Machine Learning

| Radial Basis Function (RBF) Kernel

Prof. Dr. Mostafa Elhosseini

https://youtube.com/ElhosseiniAcademy

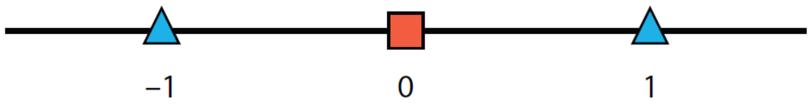
Agenda

- ~~Polynomial kernel~~
- ~~Dot Product~~
- ~~Polynomial kernel – derivation~~
- ~~Coding~~

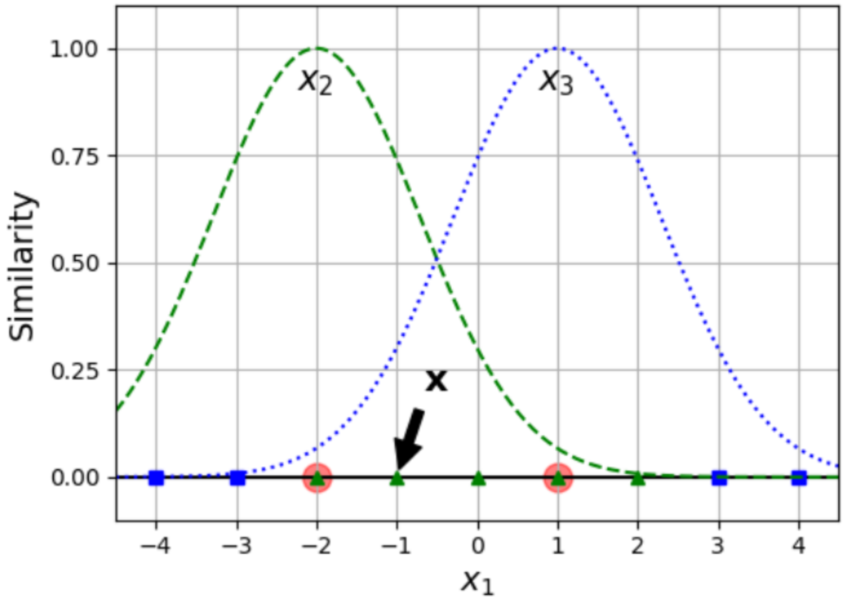
Radial Basis Function (RBF) kernel

- Using bumps in higher dimensions to our benefit
- build nonlinear boundaries using certain special functions centered at each of the data points.
- To add features computed using a similarity function, which measures how much each instance resembles a particular landmark

