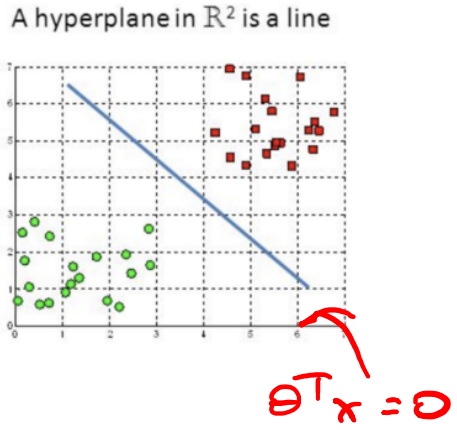
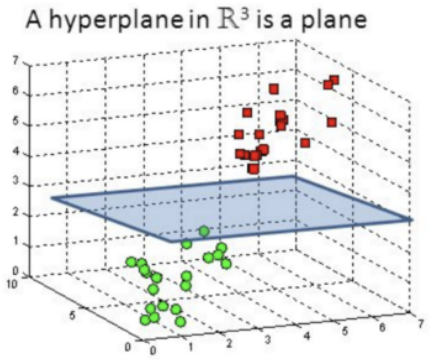
SVMs

- Support Vector Machines (SVMs) are one of the most important developments in pattern recognition in recent years
- Elegant and successful

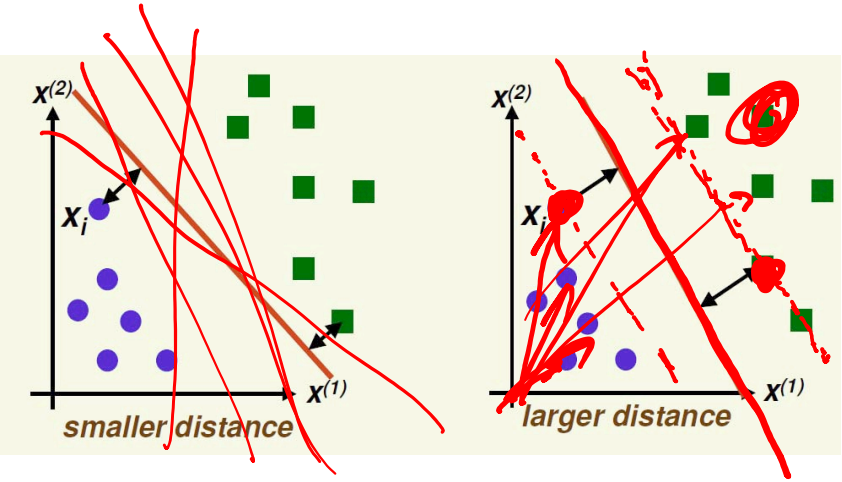
https://towardsdatascience.com





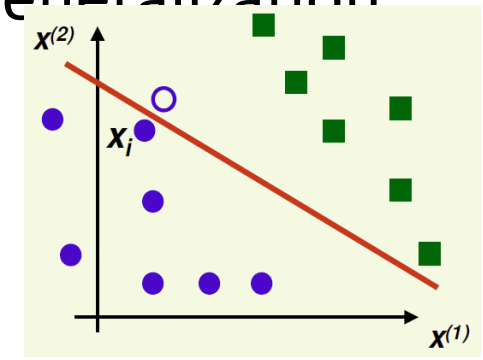
SVM Intuition

- Idea: Maximize distance to closest example (of each type)
  - For now we'll assume total separability



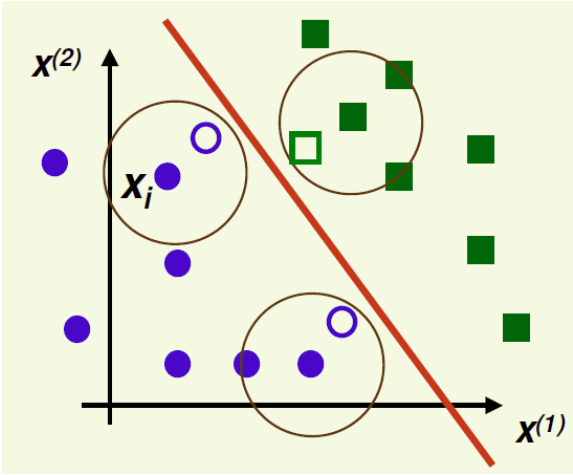
SVM Intuition

- Training data is just a subset of all possible data
  - Suppose hyperplane is close to sample X_i (open circle)
    - The *margin* is small
  - If we see a new sample close to X_i it may be on the wrong side of the hyperplane
- Therefore poor generalization



