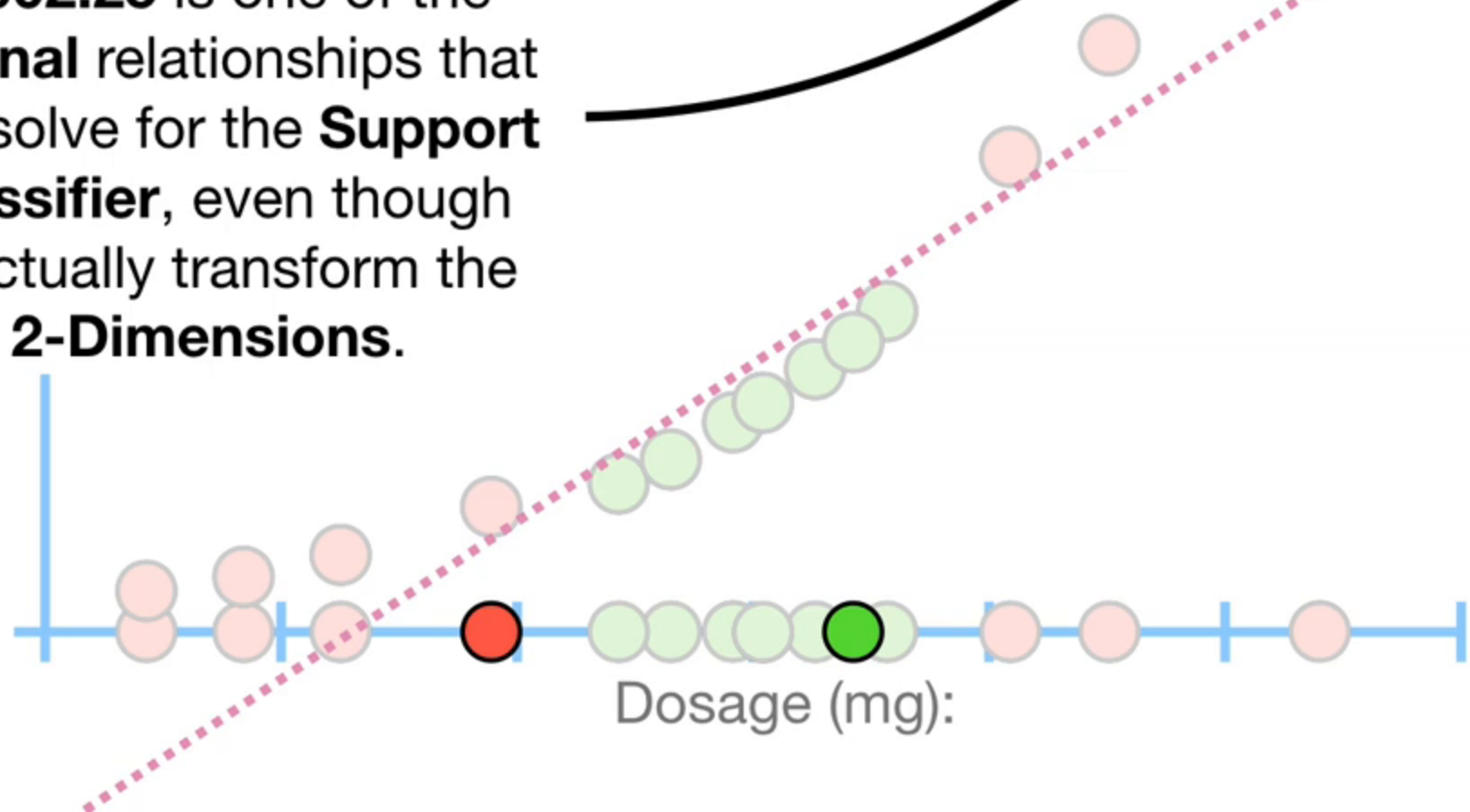
…all we need to do is plug values into the **Kernel** to get the high-dimensional relationships.




$(9 \times b + \frac{1}{2})^2$
…then we plug the **Dosages** into the **Kernel**…
Dosage (mg):

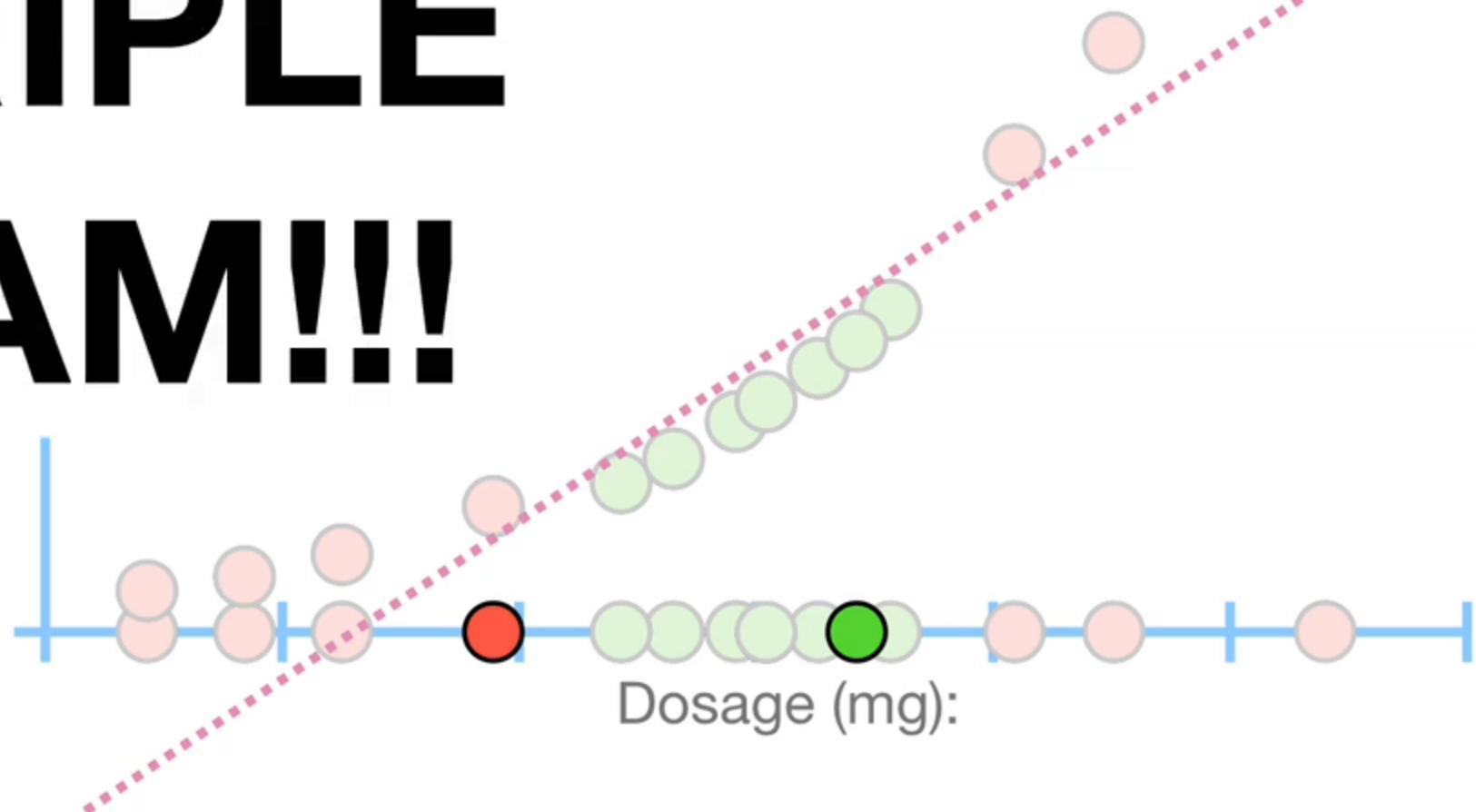
$$(9 \times 14 + \frac{1}{2})^2 = (126 + \frac{1}{2})^2 = 126.5^2 = 16{,}002.25$$

…and **16,002.25** is one of the **2-Dimensional** relationships that we need to solve for the **Support Vector Classifier**, even though we didn't actually transform the data to **2-Dimensions**.



$$(9 \times 14 + \frac{1}{2})^2 = (126 + \frac{1}{2})^2 = 126.5^2 = \boxed{16{,}002.25}$$

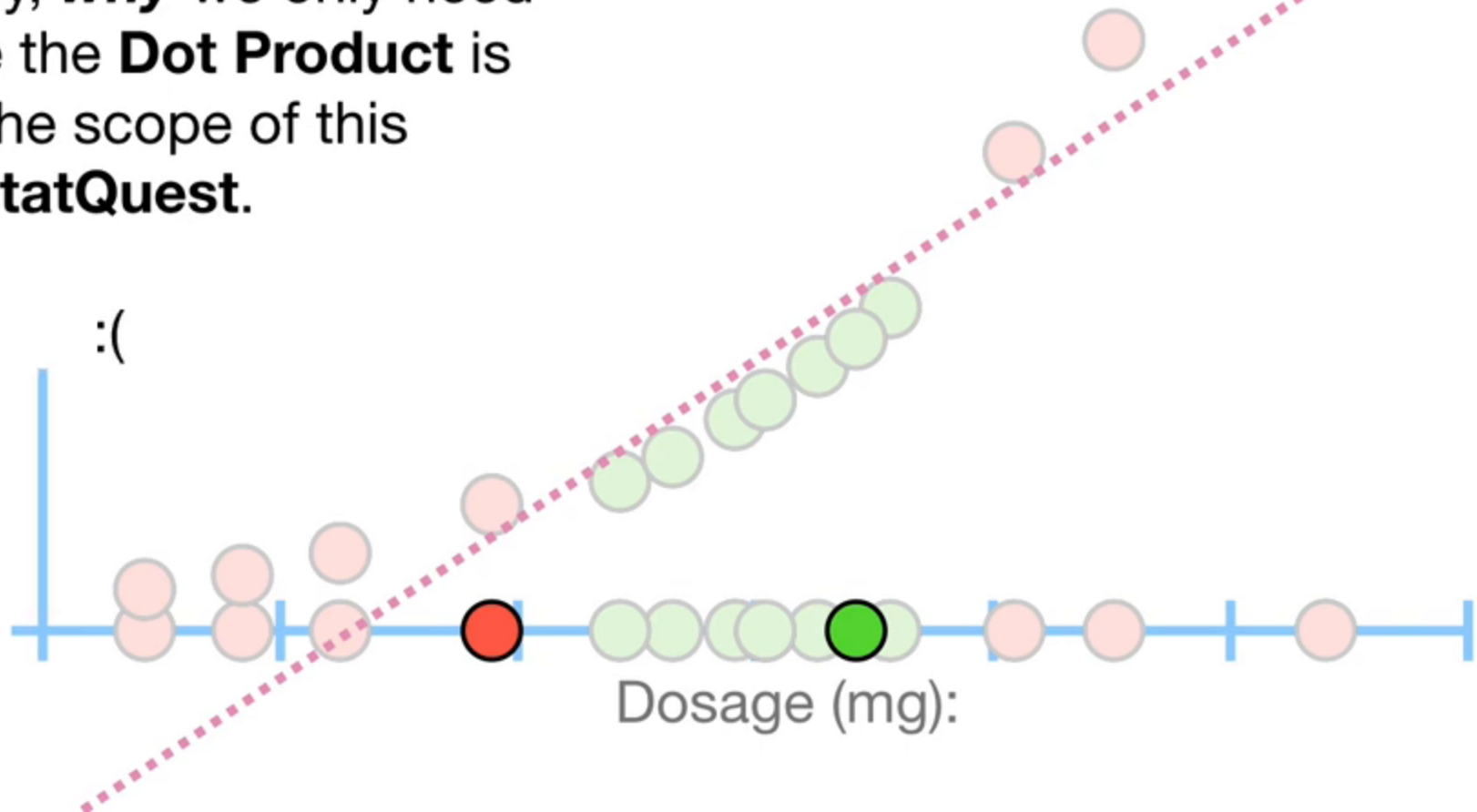
TRIPLE BAM!!!



$$(9 \times 14 + \frac{1}{2})^2 = (126 + \frac{1}{2})^2 = 126.5^2 = 16{,}002.25$$

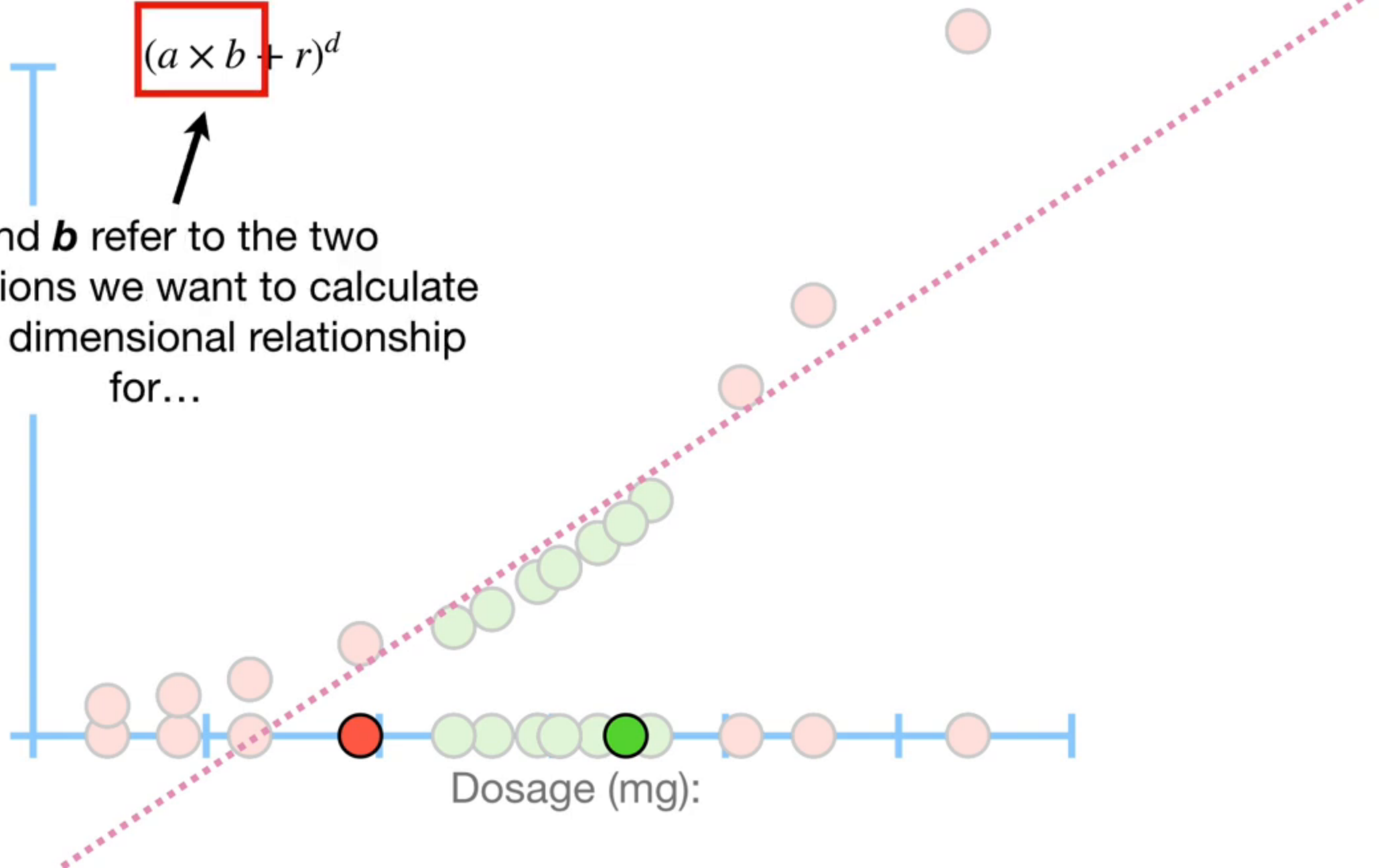
Unfortunately, ***why*** we only need to compute the **Dot Product** is out of the scope of this **StatQuest**.