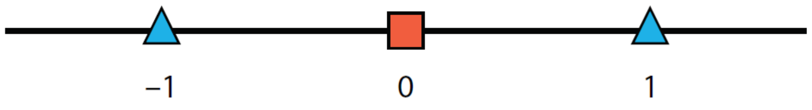
Radial Basis Function (RBF) kernel



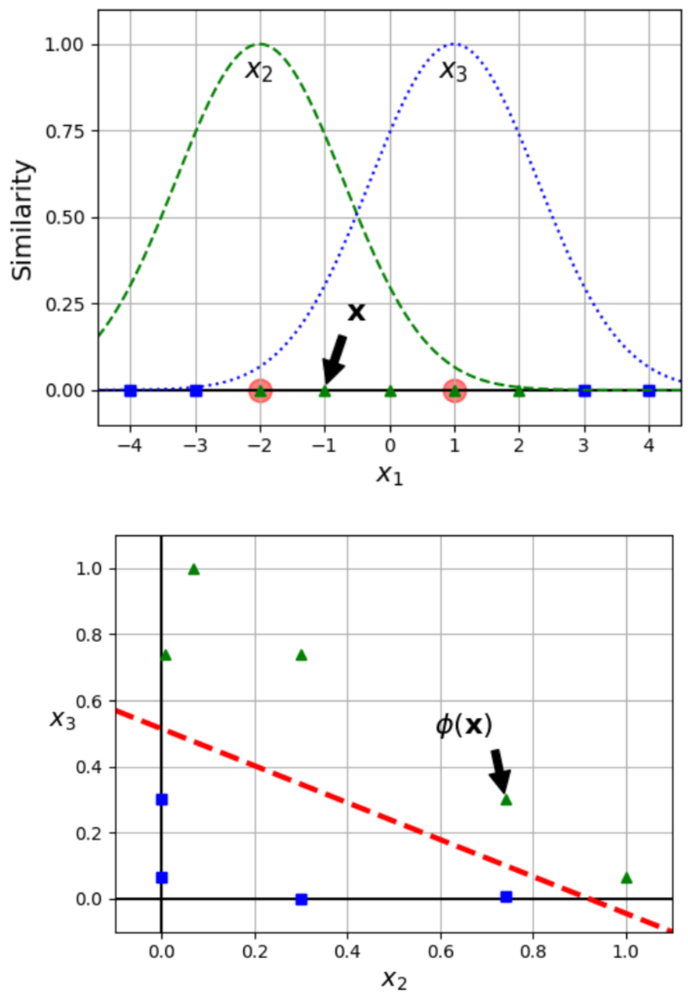
- This dataset is not linearly separable
- Add features computed using a similarity function, which measures how much each instance resembles a particular landmark
- Add two landmarks to it at $x_1 = -2$ and $x_1 = 1$. Next, we'll define the similarity function to be the Gaussian RBF with $\gamma = 0.3$.



Radial Basis Function (RBF) kernel



- Let's look at the instance $x_1 = -1$: it is located at a distance of 1 from the first landmark and 2 from the second landmark. Therefore, its new features are $x_2 = \exp(-0.3 \times 1^2) \approx 0.74$ and $x_3 = \exp(-0.3 \times 2^2) \approx 0.3$.



How to select the landmarks

- The simplest approach is to create a landmark at the location of each and every instance in the dataset.
- Doing that creates many dimensions and thus increases the chances that the transformed training set will be linearly separable.

- The downside is that a training set with m instances and n features gets transformed into a training set with m instances and m features (assuming you drop the original features).

Idea

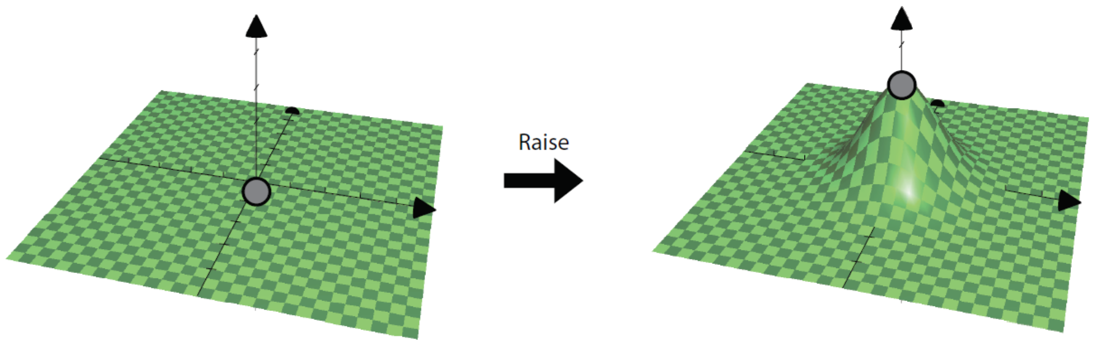
- Imagine building a **mountain** or a **valley** on each of the points
- Imagine the plane as a flat surface (a **blanket**), representing your input space (e.g., the feature space in machine learning).
- If you **pinch** the blanket at a specific point and **pull it upward**, you're creating a "**mountain**" at that point. This represents a point in the input space where the RBF function has a higher value.
- Similarly, if you **push** the blanket down at another point, you create a "**valley**," where the RBF function has a lower value.
- If we can imagine lifting and pushing down a blanket, and then building a classifier by looking at the boundary of the points that lie at a particular height, then we can understand what an RBF kernel is.

