Support Vector Machines I

Dr. Chelsea Parlett-Pelleriti

Outline

- Maximal Margin Classifiers
- Support Vector Classifiers
- Support Vector Machines: Kernels and The Kernel Trick
- Common Kernels
  - Polynomial
  - Radial Basis Function

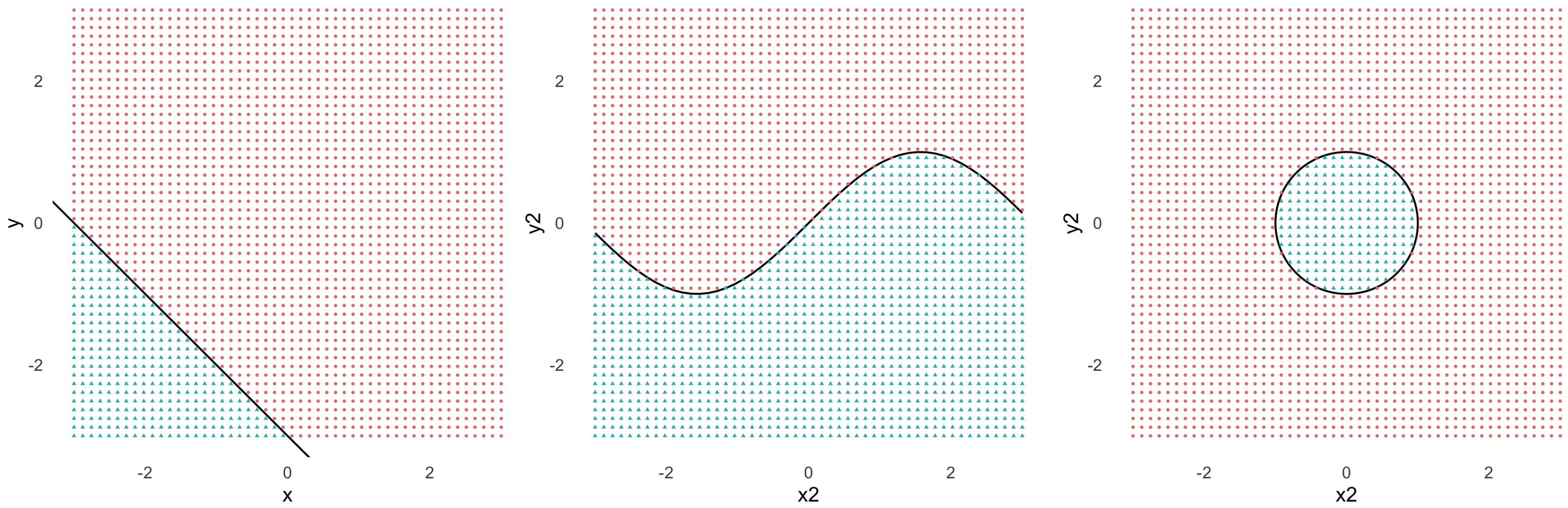
Hyperplanes

Flat: the hyperplane is not curved, it increases/decreases constantly in each direction

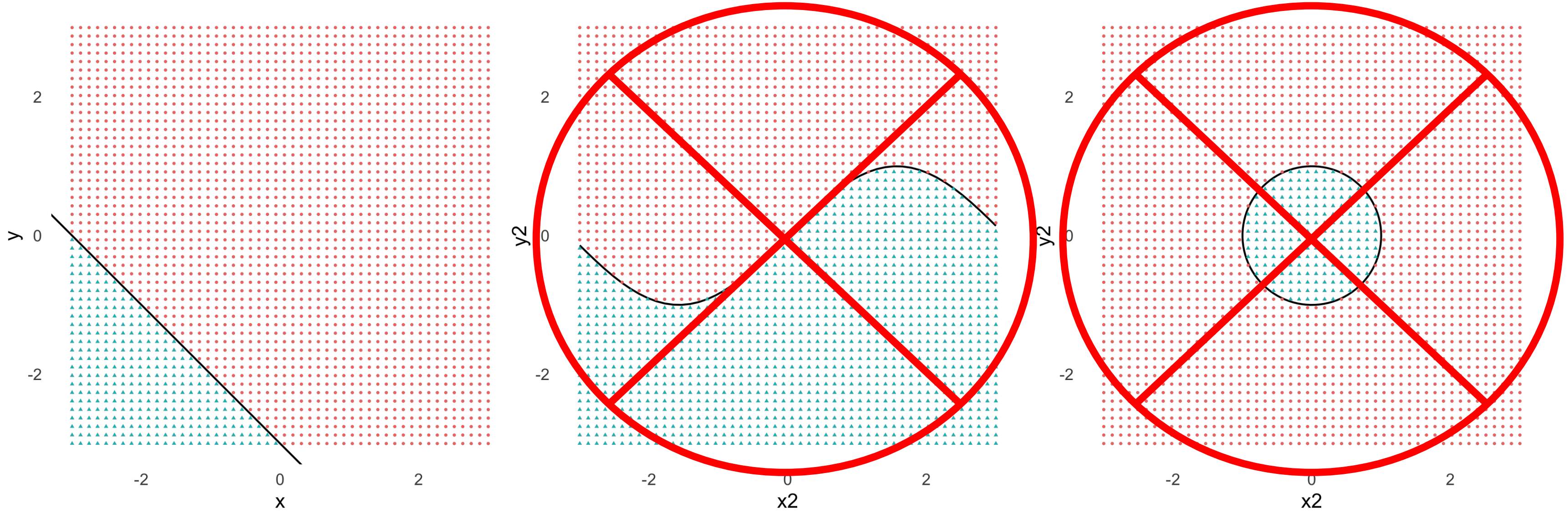
Affine: the hyperplane doesn't need to pass through the origin

Subspace: a subset of vectors in a larger vector space (closure under addition and scalar multiplication; here, a hyperplane with n-1 dimensions)

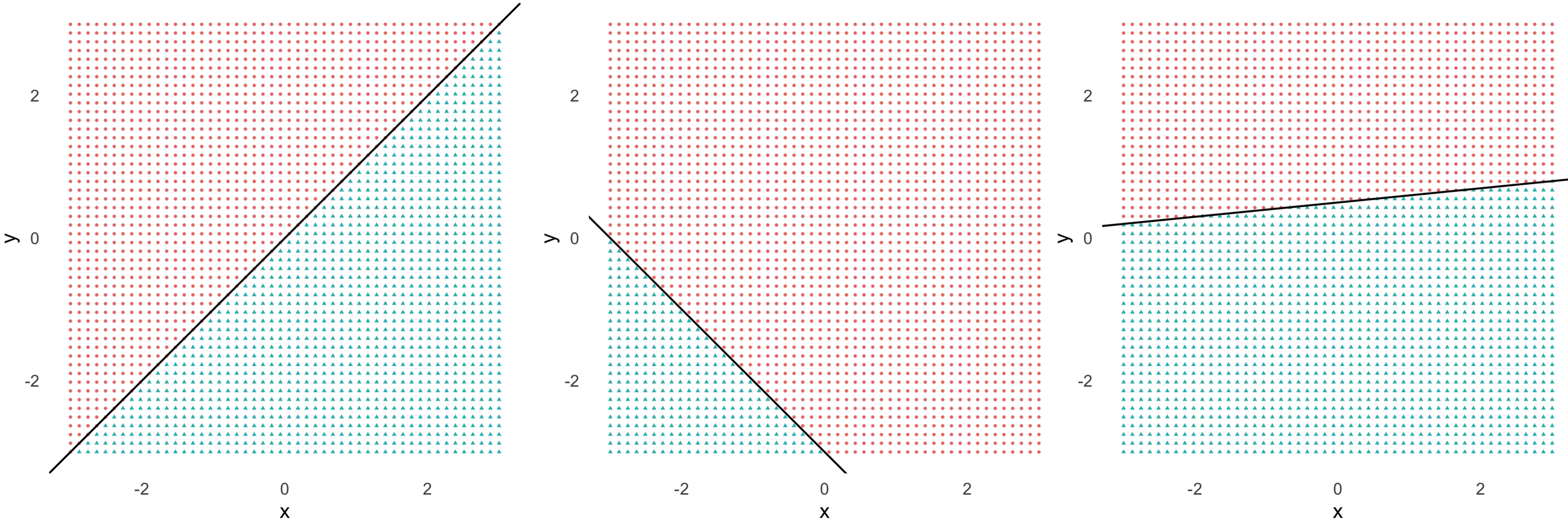
Quick Check: Which of these are flat affine subspaces?



Quick Check: Which of these are flat affine subspaces?

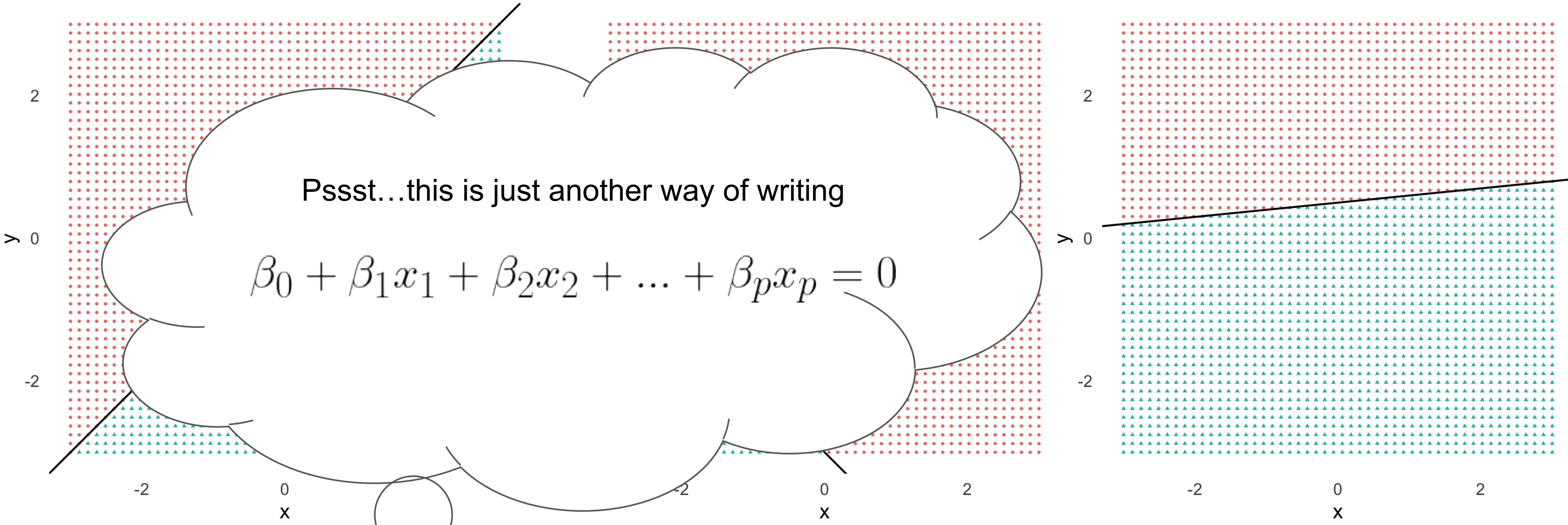


Hyperplanes Divide Spaces in Half



$$\mathbf{w}^T x + b = 0$$

Hyperplanes Divide Spaces in Half

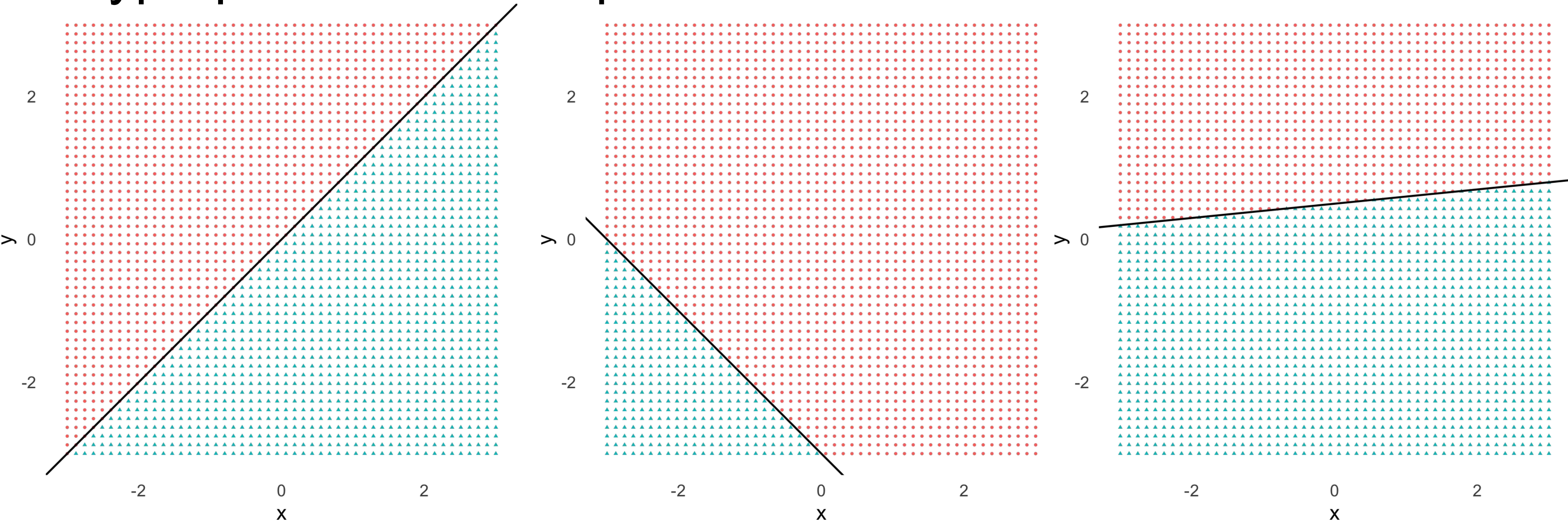


Pssst…this is just another way of writing

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p = 0$$

$$\mathbf{w}^T x + b = 0$$

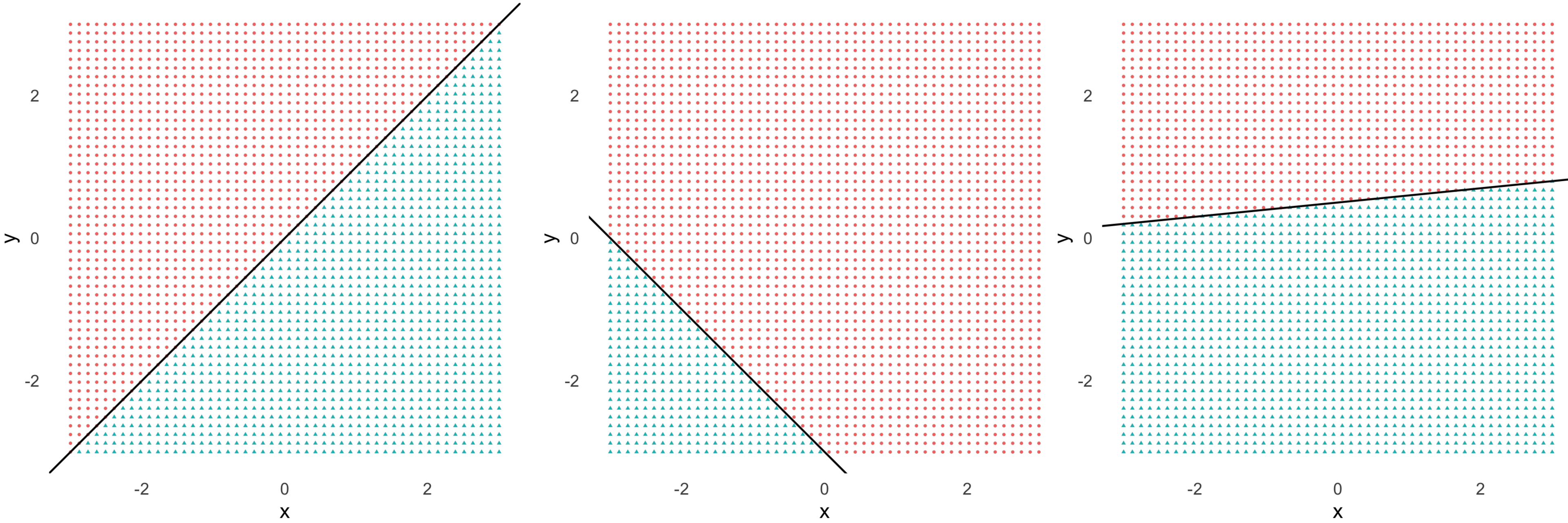
Hyperplanes Divide Spaces in Half



$$\mathbf{w}^T x_n + b > 0 \text{ if } t_n = 1$$

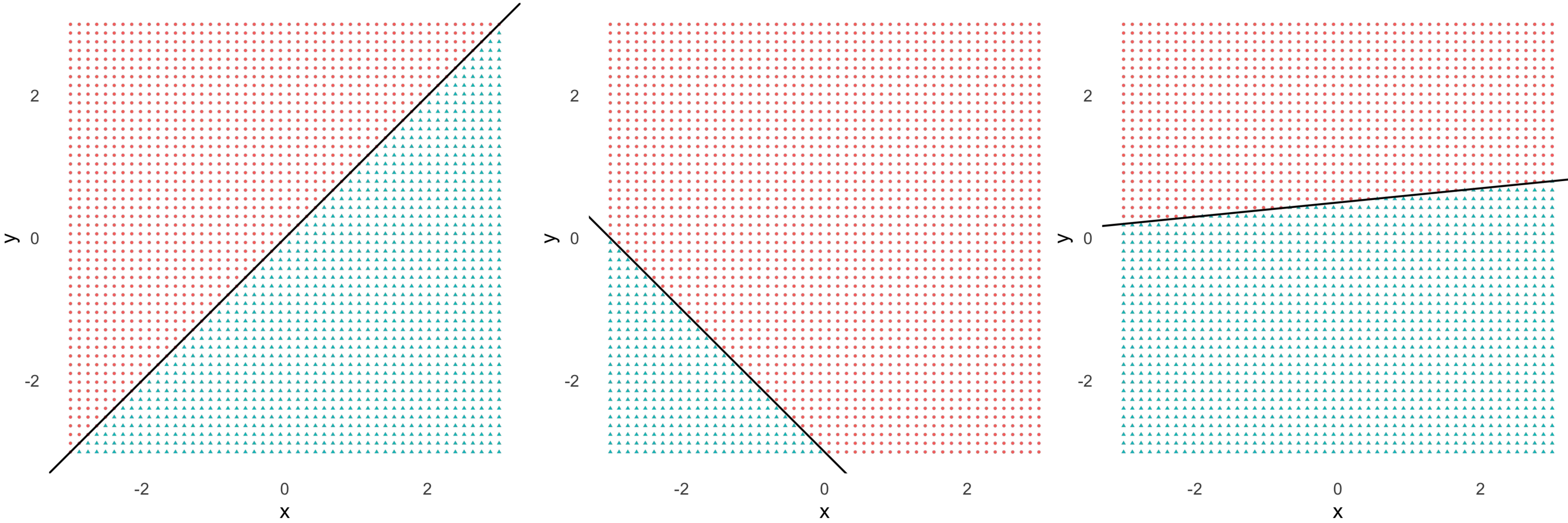
$$\mathbf{w}^T x_n + b < 0 \text{ if } t_n = -1$$