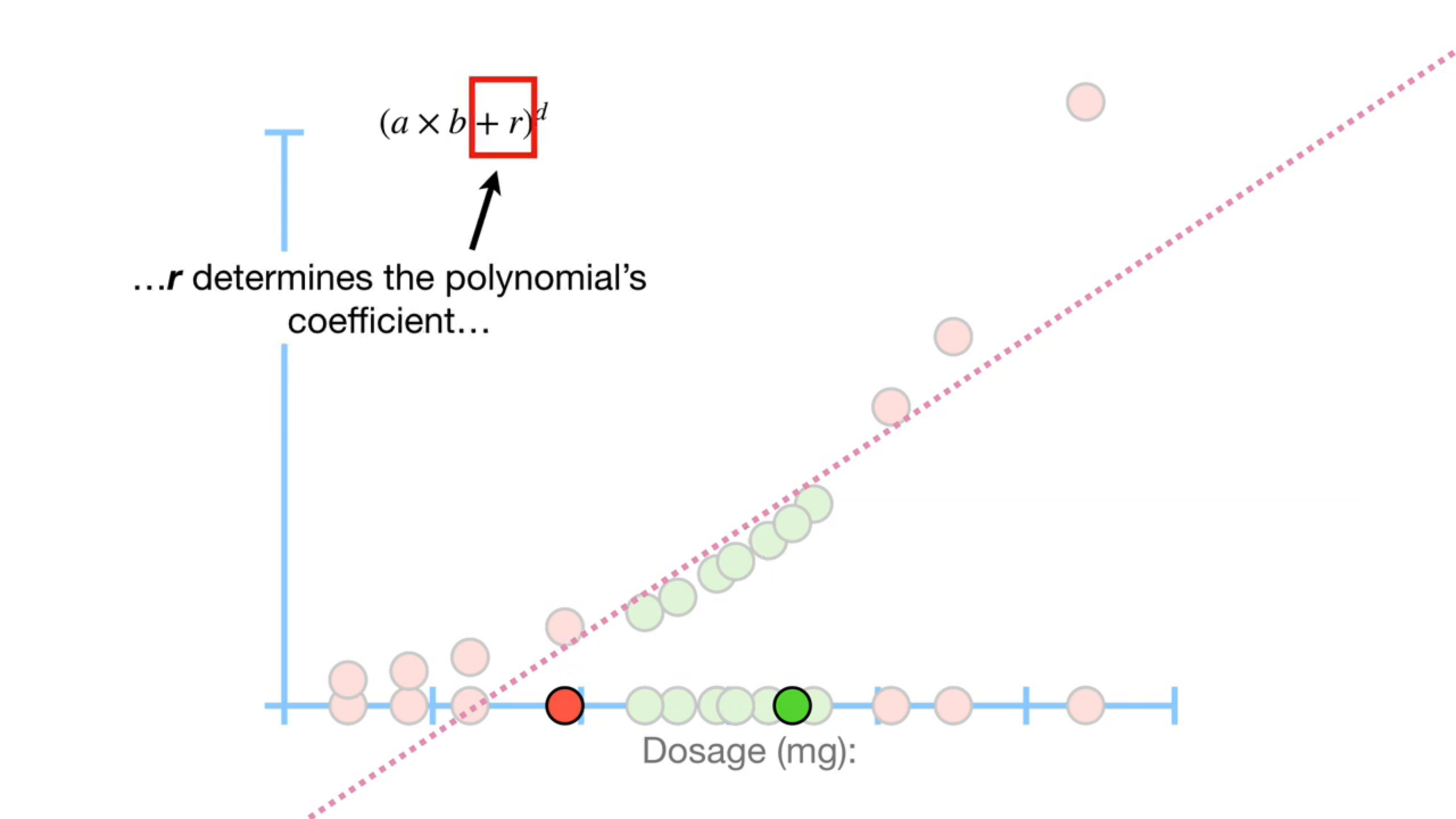
:(

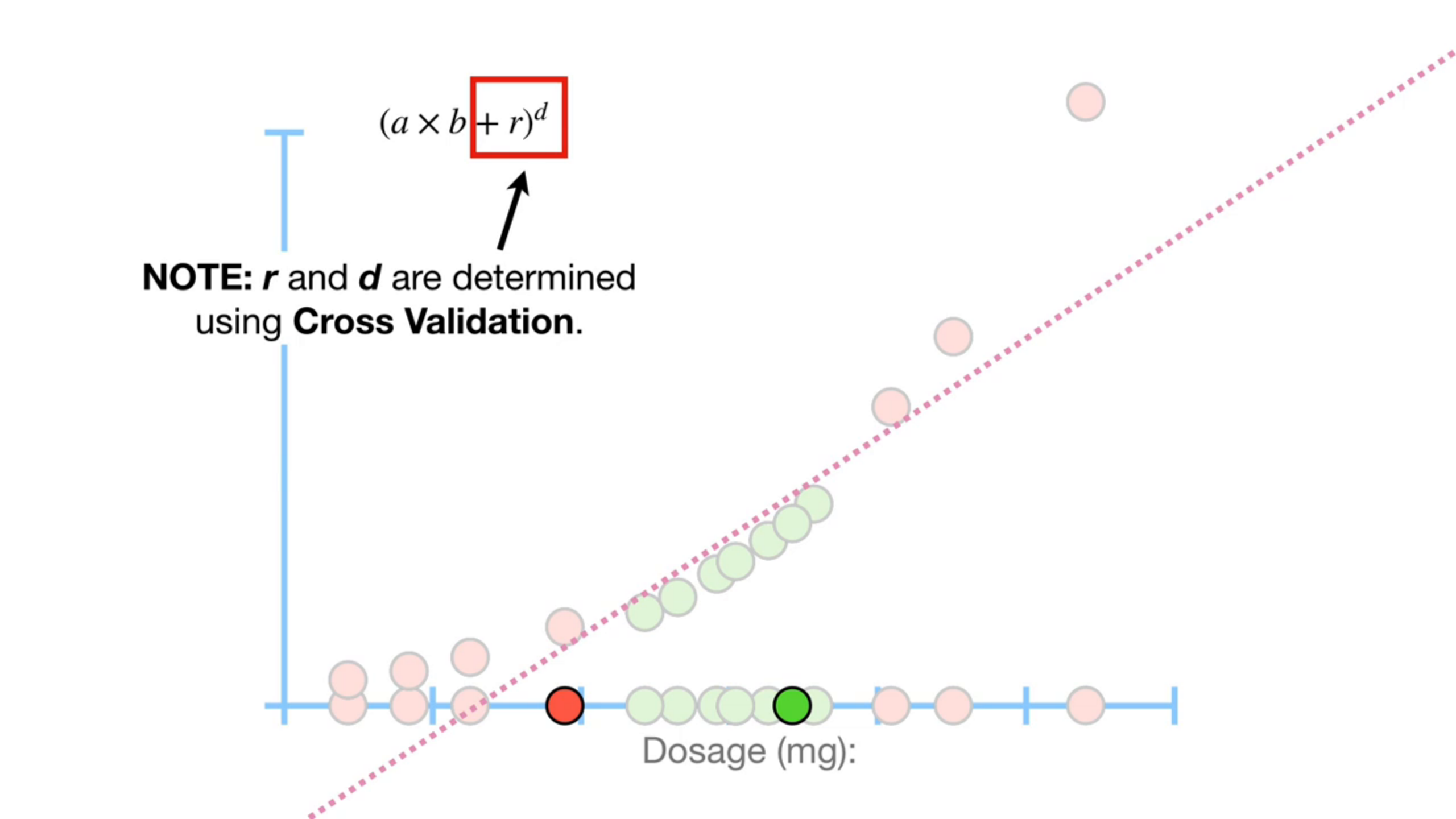


$$(a \times b + r)^d$$

a and ***b*** refer to the two observations we want to calculate the high dimensional relationship for…

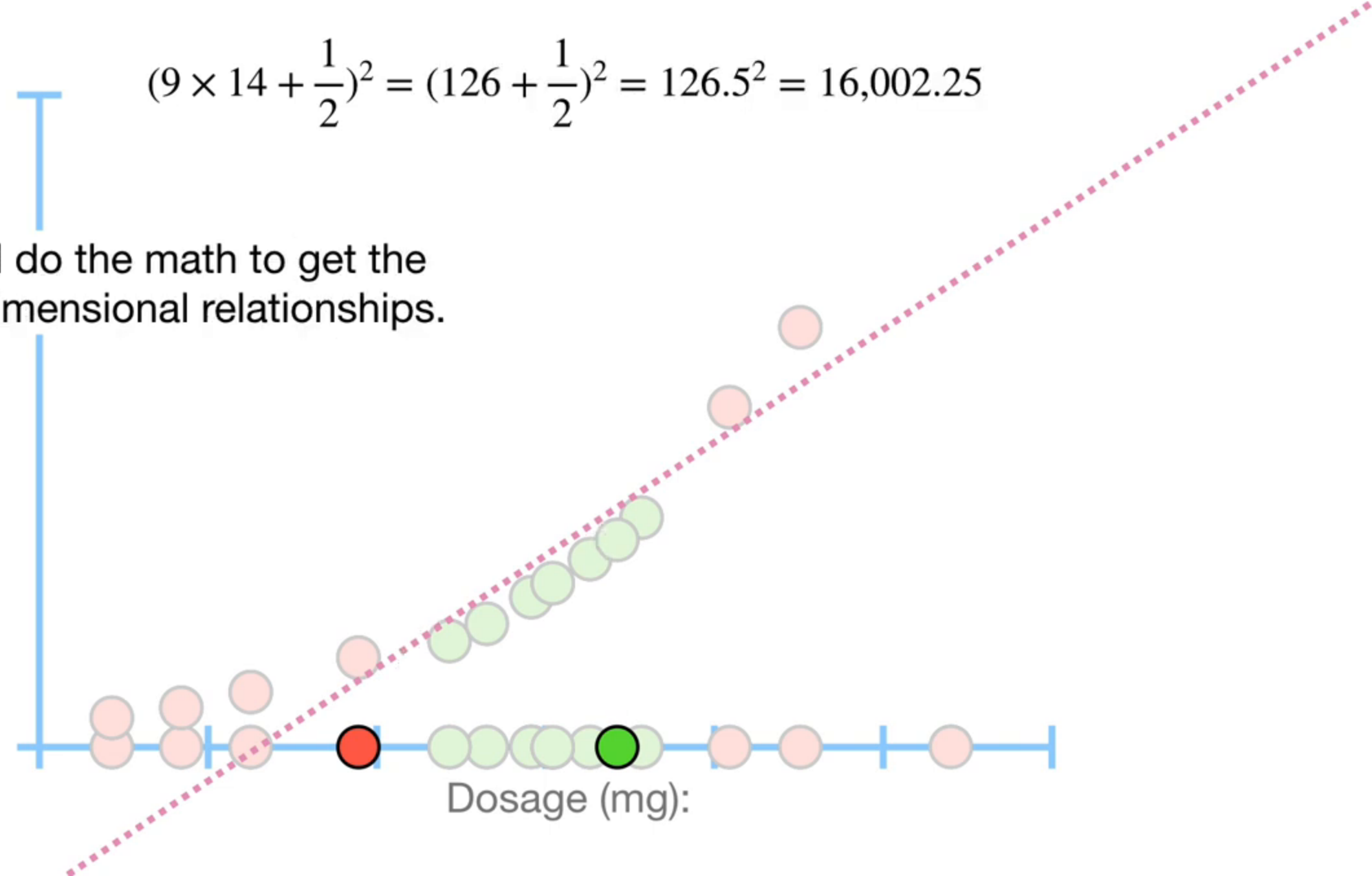



$(a \times b + r)^d$
…*r* determines the polynomial's coefficient…
Dosage (mg):


$(a \times b + r)^d$
NOTE: r and d are determined using Cross Validation.
Dosage (mg):

$$(9 \times 14 + \frac{1}{2})^2 = (126 + \frac{1}{2})^2 = 126.5^2 = 16{,}002.25$$

…and do the math to get the high-dimensional relationships.



Subscribe!!!

Support StatQuest!!! 

Subscribe!!!

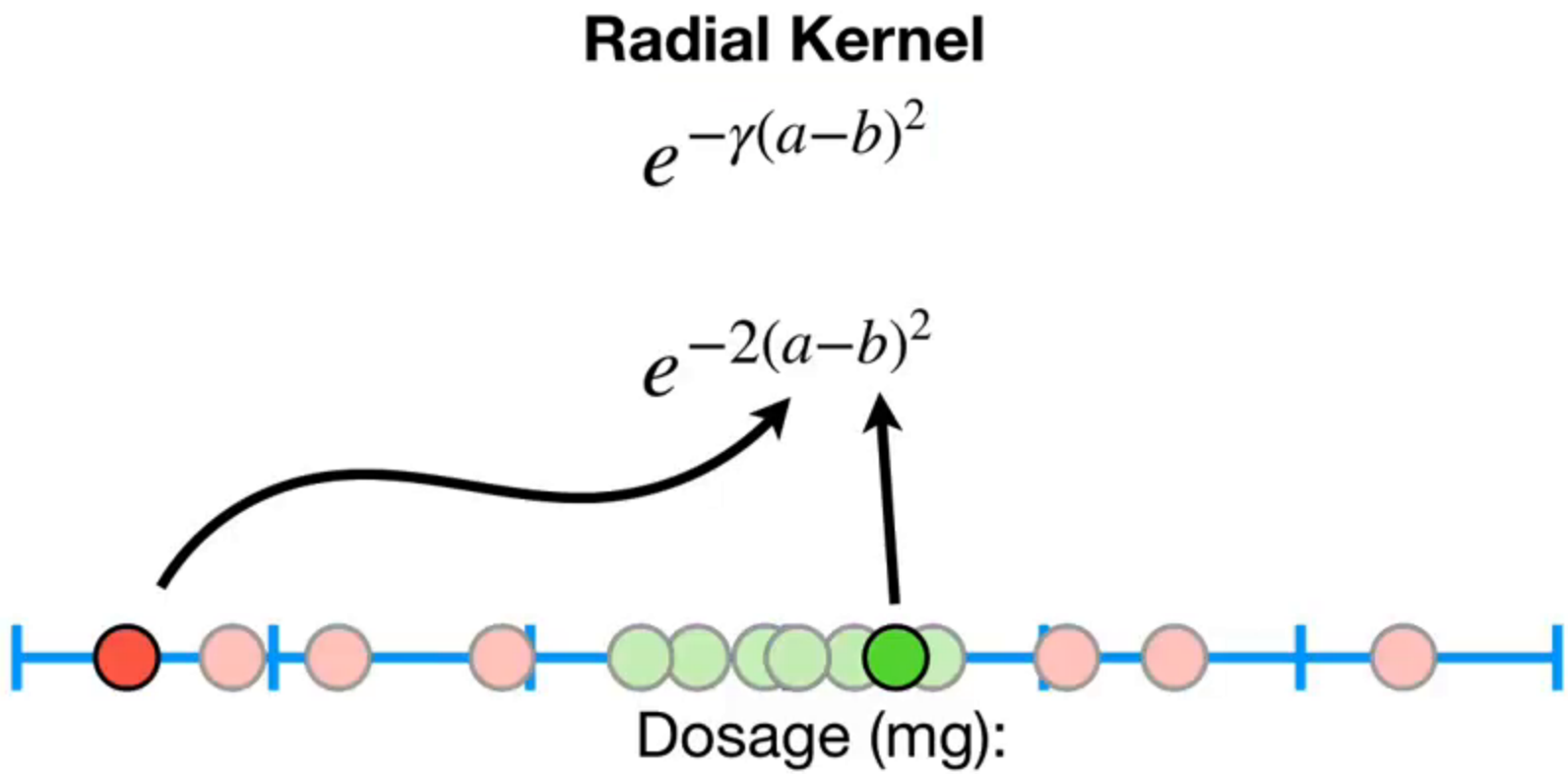
Support StatQuest!!! 

works in infinite

StatQuest!!!

Support Vector Machines
Part 3:
The Radial Kernel

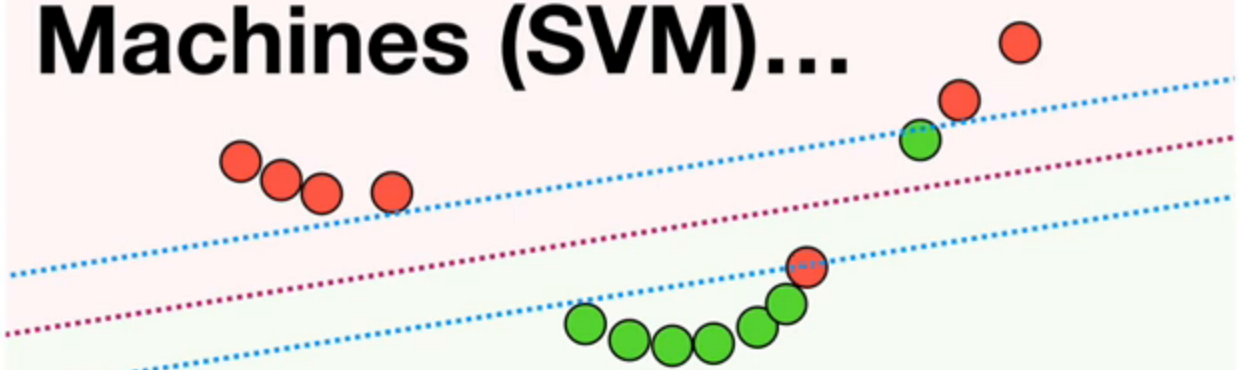
…how the **Radial Kernel** calculates high-dimensional relationships…



NOTE: This **StatQuest** assumes that you are already familiar with **Support Vector Machines** and the **Polynomial Kernel**. If not, check out the **'Quests**. The links are in the description below.

NOTE: This **StatQuest** assumes that you are already familiar with **Support Vector Machines** and the **Polynomial Kernel**. If not, check out the **'Quests**. The links are in the description below.