Idea

- These "mountains" and "valleys" are the radial basis functions themselves. The height or depth at each point (the value of the function) depends on the **distance from a central point** (the location of the pinch or push). This is why they're called "**radial**" basis functions—because the function's value depends on the radial distance from the center.
- The term "radial" comes from the word "radius," which is the distance from a central point to any point on the circumference of a circle.

Idea

- Imagine a specific point in your dataset (this is the "center").



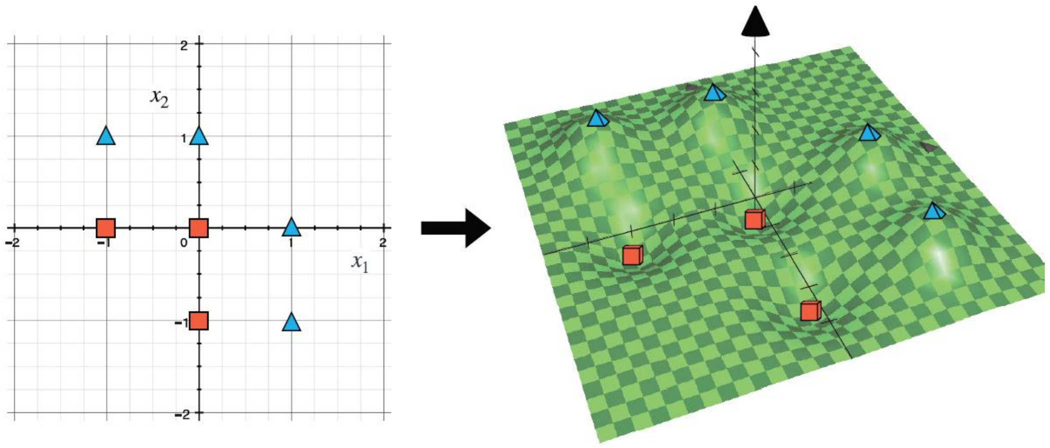
- The RBF calculates the distance of every other point in the dataset from this center.

Idea

- The RBF kernel in machine learning uses these radial functions to transform the data. By adding "mountains" and "valleys" (i.e., transforming the feature space), it can help make the data more separable (e.g., enabling linear separation in a higher-dimensional space).

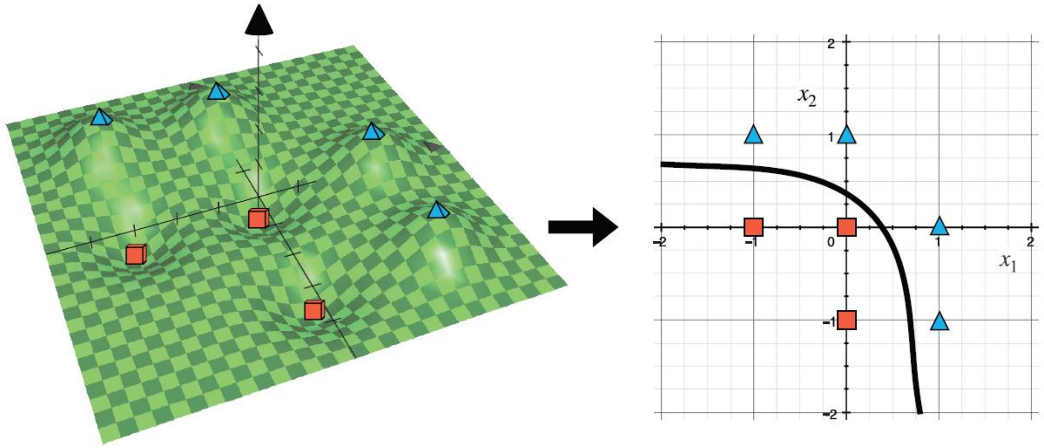
How do we use this as a classifier?

