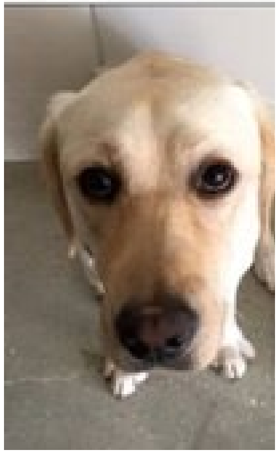
Introduction to Machine Learning

SVMs

Train: dogs and cookies

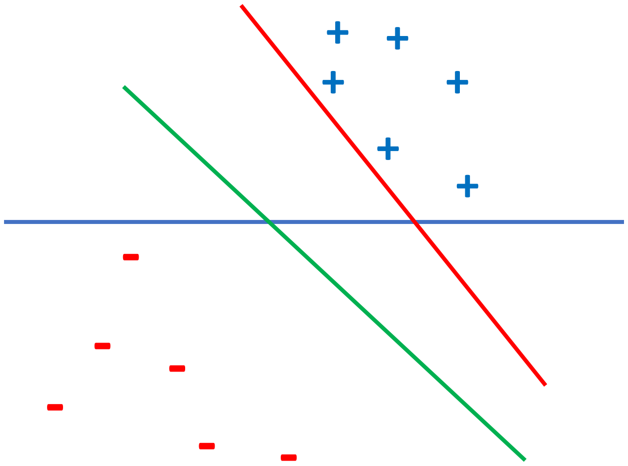
Test: dog or cookie?



SVM overview

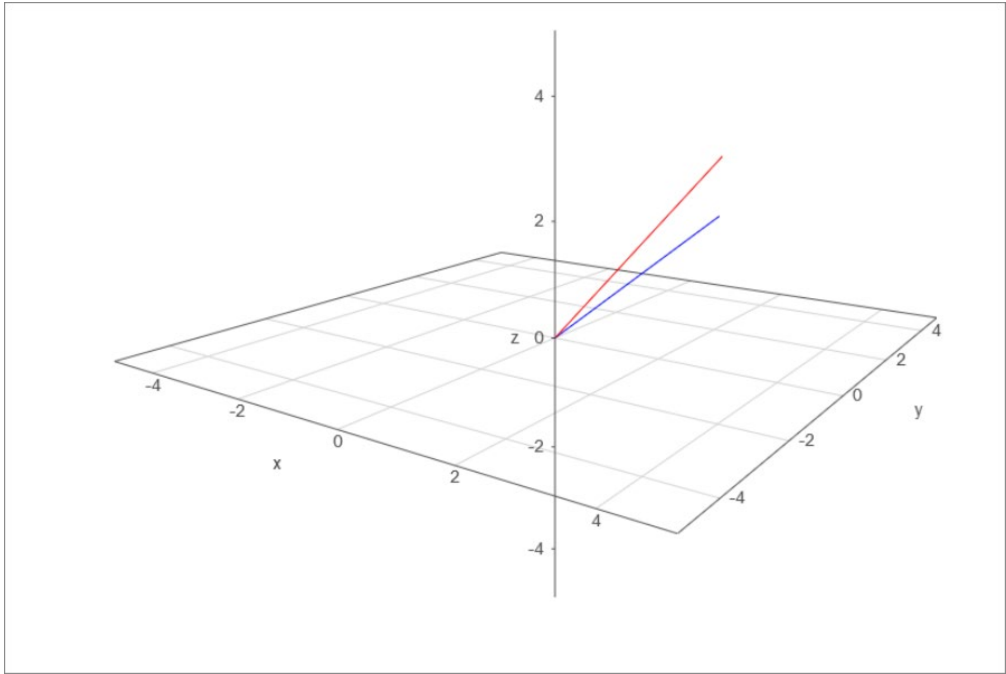
- Training
  -
  -
  -
- Model
  -
  -
- Testing
  -
  -

Which hyperplane do you like the best?



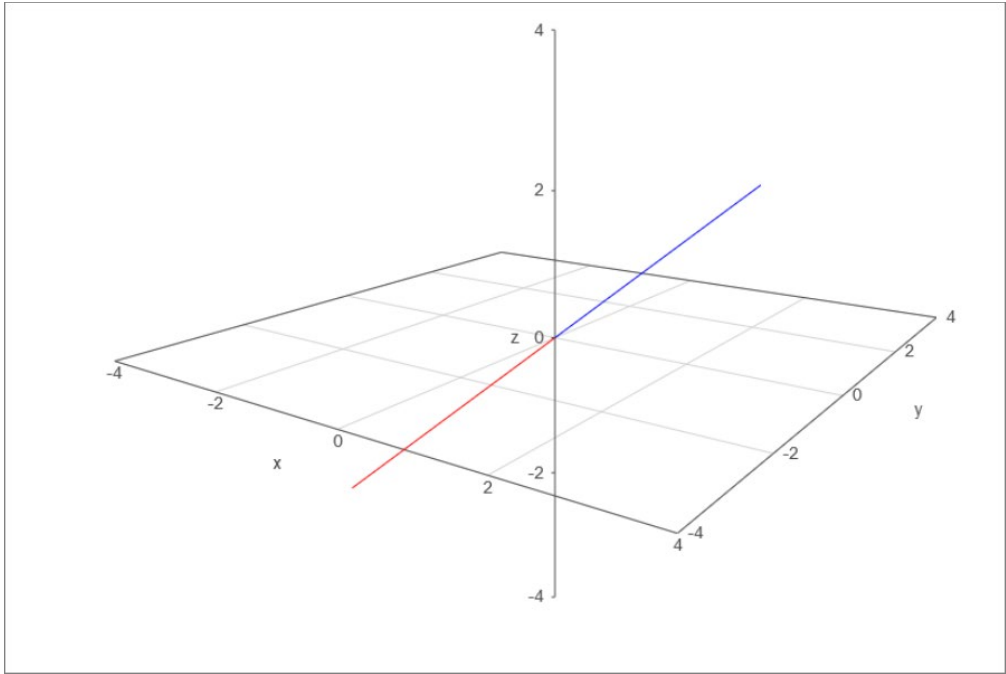
Some math review

- Given vectors *u* and *v*
- Length of vector *u* is $||u||$
- Dot product $u \cdot v$ is $\sum_d u_d v_d$