Support Vector Machines (SVM)…



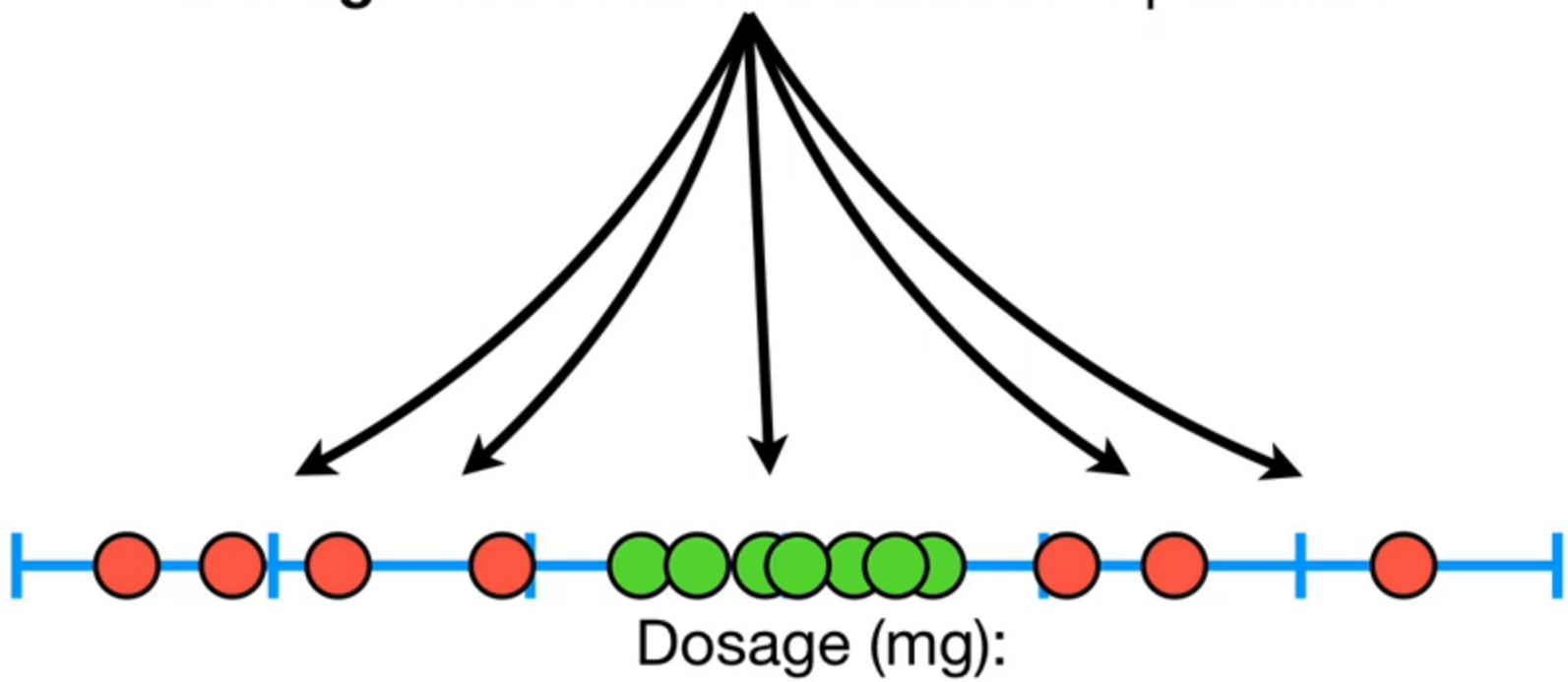
…Clearly Explained!!!

Support Vector Machines Part 2:

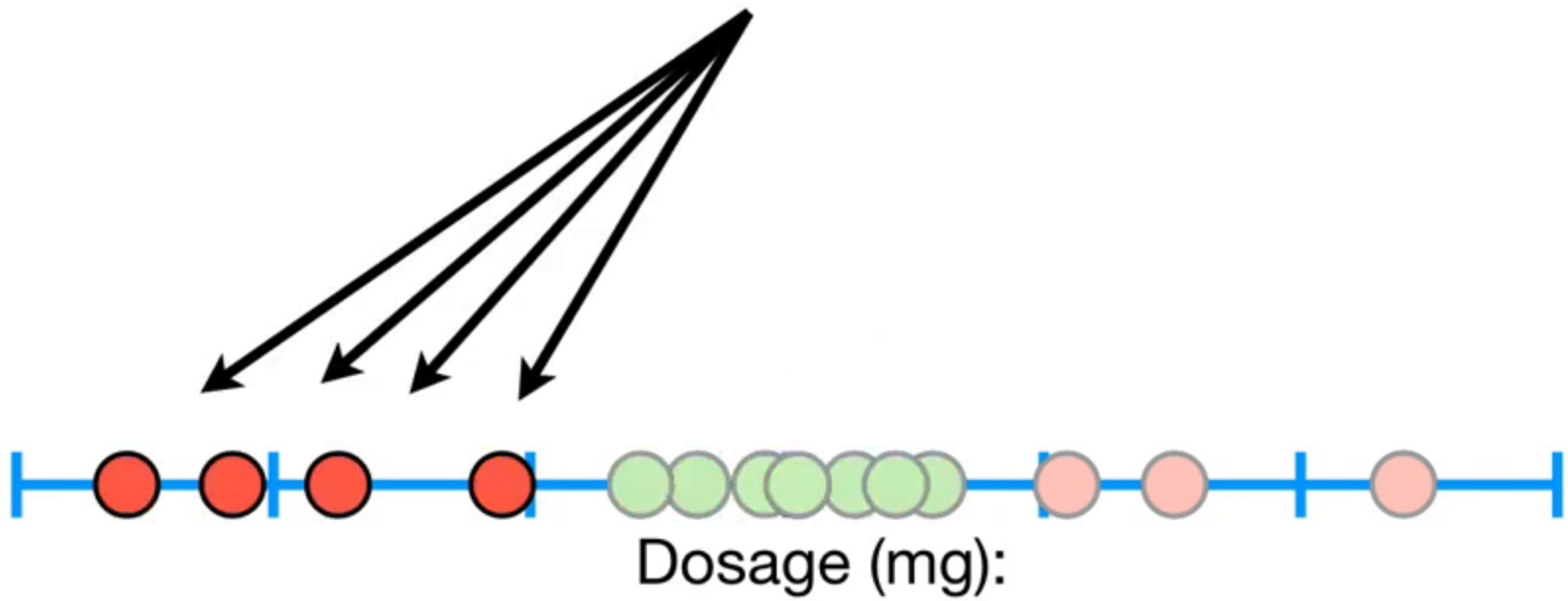
$$(a \times b + r)^d$$

…The Polynomial Kernel!!!

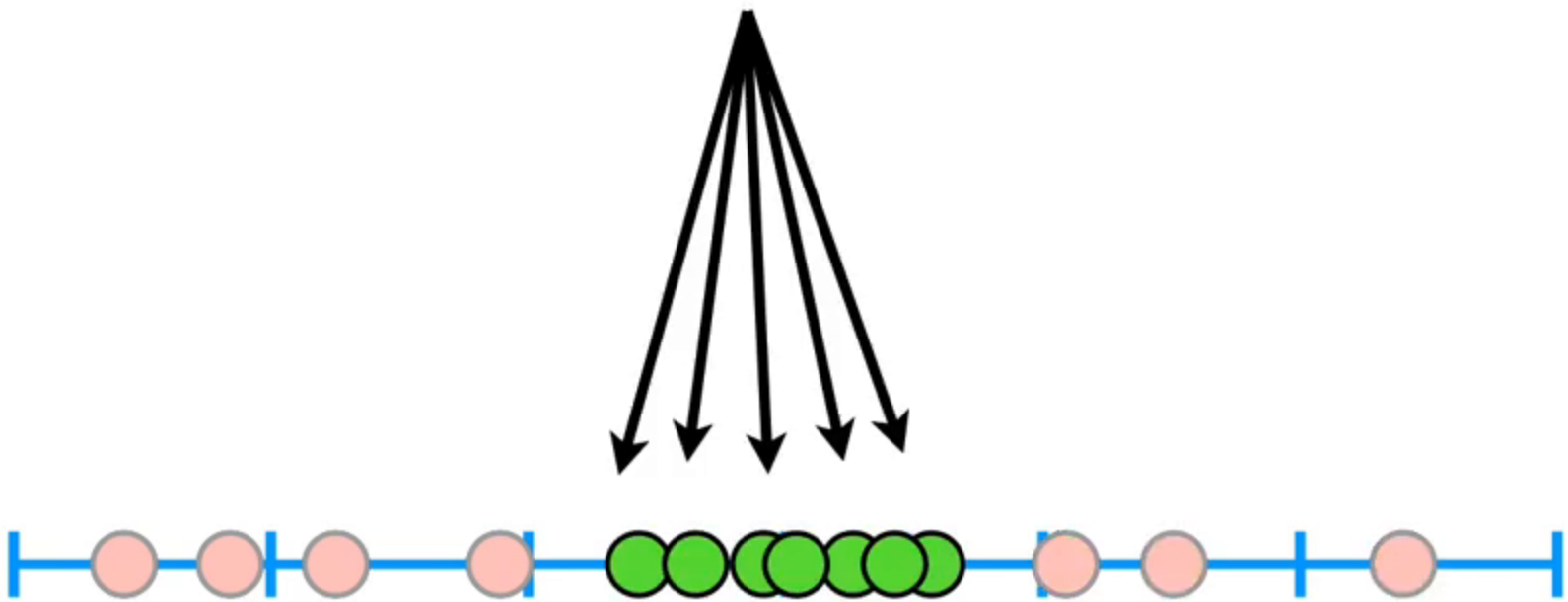
In the **StatQuest** on **Support Vector Machines**, we had a **Training Dataset** based on **Drug Dosages** measured in a bunch of patients.



In other words, the drug doesn't work if the dosage is too small…

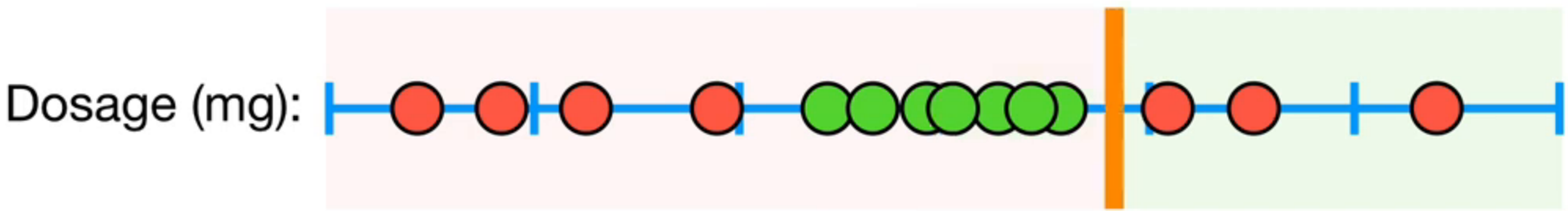


It only works when the
dosage is just right.



Dosage (mg):

Because this **Training Dataset** had so much overlap, we were unable to find a satisfying **Support Vector Classifier** to separate the patients that were cured from the patients that were not cured.

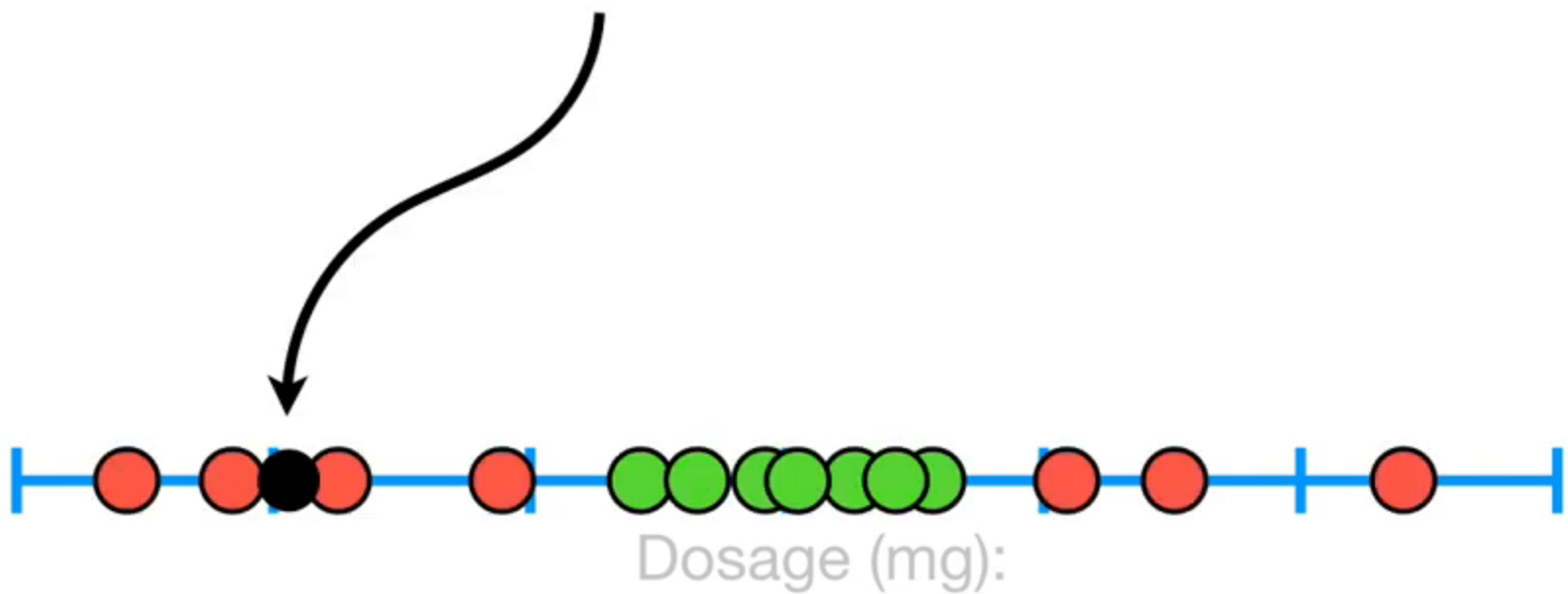


One way to deal with overlapping data is to use a **Support Vector Machine** with a **Radial Kernel**

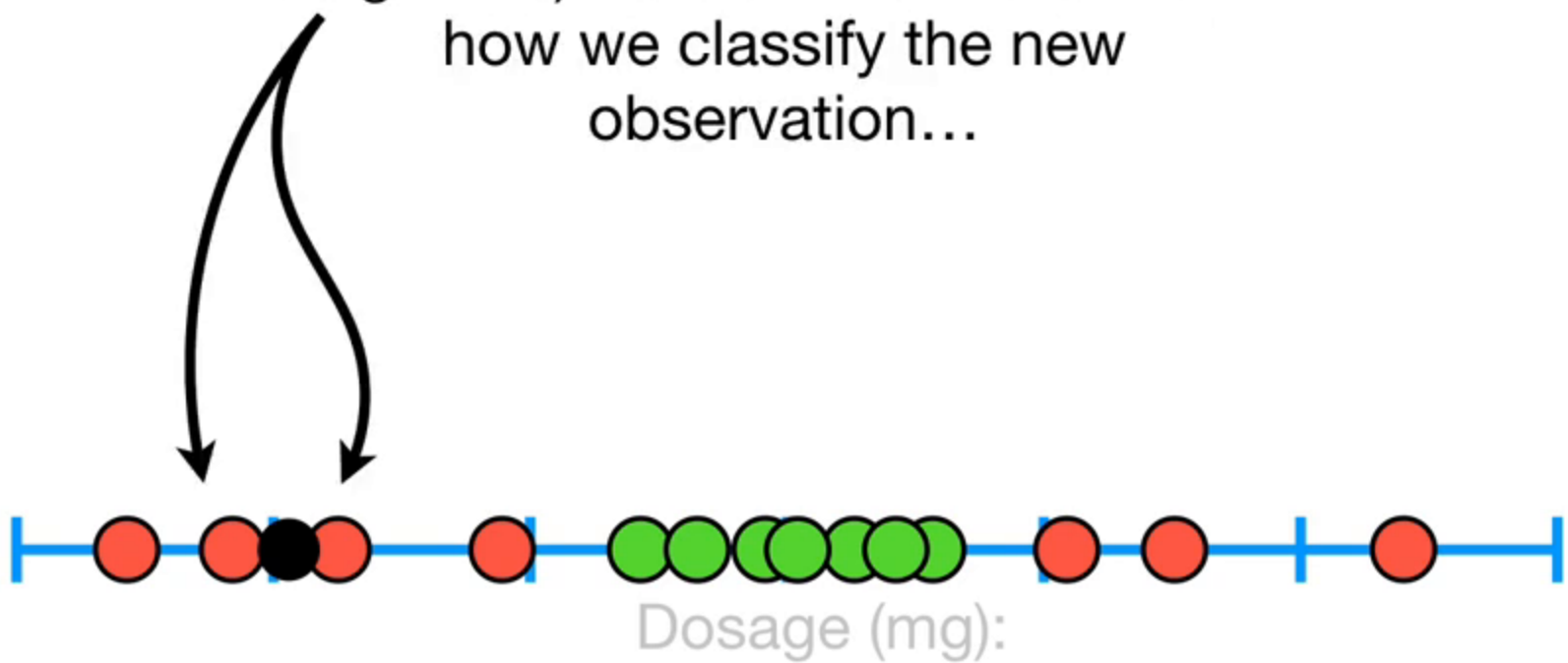
$$e^{-\gamma(a-b)^2}$$

Because the **Radial Kernel** finds **Support Vector Classifiers** in infinite dimensions, it's not possible to visualize what it does.