Hyperplanes Divide Spaces in Half



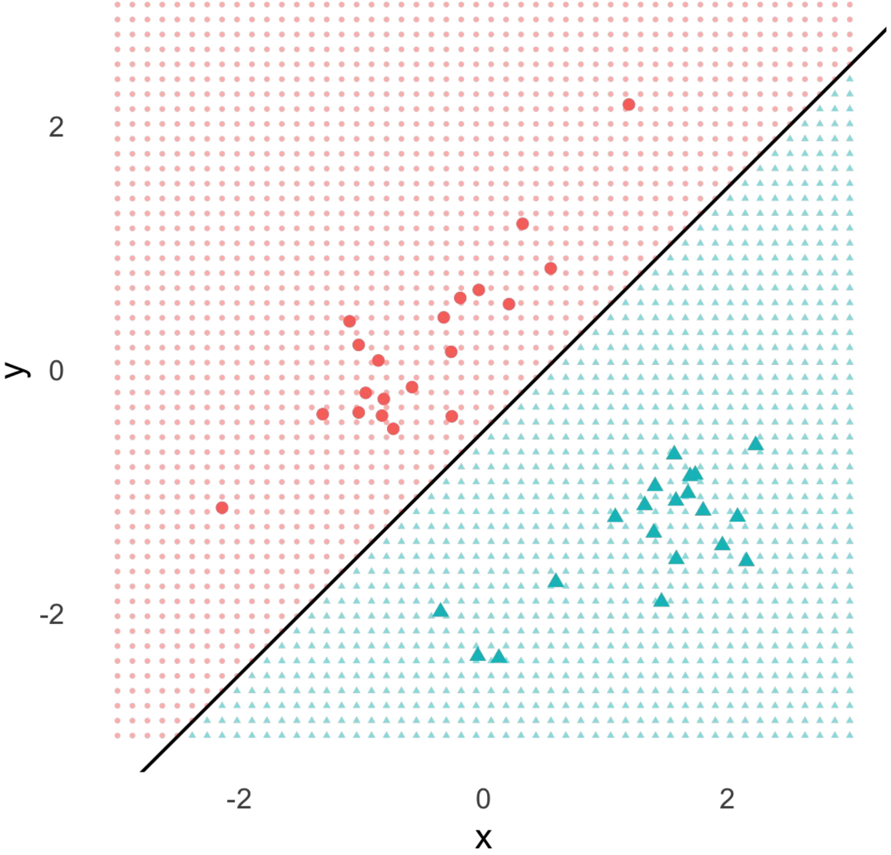
$$t_n(\mathbf{w}^T x_n + b) > 0$$

Hyperplanes Divide Spaces in Half

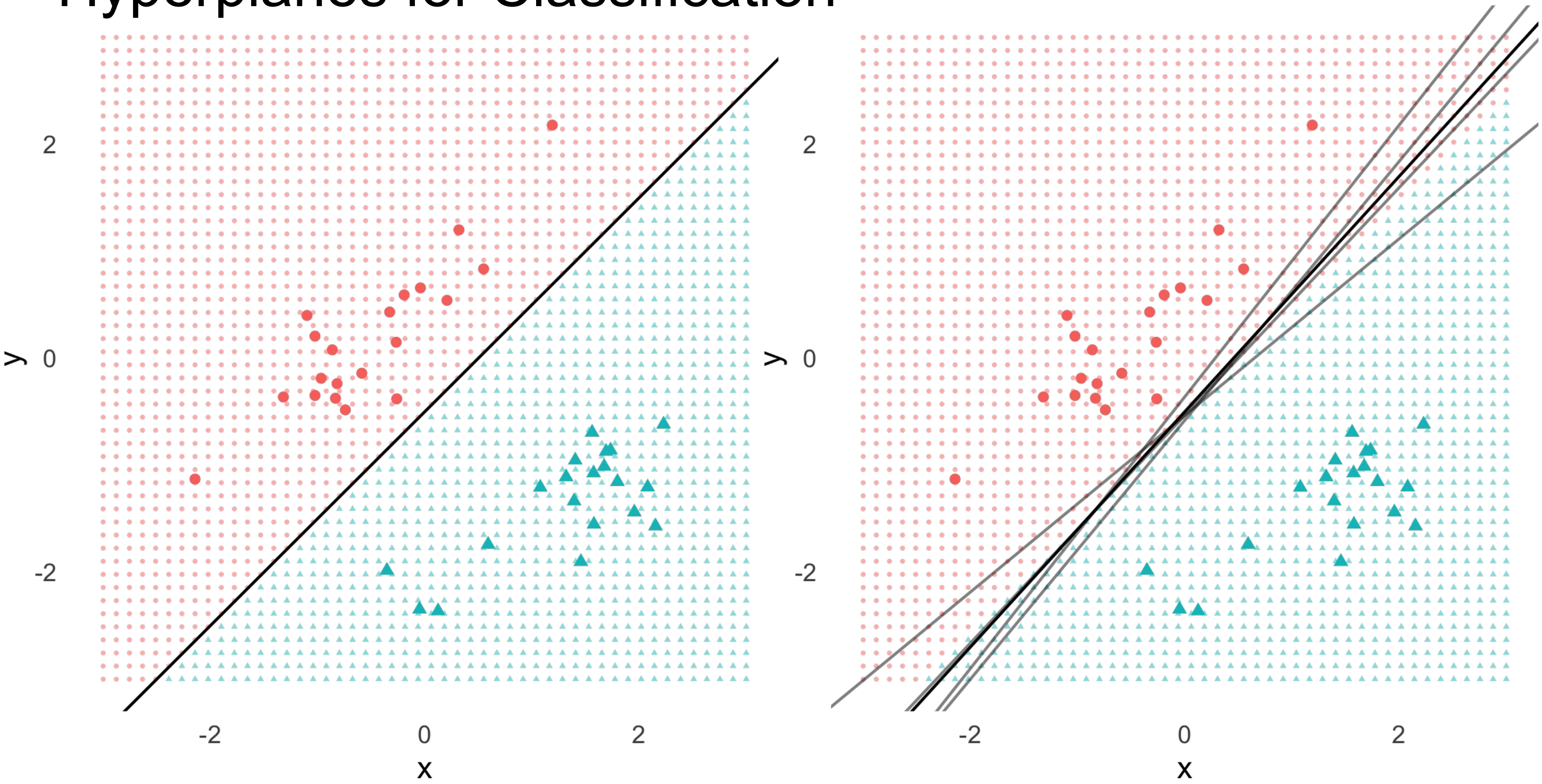


We can use these hyperplanes to classify data!

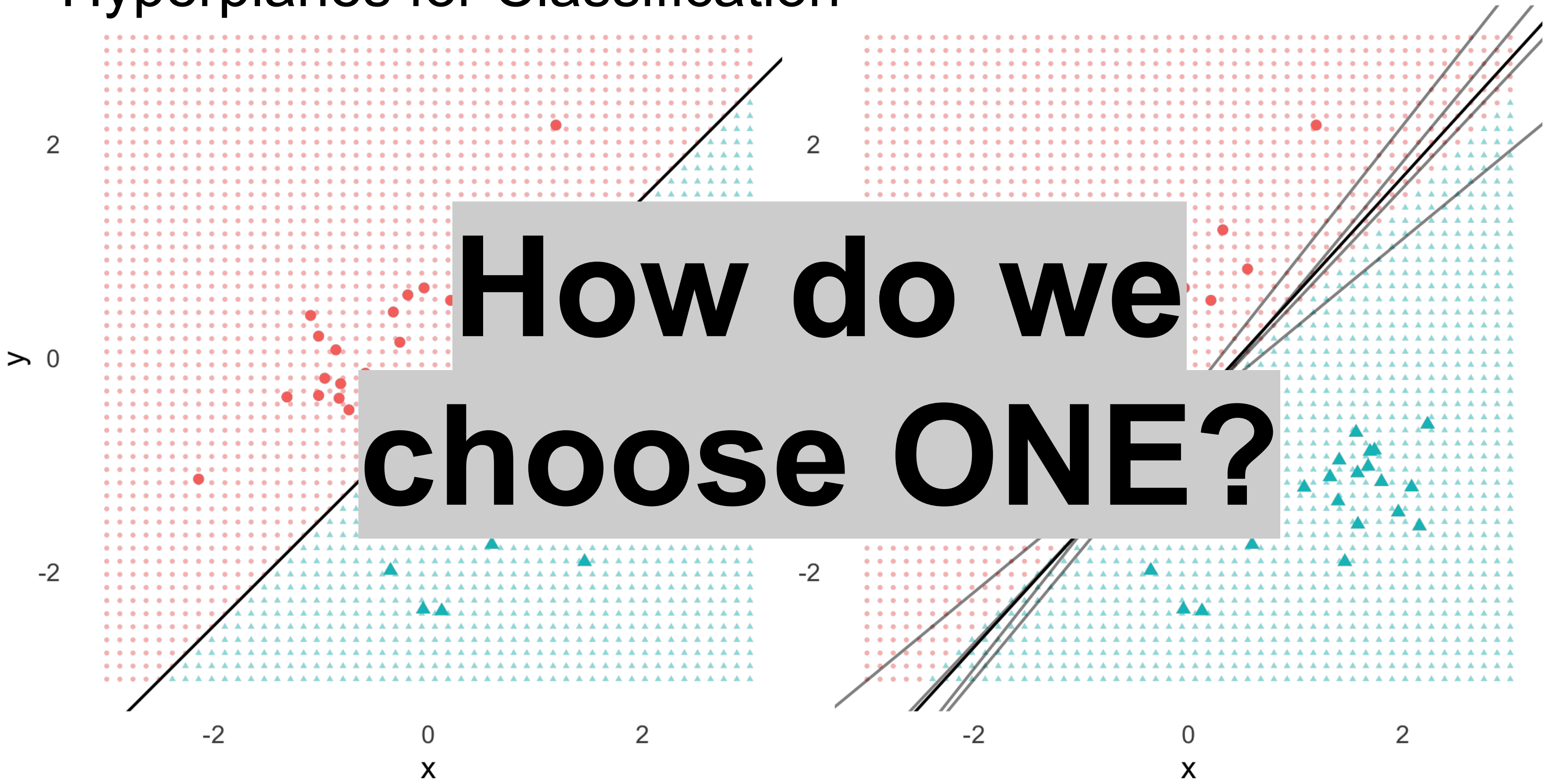
Hyperplanes for Classification



Hyperplanes for Classification



Hyperplanes for Classification



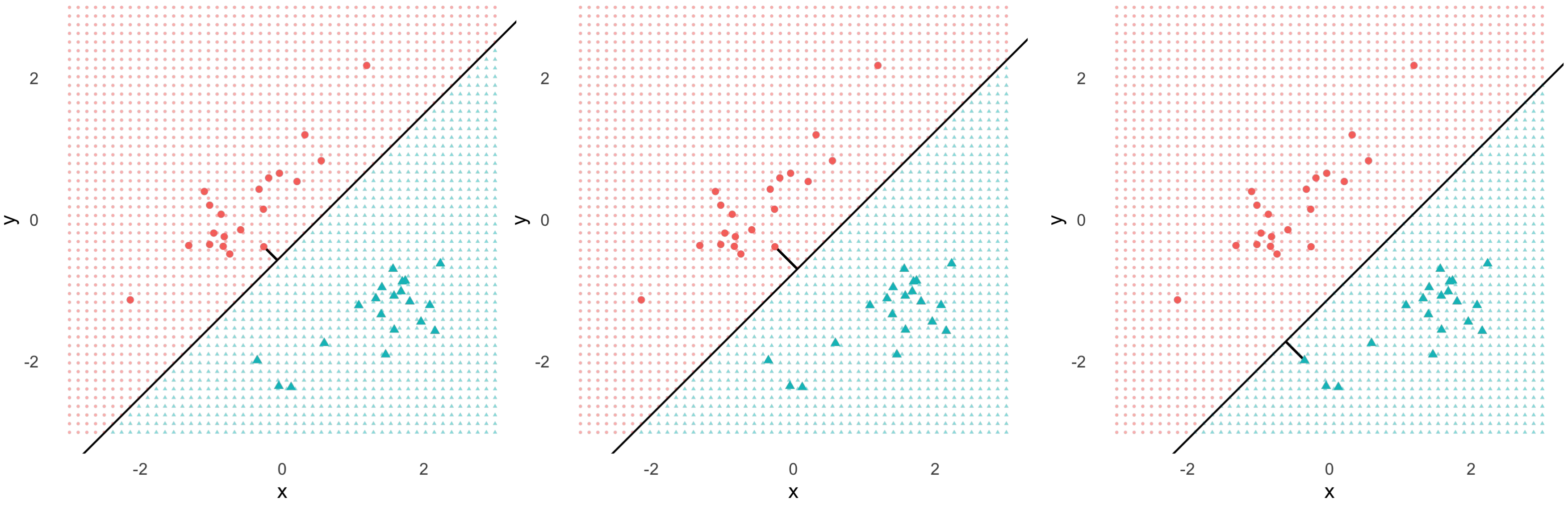
How do we choose ONE?

Maximal Margin Classifier

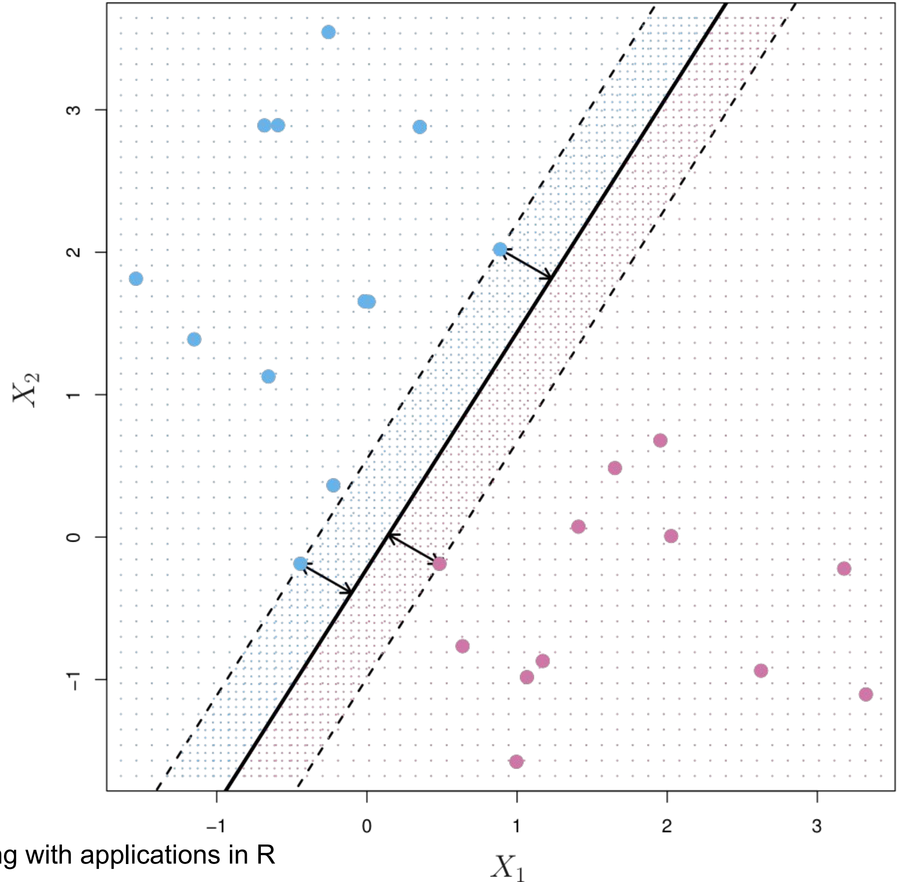
Maximal Margin Classifier

Problem: If a separating hyperplane exists, there will be ∞

Solution: Choose the one that's *furthest* from the training examples



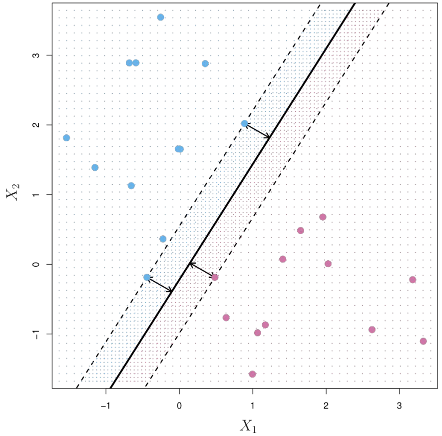
Maximal Margin Classifier



Images from an Introduction to Statistical Learning with applications in R

Maximal Margin Classifier Math

$$\underbrace{\frac{t_n\left(w^T x_n + b\right)}{\|w\|}}_{\substack{\text{distance btw} \\ x_n \text{ and line}}}$$



$$\underbrace{\underset{\mathbf{w},b}{\arg\max}\left\{\frac{1}{\|\mathbf{w}\|}\min_n\left[t_n\left(\mathbf{w}^T x_n + b\right)\right]\right\}}_{\text{maximize the minimum distance btw hyperplane and point}}$$

$$\text{subject to: } t_n\left(\mathbf{w}^T x_n + b\right) \geq 1$$

Easier to minimize

$$\underset{\mathbf{w},b}{\arg\min}\ \frac{1}{2}\|\mathbf{w}\|^2$$

Constrained Optimization

$$f(x, y) = 2x + y$$
$$g(x, y) = x^2 + y^2 = 1$$

Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$

1

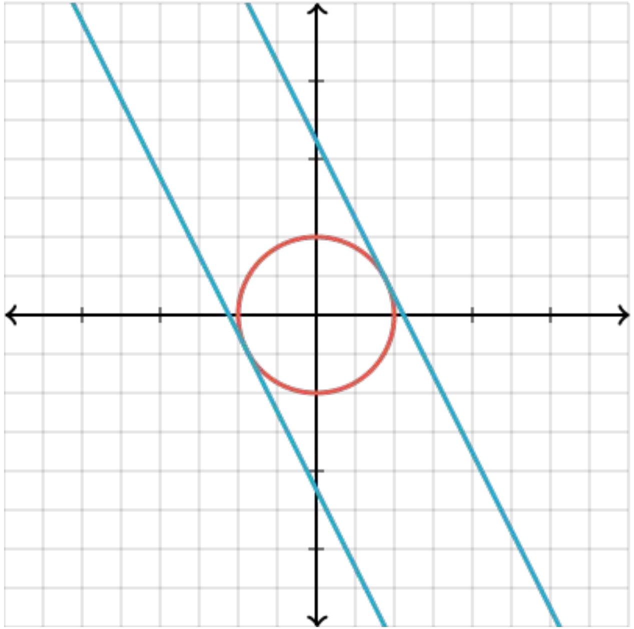
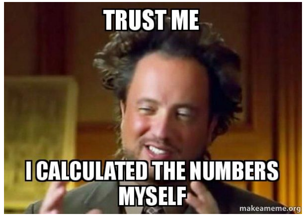


