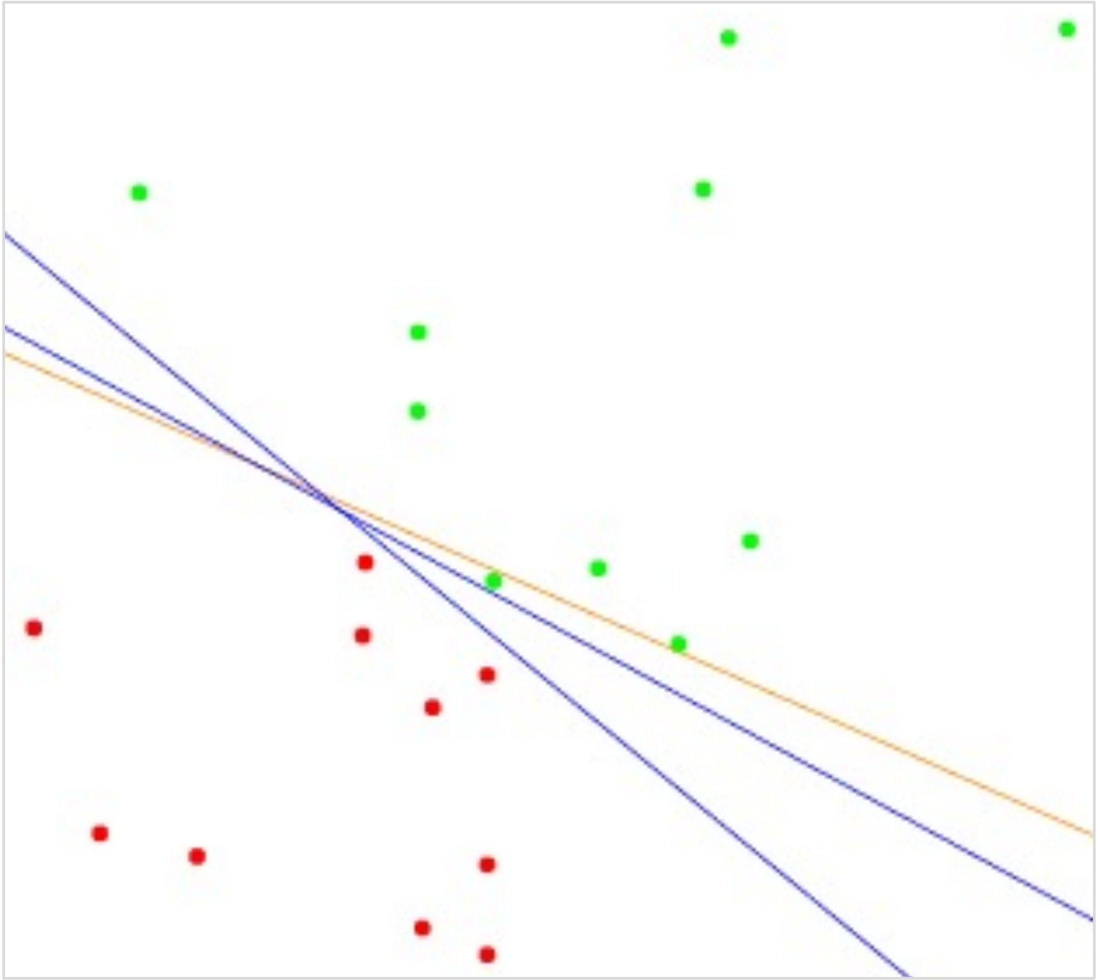
Support Vector Machines

Introduction to Support Vector Machines

Look at this plot from *The Elements of Statistical Learning*. Why don't we try to draw a line that separates the green and red dots? The orange line is the result of a logistic regression. But what about the other two lines? Do they not separate the two classes? Geometrically, what is the "best" way to separate points with a line or plane? The support vector machine is a solution to this question.



