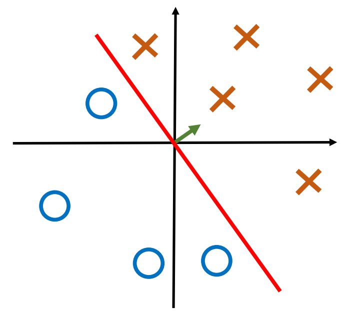
Support Vector Machines

CISC 5800

Professor Daniel Leeds

Separating boundary, defined by **w**

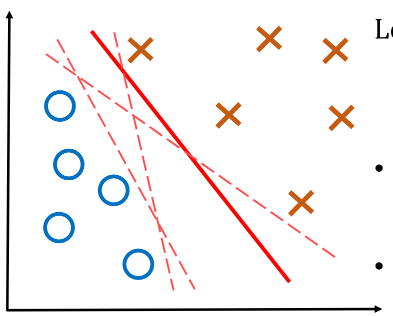


- Separating **hyperplane** splits **class 0** and **class 1**
- Plane is defined by line **w** perpendicular to plan
- Is data point **x** in class 0 or class 1? $\mathbf{w}^T\mathbf{x}+b > 0$ class **1**
  $\mathbf{w}^T\mathbf{x}+b < 0$ class **0**

2

But, where do we place the boundary?

Adjusted Log Likelihood expression



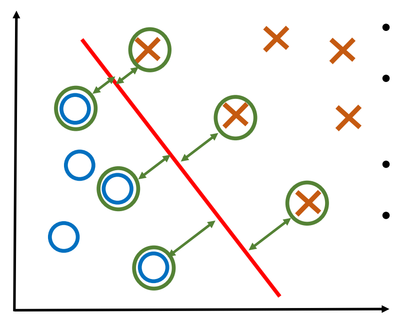
Logistic classifier:

$$LL(y|x;w): \sum_i \left((1-y^i)\ln\left(1-g(x^i;w)\right)+y^i\ln\left(g(x^i;w)\right)\right)$$

- Each data point x^i considered for boundary w
- Outlier data pulls boundary towards it

3

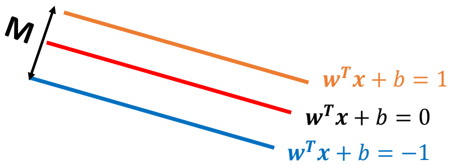
Max margin classifiers



- Focus on boundary points
- Find largest margin between boundary points on both sides
- Works well in practice
- We can call the boundary points **<u>"support vectors"</u>**

4

Maximum margin definitions



Classify as +1
if $w^Tx + b \geq 1$

Classify as -1
if $w^Tx + b \leq -1$

Undefined
if $-1 < w^Tx + b < 1$

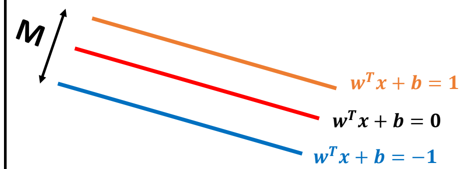
- M is the margin width
- x^+ is a +1 point closest to boundary, x^- is a -1 point closest to boundary
- $x^+ = \lambda w + x^-$
- $|x^+ - x^-| = M$

$$M = \frac{2}{\sqrt{w^Tw}}$$

maximize M **minimize w^Tw**

λ derivation

Optional extra math

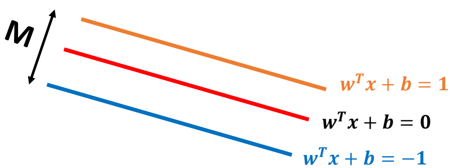


- $w^Tx^- + b = -1$
- $w^Tx^+ + b = +1$
- $x^+ = \lambda w + x^-$

- $w^Tx^+ + b = +1$
- $w^T(\lambda w + x^-) + b = +1$
- $\lambda w^Tw + w^Tx^- + b = +1$
- $\lambda w^Tw - 1 - b + b = +1$
- $\lambda = \frac{2}{w^Tw}$