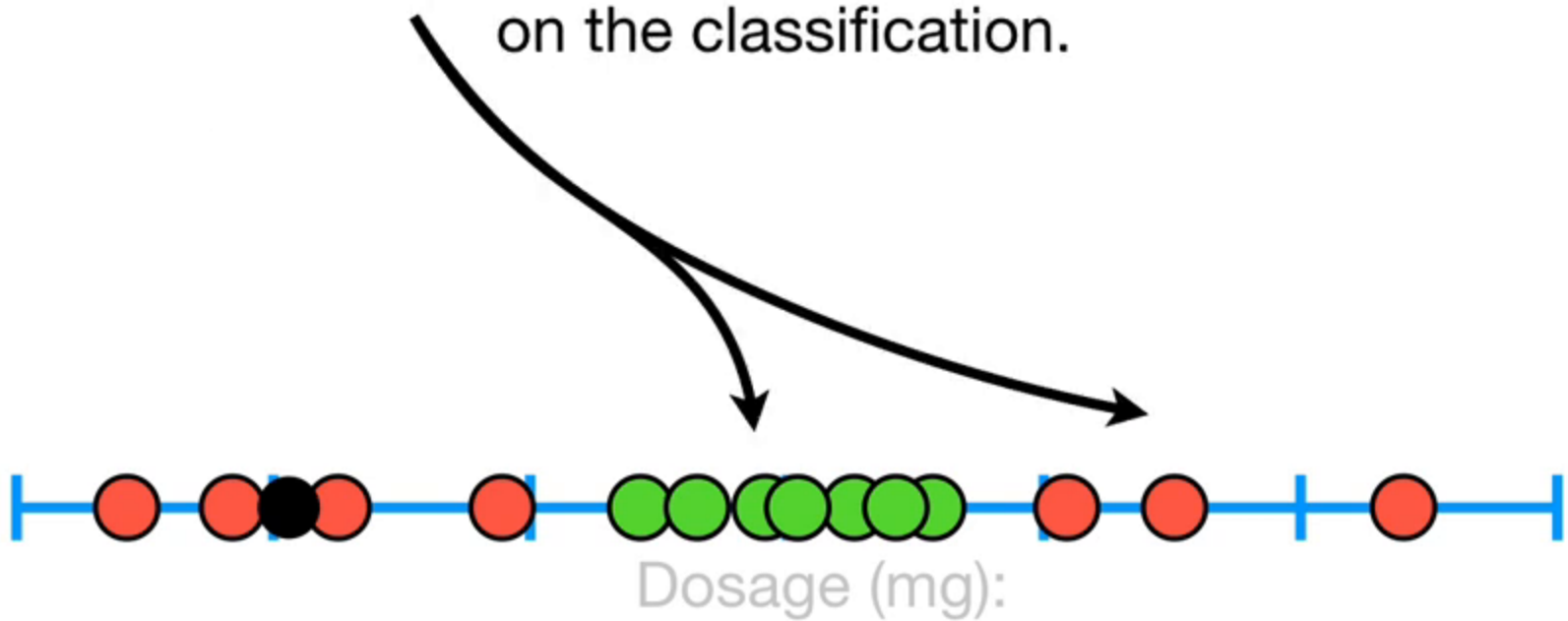
However, when using it on a new observation like this…



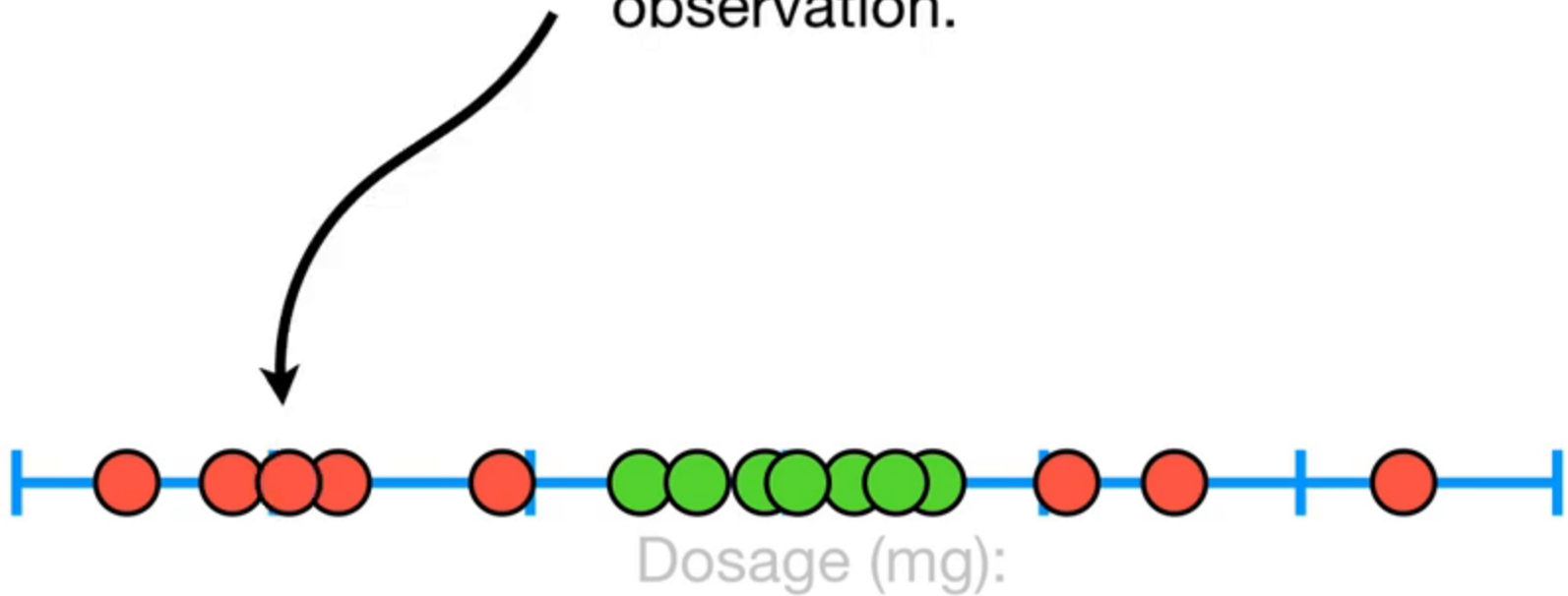
In other words, the closest observations (aka the nearest neighbors) have a lot of influence on how we classify the new observation…



…and observations that are further away have relatively little influence on the classification.



…the **Radial Kernel** uses their classification for the new observation.



Now let's talk about how the **Radial Kernel** determines how much influence each observation in the **Training Dataset** has on classifying new observations.

$$e^{-\gamma(a-b)^2}$$

Dosage (mg):

Just like with the **Polynomial Kernel**, ***a*** and ***b*** refer to two different **Dosage** measurements.

$$e^{-\gamma(a-b)^2}$$

Dosage (mg):

The difference between the measurements is then squared, giving us the squared distance between the two observations.

$$e^{-\gamma(a-b)^2}$$

Dosage (mg):

γ (gamma), which is determined by **Cross Validation**, scales the squared distance, and thus, it scales the influence.

$$e^{-\gamma(a-b)^2}$$

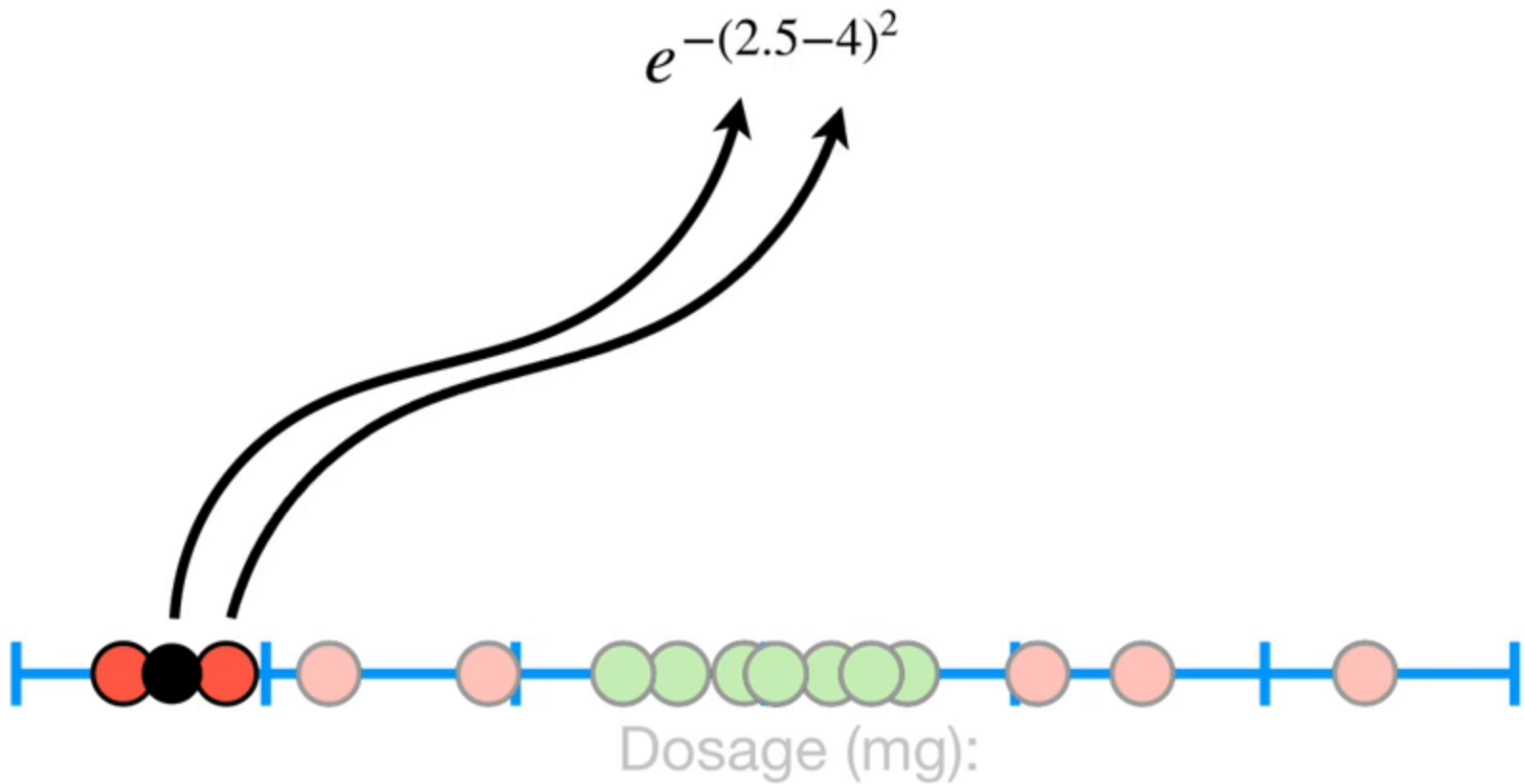
Dosage (mg):

For example, if we set $\gamma = 1$…

$$e^{-\gamma(a-b)^2}$$

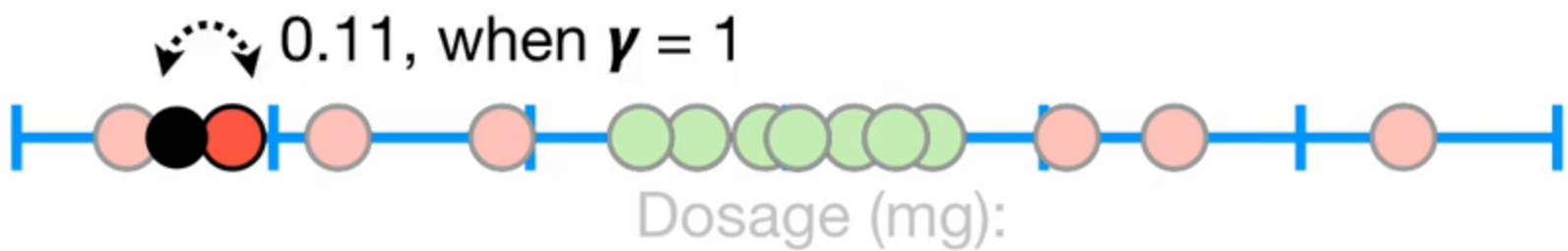
Dosage (mg):

…and plug in the **Dosages** from two observations that are relatively close to each other…

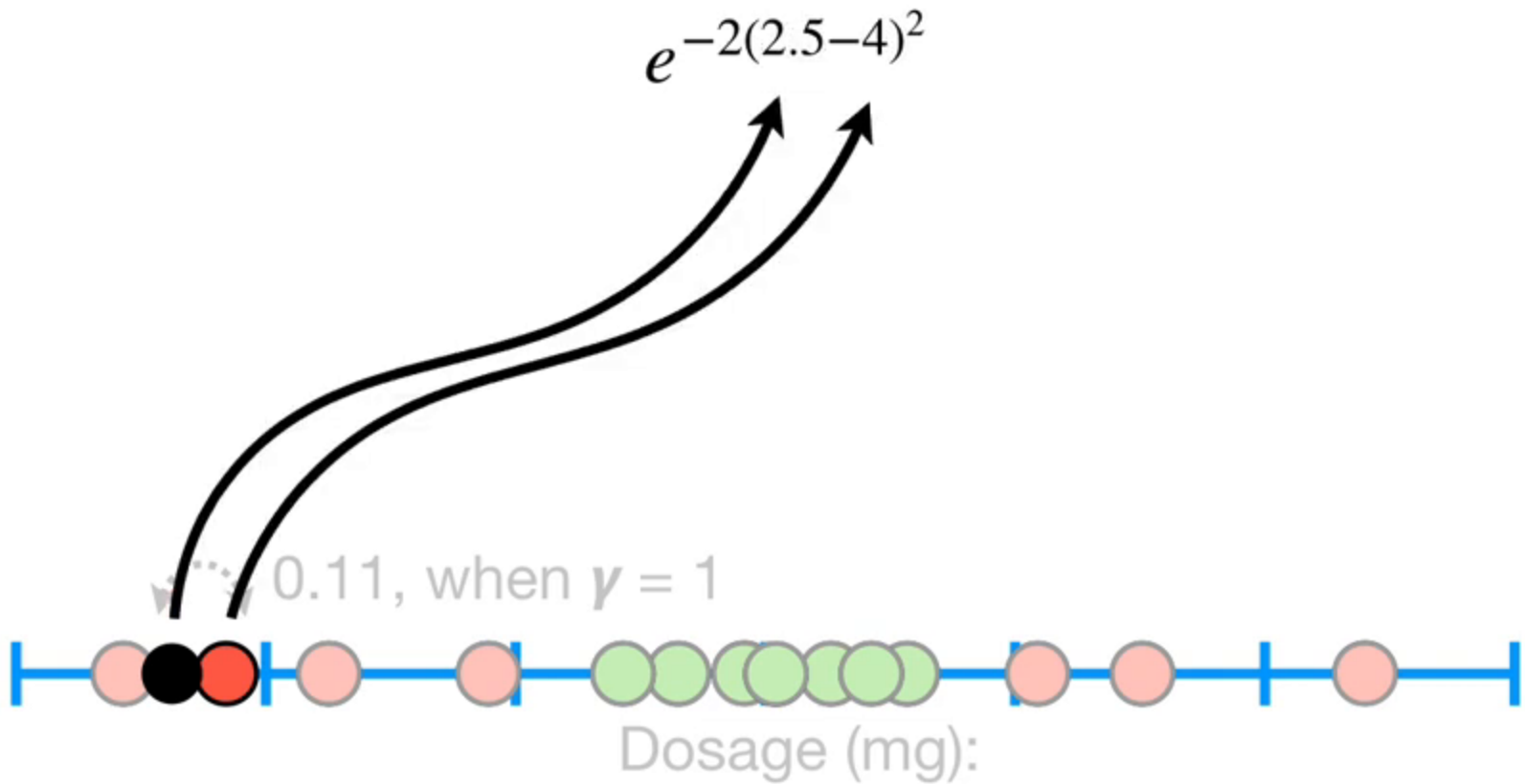


So let's put **0.11** here.