Image from Khan Academy's Lagrange Multiplier and Constrained Optimization Page: https://www.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/constrained-optimization/a/lagrange-multipliers-single-constraint

Constrained Optimization



Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$

2

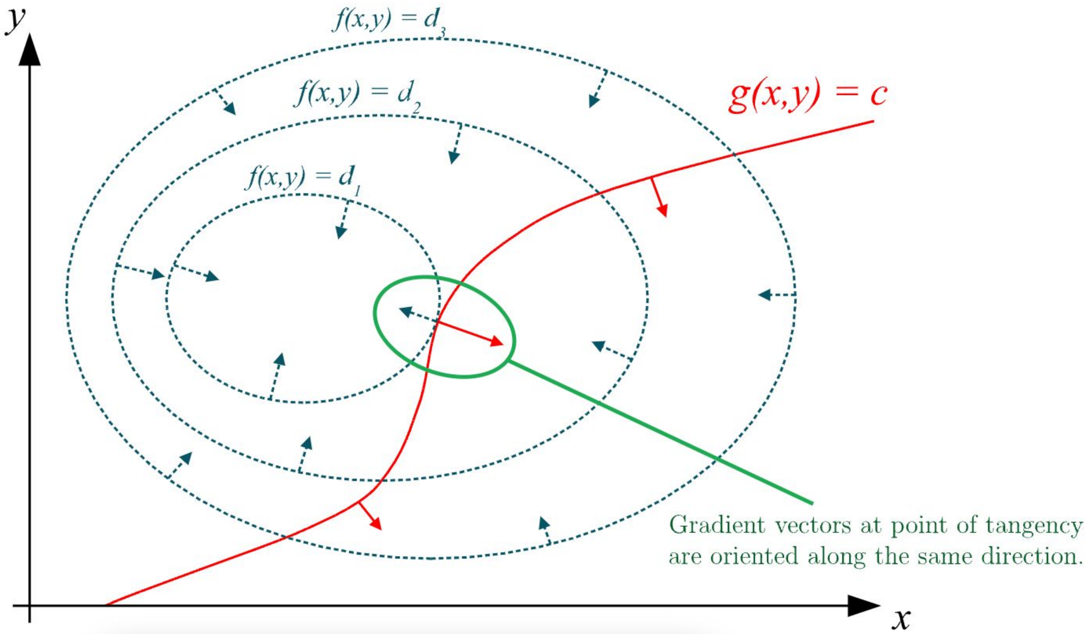


Image from Khan Academy's Lagrange Multiplier and Constrained Optimization Page: https://www.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/constrained-optimization/a/lagrange-multipliers-single-constraint

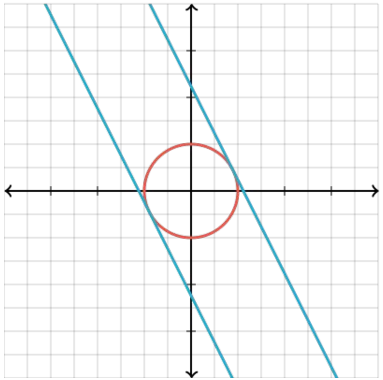
Constrained Optimization

Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$

3 $$\nabla f(x_0, y_0) = \lambda \nabla g(x_0, y_0)$$

Constrained Optimization

Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$



$$f(x, y) = 2x + y$$

$$g(x, y) = x^2 + y^2 = 1$$

3

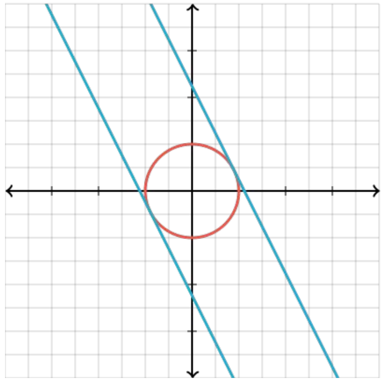
$$\nabla f(x_0, y_0) = \lambda \nabla g(x_0, y_0)$$

$$\nabla f(x, y) = \begin{bmatrix} \frac{\partial}{\partial x} 2x + y \\ \frac{\partial}{\partial y} 2x + y \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$\nabla g(x, y) = \begin{bmatrix} \frac{\partial}{\partial x} x^2 + y^2 \\ \frac{\partial}{\partial y} x^2 + y^2 \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

Constrained Optimization

Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$



$f(x, y) = 2x + y$

$g(x, y) = x^2 + y^2 = 1$

3 $$\nabla f(x_0, y_0) = \lambda \nabla g(x_0, y_0)$$

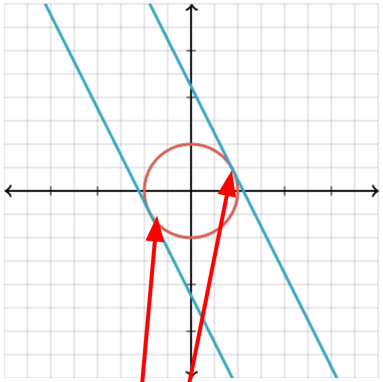
$$\nabla f(x, y) = \begin{bmatrix} \frac{\partial}{\partial x} 2x + y \\ \frac{\partial}{\partial y} 2x + y \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$\nabla g(x, y) = \begin{bmatrix} \frac{\partial}{\partial x} x^2 + y^2 \\ \frac{\partial}{\partial y} x^2 + y^2 \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

$$\boxed{\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \lambda_0 \begin{bmatrix} 2x_0 \\ 2y_0 \end{bmatrix}}$$

Constrained Optimization

Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$



3 $$\nabla f(x_0, y_0) = \lambda \nabla g(x_0, y_0)$$

$$f(x, y) = 2x + y$$

$$g(x, y) = x^2 + y^2 = 1$$

$$\nabla f(x, y) = \begin{bmatrix} \frac{\partial}{\partial x} 2x + y \\ \frac{\partial}{\partial y} 2x + y \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$\nabla g(x, y) = \begin{bmatrix} \frac{\partial}{\partial x} x^2 + y^2 \\ \frac{\partial}{\partial y} x^2 + y^2 \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

$$\boxed{\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \lambda_0 \begin{bmatrix} 2x_0 \\ 2y_0 \end{bmatrix}}$$

$$f(x,y) = 2x + y$$
$$g(x,y) = x^2 + y^2 = 1$$

Constrained Optimization

Optimize $f(x,y,z)$ subject to $g(x,y,z) = k$

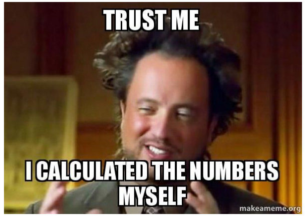
4

$$\mathcal{L}(x,y,\lambda) = f(x,y) - \lambda(g(x,y) - c)$$

$$\mathcal{L}(x,y,\lambda) = 2x + y - \lambda(x^2 + y^2 - 1)$$

For example

Constrained Optimization



Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$

4

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda(g(x, y) - c)$$

$$\mathcal{L}_\lambda(x, y, \lambda) = \frac{\partial \mathcal{L}}{\partial \lambda} = 0 - (g(x, y) - c) = -g(x, y) + c$$

$$\mathcal{L}_\lambda(x, y, \lambda) = -g(x, y) + c = 0 \text{ when } \mathrm{g(x,y)} = \mathrm{c}$$

Constrained Optimization

Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$

4

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda(g(x, y) - c)$$

$$\mathcal{L}_x(x, y, \lambda) = 0$$

$$\frac{\partial}{\partial x}(f(x, y) - \lambda(g(x, y) - c)) = 0$$

$$f_x(x, y) - \lambda g_x(x, y) = 0$$

$$f_x(x, y) = \lambda g_x(x, y)$$

Constrained Optimization

Optimize $f(x, y, z)$ subject to $g(x, y, z) = k$

4

$$\nabla \mathcal{L} = \begin{bmatrix} \dfrac{\partial}{\partial x}(2x + y - \lambda(x^2 + y^2 - 1)) \\ \\ \dfrac{\partial}{\partial y}(2x + y - \lambda(x^2 + y^2 - 1)) \\ \\ \dfrac{\partial}{\partial \lambda}(2x + y - \lambda(x^2 + y^2 - 1)) \end{bmatrix} = \begin{bmatrix} 2 - 2\lambda x \\ 1 - 2\lambda y \\ -x^2 - y^2 + 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

SVMs as Constrained Optimization

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$

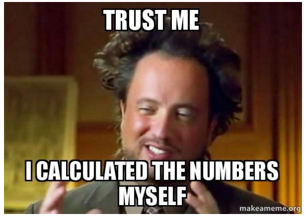
$$t_n(w^T x_n + b) \geq 1 \text{ for all n}$$

$$\mathcal{L}(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \left\{t_n(w^T x_n + b) - 1\right\}$$

SVMs as Constrained Optimization

$$\underset{\mathbf{w},b}{\arg\min}\ \frac{1}{2}\|\mathbf{w}\|^2$$

$$t_n(w^T x_n + b) \geq 1 \ \text{ for all n}$$



$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} \alpha_n \left\{ t_n(w^T x_n + b) - 1 \right\}$$

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n x_n$$

$$0 = \sum_{n=1}^{N} a_n t_n$$

$$L(\tilde{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} \underbrace{a_n a_m}_{\text{lagrange multipliers}} \overbrace{t_n t_m}^{\text{targets}} \underbrace{x_n \cdot x_m}_{\text{dot product between two vectors}}$$

$$a_n \geq 0, \ \text{ for } n = 1, \ldots, N$$

$$\sum_{n=1}^{N} a_n t_n = 0$$

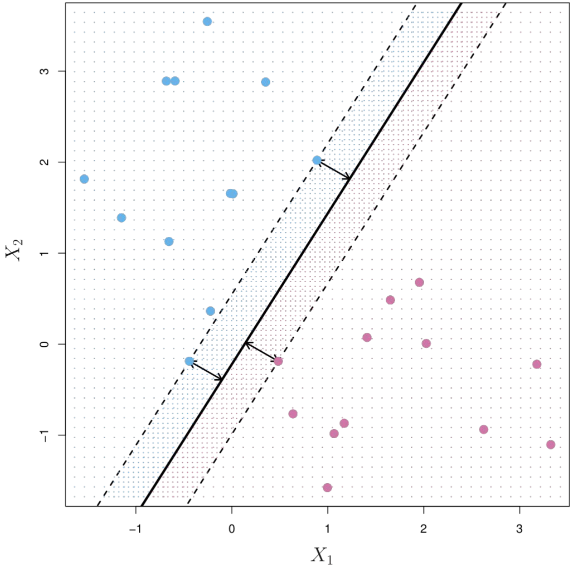
SVMs as Constrained Optimization

$$\underset{\mathbf{w},b}{\arg\min}\ \frac{1}{2}\|\mathbf{w}\|^2$$

$$t_n(w^T x_n + b) \geq 1 \quad \text{for all n}$$

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} \alpha_n \left\{ t_n(w^T x_n + b) - 1 \right\}$$

$$\overbrace{y(\mathbf{w}) = \sum_{n=1}^{N} a_n t_n \ \ x \cdot x_n \ \ + b}^{\text{formula to classify new data point } x}$$