Vector Algebra

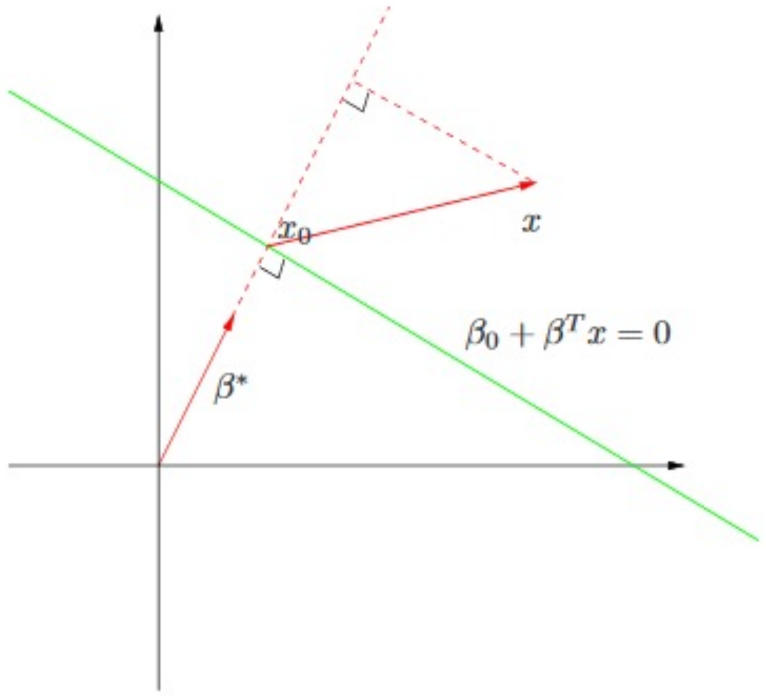


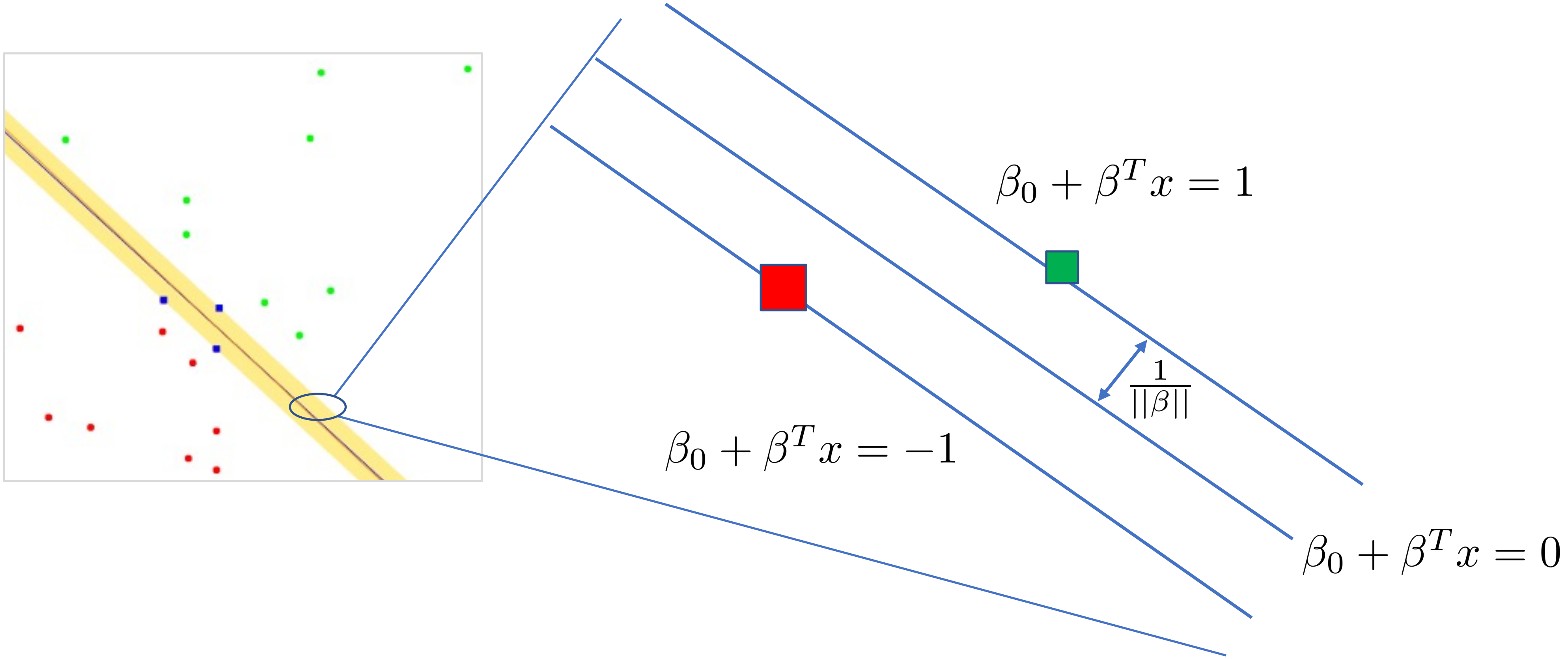
FIGURE 4.15. *The linear algebra of a hyperplane (affine set).*

β* is the normal vector of β. It points in the same direction as β with length 1.