$$e^{-(2.5-4)^2} = e^{-(-1.5)^2} = e^{-2.25} = 0.11$$



…and plug in the same two **Dosages** as before…



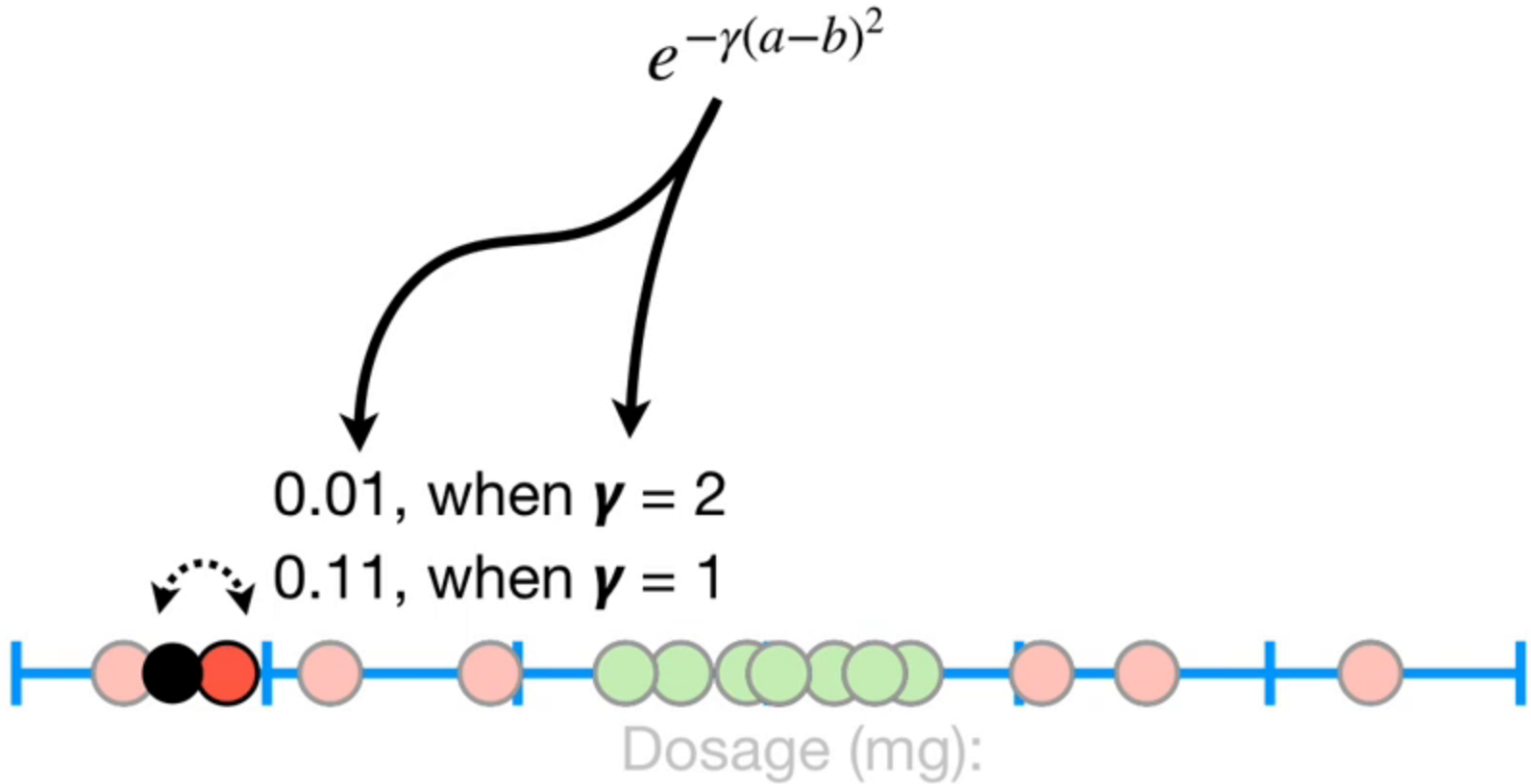
When $\gamma = 2$, we get **0.01**…

$$e^{-2(2.5-4)^2} = e^{-2(-1.5)^2} = e^{-2\times 2.25} = 0.01$$

0.11, when $\gamma = 1$

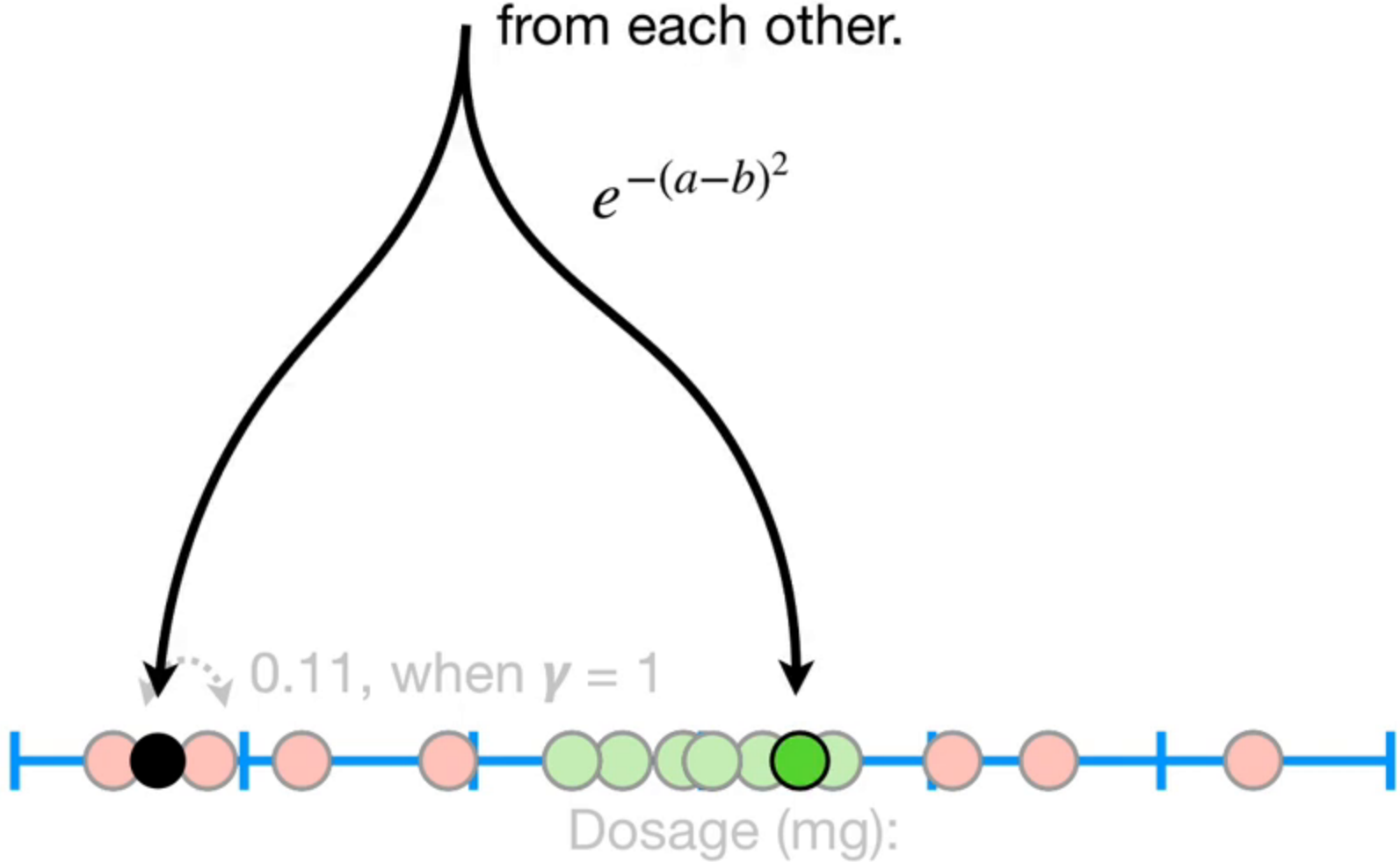
Dosage (mg):

So we see that by scaling the distance, γ scales the amount of influence two points have on each other.



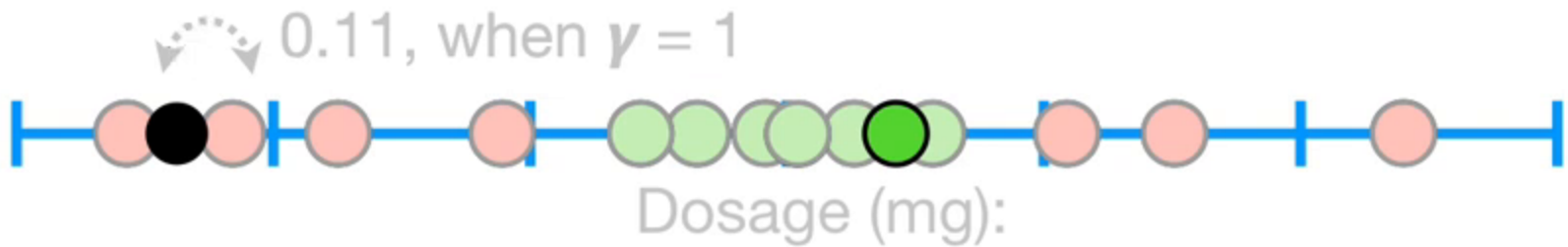
…and determine how much influence two observations have when they relatively far from each other.

$$e^{-(a-b)^2}$$



…do the math…

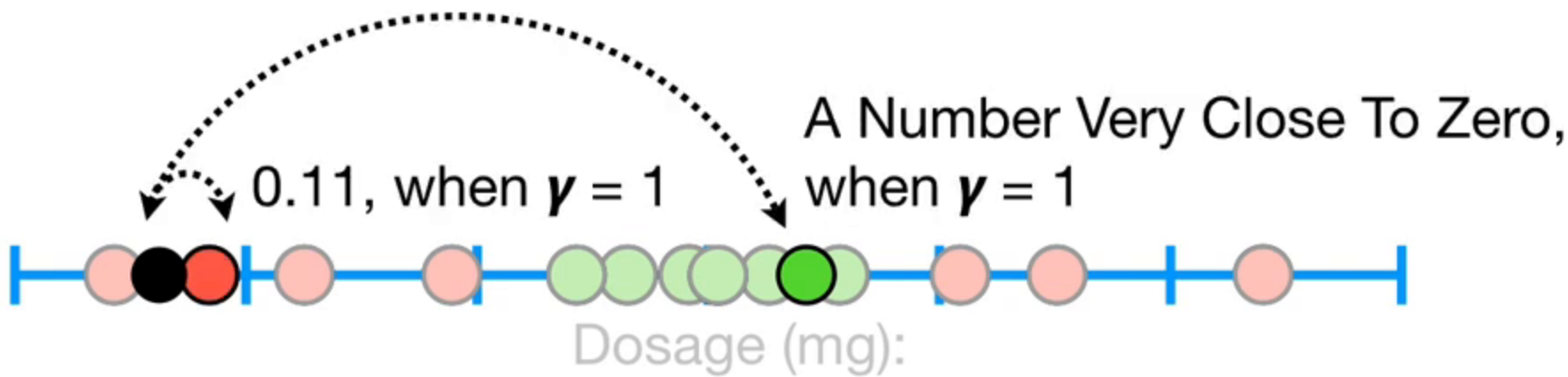
$$e^{-(2.5-16)^2} = e^{-(-13.5)^2} = e^{-182.25}$$



Thus, the further two observations are from each other, the less influence they have on each other.

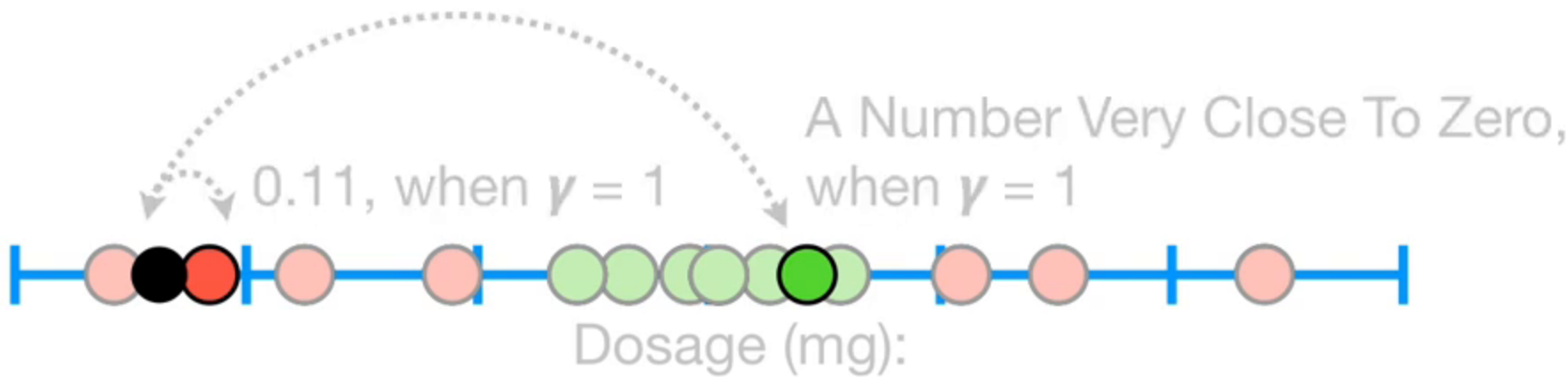
$$e^{-(2.5-16)^2} = e^{-(-13.5)^2} = e^{-182.25}$$

= A Number Very Close to Zero.



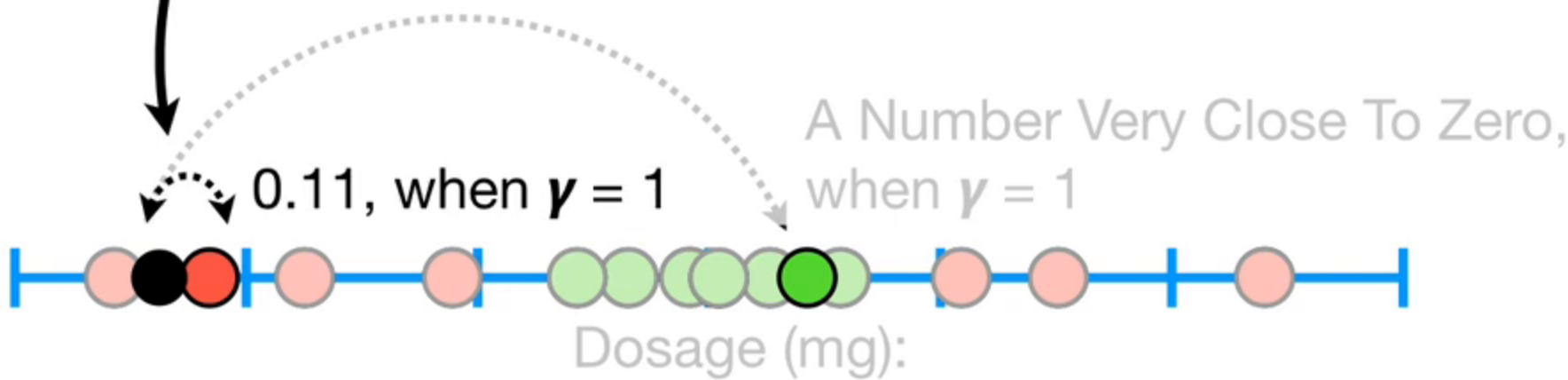
NOTE: Just like with the **Polynomial Kernel**, when we plug values into the **Radial Kernel**, we get the high-dimensional relationship.

$e^{-\gamma(a-b)^2}$ = high-dimensional relationship



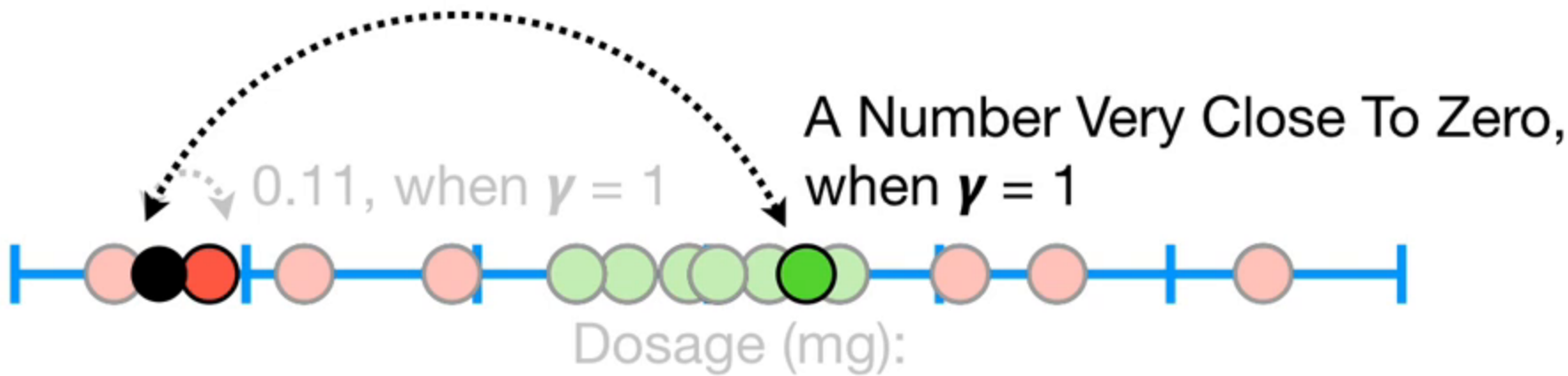
Thus, **0.11** is the high-dimensional relationship between these two observations that are relatively close to each other…

$e^{-\gamma(a-b)^2}$ = high-dimensional relationship



…and **A Number Very Close to Zero** is the high-dimensional relationship between these two observations that are relatively far from each other.