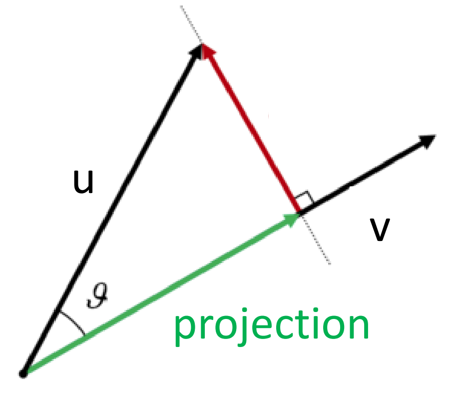
Some math review

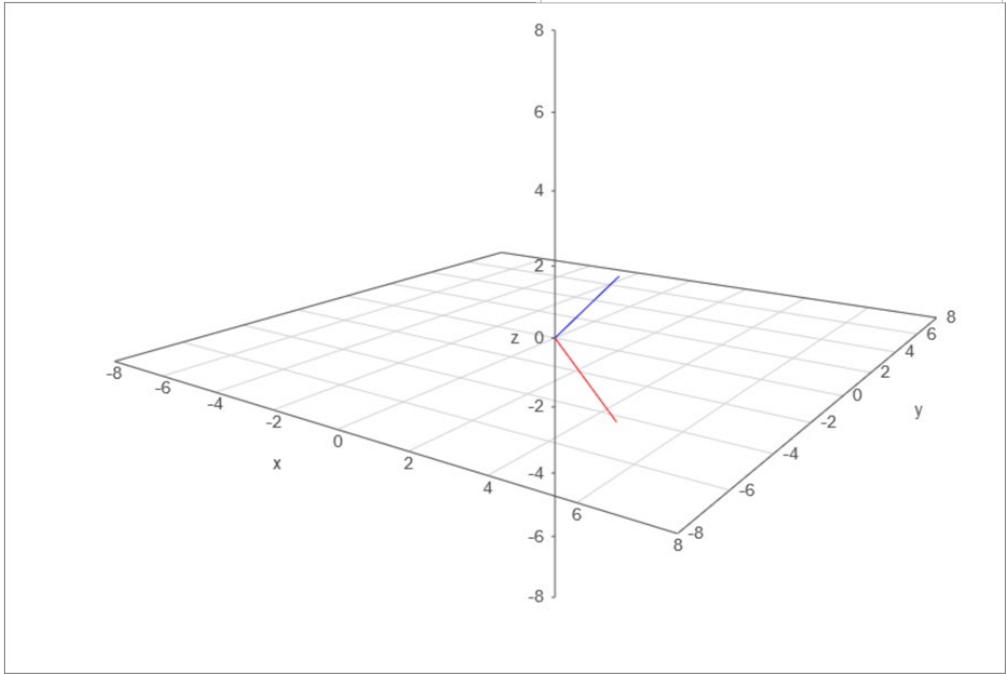
- Given vectors u and v
- Length of vector u is $||u||$
- Dot product $u \cdot v$ is $\sum_d u_d v_d$



Some math review

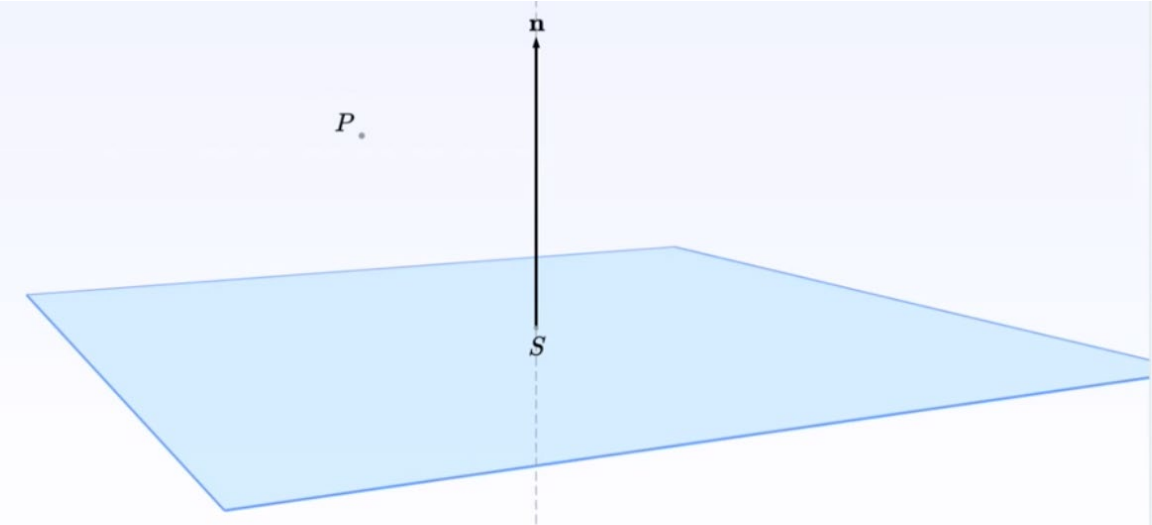
- Given vectors *u* and *v*
- Length of vector *v* is $||v||$
- Dot product $u \cdot v$ is $\sum_d u_d v_d$
- Projection of *u* onto *v*