$$a_n \geq 0$$

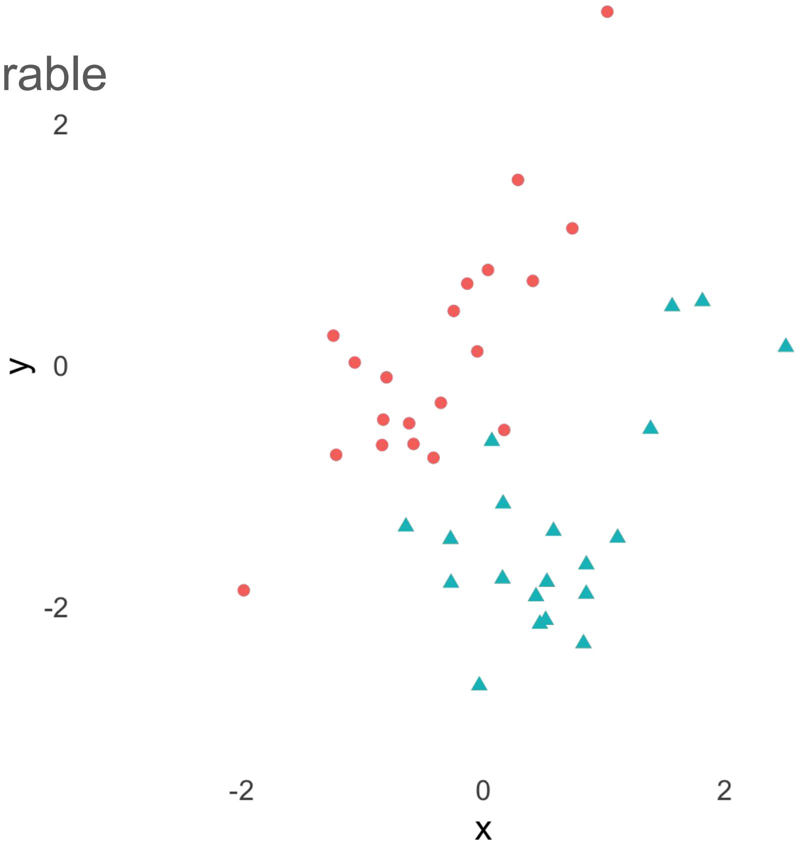
$$t_n y(x_n) - 1 \geq 0$$

2 $$a_n \left\{ t_n y(x_n) - 1 \right\} = 0$$

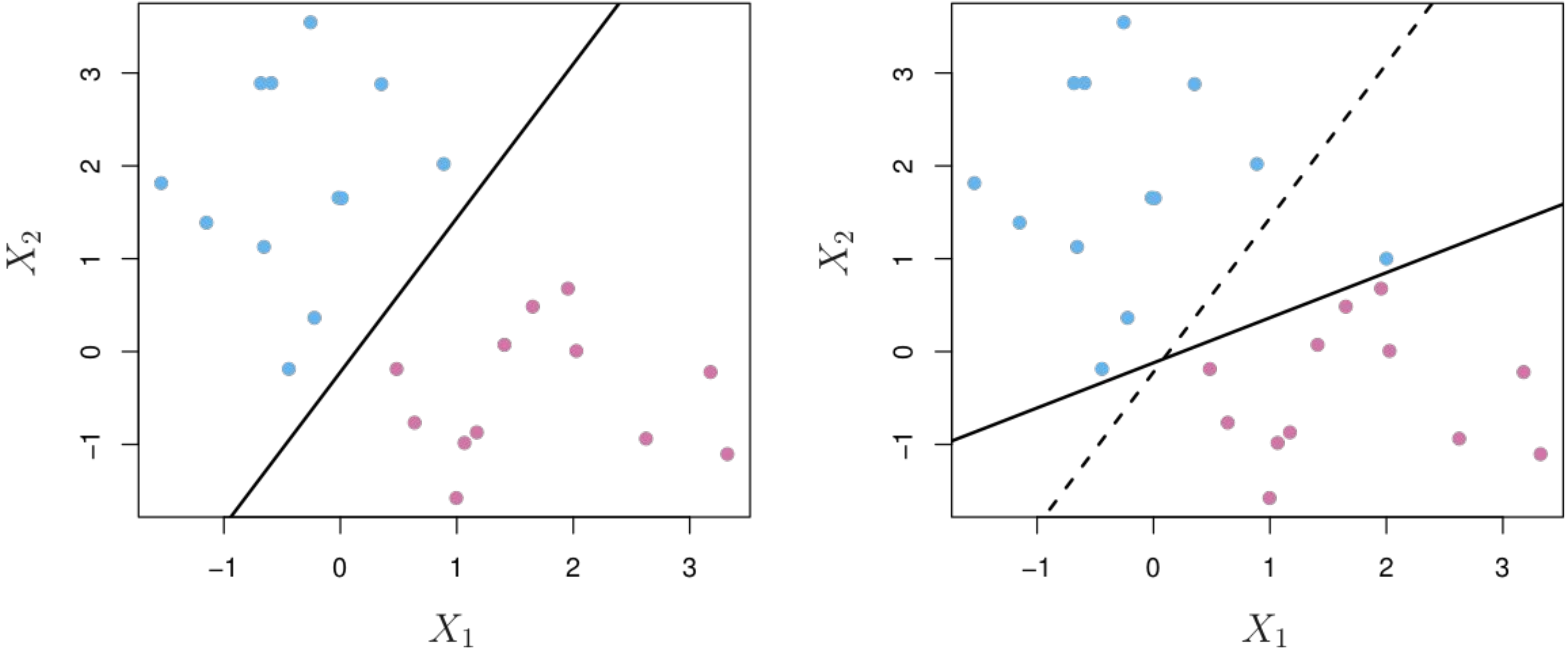
Support Vector Classifier

Support Vector Classifier

Problem: Non-Separable

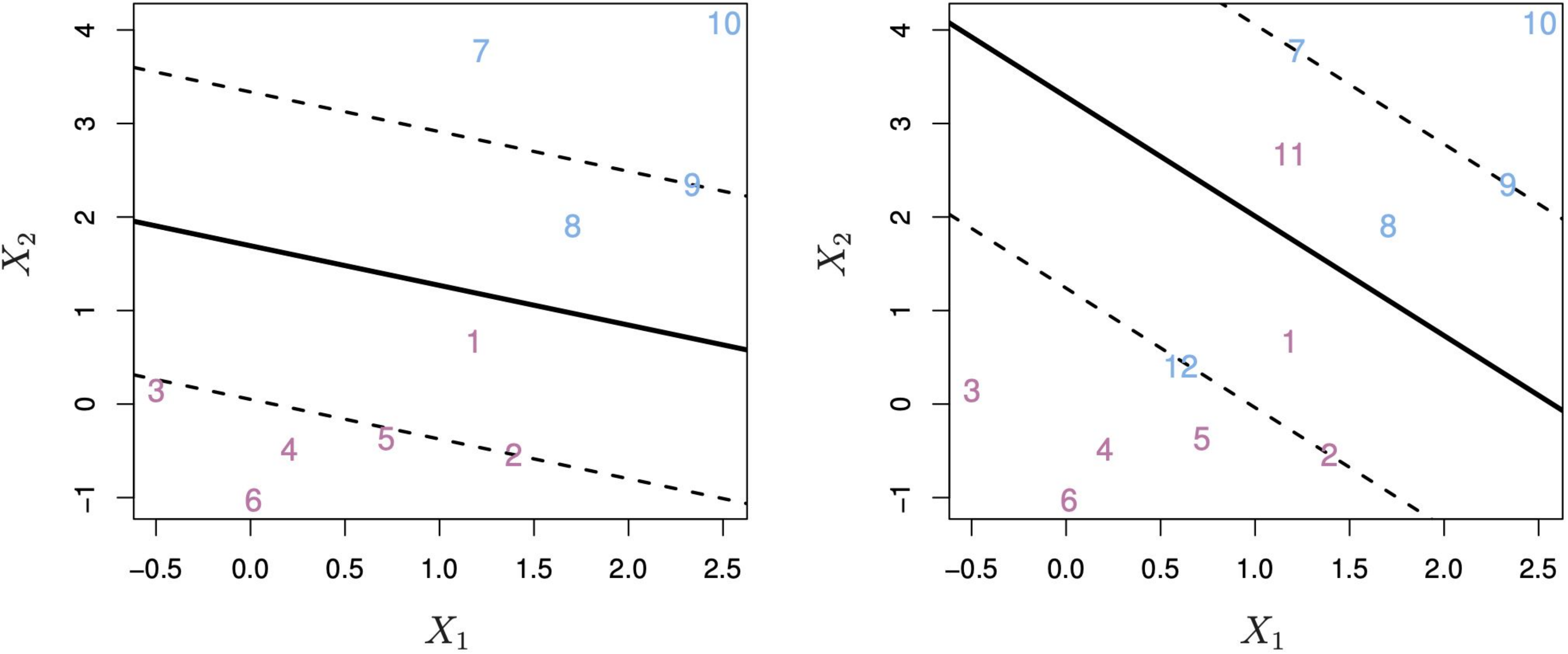


Support Vector Classifier



Images from an Introduction to Statistical Learning with applications in R

Support Vector Classifier



Images from an Introduction to Statistical Learning with applications in R

Support Vector Classifier

Before:

$$\text{minimize } \frac{1}{2}\|\mathbf{w}\|^2$$

After:

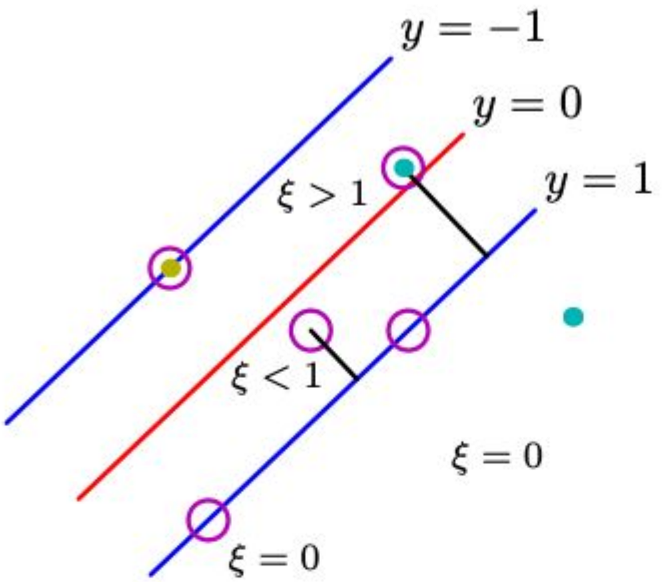
$$\text{minimize } C\sum_{n=1}^{N}\xi_n + \frac{1}{2}\|\mathbf{w}\|^2$$

Before:

$$t_n(w^T x_n + b) \geq 1$$

After:

$$t_n(w^T x_n + b) \geq 1 - \xi_n$$

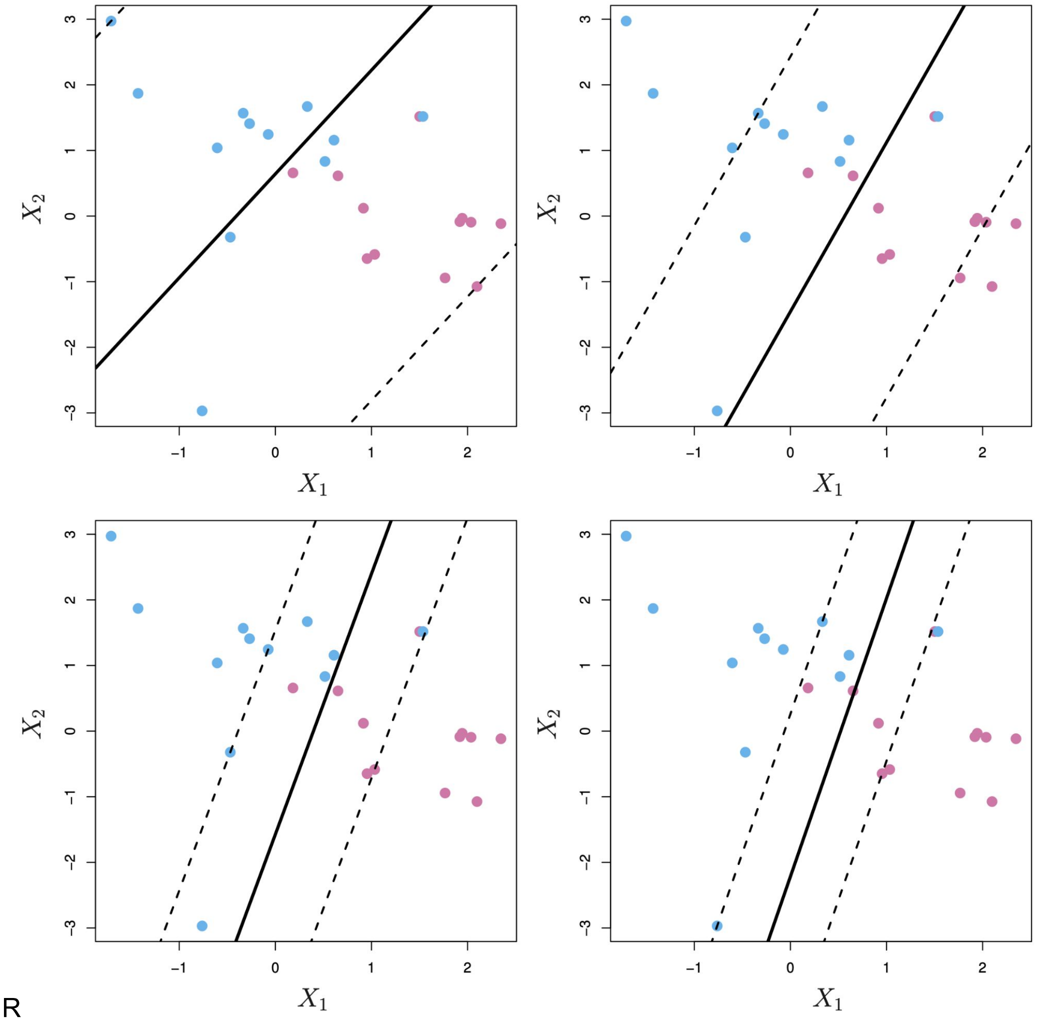


Support Vector Classifier

C: How much error to allow

Lower C: prone to overfitting

Higher C: prone to underfitting



Images from an Introduction to Statistical Learning with applications in R

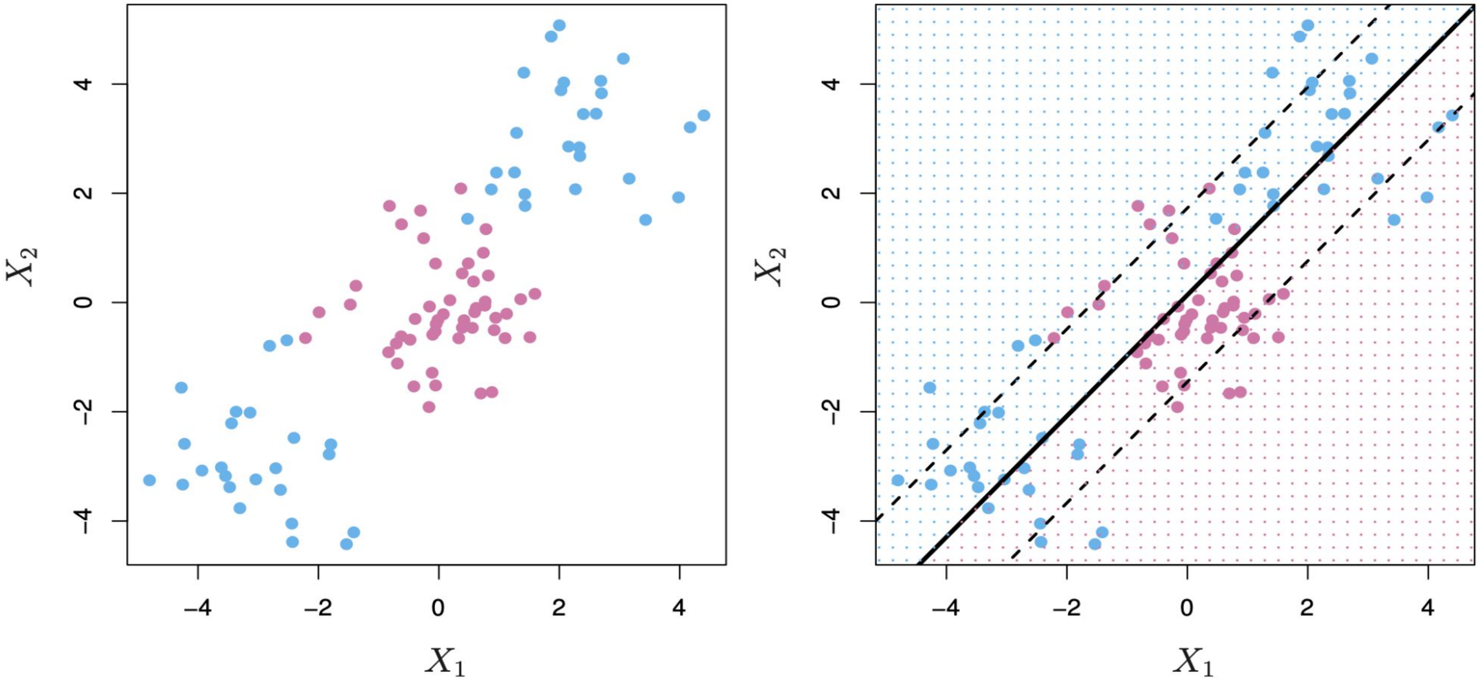
Support Vector Machines II

Dr. Chelsea Parlett-Pelleriti

Support Vector **Machines**

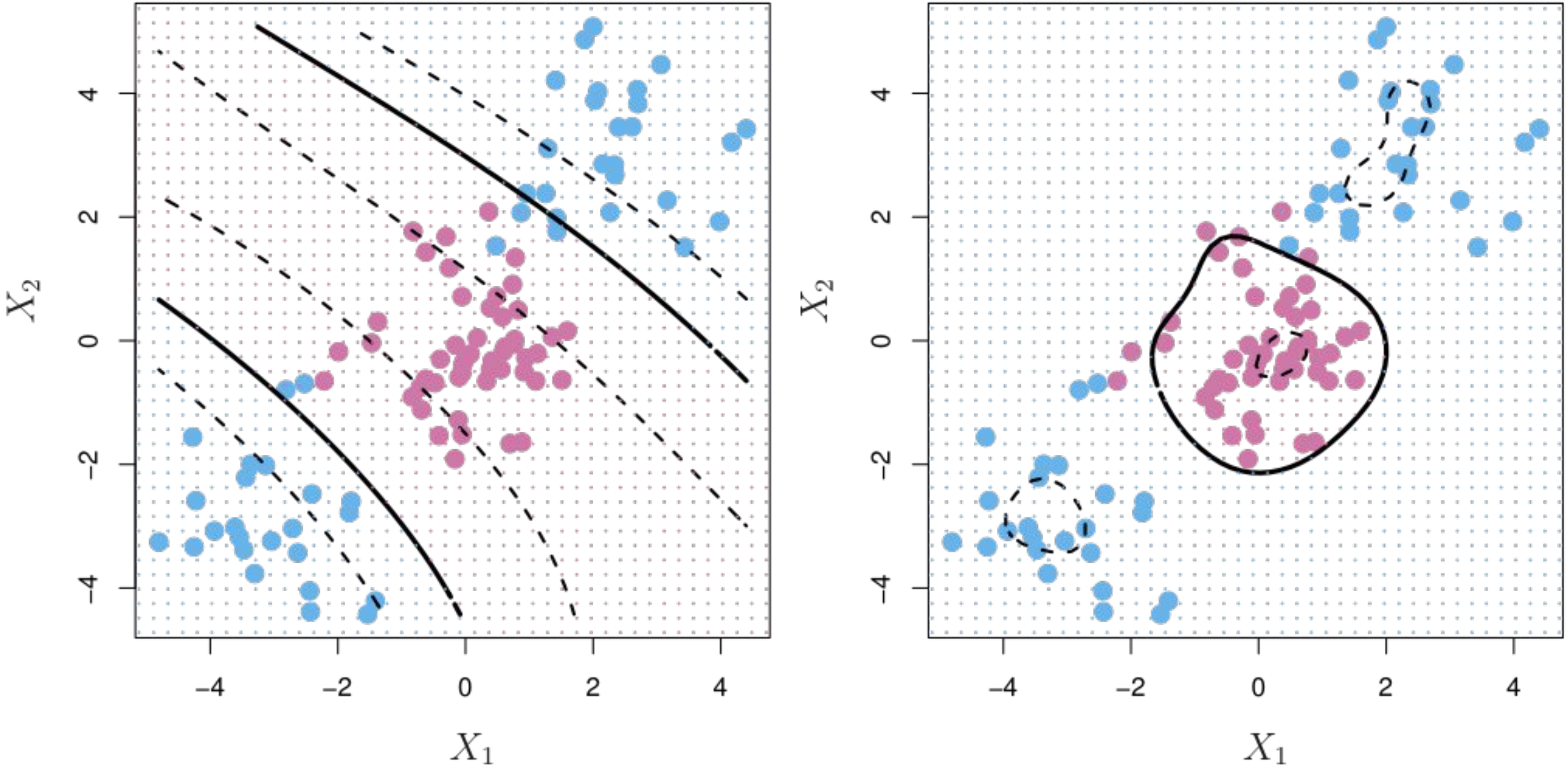
Support Vector Machines

New problem: not linearly separable even with error



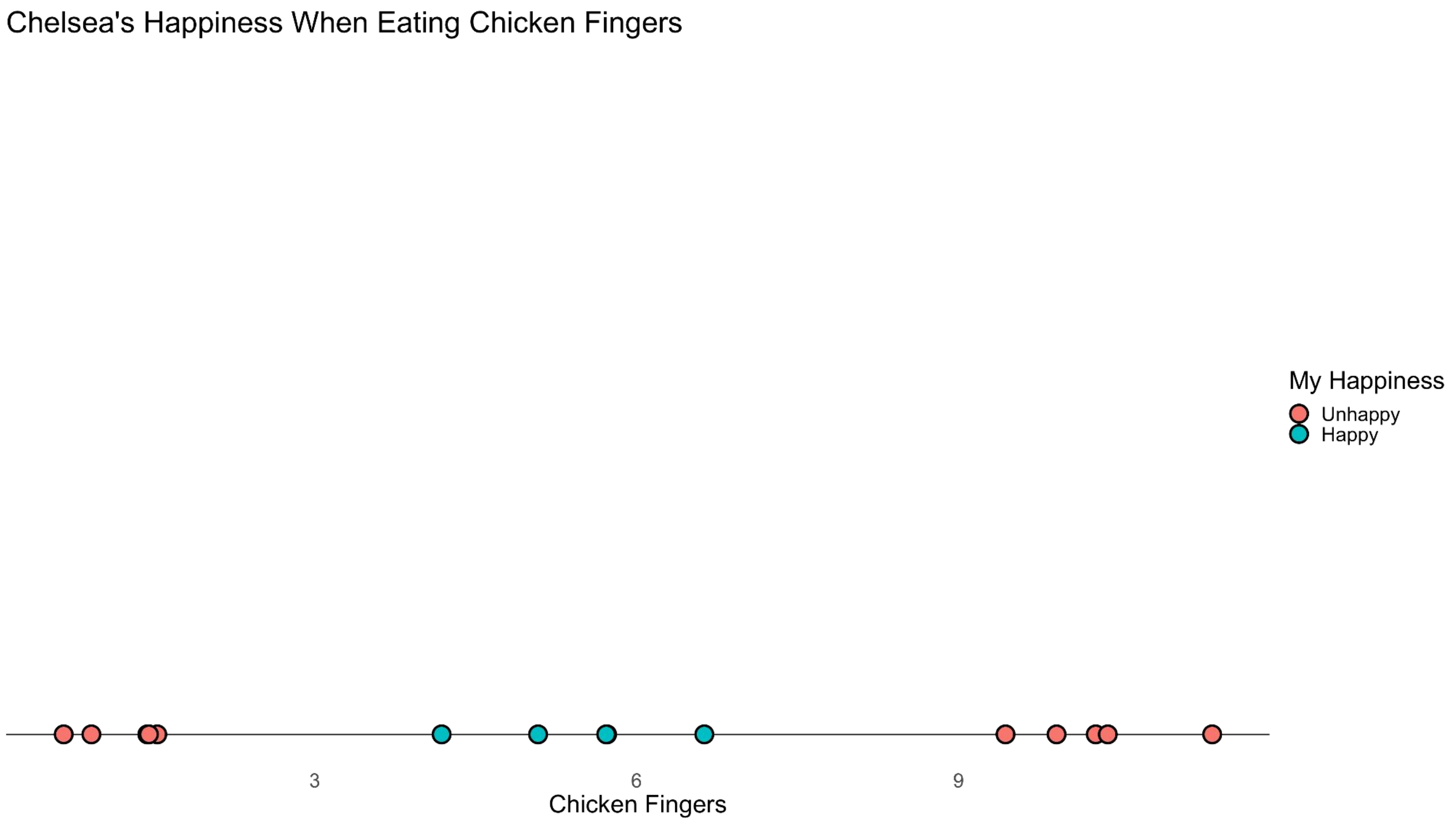
Images from an Introduction to Statistical Learning with applications in R