$e^{-\gamma(a-b)^2}$ = high-dimensional relationship



Now, before we move on, I want to simplify the **Training Dataset** to just two observations…

$$(a \times b + r)^d$$

…and use the **Polynomial Kernel** to give us intuition into how the **Radial Kernel** works in **Infinite-Dimensions**.

$$e^{-\gamma(a-b)^2}$$

Dosage (mg):

$$(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$$

When ***r* = 0**, the **Polynomial Kernel** simplifies to a single term…

…and that gives us a **Dot Product** with a single coordinate.

Dosage (mg):

When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 2$ we get... $a^2 b^2 = (a^2) \cdot (b^2)$

...which is equal to the **Dot Product** of a^2 and b^2.

Dosage (mg):

When $r = 0$… $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 2$ we get… $a^2 b^2 = (a^2) \cdot (b^2)$

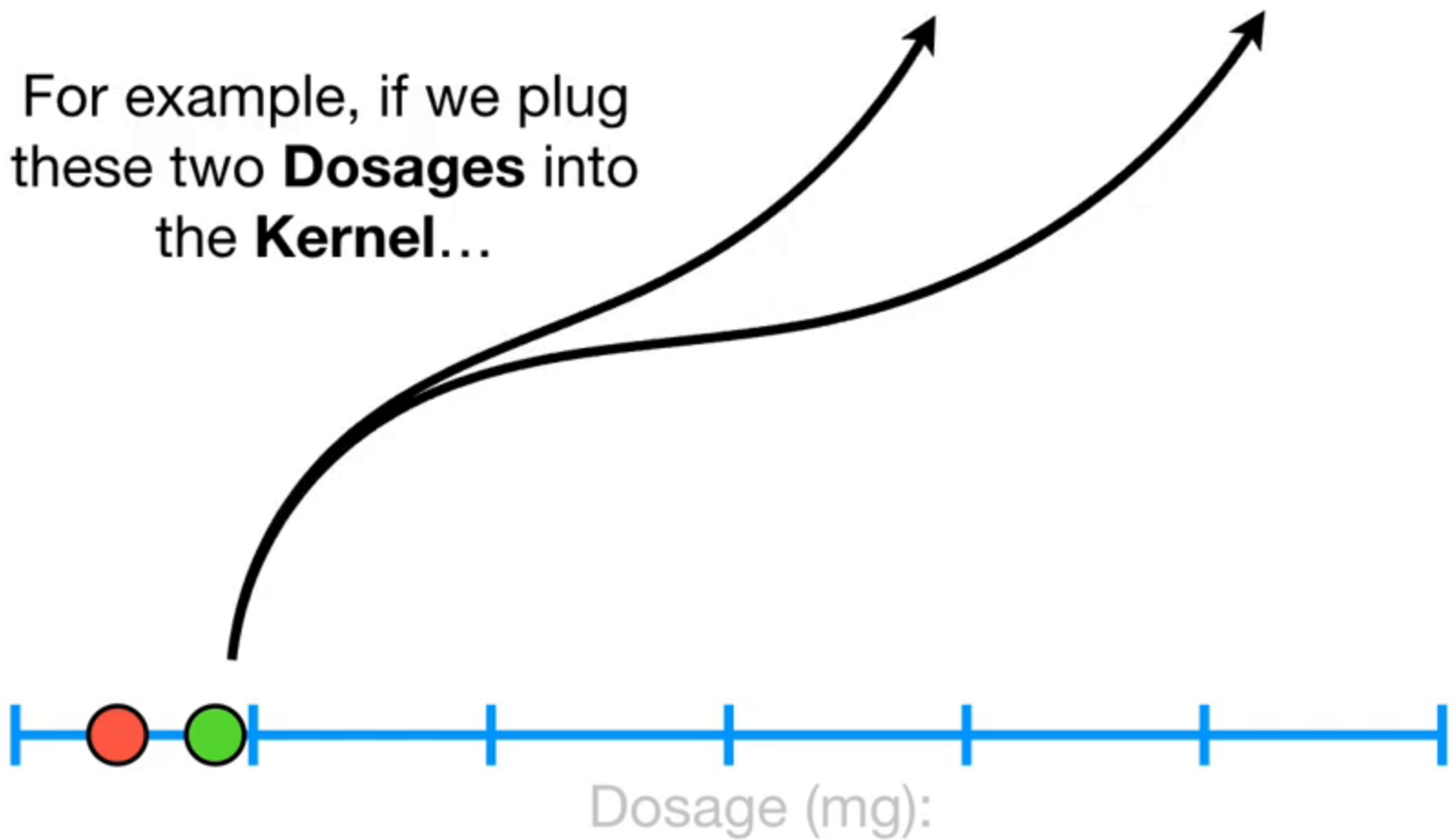
…the new coordinate is just the square of the original measurement on the original axis.

Dosage (mg):

When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 2$ we get... $2.5^2 \times 4^2 = (2.5^2) \cdot (4^2)$

For example, if we plug these two **Dosages** into the **Kernel**...



When ***r* = 0**… $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

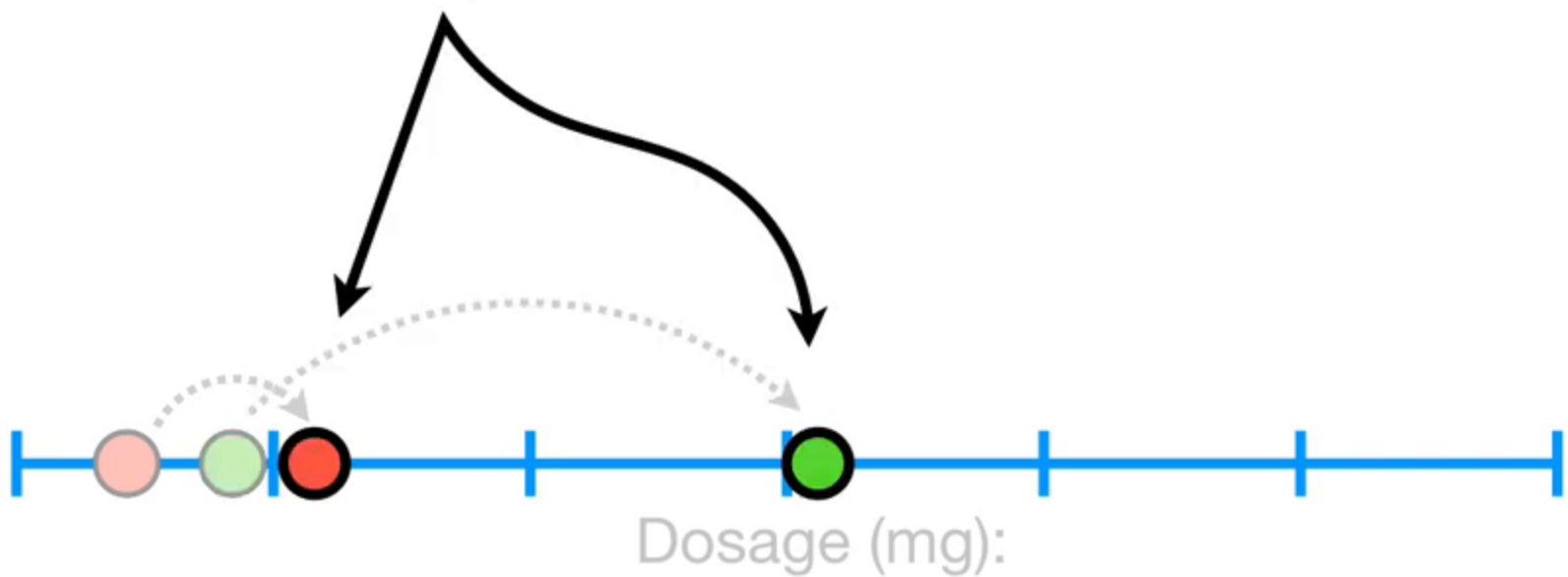
When ***d* = 2** we get… $2.5^2 \times 4^2 = (2.5^2) \cdot (4^2)$

…and that means the new ***x*-axis** coordinate for this point is **2.5^2 = 6.25**…

When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 2$ we get... $a^2 b^2 = (a^2) \cdot (b^2)$

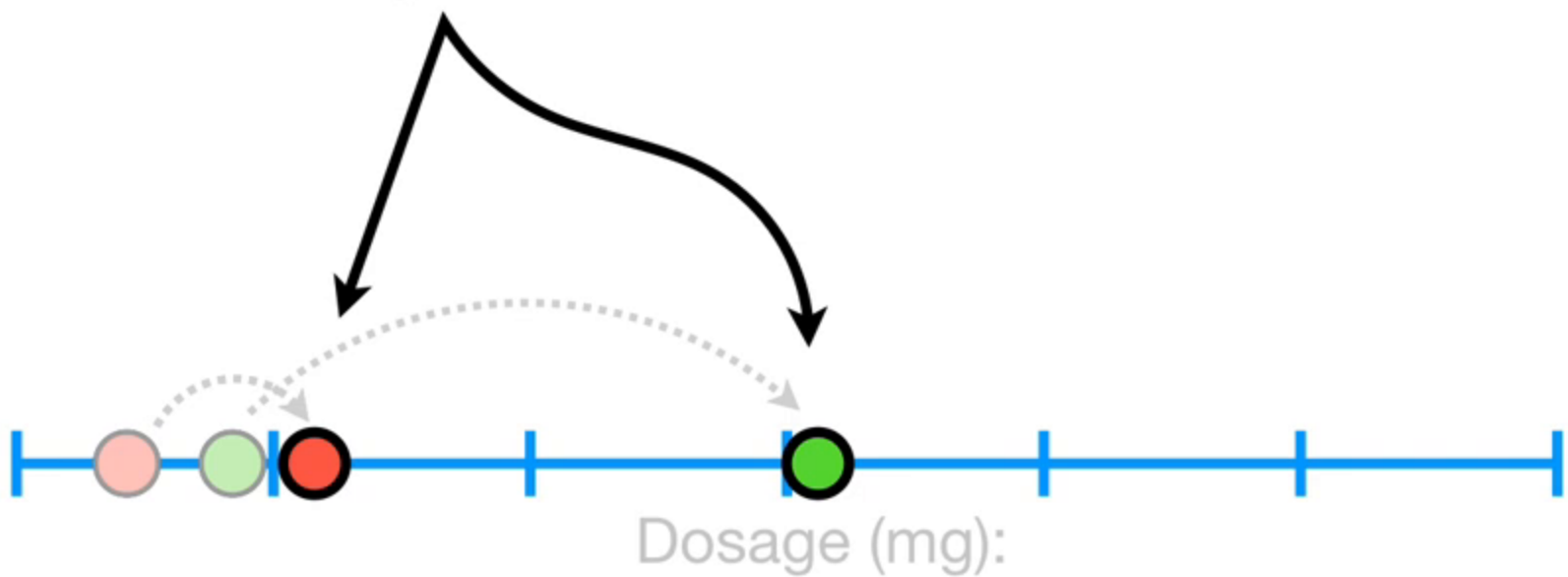
In other words, when $r = 0$ and $d = 2$, all the **Polynomial Kernel** does is shift the data down the original axis.



When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 2$ we get... $a^2 b^2 = (a^2) \cdot (b^2)$

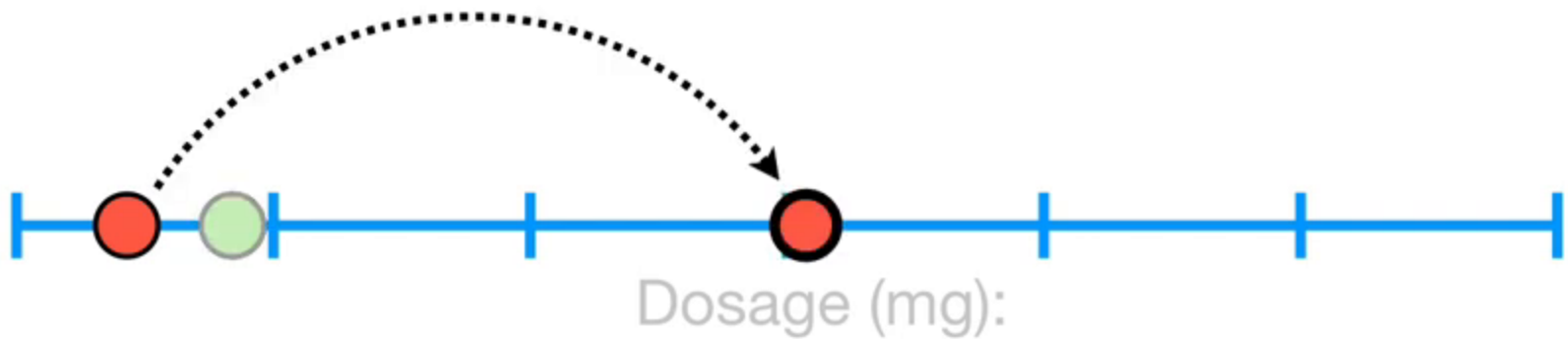
In other words, when $r = 0$ and $d = 2$, all the **Polynomial Kernel** does is shift the data down the original axis.



When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 3$ we get... $2.5^3 \times 4^3 = (2.5^3) \cdot (4^3)$

…and we shift the data down the ***x*-axis** further because now we are cubing each value.



When $r = 0$… $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 1$ we get… $a^1 b^1 = (a) \cdot (b)$

Lastly, when $r = 0$ and $d = 1$…

When $\boldsymbol{r = 0}$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

So, setting $\boldsymbol{r = 0}$ seems silly because no matter what values we use for ***d***, the **Dot Products** leave the data in the original dimension...

When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 1$ we get... $a^1 b^1 = (a) \cdot (b)$

When $d = 2$ we get... $a^2 b^2 = (a^2) \cdot (b^2)$

When $d = 3$ we get... $a^3 b^3 = (a^3) \cdot (b^3)$

...and in this example, the data stays on the same **1-Dimensional** line.

Dosage (mg):

When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

When $d = 1$ we get... $a^1 b^1 = (a) \cdot (b)$

When $d = 2$ we get... $a^2 b^2 = (a^2) \cdot (b^2)$

When $d = 3$ we get... $a^3 b^3 = (a^3) \cdot (b^3)$

However, it turns out that setting $r = 0$ can result in some pretty awesome stuff.

Dosage (mg):

When $\boldsymbol{r = 0}$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

$a^1 b^1$

..let's talk about what happens if we take a **Polynomial Kernel** with $\boldsymbol{r = 0}$ and $\boldsymbol{d = 1}$...

Dosage (mg):

When $\boldsymbol{r = 0}$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

$$a^1 b^1 + a^2 b^2$$

...and add another **Polynomial Kernel** with $\boldsymbol{r = 0}$ and $\boldsymbol{d = 2}$.