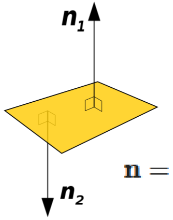
Some math review

- Projection of point P onto a plane

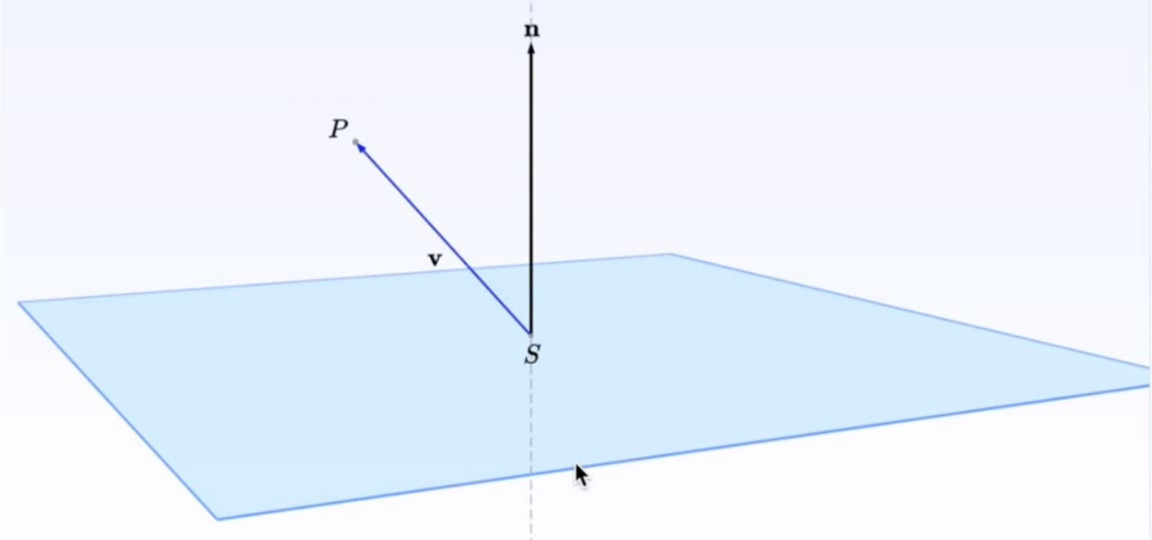


Some math review

- Projection of point P onto a plane



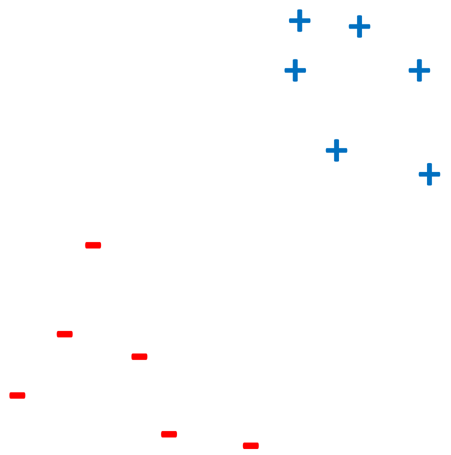
$$\mathbf{n} = \frac{\mathbf{N}}{\|\mathbf{N}\|} = \frac{(A, B, C)}{\sqrt{A^2 + B^2 + C^2}}$$



Constrained optimization problem

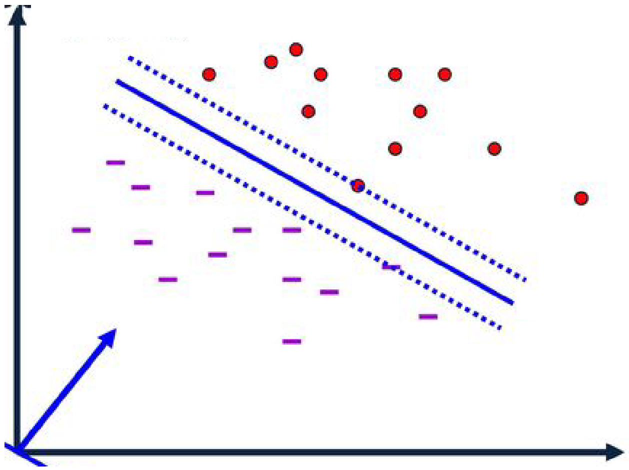
$$\min_{w,b} \quad \frac{1}{\gamma(\boldsymbol{w}, b)}$$

$$\text{subj. to} \quad y_n\left(\boldsymbol{w} \cdot \boldsymbol{x}_n + b\right) \geq 1 \quad \text{for all } n$$



Margin

- Large margin -> easy
- Small margin -> hard



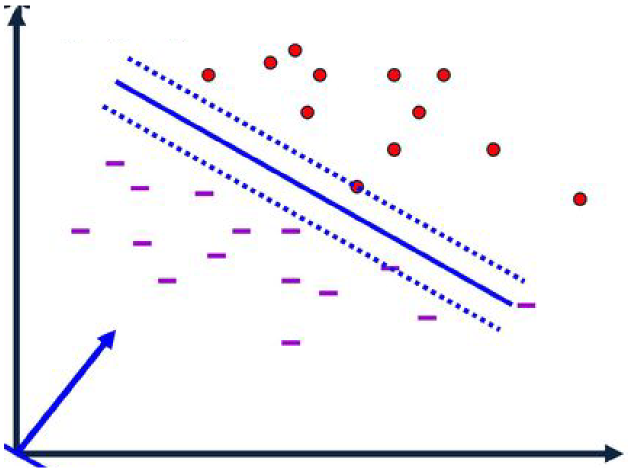
Margin

- Margin of w, b on D

$$margin(\mathbf{D}, \boldsymbol{w}, b) = \begin{cases} \min_{(x,y)\in\mathbf{D}} y(\boldsymbol{w} \cdot \boldsymbol{x} + b) & \text{if } \boldsymbol{w} \text{ separates } \mathbf{D} \\ -\infty & \text{otherwise} \end{cases}$$

- Margin of a dataset

$$margin(\mathbf{D}) = \sup_{\boldsymbol{w},b} margin(\mathbf{D}, \boldsymbol{w}, b)$$



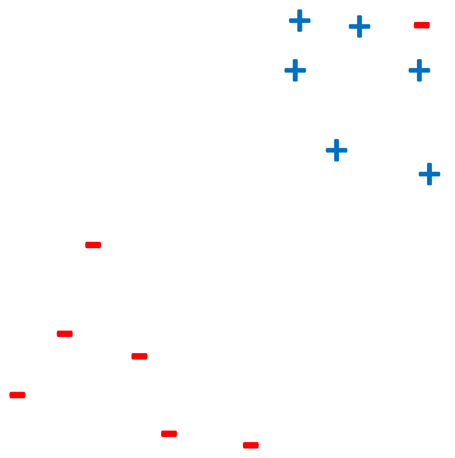
Feasible region

- Set of all parameters satisfying constraints

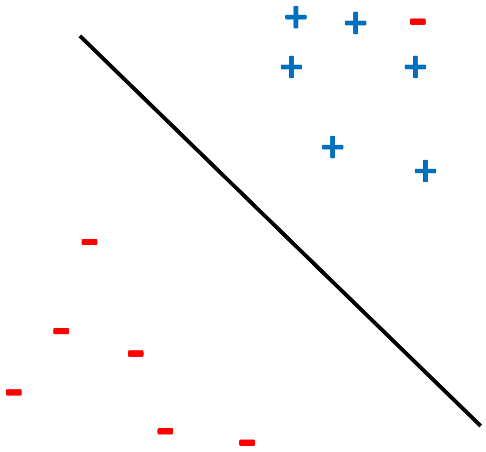
$$\min_{w,b} \quad \frac{1}{\gamma(\boldsymbol{w}, b)}$$

$$\text{subj. to} \quad y_n \left(\boldsymbol{w} \cdot \boldsymbol{x}_n + b \right) \geq 1 \qquad \text{for all } n$$