The green line is known as the hyperplane L.

$\beta_0 + \beta^T x = 0$ Which is the same as the equation as $\vec{\beta} \cdot \vec{x} + b = 0$ (Equation of a plane)

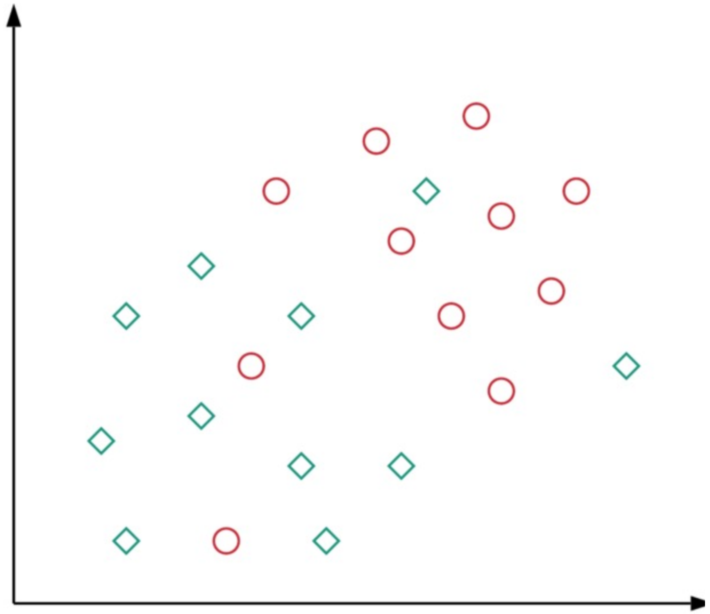
A Closeup of the Margin



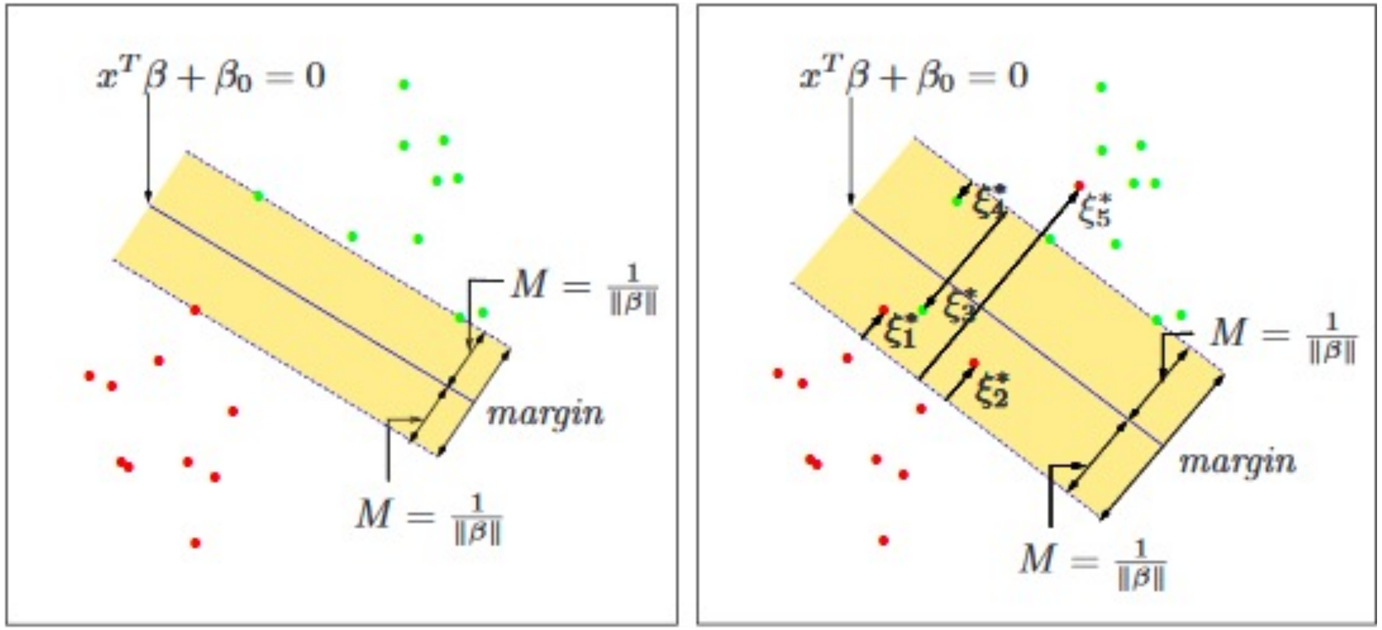
Thus the largest margin is the smallest β. This problem has been solved (see section 4.5.2 of ESL).