M derivation

Optional extra math



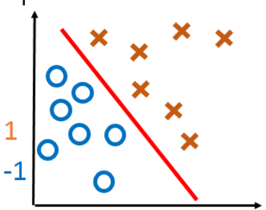
- $w^Tx^- + b = -1$
- $w^Tx^+ + b = +1$
- $x^+ = \lambda w + x^-$
- $|x^+ - x^-| = M$

- $M = |\lambda w + x^- - x^-| = |\lambda w| = \lambda|w|$
- $M = \lambda\sqrt{w^Tw}$
- $M = \frac{2}{w^Tw}\sqrt{w^Tw} = \frac{2}{\sqrt{w^Tw}}$

maximize M minimize w^Tw

Support vector machine (SVM) optimization

argmin$_w$ w^Tw
subject to
$w^Tx + b \geq 1$ for **x** in class 1
$w^Tx + b \leq -1$ for **x** in class -1



$$\text{argmax}_\lambda \ \text{argmin}_w \ w^Tw + \left(\sum_{i\in+1}\lambda_i\left(1-\left(w^Tx^i + b\right)\right) + \sum_{i\in-1}\lambda_i\left(\left(w^Tx^i + b + 1\right)\right)\right)$$

Support vector machine (SVM) optimization

$$\text{argmax}_\lambda \, \text{argmin}_w \, \boldsymbol{w}^T\boldsymbol{w} + \left(\sum_{i\in+1} \lambda_i \left(1 - \left(\boldsymbol{w}^T\boldsymbol{x}^i + b\right)\right) + \sum_{i\in-1} \lambda_i \left(\left(\boldsymbol{w}^T\boldsymbol{x}^i + b + 1\right)\right)\right)$$

Find $\boldsymbol{\lambda}$ that causes highest error

Find **w** that causes lowest error given hardest $\boldsymbol{\lambda}$

Gradient ascent: $\lambda_i \leftarrow \lambda_i + \varepsilon \frac{\partial}{\partial \lambda_i} \mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}, \boldsymbol{\lambda})$

Gradient descent: $w_j \leftarrow w_j - \varepsilon \frac{\partial}{\partial w_j} \mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}, \boldsymbol{\lambda})$

11

Support vector machine (SVM) optimization

$$\text{argmax}_\lambda \, \text{argmin}_w \, \boldsymbol{w}^T\boldsymbol{w} + \left(\sum_{i\in+1} \lambda_i \left(1 - \left(\boldsymbol{w}^T\boldsymbol{x}^i + b\right)\right) + \sum_{i\in-1} \lambda_i \left(\left(\boldsymbol{w}^T\boldsymbol{x}^i + b + 1\right)\right)\right)$$

$$\mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}, \boldsymbol{\lambda}) = \boldsymbol{w}^T\boldsymbol{w} + \left(\sum_{i\in+1} \lambda_i \left(1 - \left(\boldsymbol{w}^T\boldsymbol{x}^i + b\right)\right) + \sum_{i\in-1} \lambda_i \left(\left(\boldsymbol{w}^T\boldsymbol{x}^i + b + 1\right)\right)\right)$$

Gradient ascent: $\lambda_i \leftarrow \lambda_i + \varepsilon \frac{\partial}{\partial \lambda_i} \mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}, \boldsymbol{\lambda})$

Gradient descent: $w_j \leftarrow w_j - \varepsilon \frac{\partial}{\partial w_j} \mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}, \boldsymbol{\lambda})$

Require $\boldsymbol{\lambda} \geq 0$

If $\boldsymbol{\lambda}$ drops below 0, reset to $\boldsymbol{\lambda} = 0$

13

Support vector machine (SVM) optimization

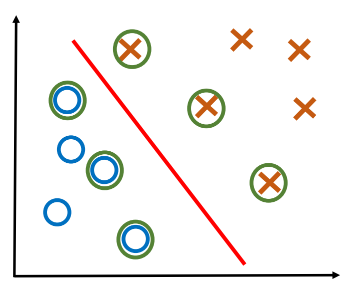
$$\text{argmax}_\lambda \, \text{argmin}_w \, \boldsymbol{w}^T\boldsymbol{w} + \left(\sum_{i\in+1} \lambda_i \left(1 - \left(\boldsymbol{w}^T\boldsymbol{x}^i + b\right)\right) + \sum_{i\in-1} \lambda_i \left(\left(\boldsymbol{w}^T\boldsymbol{x}^i + b + 1\right)\right)\right)$$