- We lift the plane at every triangle and push it down at every square



How do we use this as a classifier?

- To create the classifier, we draw a plane at height 0 and intersect it with our surface
- We then project everything back to the plane and obtain our desired classifier



Model

- $e^{-\gamma(a-b)^2}$
- Just like with the Polynomial Kernel, a and b refer to two different Dosage measurements.
- The difference between the measurements is then squared, giving us the squared distance between the two observations.
- Thus, the amount of influence one observation has on another is a function of the squared distance.
- γ (gamma), which is determined by Cross Validation, scales the squared distance, and thus, it scales the influence.

Model

- $e^{-\gamma(a-b)^2}$
- $a = 2.5 \quad b = 4 \qquad \gamma = 1$
- $e^{-\gamma(a-b)^2} = e^{-(2.5-4)^2} = 0.11$
- If we set $\gamma = 2$ $\rightarrow$ $e^{-2(2.5-4)^2} = 0.01$ Which is less than when $\gamma = 1$
- So we see that by scaling the distance, γ scales the amount of influence two points have on each other

Model

- $e^{-\gamma(a-b)^2}$
- Now let's set $\gamma = 1$ again, and determine how much influence two observations have when they are relatively far from each other
- $a = 2.5 \quad b = 16 \qquad\qquad \gamma = 1$
- $e^{-(2.5-16)^2} = e^{-(-13.5)^2} = e^{-182.25} = \text{a number very close to zero}$
- and when the points are relatively far from each other, we get A Number Very Close to Zero
- Thus, the further two observations are from each other, the less influence they have on each other.

Model

- $e^{-\gamma(a-b)^2}$
- NOTE: Just like with the Polynomial Kernel, when we plug values into the Radial Kernel, we get the high-dimensional relationship.
- $e^{-\gamma(a-b)^2} \rightarrow$ High-dimensional relationship
- Thus, 0.11 is the high-dimensional relationship between these two observations that are relatively close to each other…
- and A Number Very Close to Zero is the high-dimensional relationship between these two observations that are relatively far from each other.

Model

- what if we just kept adding **Polynomial Kernels** with $r = 0$ and increasing d until $d = $ infinity?
- $a^1b^1 + a^2b^2 + a^3b^3 + \cdots + a^\infty b^\infty = (a, a^2, a^3, \dots, a^\infty).(b, b^2, b^3, \dots, b^\infty)$
- That would give us a Dot Product with coordinates for an infinite number of dimensions!!!!
- Let's start with the radial kernel
- $e^{-\gamma(a-b)^2}$

$$e^{-\frac{1}{2}(a-b)^2} = e^{-\frac{1}{2}(a^2+b^2)}\left[\left(1, \sqrt{\frac{1}{1!}}a + \sqrt{\frac{1}{2!}}a^2 + \sqrt{\frac{1}{3!}}a^3 + \cdots + \sqrt{\frac{1}{\infty!}}a^\infty\right).\left(1, \sqrt{\frac{1}{1!}}b + \sqrt{\frac{1}{2!}}b^2 + \sqrt{\frac{1}{3!}}b^3 + \cdots + \sqrt{\frac{1}{\infty!}}b^\infty\right)\right]$$