- Hard-margin SVM



Slack parameters

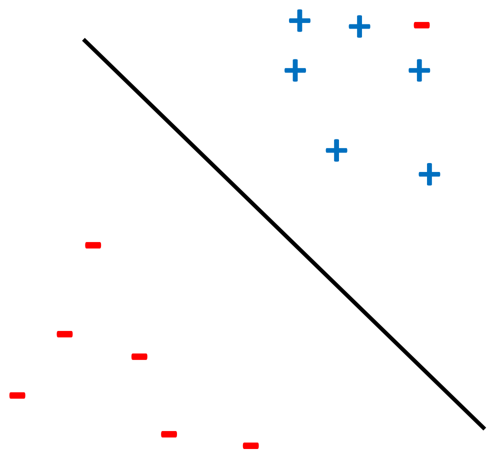


Slack parameters

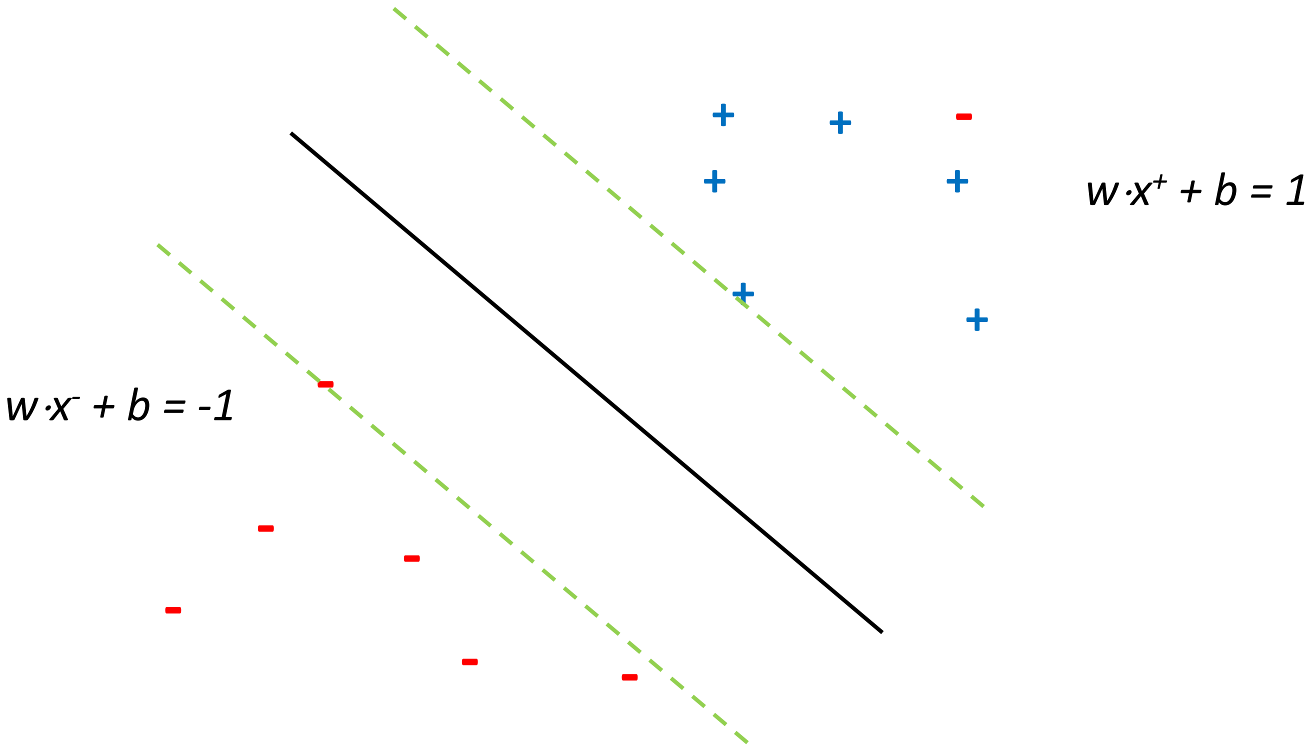
$$\min_{w,b,\xi} \quad \underbrace{\frac{1}{\gamma(\boldsymbol{w}, b)}}_{\text{large margin}} + \underbrace{C\sum_{n} \xi_n}_{\text{small slack}}$$

$$\text{subj. to} \quad y_n\left(\boldsymbol{w} \cdot \boldsymbol{x}_n + b\right) \geq 1 - \xi_n$$

$$\xi_n \geq 0$$

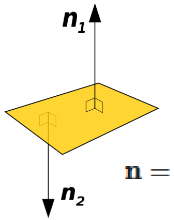


Size of the margin

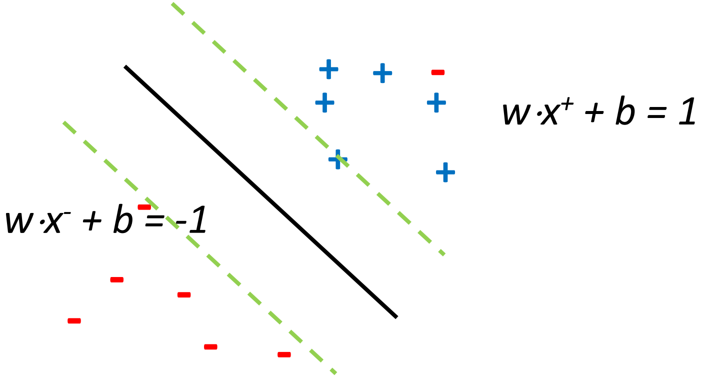


Size of the margin

$$d^{+} = \frac{1}{||w||} w \cdot x^{+} + b - 1$$
$$d^{-} = -\frac{1}{||w||} w \cdot x^{-} - b + 1$$



$$\mathbf{n} = \frac{\mathbf{N}}{\|\mathbf{N}\|} = \frac{(A, B, C)}{\sqrt{A^2 + B^2 + C^2}}$$



Compute the margin

$$d^{+} = \frac{1}{||w||} w \cdot x^{+} + b - 1 \qquad\qquad \gamma = \frac{1}{2}\left[d^{+} - d^{-}\right]$$

$$d^{-} = -\frac{1}{||w||} w \cdot x^{-} - b + 1$$

Compute slacks

$$\min_{w,b,\xi} \quad \underbrace{\frac{1}{\gamma(\boldsymbol{w}, b)}}_{\text{large margin}} + \underbrace{C \sum_n \xi_n}_{\text{small slack}}$$

$$\text{subj. to} \quad y_n \left(\boldsymbol{w} \cdot \boldsymbol{x}_n + b\right) \geq 1 - \xi_n$$