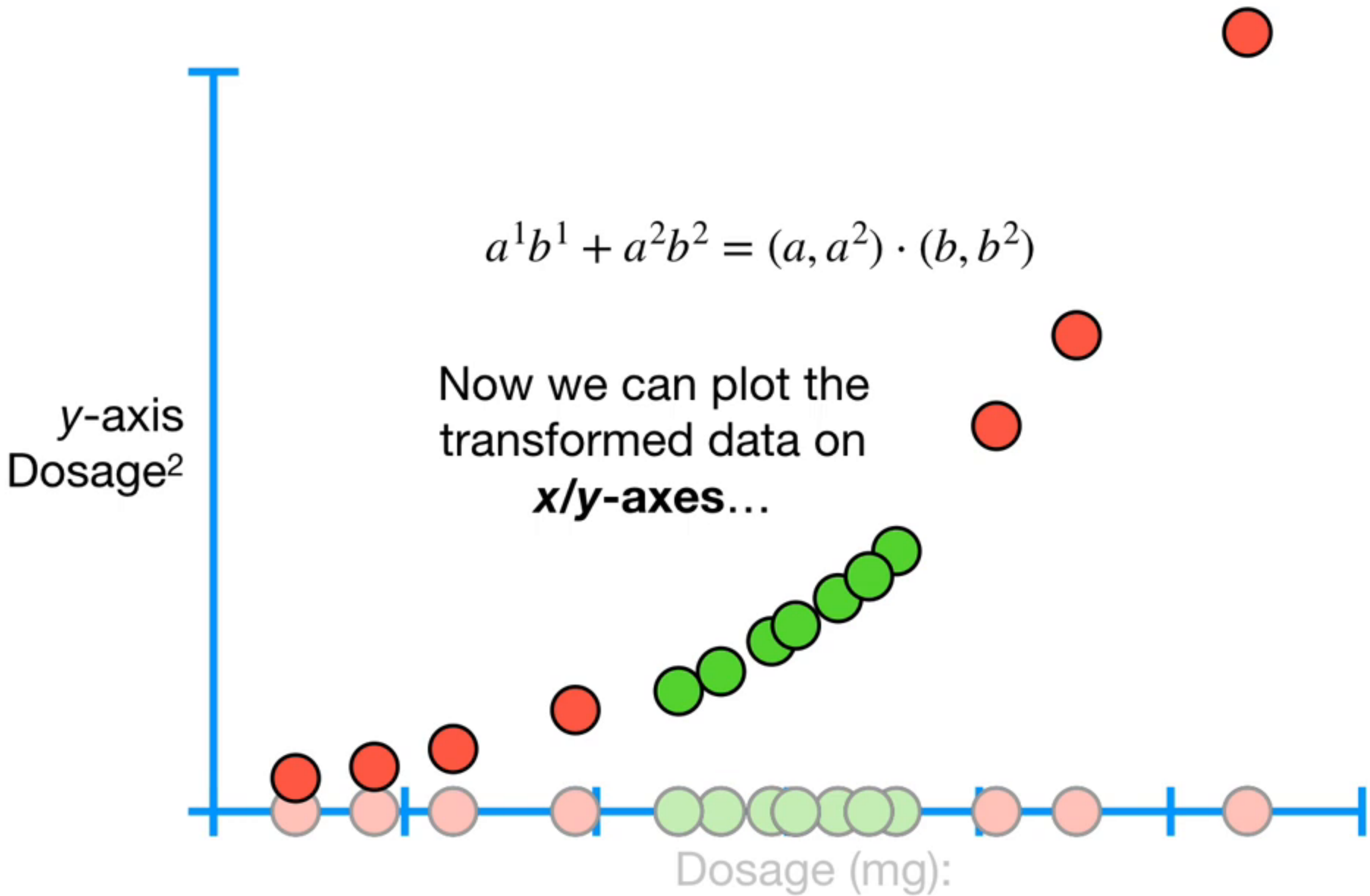
Dosage (mg):

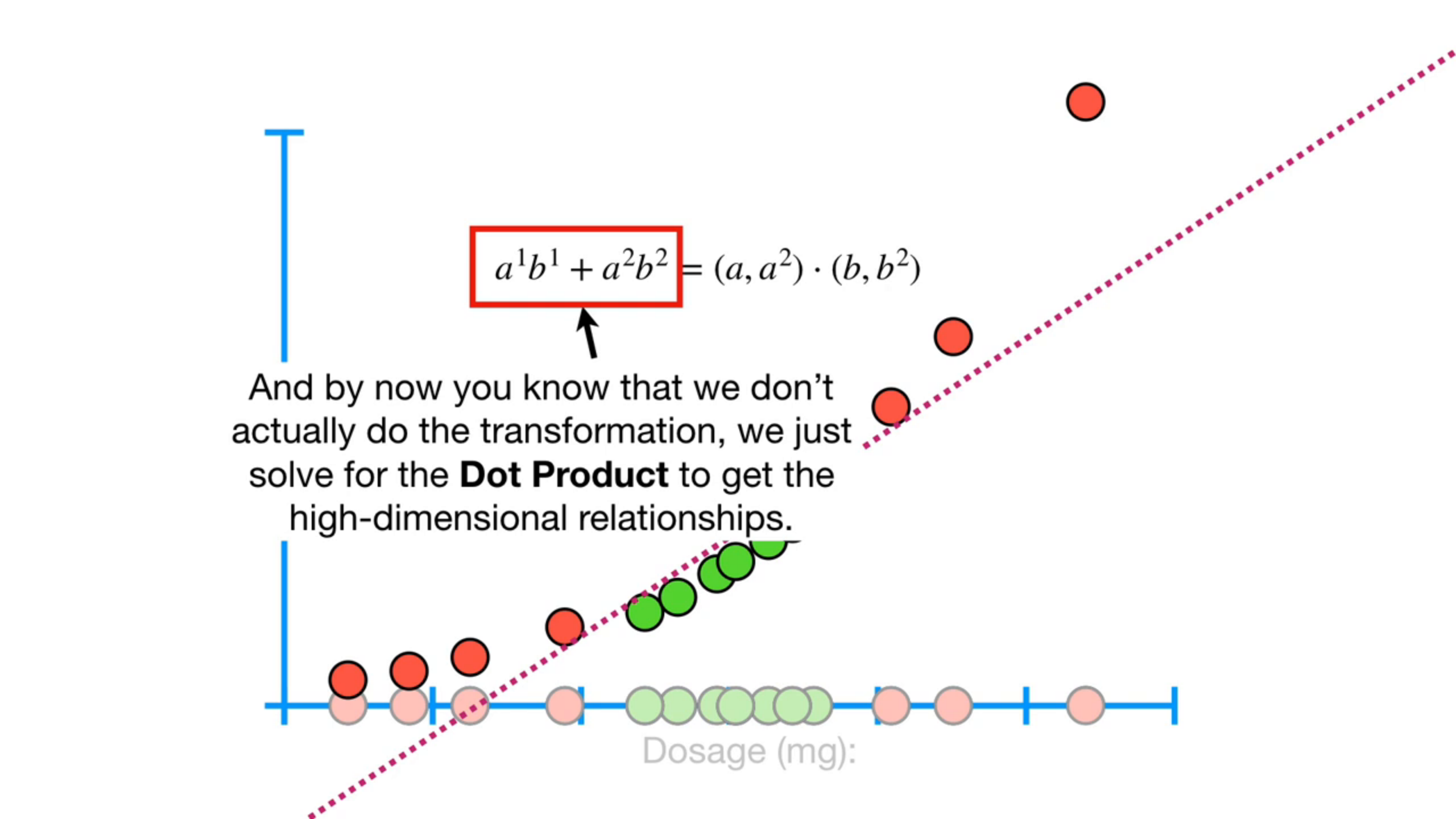
When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

$$a^1 b^1 + a^2 b^2 = (a, a^2) \cdot (b, b^2)$$

The first coordinate is the original **Dosage**...

Dosage (mg):


$a^1b^1 + a^2b^2 = (a, a^2) \cdot (b, b^2)$
Now we can plot the transformed data on ***x/y*-axes**…
y-axis
Dosage²
Dosage (mg):

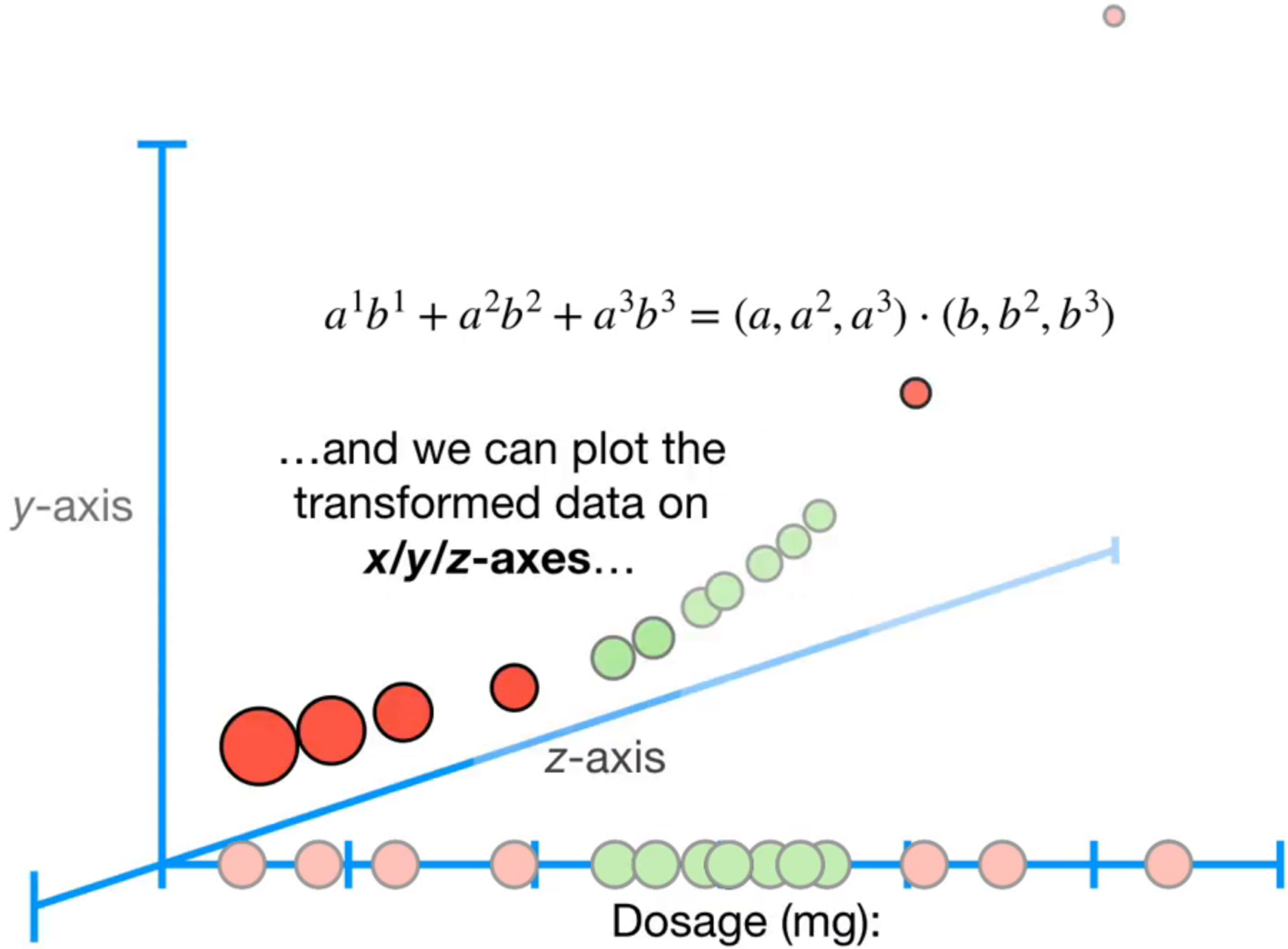

$a^1b^1 + a^2b^2 = (a, a^2) \cdot (b, b^2)$
And by now you know that we don't actually do the transformation, we just solve for the **Dot Product** to get the high-dimensional relationships.
Dosage (mg):

When $\boldsymbol{r = 0}$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

$$a^1 b^1 + a^2 b^2 + a^3 b^3$$

Now, if we added another
Polynomial Kernel with $\boldsymbol{r = 0}$
and $\boldsymbol{d = 3}$...

Dosage (mg):


$a^1b^1 + a^2b^2 + a^3b^3 = (a, a^2, a^3) \cdot (b, b^2, b^3)$
…and we can plot the transformed data on ***x/y/z*-axes**…
y-axis
z-axis
Dosage (mg):

When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

$$a^1 b^1 + a^2 b^2 + a^3 b^3 + \cdots$$

Now, what if we just kept adding **Polynomial Kernels** with $r = 0$ and increasing d until d = **infinity**?

Dosage (mg):

When $r = 0$... $(a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

$$a^1 b^1 + a^2 b^2 + a^3 b^3 + \cdots + a^\infty b^\infty = (a, a^2, a^3, \ldots, a^\infty) \cdot (b, b^2, b^3, \ldots, b^\infty)$$

That would give us a **Dot Product** with coordinates for an *infinite number of dimensions*!!!!

Dosage (mg):

When $r = 0 \ldots (a \times b + r)^d = (a \times b)^d = a^d b^d = (a^d) \cdot (b^d)$

$$a^1 b^1 + a^2 b^2 + a^3 b^3 + \cdots + a^\infty b^\infty = (a, a^2, a^3, \ldots, a^\infty) \cdot (b, b^2, b^3, \ldots, b^\infty)$$

Well, that's exactly what the **Radial Kernel** does, so let's talk about it!!!

Dosage (mg):

Warning: This part gets very mathy, so feel free to skip to the end if this is not your thing.

Now, because we can set ***γ*** to anything, let's set it to **1/2**…

$$e^{-\gamma(a-b)^2} = e^{-\gamma(a^2+b^2-2ab)} = e^{-\gamma(a^2+b^2)}e^{\gamma 2ab}$$

Now let's create the **Taylor Series Expansion** of this last term.

$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2-2ab)} = e^{-\frac{1}{2}(a^2+b^2)} e^{ab}$$

This big thing is a **Taylor Series**.

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$$

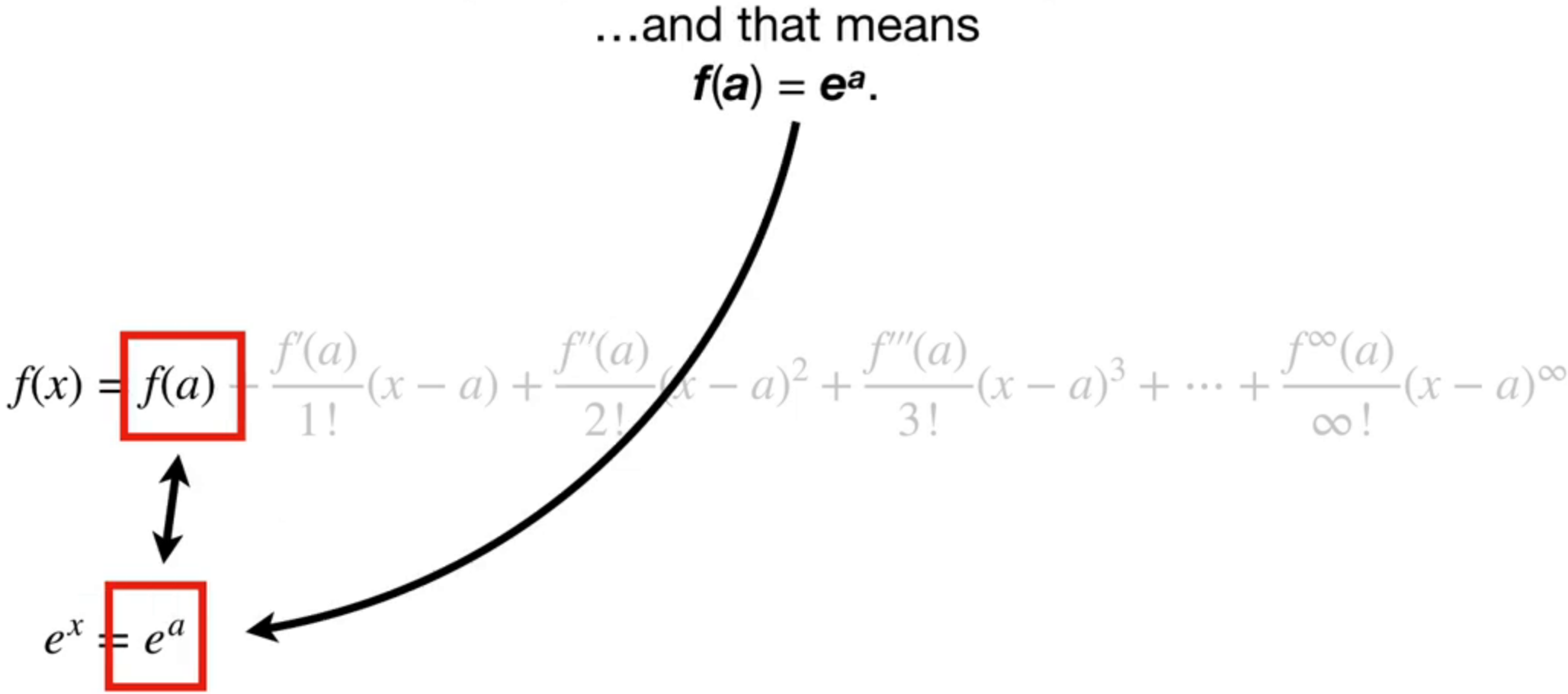
…can be split into an infinite sum.

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$$

Since this is very abstract, let's walk through how we convert e^x into an infinite sum.

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$$

e^x


…and that means
$f(a) = e^a$.
$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$
$e^x = e^a$

..and multiply by ***x*** **-** ***a***.

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$$

$$e^x = e^a + \frac{e^a}{1!}(x-a)$$

NOTE: In case you don't already know, the derivative of $e^x = e^x$, so taking the derivative of e^x is super easy.

$$\frac{d}{dx}e^x = e^x$$

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$$

$$e^x = e^a + \frac{e^a}{1!}$$

Now we plug in the second derivative of e^x evaluated at a, divide by **2** factorial…

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$$

$$e^x = e^a + \frac{e^a}{1!}(x-a) + \frac{e^a}{2!}$$

Then we keep adding terms based on higher derivatives…

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$$

$$e^x = e^a + \frac{e^a}{1!}(x-a) + \frac{e^a}{2!}(x-a)^2 + \frac{e^a}{3!}(x-a)^3 + \cdots$$

Thus, this is the **Taylor Series Expansion** of e^x.

$$e^x = e^a + \frac{e^a}{1!}(x-a) + \frac{e^a}{2!}(x-a)^2 + \frac{e^a}{3!}(x-a)^3 + \cdots + \frac{e^a}{\infty!}(x-a)^\infty$$

The definition of the **Taylor Series** says that ***a*** can be any value as long as ***f*(*a*)** exists…

$$e^x = e^a + \frac{e^a}{1!}(x - a) + \frac{e^a}{2!}(x - a)^2 + \frac{e^a}{3!}(x - a)^3 + \cdots + \frac{e^a}{\infty!}(x - a)^\infty$$

The definition of the **Taylor Series** says that ***a*** can be any value as long as ***f*(*a*)** exists…

…and since $e^0 = 1$, e^0 exists, so we will set $a = 0$…

$$e^x = e^0 + \frac{e^0}{1!}(x - 0) + \frac{e^0}{2!}(x - 0)^2 + \frac{e^0}{3!}(x - 0)^3 + \cdots + \frac{e^0}{\infty!}(x - 0)^\infty$$

Thus, if we can accept that the **Taylor Series Expansion** does what it says it does, e^x…

$$e^x = 1 + \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \cdots + \frac{1}{\infty!}x^\infty$$

LARGE BAM!!!!!!!

$$e^x = 1 + \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \cdots + \frac{1}{\infty!}x^\infty$$

…we can now create the **Taylor Series Expansion** of this last term.