Model

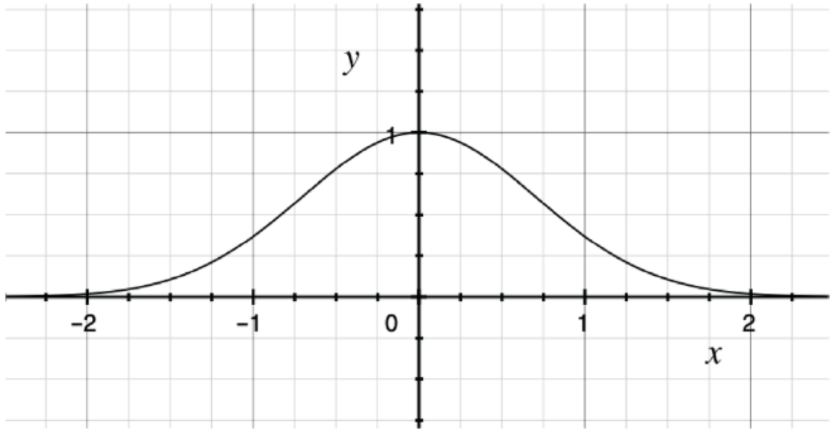
- Assume $s = \sqrt{e^{-\frac{1}{2}(a^2+b^2)}}$

$$e^{-\frac{1}{2}(a-b)^2} = \left(s, s\sqrt{\frac{1}{1!}}a + s\sqrt{\frac{1}{2!}}a^2 + s\sqrt{\frac{1}{3!}}a^3 + \cdots + s\sqrt{\frac{1}{\infty!}}a^\infty\right) \cdot \left(s, s\sqrt{\frac{1}{1!}}b + s\sqrt{\frac{1}{2!}}b^2 + s\sqrt{\frac{1}{3!}}b^3 + \cdots + s\sqrt{\frac{1}{\infty!}}b^\infty\right)$$

- 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.
- That means that when we plug numbers into the Radial Kernel…
- $e^{-(2.5-4)^2} = e^{-(2.5-4)^2} = e^{-2.25} = 0.11$
- the value we get at the end is the relationship between the two points in infinite-dimensions.

A more in-depth look at radial basis functions

- For one variable, the simplest radial basis function has the formula $\boldsymbol{y} = \boldsymbol{e}^{-x^2}$



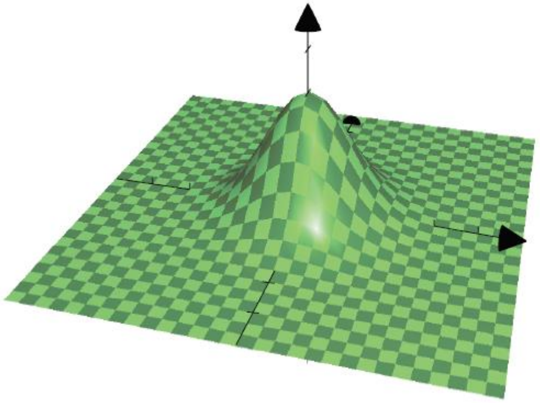
- This looks like a bump over the line. It looks a lot like a standard normal (Gaussian) distribution.

A more in-depth look at radial basis functions

- Notice that this bump happens at 0. If we wanted it to appear at any different point, say p, we can translate the formula and get $\boldsymbol{y} = \boldsymbol{e}^{-(\boldsymbol{x}-\boldsymbol{p})^2}$. For example, the radial basis function centered at the point 5 is precisely $y = e^{-(x-5)^2}$.

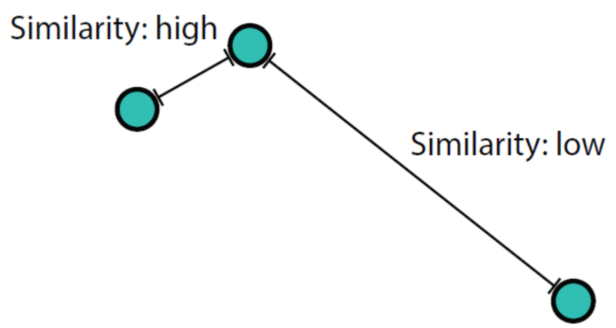
A more in-depth look at radial basis functions