The Kernel Trick

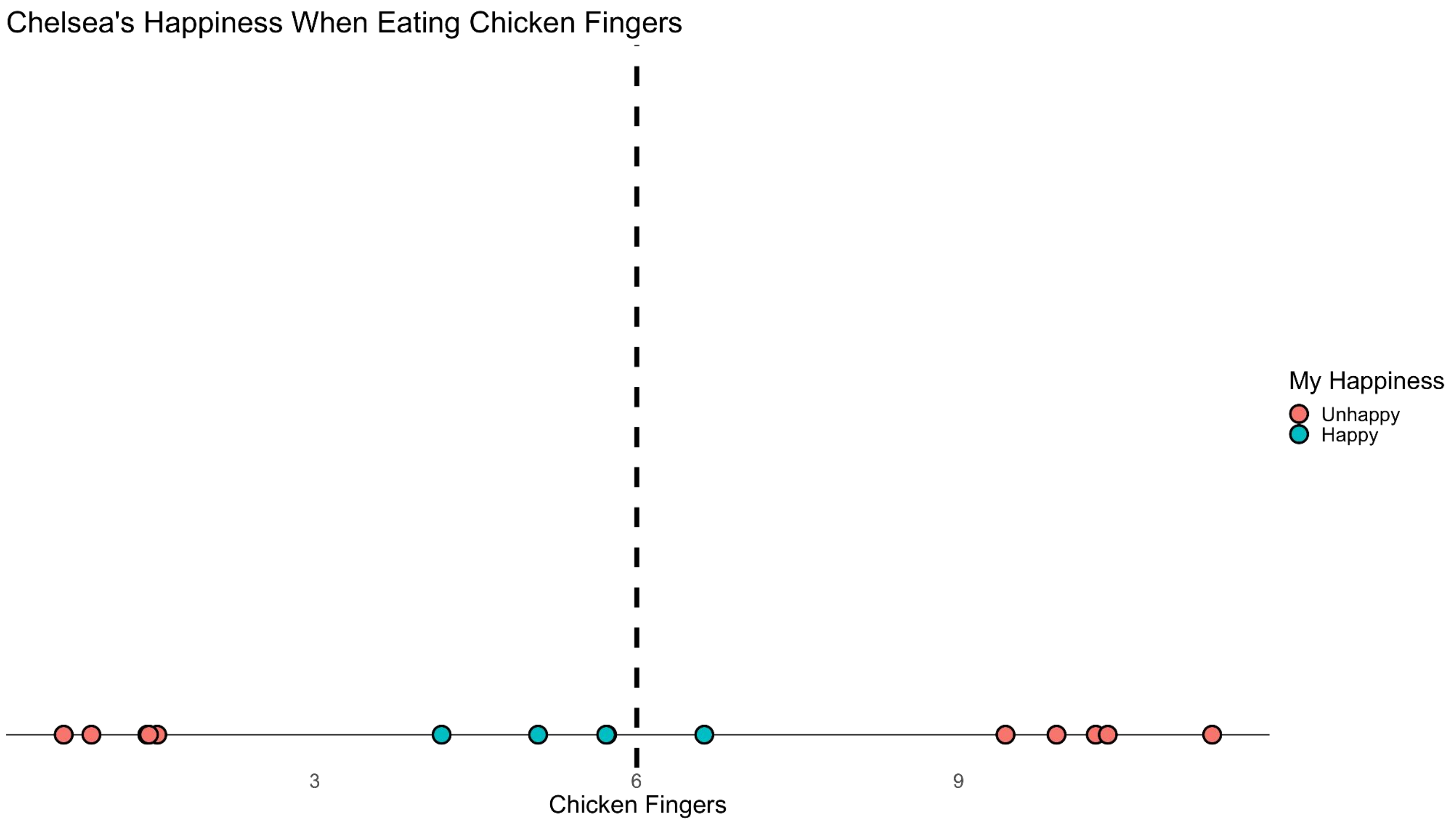


Images from an Introduction to Statistical Learning with applications in R

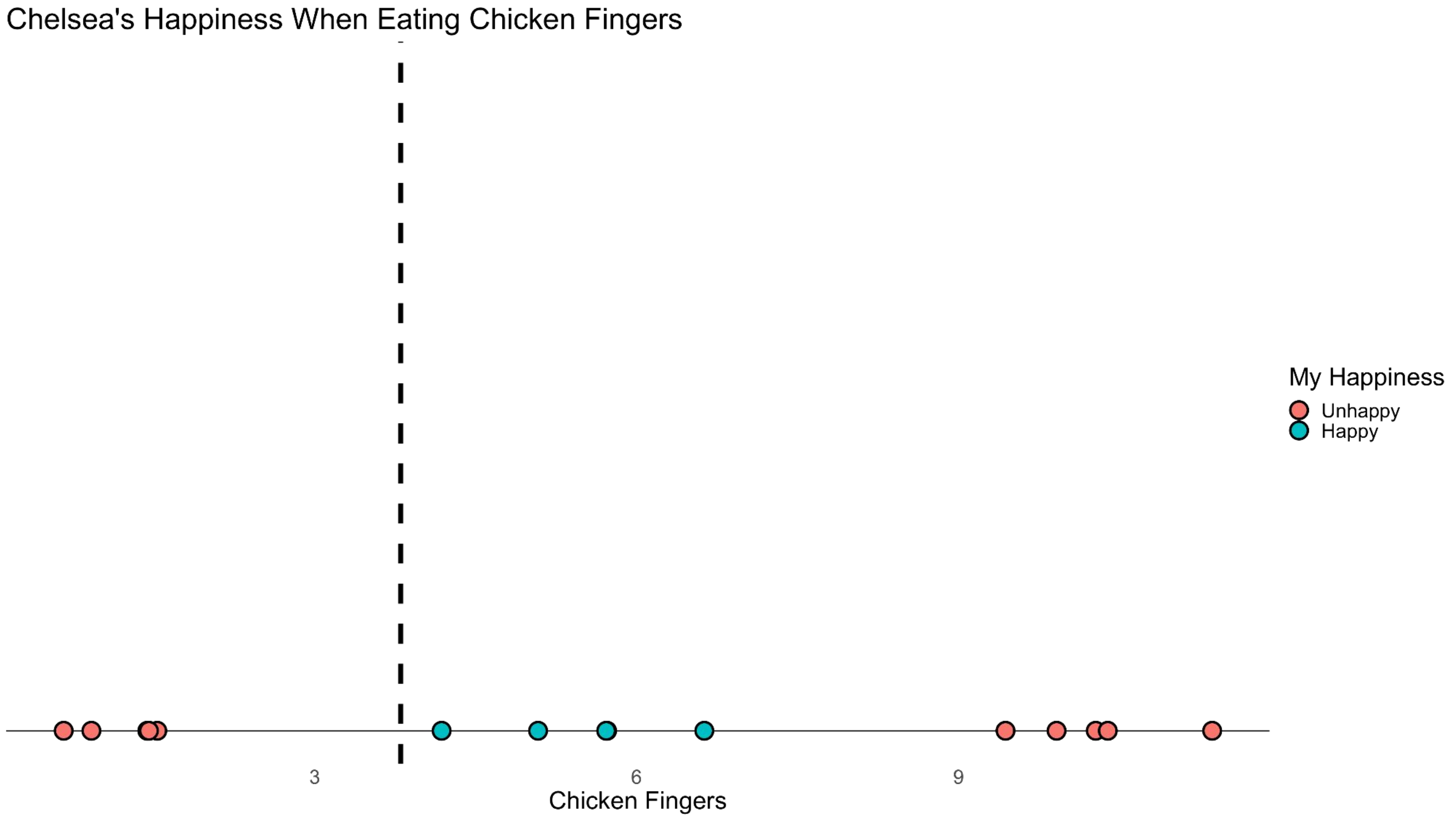
The Kernel Trick



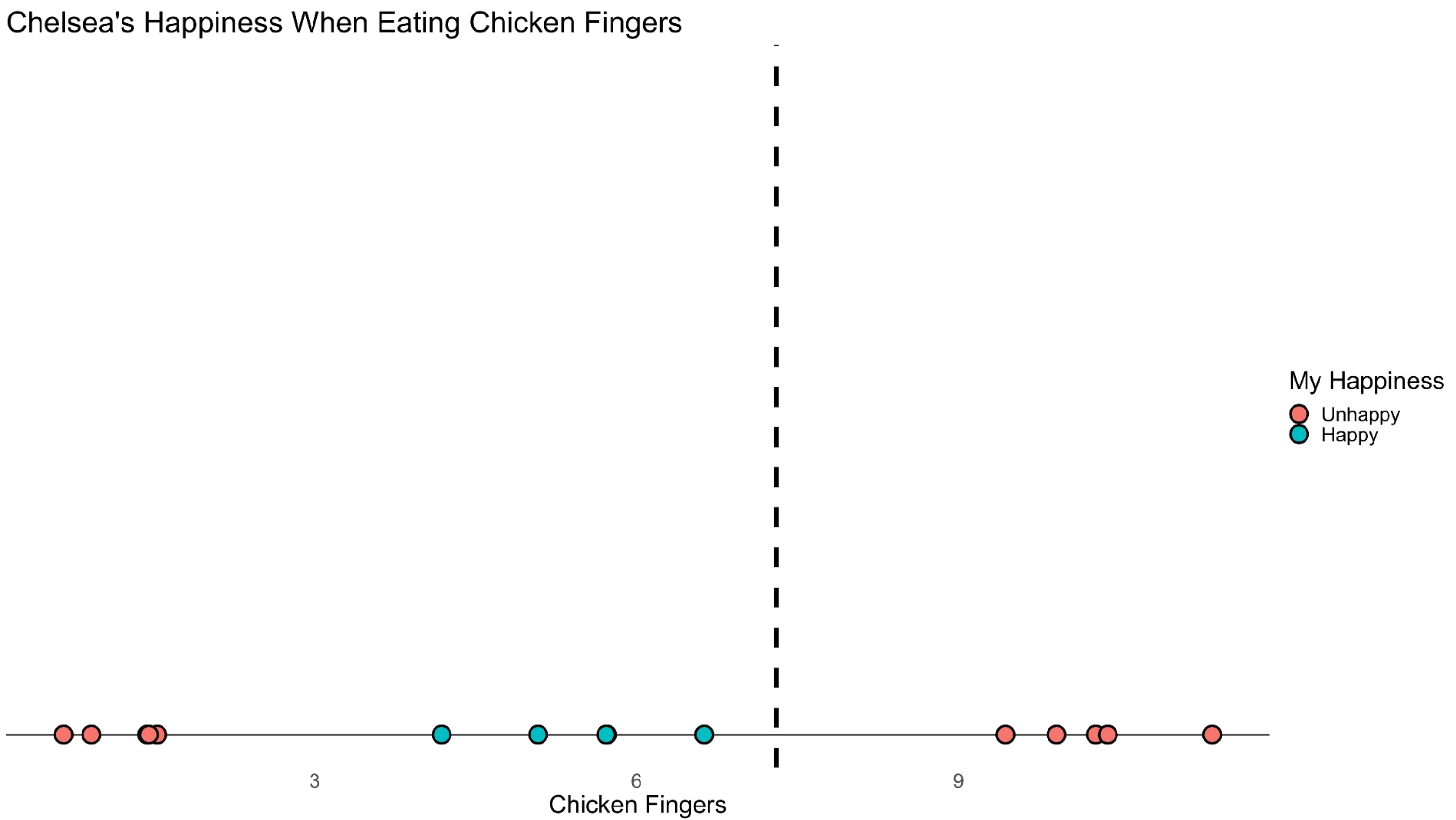
The Kernel Trick



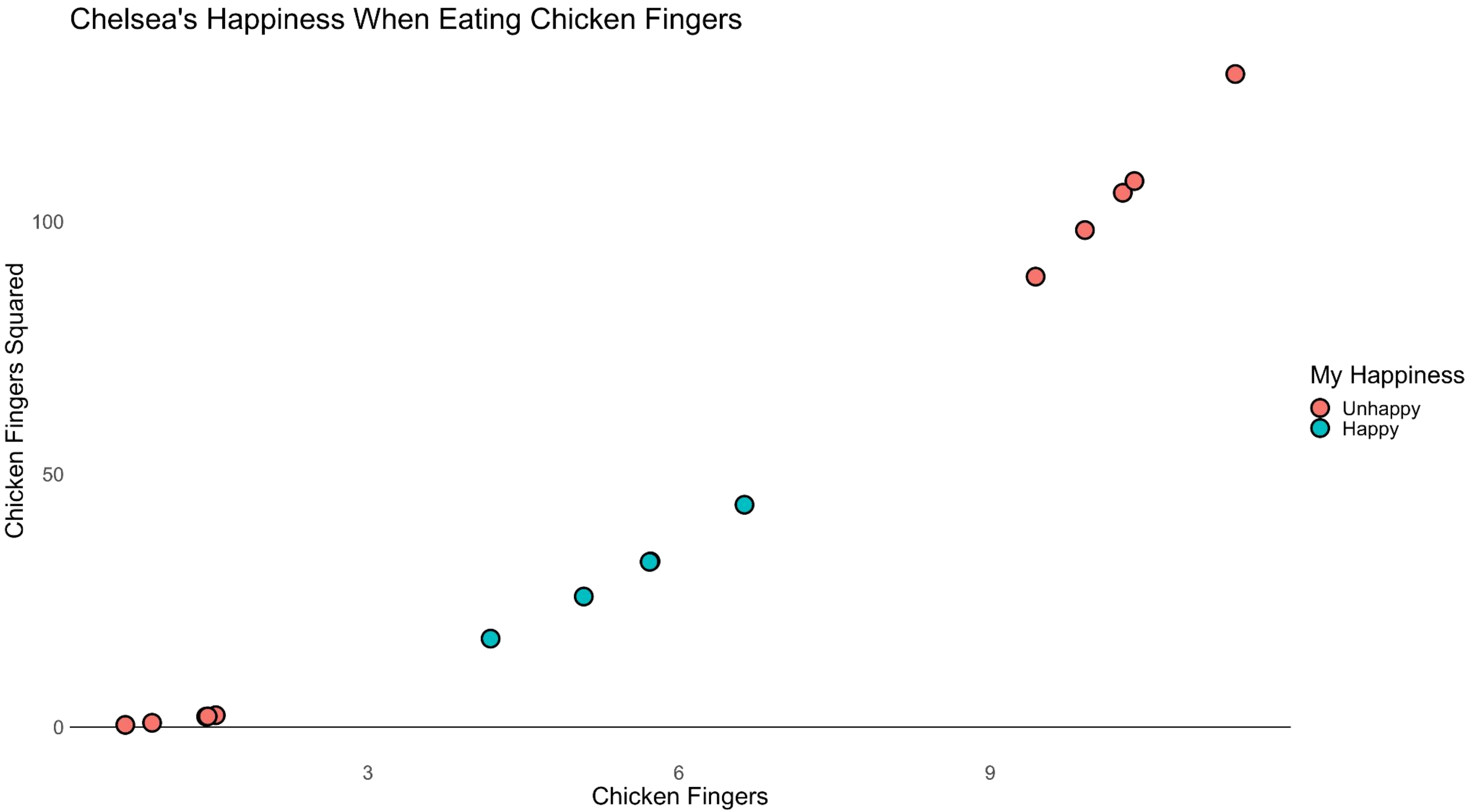
The Kernel Trick



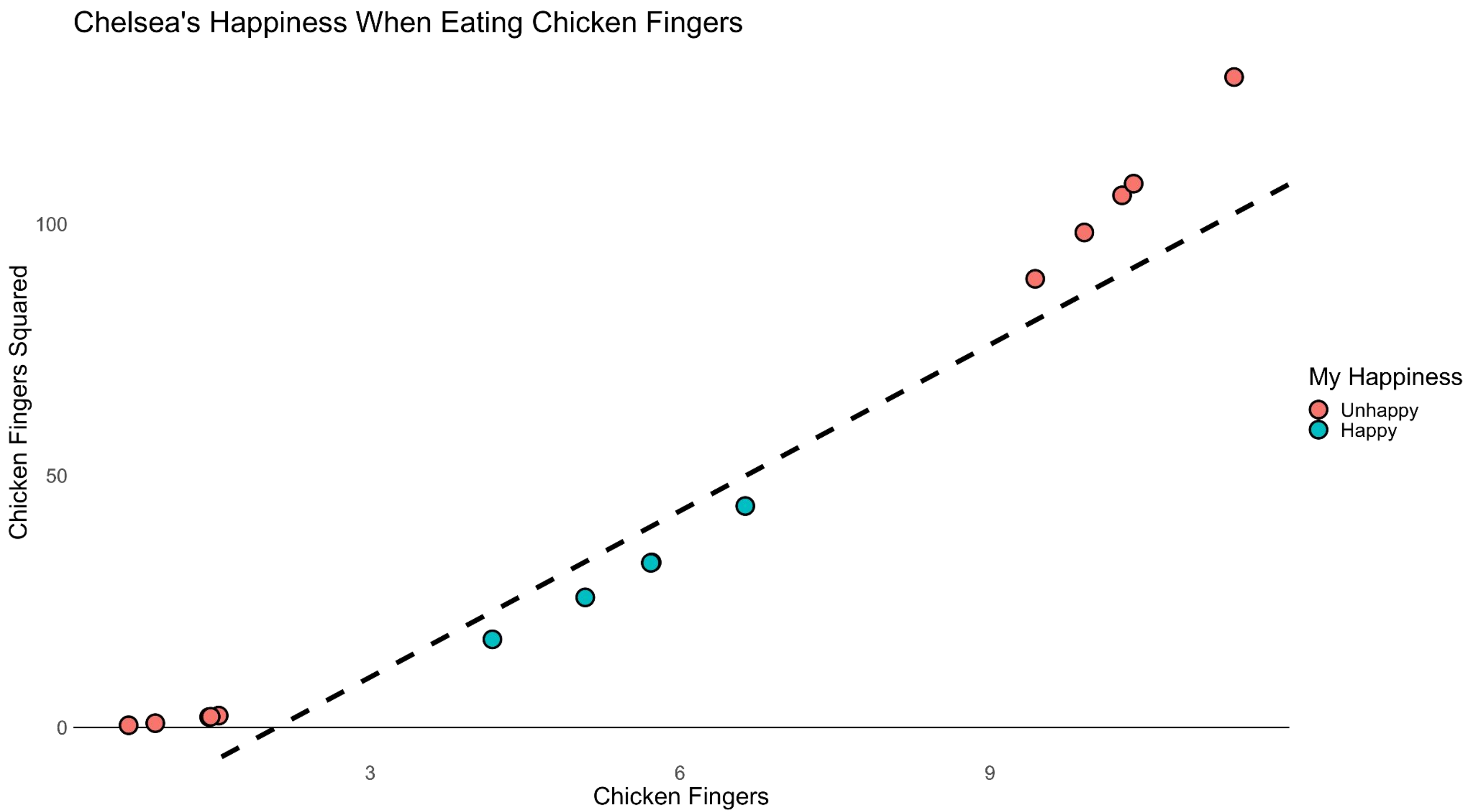
The Kernel Trick



The Kernel Trick



The Kernel Trick



The Kernel Trick

What is a Kernel (in SVMs)?

A function that calculates the the relationship between two vectors in multiple dimensions (without *actually* having to calculate the coordinates for those dimensions)

The Kernel Trick

$$\underbrace{K(x, y)}_{\text{kernel}} = (x * y + r)^d$$

- We used a Polynomial Kernel with **r** (coefficient) and **d** (degree)

The Kernel Trick

$$\underbrace{K(x, y)}_{\text{kernel}} = (x * y + r)^d$$

A point Another point

- We used a Polynomial Kernel with **r** (coefficient) and **d** (degree)
- Let’s try d = 2, r = ½

The Kernel Trick

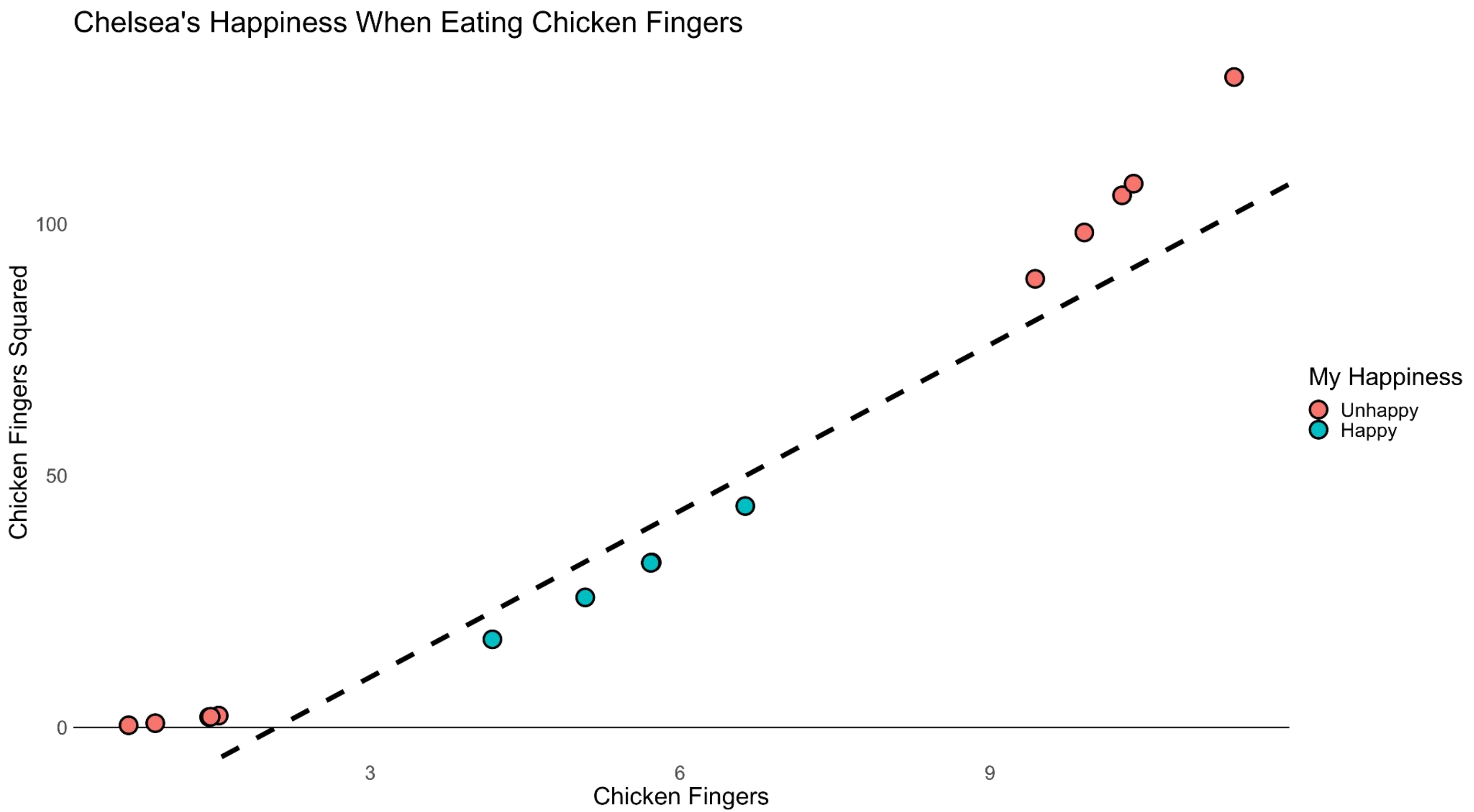
$$\underbrace{K(x, y)}_{\text{kernel}} = (x * y + r)^d$$

- We used a Polynomial Kernel with **r** (coefficient) and **d** (degree)
- Let's try d = 2, r = ½

Polynomial Kernel

$$\underbrace{K(x, y)}_{\text{kernel}} = (x * y + r)^d$$

The Kernel Trick

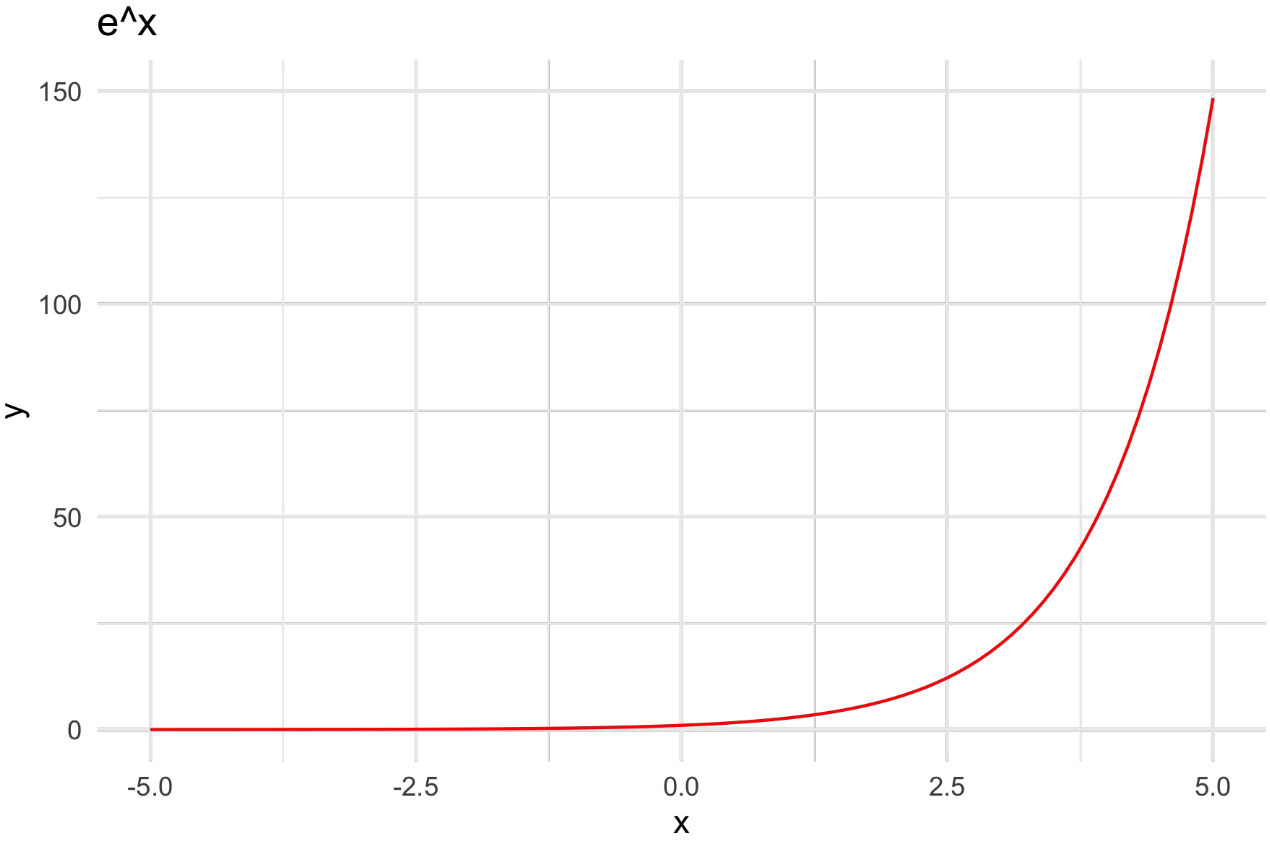


Radial Kernel (RBF)

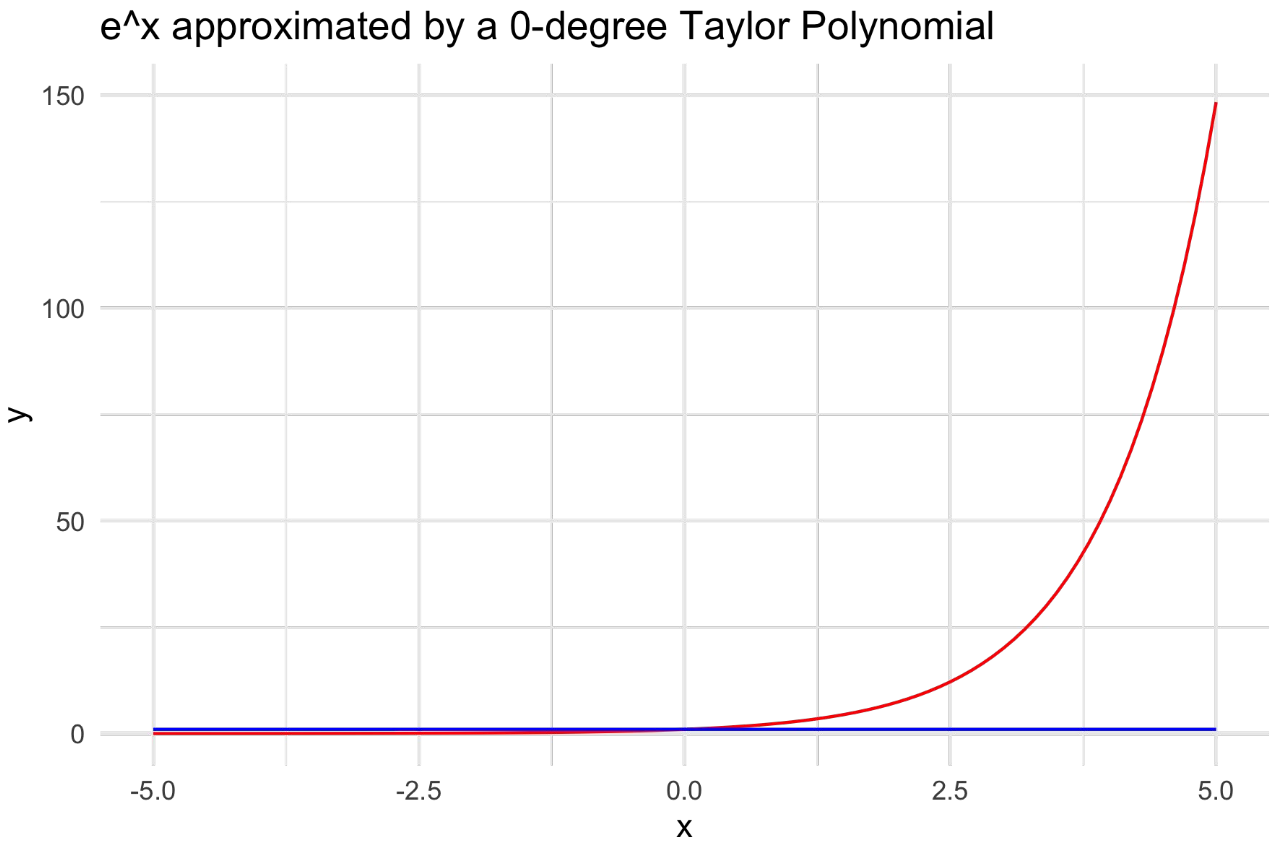
$$\underbrace{K(x, y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

- Projects to *infinite dimensional* space
- Similar to a nearest neighbors classifier (points that are far away have less influence)