$y_i(x_i^T \beta + \beta_0) \geq M||\beta||.$ This is the equation solved for a SVM that is perfectly separable.

What If We Cannot Separate?



There is no straight line that is able to separate the different point shapes with 100% accuracy.



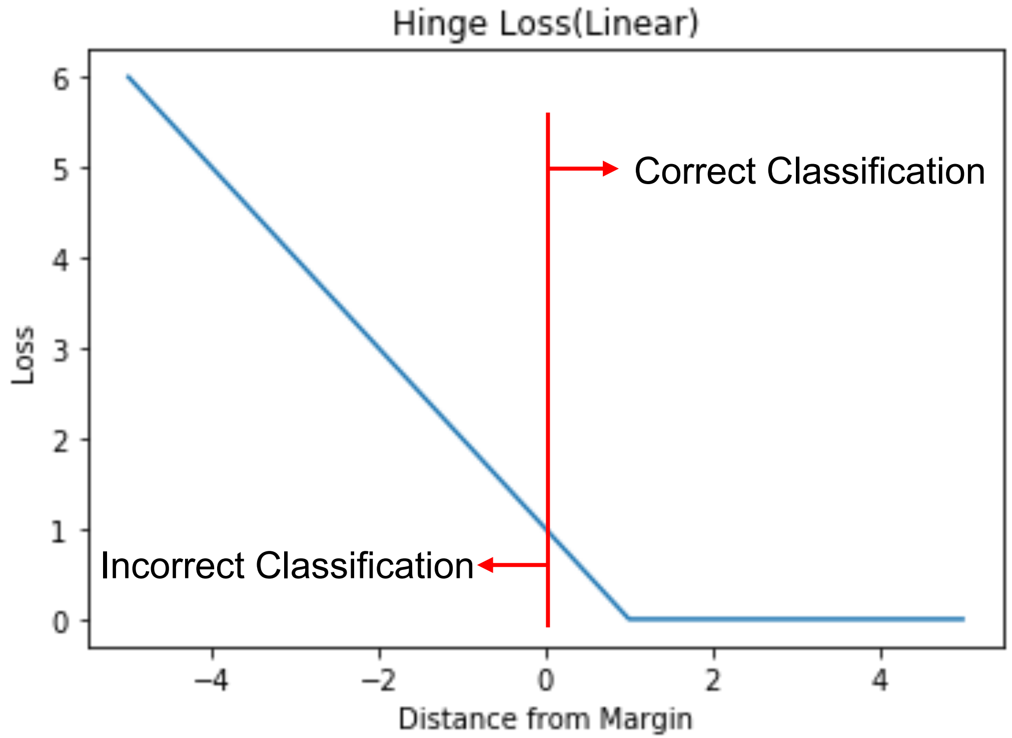
This gives us a modified equation to solve:

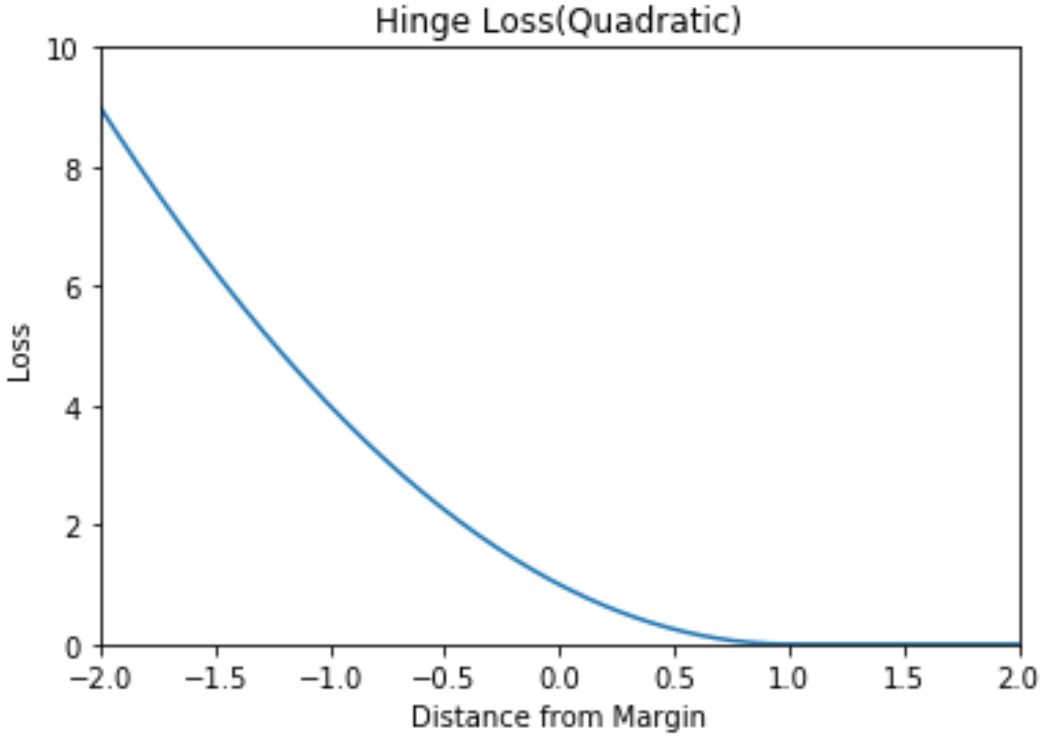
$$
\begin{aligned}
y_i(x_i^T\beta + \beta_0) &\geq M - \xi_i, \\
&\text{or} \\
y_i(x_i^T\beta + \beta_0) &\geq M(1 - \xi_i).
\end{aligned}
$$

DataScience@SMU

Margins and Loss

A New Type of Loss: Hinge

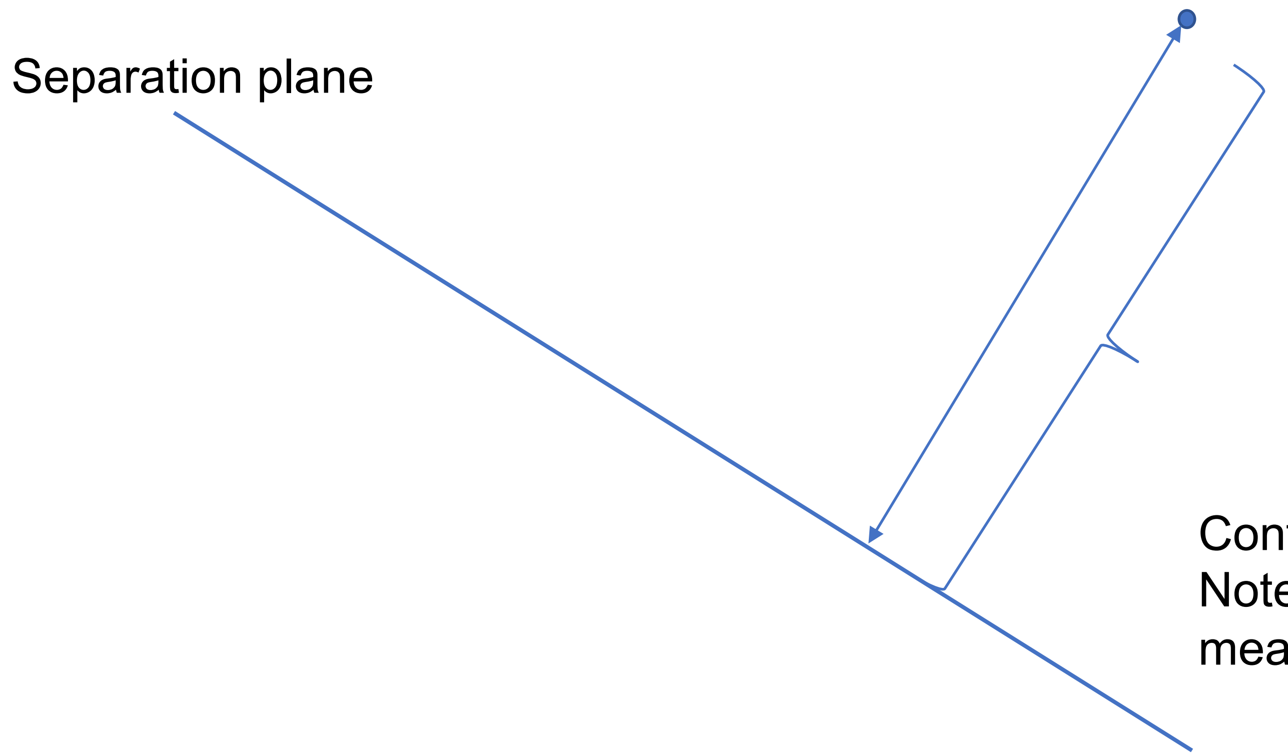




Negative distance means on the wrong side of the hyperplane (misclassification). Note that in this loss, the classifier can correctly classify a point and have that point contribute to the loss. Points far away from the boundary (correctly classified) contribute no loss.