- For two variables, the formula for the most basic radial basis function is $\boldsymbol{z} = \boldsymbol{e}^{-(\boldsymbol{x}^2+\boldsymbol{y}^2)}$.
- This bump happens exactly at the point (0,0). If we wanted it to appear at any different point, say (p, q), we can translate the formula, and get $\boldsymbol{z} = \boldsymbol{e}^{-[(\boldsymbol{x}-\boldsymbol{p})^2+(\boldsymbol{y}-\boldsymbol{q})^2]}$.
- For example, the radial basis function centered at the point $(2, -3)$ is precisely $z = e^{-[(x-2)^2+(y+3)^2]}$



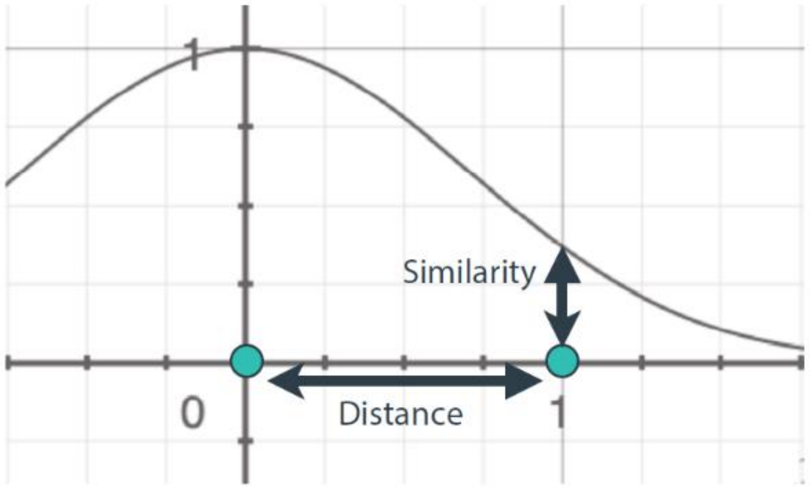
Similarity

- We say that two points are similar if they are close to each other, and not similar if they are far away
- The similarity between two points with x-coordinates x_1, x_2 is measured using the following function: Similarity $(\boldsymbol{x_1}, \boldsymbol{x_2}) = \boldsymbol{e^{-(x_1 - x_2)^2}}$
- This is a Gaussian similarity function, where the value decreases as the distance between the points increases. This function is often used in Radial Basis Function (RBF) kernels.



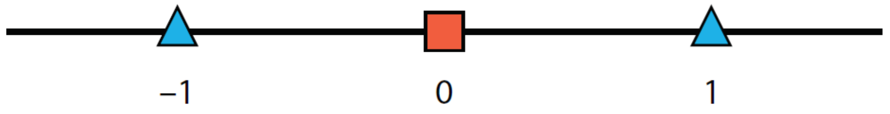
Similarity

- If we want to find the similarity between two points, say p and q, this similarity is precisely the height of the radial basis function centered at p and applied at the point q.
- This is, if we pinch the blanket at point p and lift it, then the height of the blanket at point q is high if the q is close to p and low if q is far from p



Training an SVM with the RBF kernel

Point	*x*	y (label)
1	–1	1
2	0	0
3	1	1



- This dataset is not linearly separable. To make it linearly separable, we'll add a few columns.
- The three columns we are adding are similarity columns, and they record the similarity between the points.