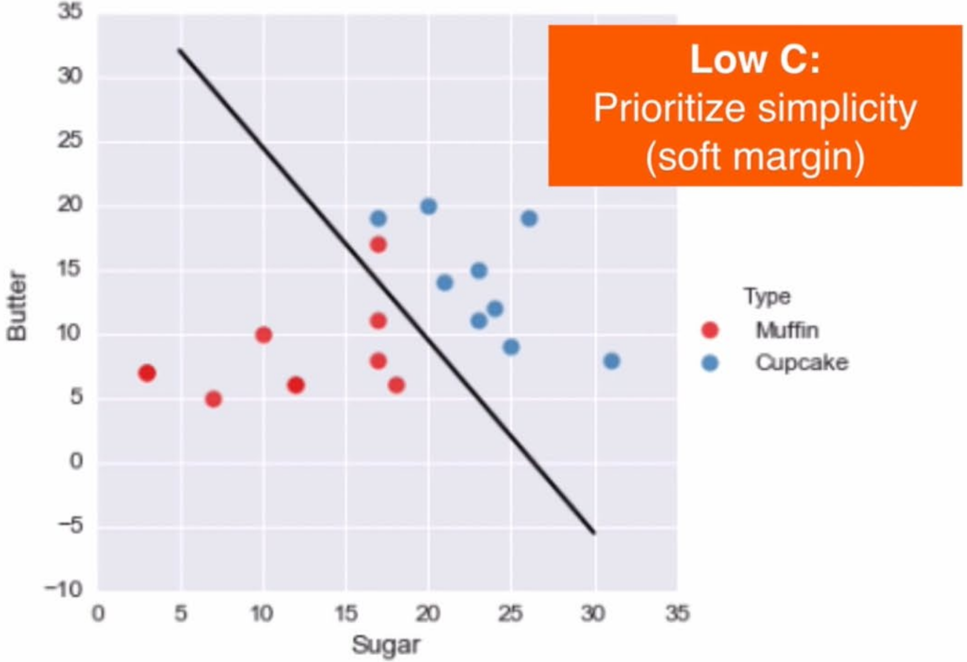
$$\xi_n \geq 0$$

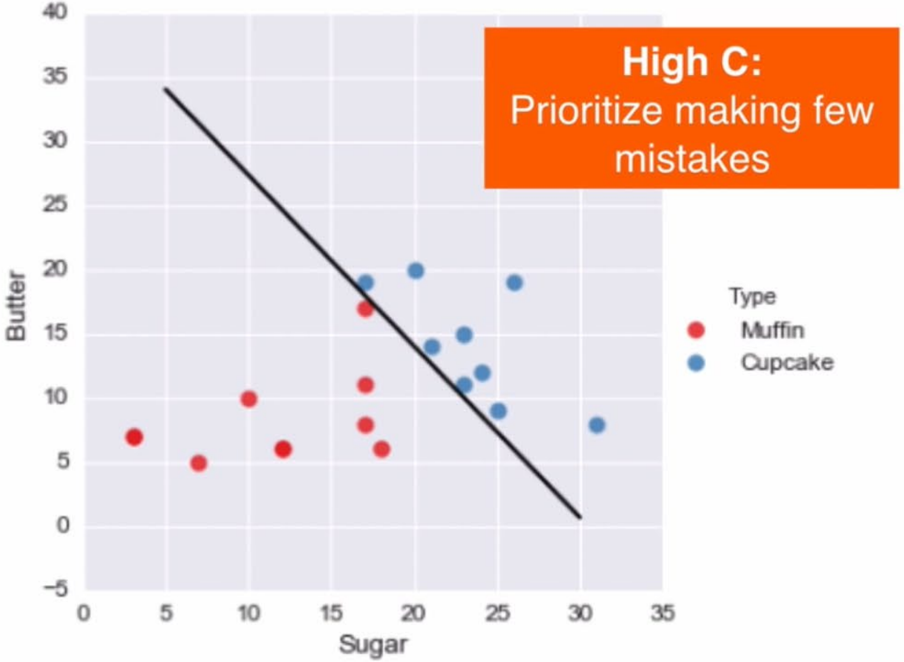
$$\xi_n = \begin{cases} 0 & \text{if } y_n(\boldsymbol{w} \cdot \boldsymbol{x}_n + b) \geq 1 \\ 1 - y_n(\boldsymbol{w} \cdot \boldsymbol{x}_n + b) & \text{otherwise} \end{cases}$$

(hinge loss, or $\ell^{(\text{hin})}$)

SVM objective:

$$\min_{w,b} \quad \underbrace{\frac{1}{2}||\boldsymbol{w}||^2}_{\text{large margin}} + \underbrace{C \sum_n \ell^{(\text{hin})}(y_n, \boldsymbol{w} \cdot \boldsymbol{x}_n + b)}_{\text{small slack}}$$


Low C:
Prioritize simplicity (soft margin)
Butter
Sugar
Type
Muffin
Cupcake


High C:
Prioritize making few mistakes
Butter
Sugar
Type
Muffin
Cupcake

Support vector machines

$$
\begin{aligned}
\min_{w,b,\xi} \quad & \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_n \xi_n \\
\text{subj. to} \quad & y_n\left(\boldsymbol{w}\cdot\boldsymbol{x}_n + b\right) \geq 1 - \xi_n \\
& \xi_n \geq 0
\end{aligned}
$$

SVM optimization problem

$$\min_{w,b,\xi} \max_{\alpha \geq 0} \max_{\beta \geq 0} \mathcal{L}(w, b, \xi, \alpha, \beta)$$

SVM optimization problem

$$\min_{w,b,\xi} \max_{\alpha \geq 0} \max_{\beta \geq 0} \mathcal{L}(w, b, \xi, \alpha, \beta)$$

$$\nabla_w \mathcal{L} = w - \sum_n \alpha_n y_n x_n = 0 \quad \Longleftrightarrow \quad w = \sum_n \alpha_n y_n x_n$$

$$\mathcal{L}(b, \xi, \alpha, \beta) = \frac{1}{2} \left|\left| \sum_m \alpha_m y_m x_m \right|\right|^2 + C \sum_n \xi_n - \sum_n \beta_n \xi_n - \sum_n \alpha_n \left[y_n \left(\left[\sum_m \alpha_m y_m x_m \right] \cdot x_n + b \right) - 1 + \xi_n \right]$$