A Slightly New Parameter: Confidence

If we find a value called “confidence” in SVMs, it is the distance to the separation plane (not the margin!!).



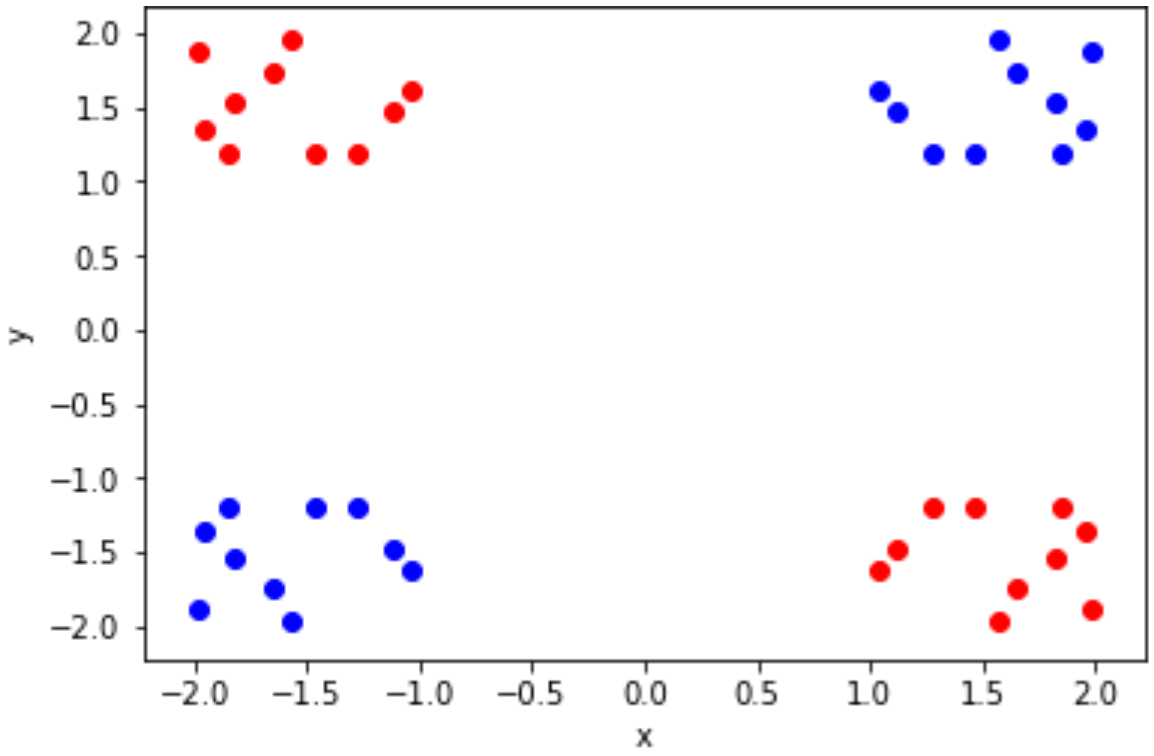
Confidence (positive)
Note: A negative confidence means **misclassification.**