Training an SVM with the RBF kernel

- In the Sim1 column, we'll record the similarity between point 1 and the other three points, and so on.
- Sim1 for Point 2:

$$-(x_1, x_2) = e^{-(x_1-x_2)^2}$$

$$-(-1,0) = e^{-(-1-0)^2} = e^{-1} \approx 0.368$$

Point	*x*	Sim1	Sim2	Sim3	*y*
1	–1	1	0.368	0.018	1
2	0	0.368	1	0.368	0
3	1	0.018	0.368	1	1

Training an SVM with the RBF kernel

- Sim1 for Point 3:

$$-(x_1, x_3) = e^{-(x_1 - x_3)^2}$$

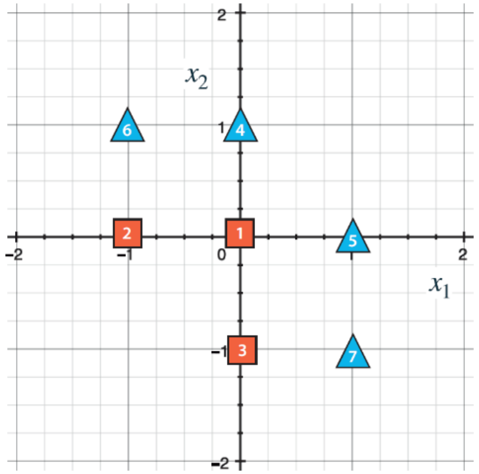
$$-(-1,1) = e^{-(-1-1)^2} = e^{-4} \approx 0.018$$

- This extended dataset is now linearly separable! Many classifiers will separate this set, but in particular, the one with the following boundary equation will:
- $\hat{y} = step(sim1 - sim2 + sim3)$
- $\hat{y} = step(e^{-(x+1)^2} - e^{-x^2} + e^{-(x-1)^2})$

Example 02

- Now, let's do this same procedure but in two dimensions

Point	x_1	x_2	y
1	0	0	0
2	–1	0	0
3	0	–1	0
4	0	1	1
5	1	0	1
6	–1	1	1
7	1	–1	1



- We will now add seven columns to this table. The columns are the similarities with respect to every point. For example, for point 1, we add a similarity column named Sim1. The entry for every point in this column is the amount of similarity between that point and point 1.

Example 02

- The distance between point 1 and point 6, by the Pythagorean theorem follows:
- Distance (Point 1, Point 6) $= \sqrt{(0+1)^2 + (0-1)^2} = \sqrt{2}$
- Similarity (Point 1, Point 6) = $e^{-distance^2} = e^{-(\sqrt{2})^2} = e^{-2} = 0.135$

Point	x_1	x_2	Sim1	Sim2	Sim3	Sim4	Sim5	Sim6	Sim7	y
1	0	0	1	0.368	0.368	0.368	0.368	0.135	0.135	0
2	–1	0	0.368	1	0.135	0.135	0.018	0.368	0.007	0
3	0	–1	0.368	0.135	1	0.018	0.135	0.007	0.368	0
4	0	1	0.368	0.135	0.018	1	0.135	0.368	0.007	1
5	1	0	0.368	0.018	0.135	0.135	1	0.007	0.368	1
6	–1	1	0.135	0.368	0.007	0.367	0.007	1	0	1
7	1	–1	0.135	0.007	0.368	0.007	0.368	0	1	1

Example 02

Point	x_1	x_2	Sim1	Sim2	Sim3	Sim4	Sim5	Sim6	Sim7	y
1	0	0	1	0.368	0.368	0.368	0.368	0.135	0.135	0
2	–1	0	0.368	1	0.135	0.135	0.018	0.368	0.007	0
3	0	–1	0.368	0.135	1	0.018	0.135	0.007	0.368	0
4	0	1	0.368	0.135	0.018	1	0.135	0.368	0.007	1
5	1	0	0.368	0.018	0.135	0.135	1	0.007	0.368	1
6	–1	1	0.135	0.368	0.007	0.367	0.007	1	0	1
7	1	–1	0.135	0.007	0.368	0.007	0.368	0	1	1

- Now, on to building our classifier
- This classifier has the following weights:
  - The weights of x_1 and x_2 are 0.
  - The weight of Sim p is 1, for p = 1, 2, and 3.
  - The weight of Sim p is –1, for p = 4, 5, 6, and 7.
  - The bias is $b = 0$.