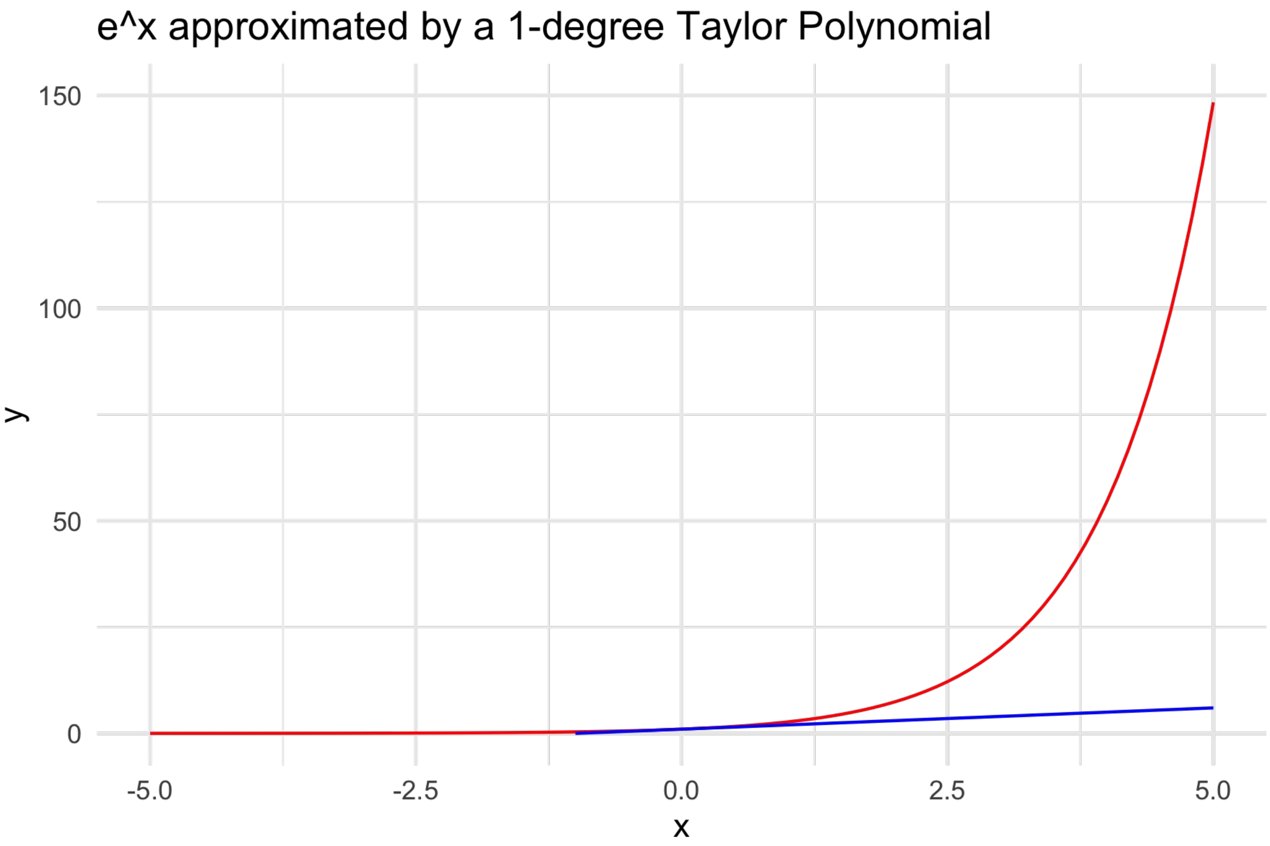
Taylor Series (visually)



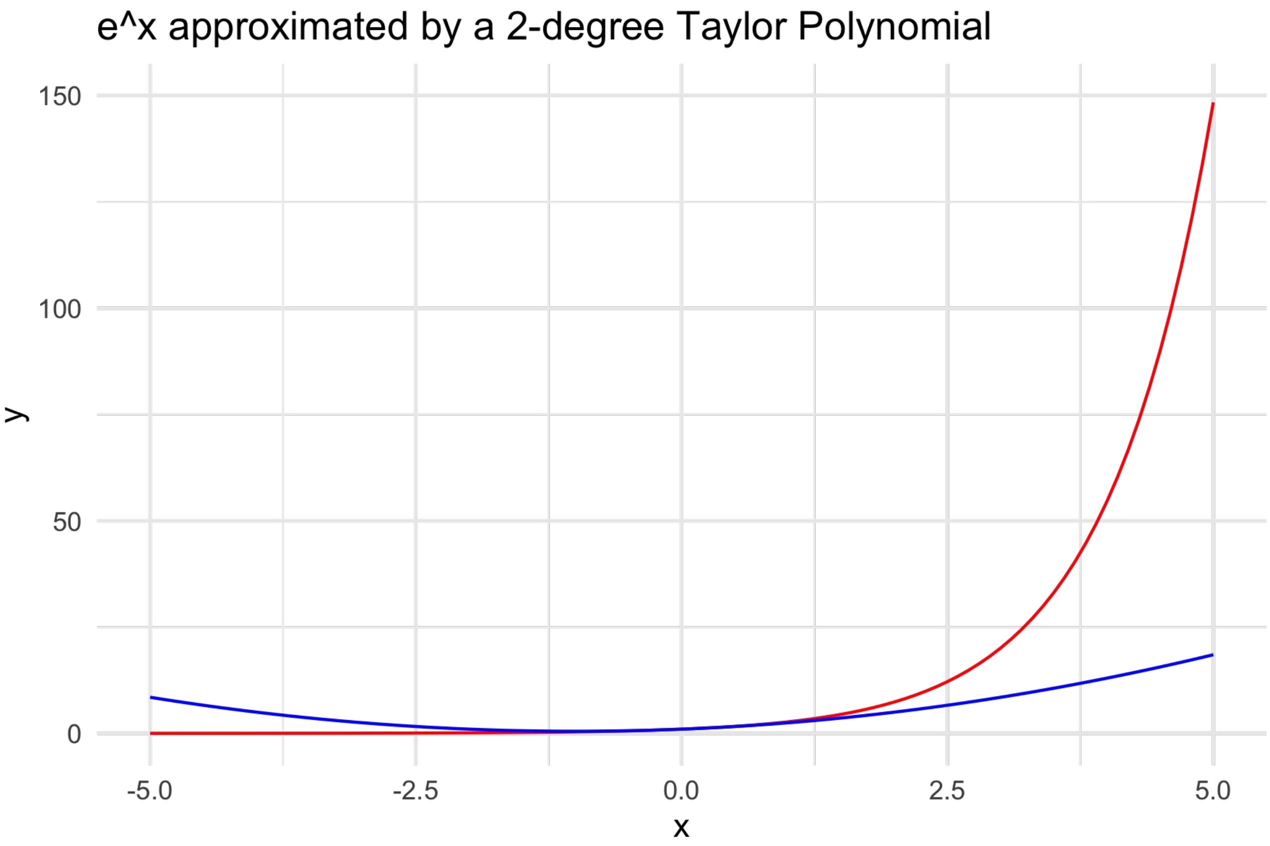
Taylor Series (visually)



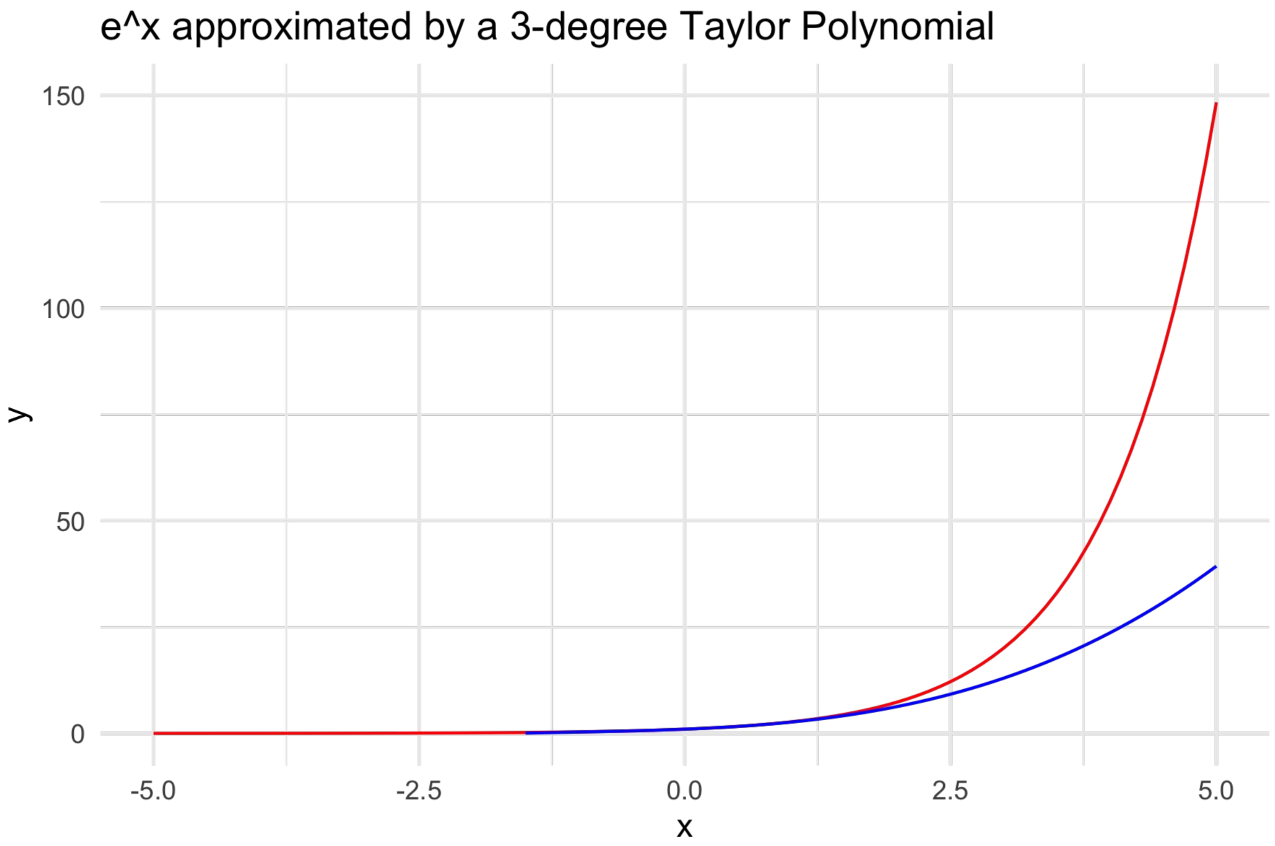
Taylor Series (visually)



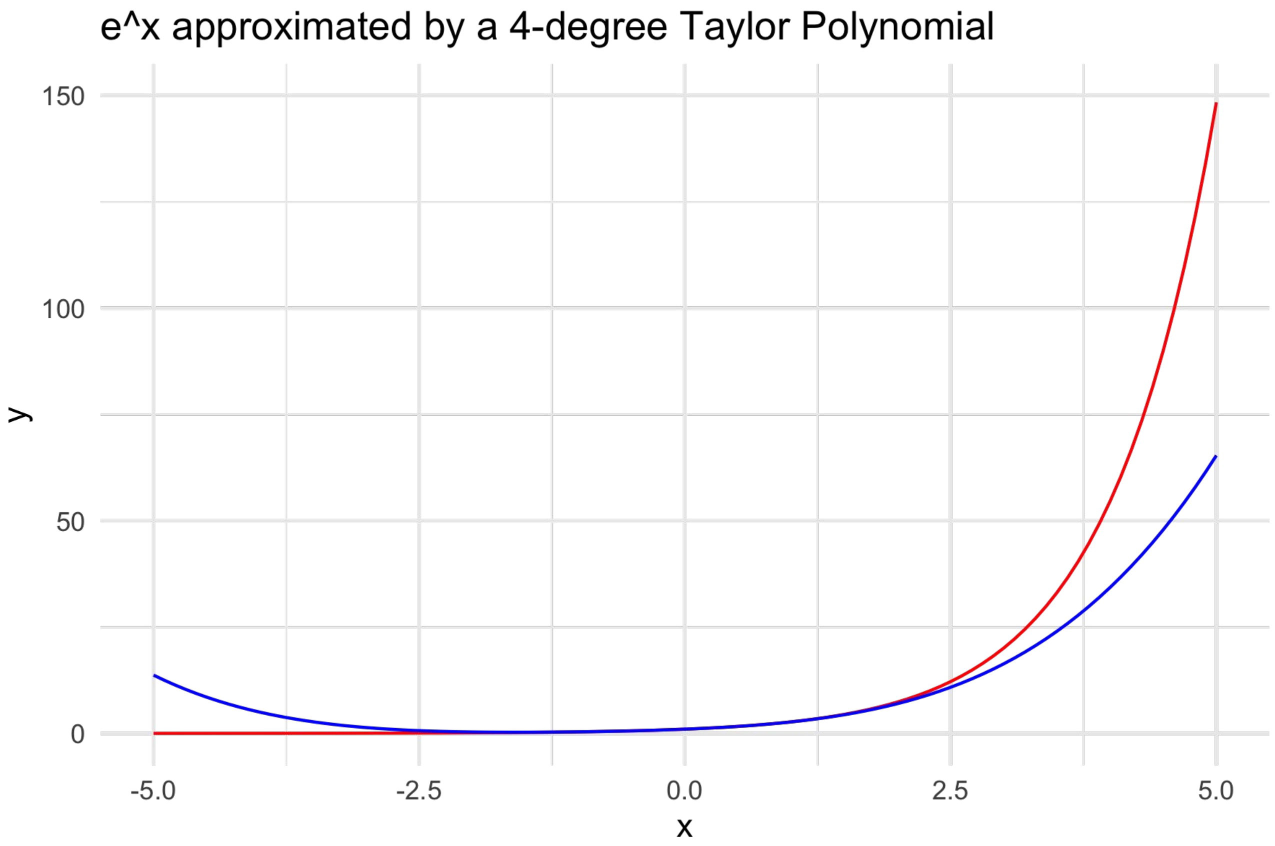
Taylor Series (visually)



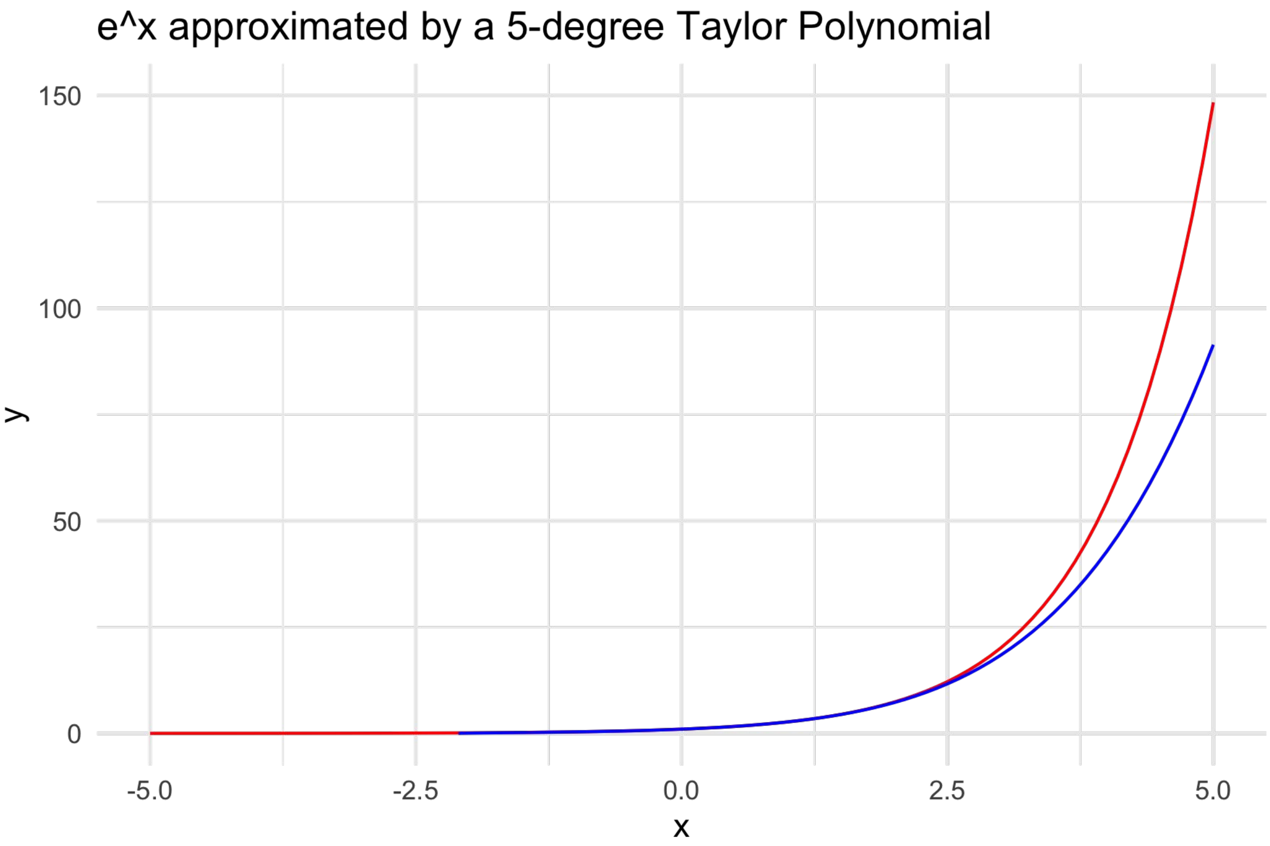
Taylor Series (visually)