SVM optimization

$$\mathcal{L}(b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\sum_n \sum_m \alpha_n \alpha_m y_n y_m \boldsymbol{x}_n \cdot \boldsymbol{x}_m + \sum_n (C - \beta_n)\xi_n$$

$$- \sum_n \sum_m \alpha_n \alpha_m y_n y_m \boldsymbol{x}_n \cdot \boldsymbol{x}_m - \sum_n \alpha_n \left(y_n b - 1 + \xi_n \right)$$

$$= -\frac{1}{2}\sum_n \sum_m \alpha_n \alpha_m y_n y_m \boldsymbol{x}_n \cdot \boldsymbol{x}_m + \sum_n (C - \beta_n)\xi_n$$

$$-b \sum_n \alpha_n y_n - \sum_n \alpha_n (\xi_n - 1)$$

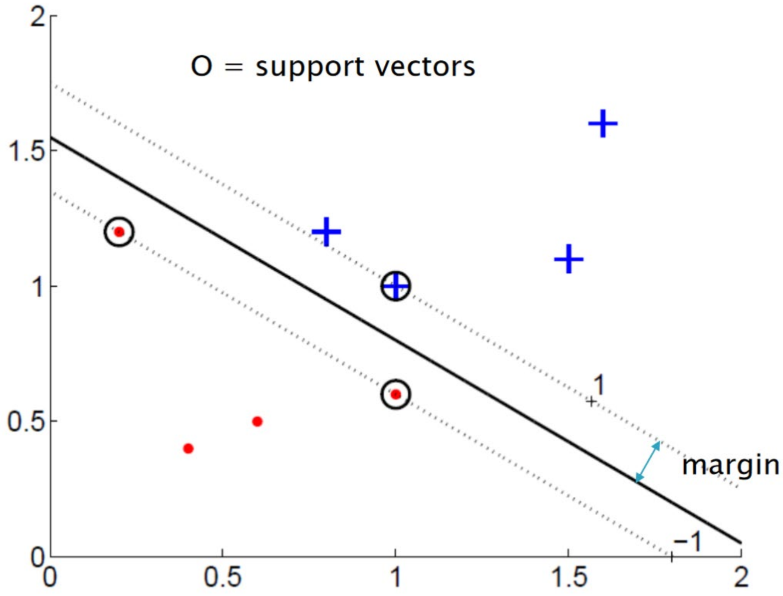
SVM optimization

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_n \alpha_n - \frac{1}{2}\sum_n\sum_m \alpha_n\alpha_m y_n y_m K(\boldsymbol{x}_n, \boldsymbol{x}_m)$$

- Maximize $\mathcal{L}(\boldsymbol{\alpha})$ subj. to $0 \leq \alpha_n \leq C$
- Prediction function is $f(\hat{\boldsymbol{x}}) = \text{sign}(\sum_n \alpha_n y_n K(\boldsymbol{x}_n, \hat{\boldsymbol{x}}))$
- Complexity $O(N^3)$

Which data points should we keep?

- Keep training examples that lie 1 unit away from maximum margin decision boundary
- These are the support vectors
- Intuitively they are the hardest to classify



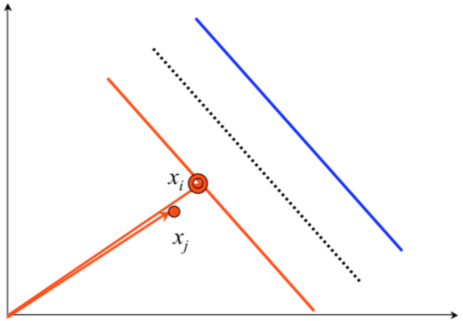
SVM optimization

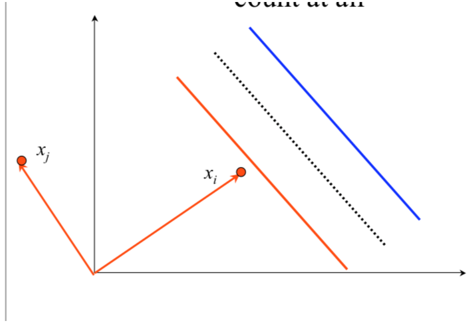
- During optimization, constraints for almost all points disappear
- A small set remain with $\alpha_n \geq 0$
- Generate class label for x
  - sign($w_n x + w_0$)
  - w_0 is average over all support vectors of $y_n - w_n x_n$

Consider similarity of pairs of points

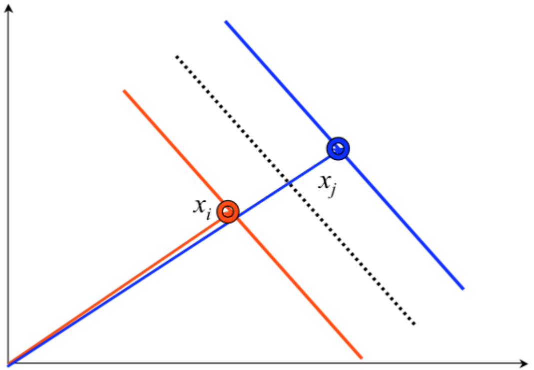
- Consider $y_n = y_m$

$$f(\hat{x}) = \text{sign}(\textstyle\sum_n \alpha_n y_n K(\boldsymbol{x}_n, \hat{\boldsymbol{x}}))$$