$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2-2ab)} = e^{-\frac{1}{2}(a^2+b^2)}e^{ab}$$

$$e^x = 1 + \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \cdots + \frac{1}{\infty!}x^\infty$$

Now we have the **Taylor Series Expansion** of the last part of the **Radial Kernel**.

$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2-2ab)} = e^{-\frac{1}{2}(a^2+b^2)} e^{ab}$$

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^{\infty}$$

OK. Time to take a deep breath. We've done a lot, but we still have a few more steps before we are done and get to eat snacks.

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^{\infty}$$

Before we move on, let's remember that when we added up a bunch of **Polynomial Kernels** with ***r* = 0** and ***d*** going from **0** to **infinity**…

$$a^0b^0 + a^1b^1 + a^2b^2 + \cdots + a^\infty b^\infty$$

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$$

…we got a **Dot Product** with coordinates for an infinite number of dimensions.

$$a^0b^0 + a^1b^1 + a^2b^2 + \cdots + a^\infty b^\infty = (1,\, a,\, a^2,\, \ldots,\, a^\infty) \cdot (1,\, b^1,\, b^2,\, \ldots,\, b^\infty)$$

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$$

Now, observe that a
Polynomial Kernel with ***r* = 0**
and ***d* = 0** is equal to **1**…

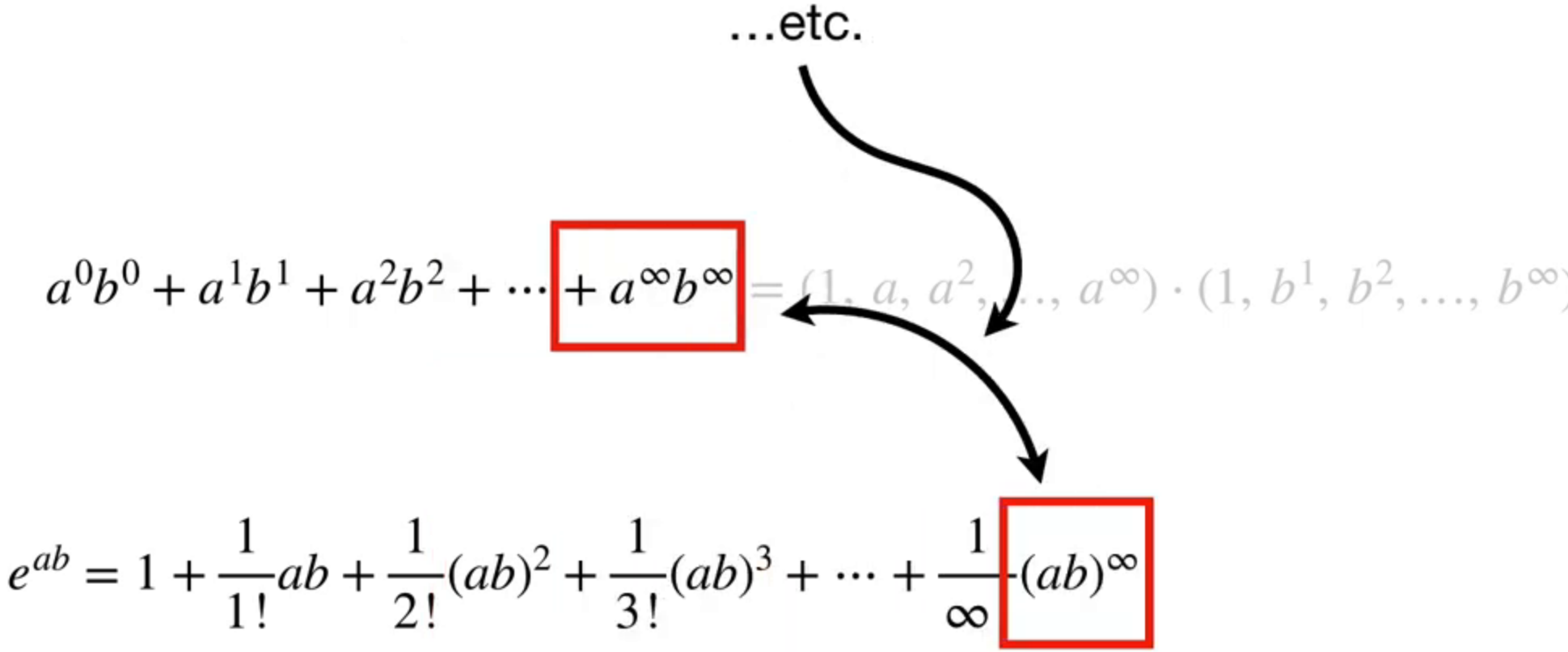
$$a^0b^0 + a^1b^1 + a^2b^2 + \cdots + a^\infty b^\infty = (1,\, a,\, a^2,\, \ldots,\, a^\infty) \cdot (1,\, b^1,\, b^2,\, \ldots,\, b^\infty)$$

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$$

…and a **Polynomial Kernel** with
r* = 0** and ***d* = 2** is equal to (ab***)²…

$$a^0b^0 + a^1b^1 + a^2b^2 + \cdots + a^\infty b^\infty = (1,\ a,\ a^2,\ \ldots,\ a^\infty) \cdot (1,\ b^1,\ b^2,\ \ldots,\ b^\infty)$$

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$$

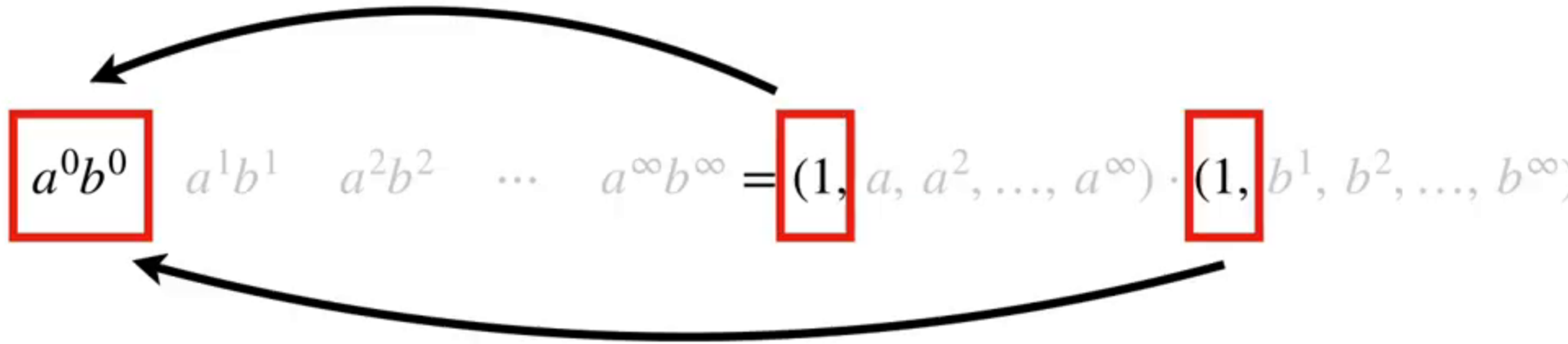

…etc.
$a^0b^0 + a^1b^1 + a^2b^2 + \cdots + a^\infty b^\infty = (1, a, a^2, \ldots, a^\infty) \cdot (1, b^1, b^2, \ldots, b^\infty)$
$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty}(ab)^\infty$

Thus, each term in this **Taylor Series Expansion** contains a **Polynomial Kernel** with ***r* = 0** and ***d*** going from **0** to **infinity**.

$$a^0b^0 + a^1b^1 + a^2b^2 + \cdots + a^\infty b^\infty = (1,\, a,\, a^2,\, \ldots,\, a^\infty) \cdot (1,\, b^1,\, b^2,\, \ldots,\, b^\infty)$$

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$$

…because the **Dot Product** tells us
to multiply each term together…



$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$$

With that in mind, the **Dot Product** for e^{ab} is…

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$$

We can verify that the **Dot Product** is correct by multiplying each term together…

$$e^{ab} = (1, \sqrt{\frac{1}{1!}}a, \sqrt{\frac{1}{2!}}a^2, \boxed{\sqrt{\frac{1}{3!}}a^3}, \ldots, \sqrt{\frac{1}{\infty!}}a^\infty) \cdot (1, \sqrt{\frac{1}{1!}}b, \sqrt{\frac{1}{2!}}b^2, \boxed{\sqrt{\frac{1}{3!}}b^3}, \ldots, \sqrt{\frac{1}{\infty!}}b^\infty)$$

$$1 \quad \frac{1}{1!}ab \quad \frac{1}{2!}(ab)^2 \quad \boxed{\frac{1}{3!}(ab)^3} \quad \frac{1}{\infty!}(ab)^\infty$$

…to get the **Taylor Series Expansion** of e^{ab}.

$$e^{ab} = (1, \sqrt{\frac{1}{1!}}a, \sqrt{\frac{1}{2!}}a^2, \sqrt{\frac{1}{3!}}a^3, \ldots, \sqrt{\frac{1}{\infty!}}a^\infty) \cdot (1, \sqrt{\frac{1}{1!}}b, \sqrt{\frac{1}{2!}}b^2, \sqrt{\frac{1}{3!}}b^3, \ldots, \sqrt{\frac{1}{\infty!}}b^\infty)$$

$$e^{ab} = 1 + \frac{1}{1!}ab + \frac{1}{2!}(ab)^2 + \frac{1}{3!}(ab)^3 + \cdots + \frac{1}{\infty!}(ab)^\infty$$

…we can plug in the **Dot Product** for e^{ab}.

$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2-2ab)} = e^{-\frac{1}{2}(a^2+b^2)} e^{ab}$$

$$e^{ab} = (1, \sqrt{\frac{1}{1!}}a, \sqrt{\frac{1}{2!}}a^2, \sqrt{\frac{1}{3!}}a^3, \ldots, \sqrt{\frac{1}{\infty!}}a^\infty) \cdot (1, \sqrt{\frac{1}{1!}}b, \sqrt{\frac{1}{2!}}b^2, \sqrt{\frac{1}{3!}}b^3, \ldots, \sqrt{\frac{1}{\infty!}}b^\infty)$$

To make the **Radial Kernel** all one **Dot Product** instead of something times a **Dot Product**...

$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)} \left[(1, \sqrt{\frac{1}{1!}}a, \sqrt{\frac{1}{2!}}a^2, \ldots, \sqrt{\frac{1}{\infty!}}a^{\infty}) \cdot (1, \sqrt{\frac{1}{1!}}b, \sqrt{\frac{1}{2!}}b^2, \ldots, \sqrt{\frac{1}{\infty!}}b^{\infty}) \right]$$

…we just multiply both parts of the **Dot Product** by the square root of this term.

$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)} \left[(1, \sqrt{\frac{1}{1!}}a, \sqrt{\frac{1}{2!}}a^2, \ldots, \sqrt{\frac{1}{\infty!}}a^\infty) \cdot (1, \sqrt{\frac{1}{1!}}b, \sqrt{\frac{1}{2!}}b^2, \ldots, \sqrt{\frac{1}{\infty!}}b^\infty) \right]$$

So we can fit everything on to the screen, let's let **s** equal the square root of the first term.

$$s = \sqrt{e^{-\frac{1}{2}(a^2+b^2)}}$$

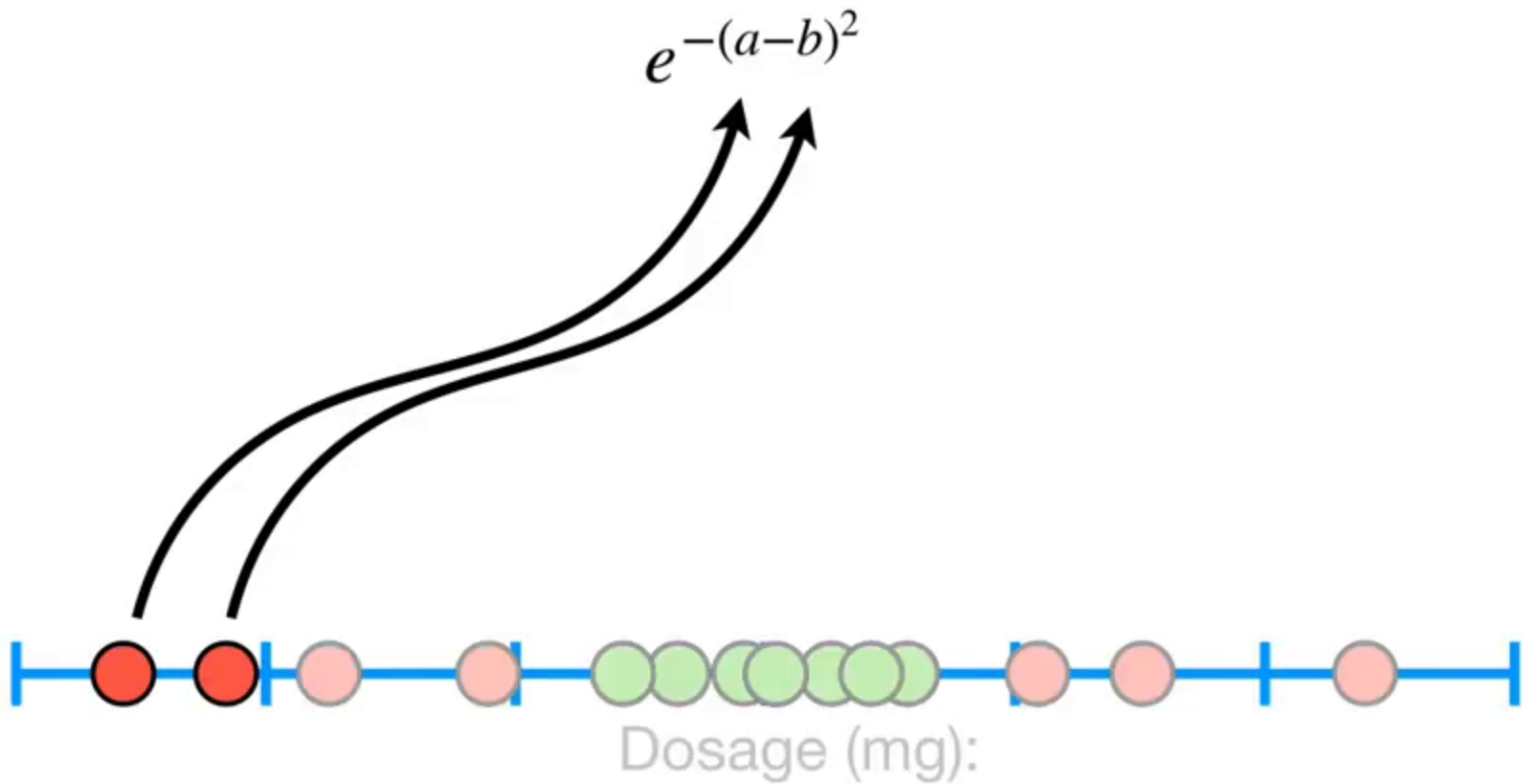
$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)} \left[(1, \sqrt{\frac{1}{1!}}a, \sqrt{\frac{1}{2!}}a^2, \ldots, \sqrt{\frac{1}{\infty!}}a^\infty) \cdot (1, \sqrt{\frac{1}{1!}}b, \sqrt{\frac{1}{2!}}b^2, \ldots, \sqrt{\frac{1}{\infty!}}b^\infty) \right]$$

…and, at long last, we see that the **Radial Kernel** is equal to a **Dot Product** that has coordinates for an infinite number of dimensions.

$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)}\left[(1,\sqrt{\frac{1}{1!}}a,\sqrt{\frac{1}{2!}}a^2,\ldots,\sqrt{\frac{1}{\infty!}}a^\infty)\cdot(1,\sqrt{\frac{1}{1!}}b,\sqrt{\frac{1}{2!}}b^2,\ldots,\sqrt{\frac{1}{\infty!}}b^\infty)\right]$$

$$e^{-\frac{1}{2}(a-b)^2} = (s, s\sqrt{\frac{1}{1!}}a, s\sqrt{\frac{1}{2!}}a^2,\ldots,s\sqrt{\frac{1}{\infty!}}a^\infty)\cdot(s, s\sqrt{\frac{1}{1!}}b, s\sqrt{\frac{1}{2!}}b^2,\ldots,s\sqrt{\frac{1}{\infty!}}b^\infty)$$

That means that when we plug numbers into the **Radial Kernel**…



…the value we get at the end is the relationship between the two points in **infinite-dimensions**.

$$e^{-(2.5-4)^2} = e^{-(-1.5)^2} = e^{-2.25} = 0.11$$

Dosage (mg):

Now we can go eat snacks!!!

Dosage (mg):

Subscribe!!!

Support StatQuest!!! 

Subscribe!!!

Support StatQuest!!!