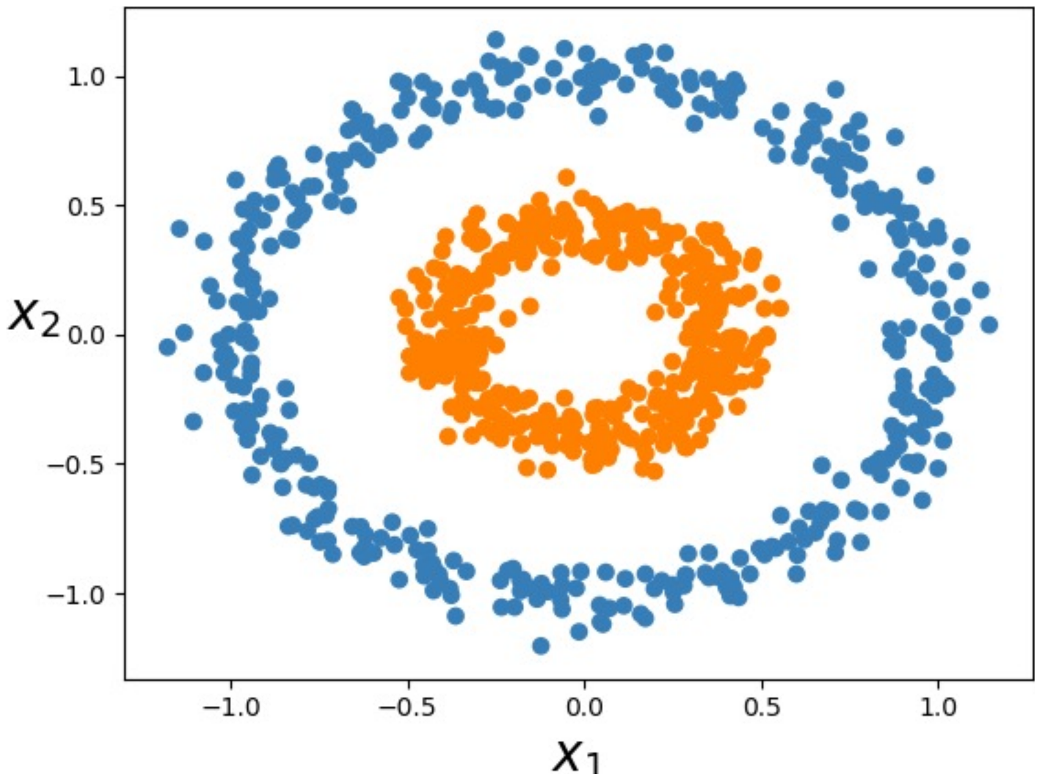
DataScience@SMU

The Kernel Trick

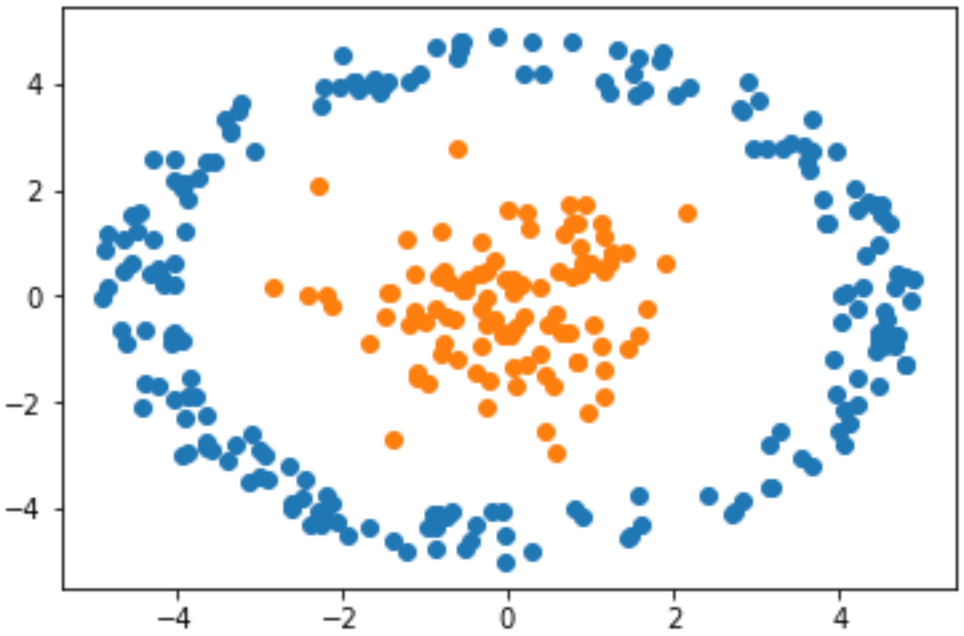
The XOR Problem

There is no good dividing line for problems such as these. A hyperplane in this space will never produce a good result due to the nonlinear nature of the data.