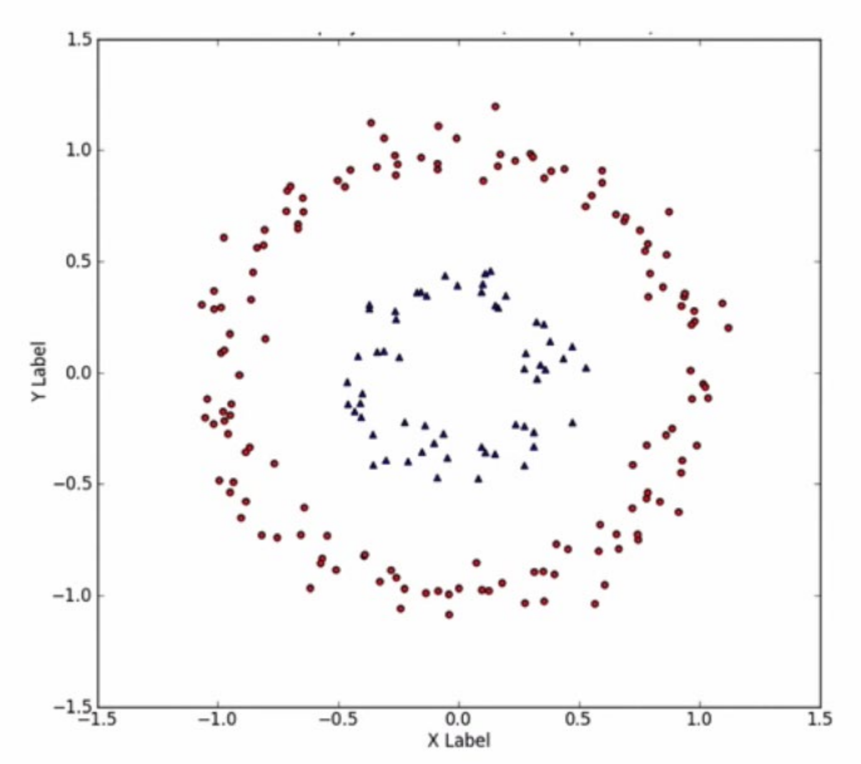




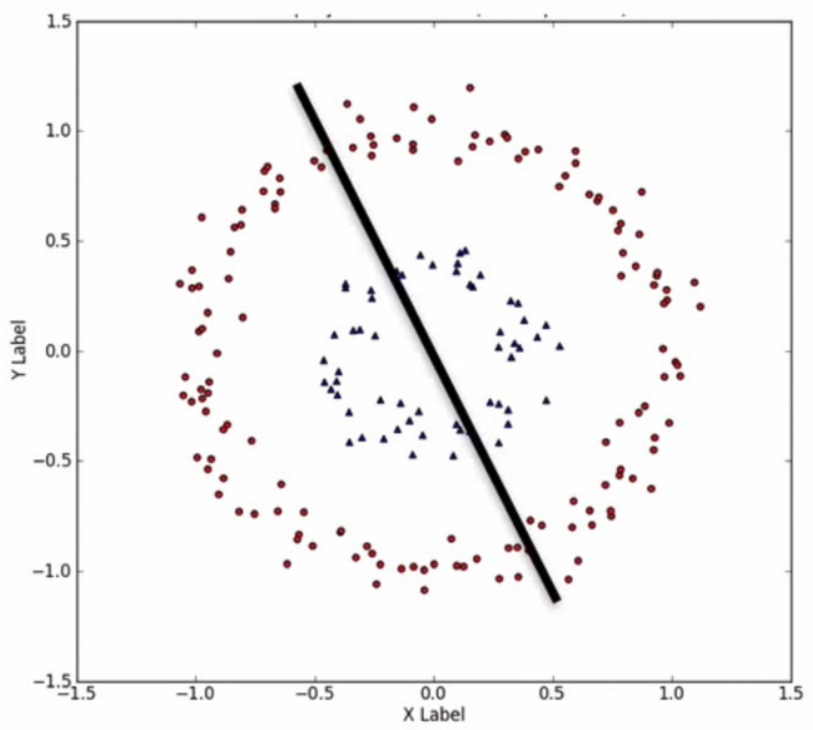
- Consider $y_n \neq y_m$

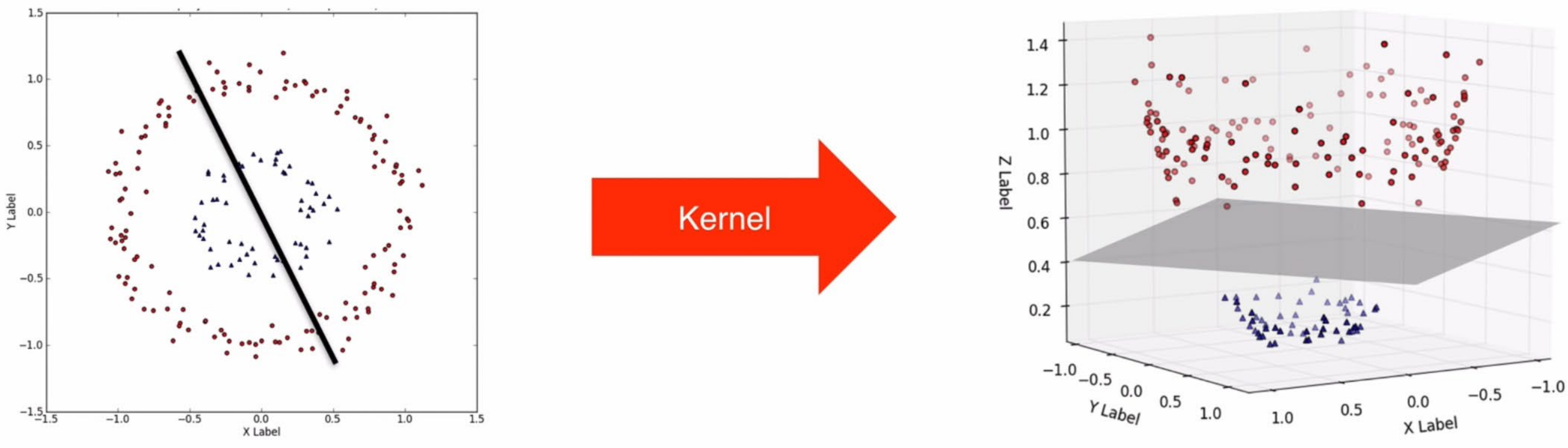


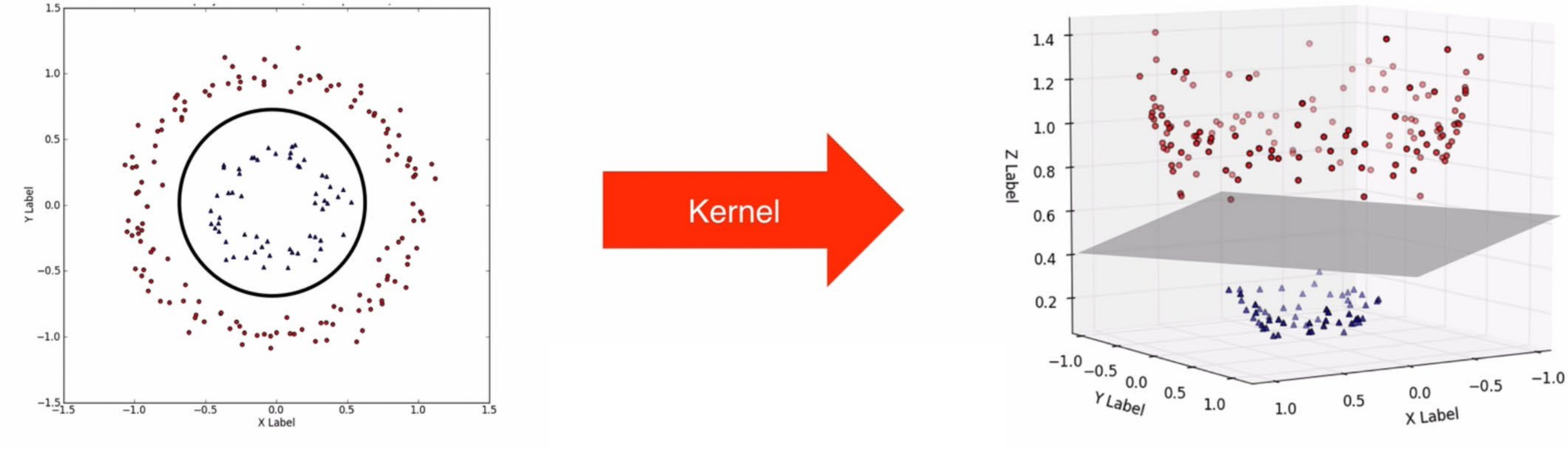
Enhancing learners through kernels



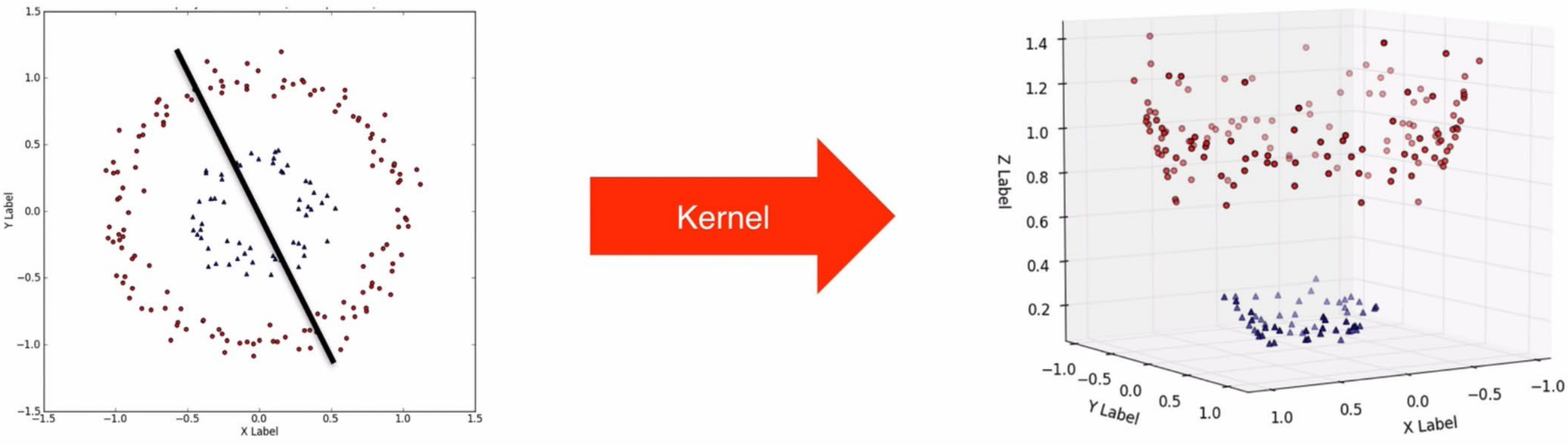
Enhancing learners through kernels




Y Label
X Label
Kernel
Z Label
Y Label
X Label



Kernel
X Label
Y Label
Z Label

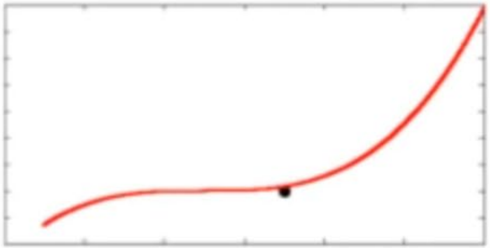

1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
Y Label
X Label
Kernel
1.4
1.2
1.0
0.8
0.6
0.4
0.2
Z Label
−1.0
−0.5
0.0
0.5
1.0
Y Label
1.0
0.5
0.0
−0.5
−1.0
X Label

Creating new features

- Can be computationally expensive
- Suppose f maps from n-dimensional to m-dimensional space, m>>n
- Dot product of x and y in this new space is $f(x)^T f(y)$
- Kernel is function k that corresponds to this dot product
  - $k(x,y) = f(x)^T f(y)$
  - A kernel computes a similarity function