Gradient descent: $w_j \leftarrow w_j - \varepsilon \frac{\partial}{\partial w_j} \mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}, \boldsymbol{\lambda})$

$\frac{\partial}{\partial w_j} \mathcal{L}(\boldsymbol{x}, y; \boldsymbol{w}, \boldsymbol{\lambda})$: $2w_j + \left(\sum_{i\in+1} -\lambda_i x_j^i + \sum_{i\in-1} \lambda_i x_j^i\right)$

15

Alternate SVM formulation



$$\boldsymbol{w} = \sum_i \alpha^i \boldsymbol{x}^i y^i$$

Support vectors $\boldsymbol{x}_i$ have $\alpha_i > 0$

y_i are the data labels +1 or -1

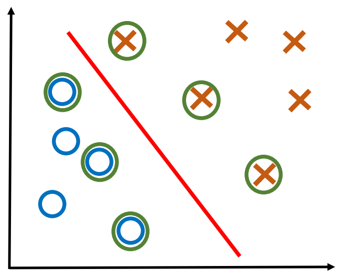
$\boldsymbol{\alpha}^i \geq \mathbf{0} \; \forall \boldsymbol{i}$ $\qquad$ $\sum_i \boldsymbol{\alpha}^i \, \boldsymbol{y}^i = \mathbf{0}$

17

Example

$\boldsymbol{\alpha^i} \geq \mathbf{0}\ \forall \boldsymbol{i}$

$\sum_i \boldsymbol{\alpha^i}\ \boldsymbol{y^i} = \mathbf{0}$



$$\boldsymbol{w} = \sum_i \alpha^i \boldsymbol{x}^i y^i$$

$$\boldsymbol{x}^1 = \begin{bmatrix}-1\\1\end{bmatrix}, y^1 = +1, \alpha^1 = 0.5$$

$$\boldsymbol{x}^2 = \begin{bmatrix}0\\0\end{bmatrix}, y^2 = +1, \alpha^2 = 0.7$$

$$\boldsymbol{x}^3 = \begin{bmatrix}-1\\-1\end{bmatrix}, y^3 = -1, \alpha^3 = 1$$

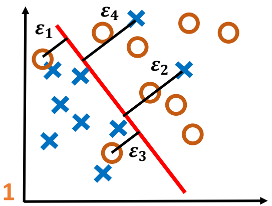
$$\boldsymbol{x}^4 = \begin{bmatrix}-0.5\\-3\end{bmatrix}, y^4 = -1, \alpha^4 = 0.2$$

$$\boldsymbol{w} = 0.5 \times \begin{bmatrix}-1\\1\end{bmatrix} + 0.7 \times \begin{bmatrix}0\\0\end{bmatrix} - 1 \times \begin{bmatrix}-1\\-1\end{bmatrix} - 0.2 \times \begin{bmatrix}-0.5\\-3\end{bmatrix} = \begin{bmatrix}-0.5+1+0.1\\0.5+1+0.6\end{bmatrix} = \begin{bmatrix}\mathbf{0.6}\\\mathbf{2.1}\end{bmatrix}$$

19

Support vector machine (SVM) optimization
with slack variables

What if data not completely ~~linearly~~ separable?



$$\text{argmin}_{\text{w,b}}\ \boldsymbol{w}^T\boldsymbol{w} + C\sum_i \varepsilon^i$$

subject to

$\boldsymbol{w^T x} + b \geq 1 - \varepsilon^i$ **for x in class 1**

$\boldsymbol{w^T x} + b \leq -1 + \varepsilon^i$ **for x in class -1**

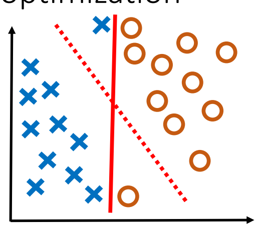
$\varepsilon^i \geq 0 \quad \forall i$

Each error ε^i is penalized based on distance from separator

20

Support vector machine (SVM) optimization
with slack variables

Example: Linearly separable but with narrow margins



$$\text{argmin}_{\text{w,b}}\ \boldsymbol{w}^T\boldsymbol{w} + C\sum_i \varepsilon^i$$