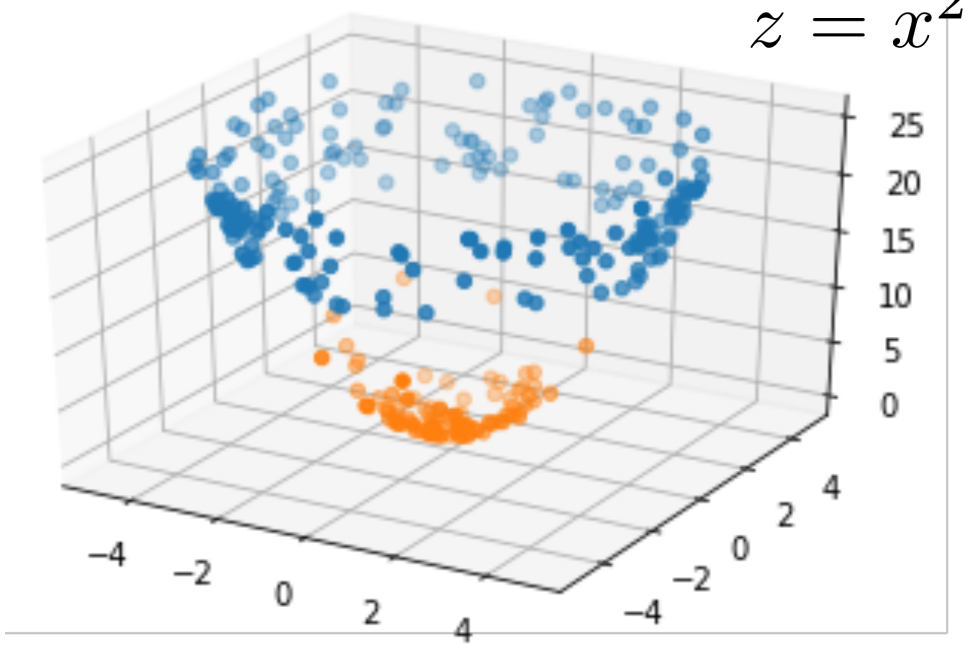




Project Into a Higher Dimension



$$z = x^2 + y^2$$



We can see that in three dimensions, it is possible to use a 2D plane to separate the two groups.

So Why Does This Work?

- Introduce the “kernel trick”
- The kernel trick relies on two things
  1. A transformation: $x \rightarrow \phi(x)$
  2. The dot product: $\phi(x_1) \cdot \phi(x_2)$

The Transformation

- The key to the kernel trick is **we do not need to know** $\phi(x)$.
- If we want to project our data into higher dimensions, $\phi(x)$ is just the map of the projection.

The Dot Product

- $\phi(x_1) \cdot \phi(x_2)$ is the dot product in the higher dimension
- How do we get this without knowing $\phi(x)$?
- That is the “trick” part of the kernel trick; by picking certain types of transformations (ϕ), we never actually have to project our data into the higher dimension—we can just calculate $\phi(x_1) \cdot \phi(x_2)$
- Since $\phi(x_1) \cdot \phi(x_2)$ turns up in the loss function we optimize, this shortcut allows us to calculate the loss in higher dimensions without actually projecting/transforming the data!

What Are Some Useful Kernels (Φ(x)·Φ(y)) Then

- Polynomial
- Gaussian
- Linear

Kernel	Mathematical form
linear	(x, y)
polynomial	$(\gamma(x, y) + r)^d$
RBF	$\exp(-\gamma\|x - y\|^2)$
sigmoid	$\tanh(\gamma(x, y) + r)$

DataScience@SMU

Scaling

SVMs Do Not Scale to Big Data

- While SVMs present a great resource for nonlinear problems, they suffer from $O(n^2)$ scaling.
- An SVM must store a matrix of all the dot products.
  - The SVM needs to find the “nearest” points—so each point must be compared to all others.
  - This is why the SVM is not typically used for “big data.”
    - About 50,000 samples or more is the typical limit
    - The limit is storing all the dot products in memory
    - Back of envelope calculation: 16 GB memory is approximately equal to 128,000 samples
    - 128,000/2 (Matrix is symmetric)
    - $64{,}000^2$ is about 16 gigabytes of memory at 32 bits per number

Why Can Other Algorithms Scale?

- SVM has to store a matrix of all the dot products ($n \times n$).
- Regression needs only the data to count the slope (n or less).
- Trees need only the data used to build the tree (n or less).
- Gaussian needs only the data distributions ($<<n$).
- Clustering needs distances ($n \times n$).
- SVM scales like a clustering problem—there are adaptations, but that is why we tend not to use SVM for larger datasets.

Other SVM Issues

- Regression is not a natural outgrowth.
- Probabilities are not a natural outgrowth.
  - SVM assigns by hard limits: either class -1 or class 1
  - We do get “confidence” from how far from the boundary
    - **No probability!**

Summary

- SVMs are powerful tools for small and medium size datasets for classification.
- Regression and probability are afterthoughts.
  - Adaptations exist, but better, more reliable results with other algorithms
- SVMs suffer scaling issues.
- SVMs need to guard against overfitting in high dimensions.

DataScience@SMU