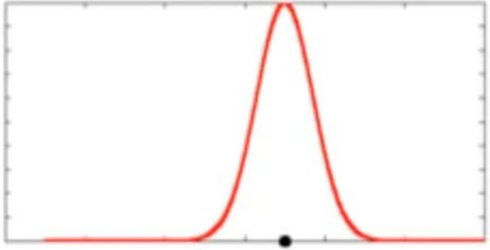
$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_n \alpha_n - \frac{1}{2} \sum_n \sum_m \alpha_n \alpha_m y_n y_m K(\boldsymbol{x}_n, \boldsymbol{x}_m)$$

- Polynomial $K(a,b) = (1 + \sum_j a_j b_j)^d$



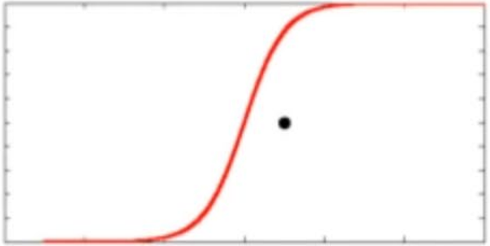
- Radial Basis Functions

$$K(a,b) = \exp(-(a-b)^2/2\sigma^2)$$

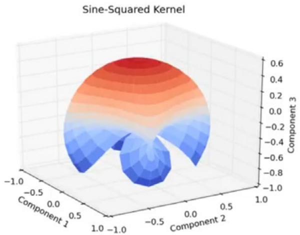


- Saturating, sigmoid-like:

$$K(a,b) = \tanh(ca^T b + h)$$



- Many for special data types:
  - String similarity for text, genetics



SVM highlights

- Built on theoretical machine learning
- Maximize margin
- Only keep support vectors
- Add slack parameters that minimize hinge loss
- Add kernels to introduce new dimensions and minimize computation

Pros

- Effective in high-dimensional spaces
- Alternative kernel functions

Cons

- Poor performance when #features > #samples
- Do not output probability distribution

Let's try this out