subject to

$\boldsymbol{w^T x} + b \geq 1 - \varepsilon^i$ **for x in class 1**

$\boldsymbol{w^T x} + b \leq -1 + \varepsilon^i$ **for x in class -1**

$\varepsilon_i \geq 0 \quad \forall i$

21

Hyper-parameters for learning

$$\text{argmin}_{\text{w,b}}\ \boldsymbol{w}^T\boldsymbol{w} + C\sum_i \varepsilon_i$$

Optimization constraints: ***C*** influences tolerance for label errors versus narrow margins

$$w_j \leftarrow w_j + \varepsilon \boldsymbol{x}_j^i\left(y^i - g(\boldsymbol{w}^T\boldsymbol{x}^i)\right) - \frac{w_j}{\lambda}$$

Gradient ascent:

- $\boldsymbol{\varepsilon}$ influences effect of individual data points in learning
- ***T*** number of training examples, ***L*** number of loops through data – balance learning and over-fitting

Regularization: $\boldsymbol{\lambda}$ influences the strength of your prior belief

22

Parameter counts

Each data point $\boldsymbol{x}^i$ has N features (presuming classify with $\boldsymbol{w}^T\boldsymbol{x}^i+b$)

Separator: $\boldsymbol{w}$ and b

- N elements of **w**, 1 value for b: $N+1$ parameters **OR**
- t support vectors -> t non-zero α^i, 1 value for b: $t+1$ parameters

23

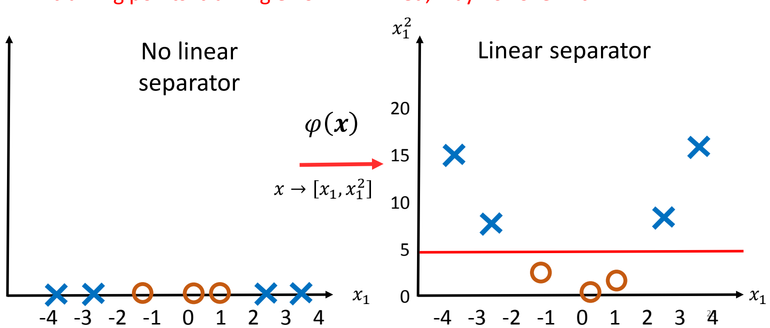
Binary -> *M*-class classification

- Learn boundary for class m vs all other classes
  - Only need M-1 separators for M classes – Mth class is for data outside of classes 1, 2, 3, …, M-1
- Find boundary that gives highest margin for data points $\mathbf{x}^i$

24

Classifying with additional dimensions

Note: More dimensions makes it easier to separate T training points: training error minimized, may risk over-fit



Quadratic mapping function (math)

$\boldsymbol{w}^T\boldsymbol{x}^k + b = \sum_i \alpha^i\, y^i (\boldsymbol{x}^i)^T \boldsymbol{x}^k + b$

x_1, x_2, x_3, x_4 -> $x_1, x_2, x_3, x_4, x_1^2, x_2^2, \ldots, x_1x_2, x_1x_3, \ldots, x_2x_4, x_3x_4$

N features -> $N + N + \frac{N\times(N-1)}{2} \approx N^2$ features

N^2 values to learn for w in higher-dimensional space

Or, observe: $(\boldsymbol{v}^T\boldsymbol{x}+1)^2 = \boldsymbol{v}_1^2 x_1^2 + \cdots + \boldsymbol{v}_N^2 x_N^2$
$+ \boldsymbol{v}_1\boldsymbol{v}_2 x_1 x_2 + \cdots + \boldsymbol{v}_{N-1}\boldsymbol{v}_N x_{N-1} x_N$
$+ \boldsymbol{v}_1 x_1 + \cdots + \boldsymbol{v}_N x_N$

v with N elements operating in quadratic space

26

Quadratic mapping function *Simplified*

x= [x_1, x_2] -> [$\sqrt{2}x_1$, $\sqrt{2}x_2$, x_1^2, x_2^2, $\sqrt{2}x_1x_2$, 1]

x^i=[5,-2] -> [10, -4, 25, 4, -20, 1] x^k=[3,-1] -> [6, -2, 9, 1, -6, 1]

$$\varphi(\boldsymbol{x}^i)^T\varphi(\boldsymbol{x}^k) = 30 + 4 + 225 + 4 + 60 + 1 = 324$$