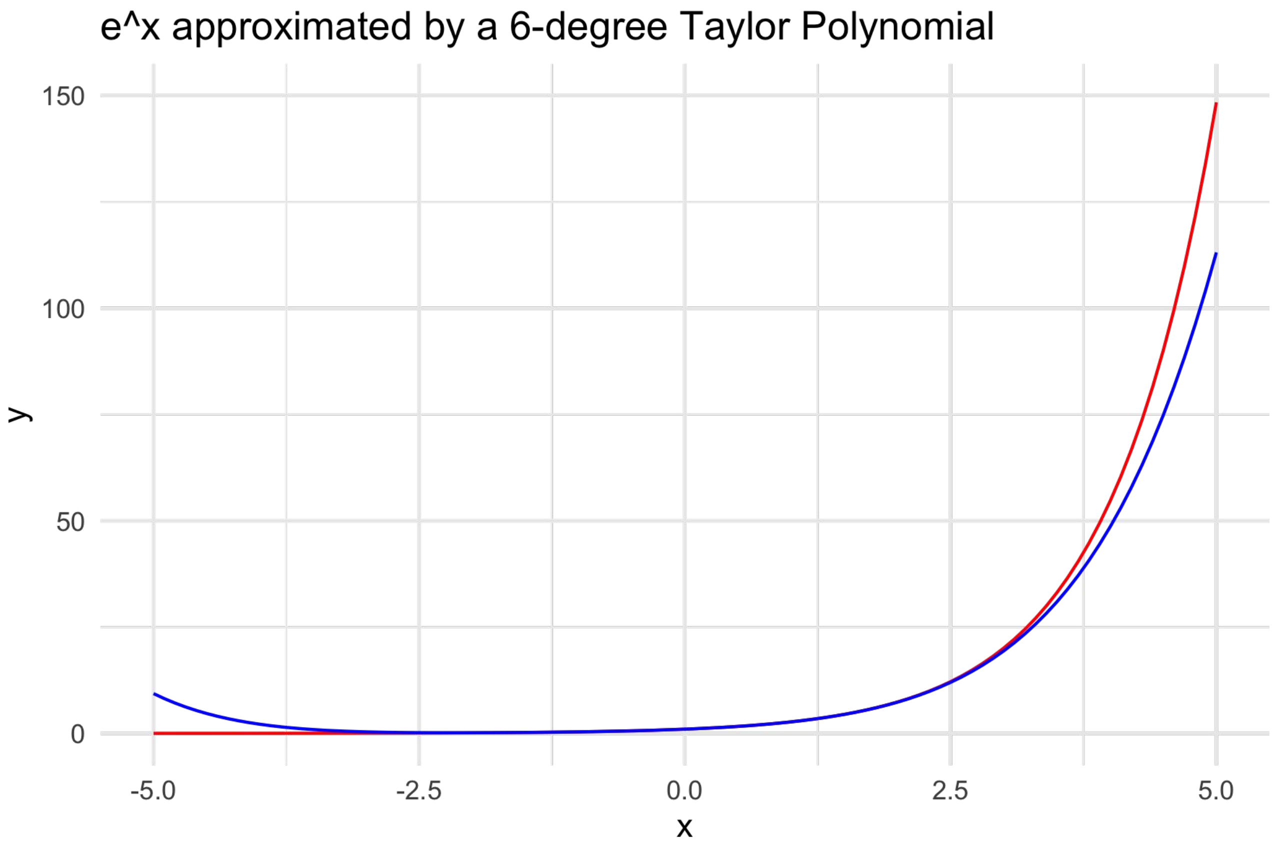
Taylor Series (visually)



Taylor Series (visually)



Taylor Series (visually)



Taylor Series

$$\sum_{n=0}^{\infty} \frac{f^n(a)}{n!}(x-a)^n = f(a) + \frac{f'(a)}{1}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \ldots + \frac{f^\infty(a)}{\infty!}(x-a)^\infty$$

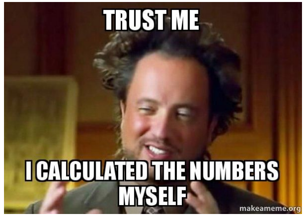
$$\sum_{n=0}^{\infty} \frac{f^n(a)}{n!}(x-a)^n = \boxed{\underbrace{f(a) + \frac{f'(a)}{1}(x-a) + \frac{f''(a)}{2!}(x-a)^2}_{\text{use just this for approximation of f(x))}}} + \ldots + \frac{f^\infty(a)}{\infty!}(x-a)^\infty$$

Radial Kernel (Taylor's Version)

$$\underbrace{K(x, y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$\underbrace{K(x, y)}_{\text{kernel}} = (x * y + r)^d$$

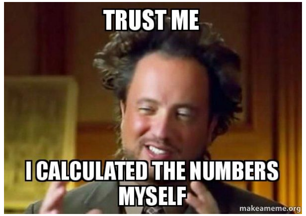
Radial Kernel (Taylor's Version)



$$K(x, y) = (x * y)^1 = x^1 y^1 = x^1 \cdot y^1$$

$$K(x, y) = (x * y)^2 = x^2 y^2 = x^2 \cdot y^2$$

Radial Kernel (Taylor's Version)



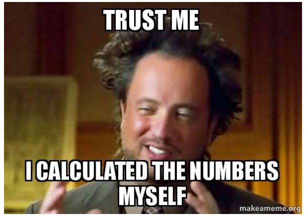
$$K(x, y) = (x * y)^1 = x^1 y^1 = x^1 \cdot y^1$$

$$K(x, y) = (x * y)^2 = x^2 y^2 = x^2 \cdot y^2$$

$$x^1 y^1 + x^2 y^2 = (x^1, x^2) \cdot (y^1, y^2)$$

$$x^1 y^1 + x^2 y^2 + x^3 y^3 = (x^1, x^2, x^3) \cdot (y^1, y^2, y^3)$$

Radial Kernel (Taylor's Version)



$$K(x, y) = (x * y)^1 = x^1 y^1 = x^1 \cdot y^1$$

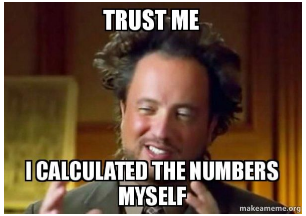
$$K(x, y) = (x * y)^2 = x^2 y^2 = x^2 \cdot y^2$$

$$x^1 y^1 + x^2 y^2 = (x^1, x^2) \cdot (y^1, y^2)$$

$$x^1 y^1 + x^2 y^2 + x^3 y^3 = (x^1, x^2, x^3) \cdot (y^1, y^2, y^3)$$

$$x^1 y^1 + x^2 y^2 + x^3 y^3 + \ldots + x^\infty y^\infty = (x^1, x^2, x^3, \ldots, x^\infty) \cdot (y^1, y^2, y^3, \ldots, y^\infty)$$

Radial Kernel (Taylor's Version)



$$K(x,y) = (x * y)^1 = x^1 y^1 = x^1 \cdot y^1$$

$$K(x,y) = (x * y)^2 = x^2 y^2 = x^2 \cdot y^2$$

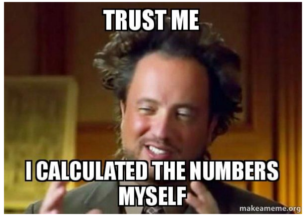
$$\underbrace{K(x,y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$x^1 y^1 + x^2 y^2 = (x^1, x^2) \cdot (y^1, y^2)$$

$$x^1 y^1 + x^2 y^2 + x^3 y^3 = (x^1, x^2, x^3) \cdot (y^1, y^2, y^3)$$

$$x^1 y^1 + x^2 y^2 + x^3 y^3 + \ldots + x^\infty y^\infty = (x^1, x^2, x^3, \ldots, x^\infty) \cdot (y^1, y^2, y^3, \ldots, y^\infty)$$

Radial Kernel (Taylor's Version)



$$\underbrace{K(x, y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$e^{-\gamma(x-y)^2} = e^{-\gamma(x^2+y^2-2xy)}$$

$$e^{-\gamma(x^2+y^2)}e^{\gamma 2xy}$$

Radial Kernel (Taylor's Version)

$$\underbrace{K(x,y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$e^{-\gamma(x-y)^2} = e^{-\gamma(x^2+y^2-2xy)}$$

$$e^{-\gamma(x^2+y^2)}e^{\gamma 2xy}$$

$$e^{-\frac{1}{2}(x^2+y^2)}e^{xy}$$

Radial Kernel (Taylor's Version)

$$\underbrace{K(x,y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$e^{-\gamma(x-y)^2} = e^{-\gamma(x^2+y^2-2xy)}$$

$$e^{-\gamma(x^2+y^2)}e^{\gamma 2xy}$$

$$e^{-\frac{1}{2}(x^2+y^2)}\boxed{e^{xy}}$$

Radial Kernel (Taylor's Version)

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

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

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

$$\underbrace{K(x, y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

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

$$x^0y^0 + x^1y^1 + x^2y^2 + \ldots + x^\infty y^\infty = (1, x, x^2, \ldots, x^\infty) \cdot (1, y, y^2, \ldots, y^\infty)$$

Radial Kernel (Taylor's Version)

$$\underbrace{K(x, y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$e^{xy} = \boxed{1} + \frac{1}{1!}(xy) + \frac{1}{2!}(xy)^2 + \frac{1}{3!}(xy)^3 + \ldots + \frac{1}{\infty!}(xy)^{\infty}$$

$$\boxed{x^0 y^0} + x^1 y^1 + x^2 y^2 + \ldots + x^{\infty} y^{\infty} = (1, x, x^2, \ldots, x^{\infty}) \cdot (1, y, y^2, \ldots, y^{\infty})$$

Radial Kernel (Taylor's Version)

$$\underbrace{K(x, y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$e^{xy} = \boxed{1} + \frac{1}{1!}\boxed{(xy)} + \frac{1}{2!}(xy)^2 + \frac{1}{3!}(xy)^3 + \ldots + \frac{1}{\infty!}(xy)^\infty$$

$$\boxed{x^0y^0} + \boxed{x^1y^1} + x^2y^2 + \ldots + x^\infty y^\infty = (1, x, x^2, \ldots, x^\infty) \cdot (1, y, y^2, \ldots, y^\infty)$$

Radial Kernel (Taylor's Version)

$$\underbrace{K(x, y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$e^{xy} = \boxed{1} + \frac{1}{1!}\boxed{(xy)} + \frac{1}{2!}\boxed{(xy)^2} + \frac{1}{3!}(xy)^3 + \ldots + \frac{1}{\infty!}(xy)^\infty$$

$$\boxed{x^0 y^0} + \boxed{x^1 y^1} + \boxed{x^2 y^2} + \ldots + x^\infty y^\infty = (1, x, x^2, \ldots, x^\infty) \cdot (1, y, y^2, \ldots, y^\infty)$$

Radial Kernel (Taylor's Version)

$$\underbrace{K(x, y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$$e^{xy} = \boxed{1} + \frac{1}{1!}\boxed{(xy)} + \frac{1}{2!}\boxed{(xy)^2} + \frac{1}{3!}(xy)^3 + \ldots + \frac{1}{\infty!}\boxed{(xy)^\infty}$$

$$\boxed{x^0y^0} + \boxed{x^1y^1} + \boxed{x^2y^2} + \ldots + \boxed{x^\infty y^\infty} = (1, x, x^2, \ldots, x^\infty)\cdot(1, y, y^2, \ldots, y^\infty)$$

Radial Kernel (Taylor's Version)

$$\underbrace{K(x,y)}_{\text{kernel}} = e^{-\gamma(x-y)^2}$$

$e^{xy} = \boxed{1} + \frac{1}{1!}($ …

$\boxed{x^0y^0} + \boxed{x^1y^1} +$ … $, \ldots, y^\infty)$

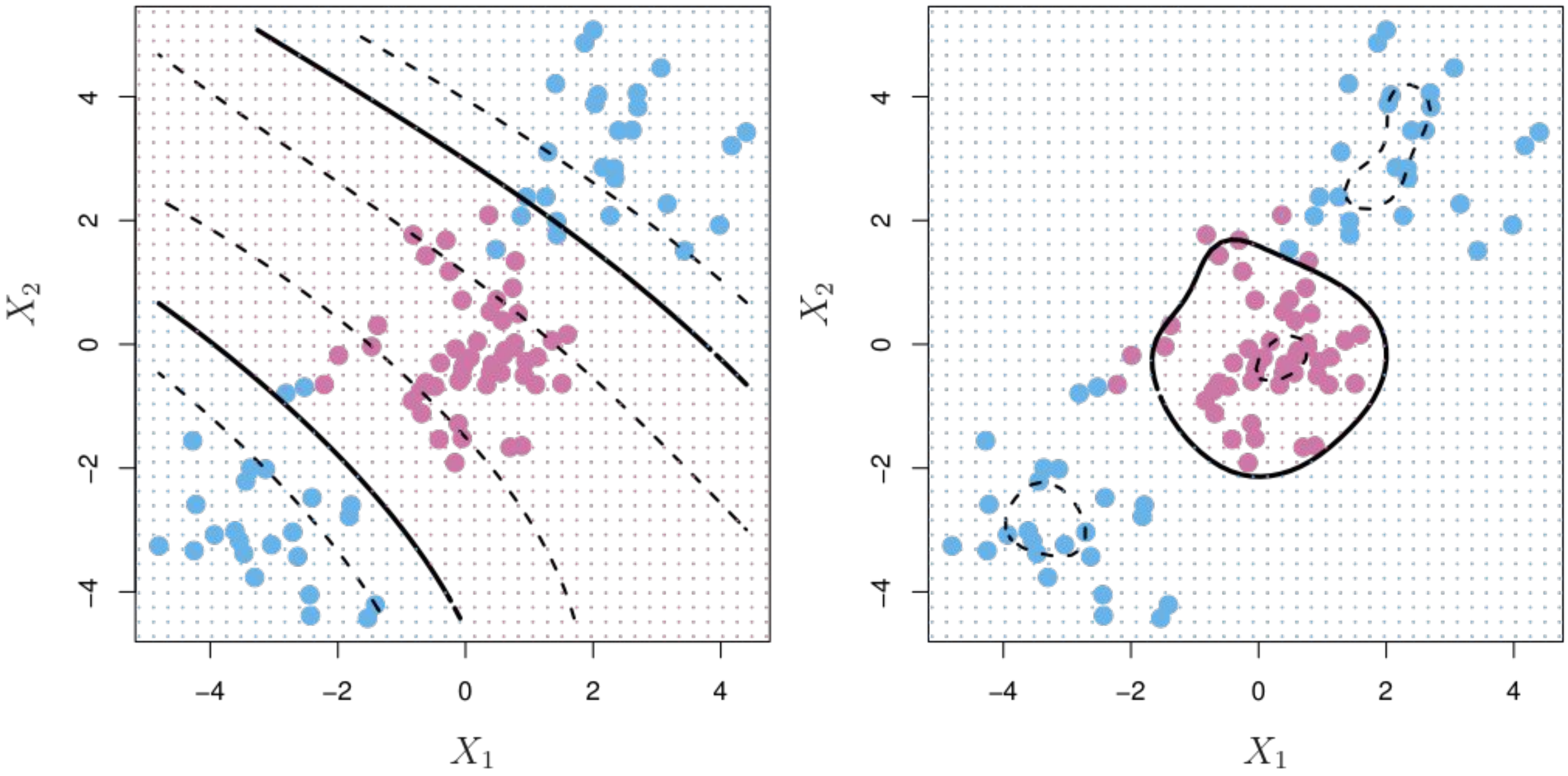
Radial Basis Kernels have coordinates for infinite dimensions!

Support Vector Machines Part 3:

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

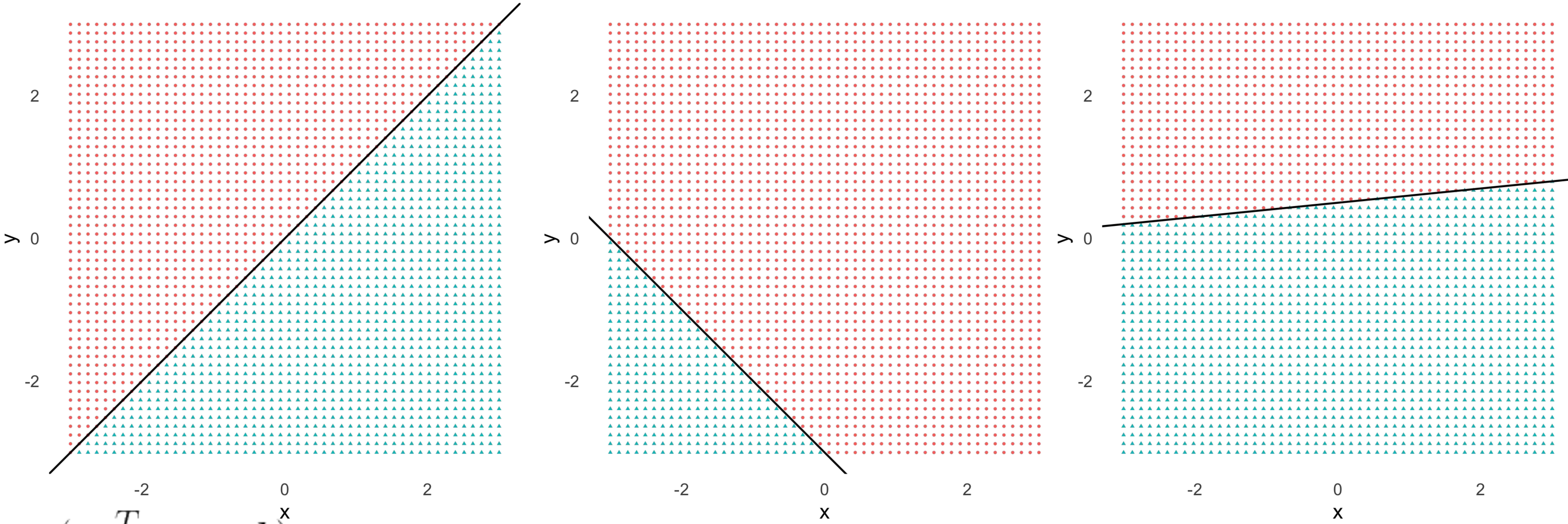
...The Radial Kernel!!!

Radial Basis Kernel Example



Images from an Introduction to Statistical Learning with applications in R

Kernels



$$t_n(\mathbf{w}^T x_n + b)$$

$$t_n(\mathbf{w}^T \phi(x_n) + b)$$

Kernels

$$\overbrace{y(\mathbf{w}) = \sum_{n=1}^{N} a_n t_n \;\; \phi(x) \cdot \phi(x_n) \;\; + b}^{\text{formula to classify new data point } x}$$

$$\overbrace{y(\mathbf{w}) = \sum_{n=1}^{N} a_n t_n \;\; \underbrace{K(x, y)}_{\text{kernel}} \;\; +b}^{\text{formula to classify new data point } x}$$