Or, observe: $\left(\boldsymbol{x}^{i^T}\boldsymbol{x}^k + 1\right)^2 = \left((15+2)+1\right)^2 = (18)^2$ =324

27

Mapping function(s)

- Map from low-dimensional space $\boldsymbol{x} = (x_1, x_2)$ to higher dimensional space $\varphi(\boldsymbol{x}) = (\sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2, 1)$
- N data points guaranteed to be separable in space of N-1 dimensions or more

$$\boldsymbol{w} = \sum_i \alpha_i \varphi(\boldsymbol{x}^i) y^i$$

Classifying $\boldsymbol{x}^k$:

$$\sum_i \alpha_i y^i \varphi(\boldsymbol{x}^i)^T \varphi(\boldsymbol{x}^k) + b$$

29

Kernels

Classifying $\boldsymbol{x}^k$:

$$\sum_i \alpha_i y^i \varphi(\boldsymbol{x}^i)^T \varphi(\boldsymbol{x}^k) + b$$

Kernel trick:

- Estimate high-dimensional dot product with function
- $K(\boldsymbol{x}^i, \boldsymbol{x}^k) = \varphi(\boldsymbol{x}^i)^T \varphi(\boldsymbol{x}^k)$

Now classifying $\boldsymbol{x}^k$

$$\sum_i \alpha_i y^i K(\boldsymbol{x}^i, \boldsymbol{x}^k) + b$$

30

Radial Basis Kernel

Try projection to infinite dimensions

$$\varphi(\boldsymbol{x}) = \left[x_1, \cdots, x_n, x_1^2, \cdots, x_n^2, \cdots, x_1^\infty \cdots, x_n^\infty\right]$$

Taylor expansion: $e^x = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^\infty}{\infty!}$

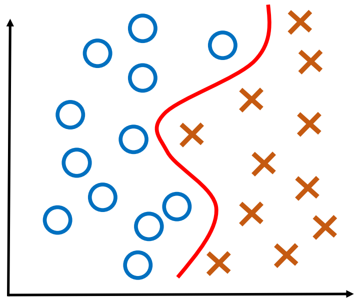
$$K(\boldsymbol{x}^i, \boldsymbol{x}^k) = \exp\left(-\frac{(\boldsymbol{x}^i - \boldsymbol{x}^k)^2}{2\sigma^2}\right)$$

Note: $(\boldsymbol{x}^i - \boldsymbol{x}^k)^2 = (\boldsymbol{x}^i - \boldsymbol{x}^k)^T(\boldsymbol{x}^i - \boldsymbol{x}^k)$

Draw separating plane to curve around all support vectors

31

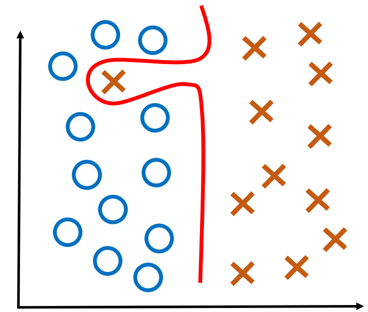
Example RBF-kernel separator



Large margin

Non-linear separation

Potential dangers of RBF-kernel separator



Small margin - **overfitting**

Non-linear separation

The power of SVM (+kernels)

Boundary defined by a few support vectors

- Caused by: maximizing margin
- Causes: less overfitting
- Similar to: regularization

Kernels keep number of learned parameters in check

Benefits of generative methods

- $P(\boldsymbol{D}|\boldsymbol{\theta})$ and $P(\boldsymbol{\theta}|\boldsymbol{D})$ can generate non-linear boundary
- E.g.: Gaussians with multiple variances