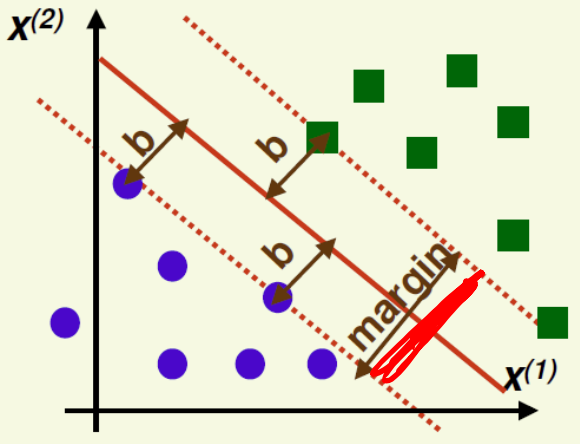
SVM Intuition

- Intuition: We want hyperplane as far as possible from any sample
- New samples close to old samples will then be classified correctly
- Good generalization



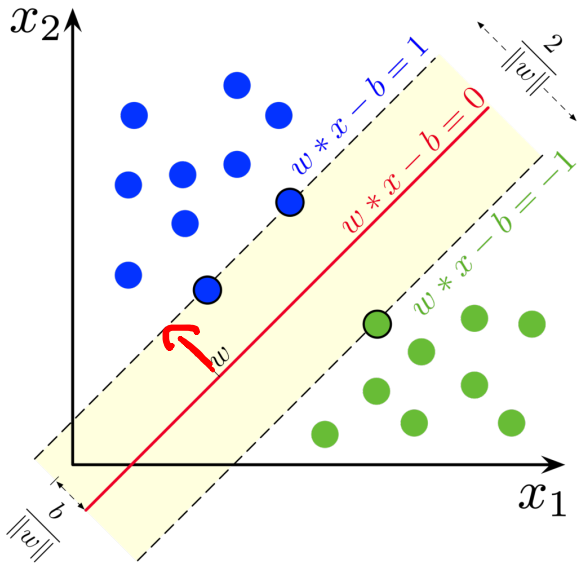
SVM – Linearly Separable Case

- Definition: Margin
  - The margin is twice the absolute value of distance b of the closest example to the hyperplane
- Our goal is to maximize the margin



Linear Discriminant Functions

- The general equation for a line is
$$Ax + By + C = 0$$
- This equation holds for any point (x, y) on the line.
- We can write this as a *discriminant function:*
$$g(x, y) = Ax + By + C$$
  - $g(x, y) = 0$ for an point (x, y) **on** the line.
  - $g(x, y) > 0$ for any point **above** the line
  - $g(x, y) < 0$ for any point **below** the line.
- Or in a linear algebra sense
$$g(x) = [x, y, 1] \begin{bmatrix} A \\ B \\ C \end{bmatrix}$$
- If we add a bias feature…
$$g(x) = xw$$

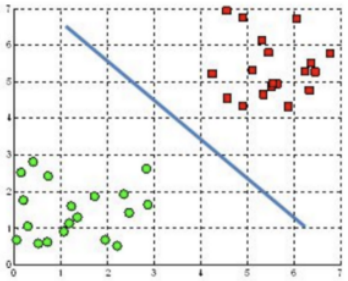


Linear Discriminant Functions

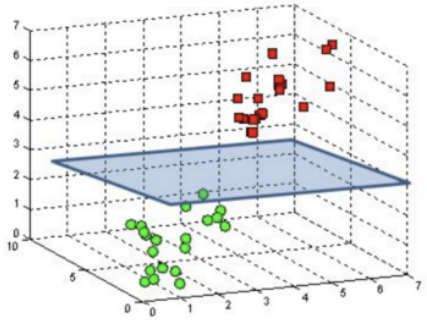
$$g(x) = xw$$

- If x and w have more than three features (including the bias feature) then this is the discriminant function for a **hyperplane.**
- If $g(x) = 0$ then we're on the hyperplane.
- If $g(x) > 0$ then we're on one side of the hyperplane
- If $g(x) < 0$ then we're on the other side of it.

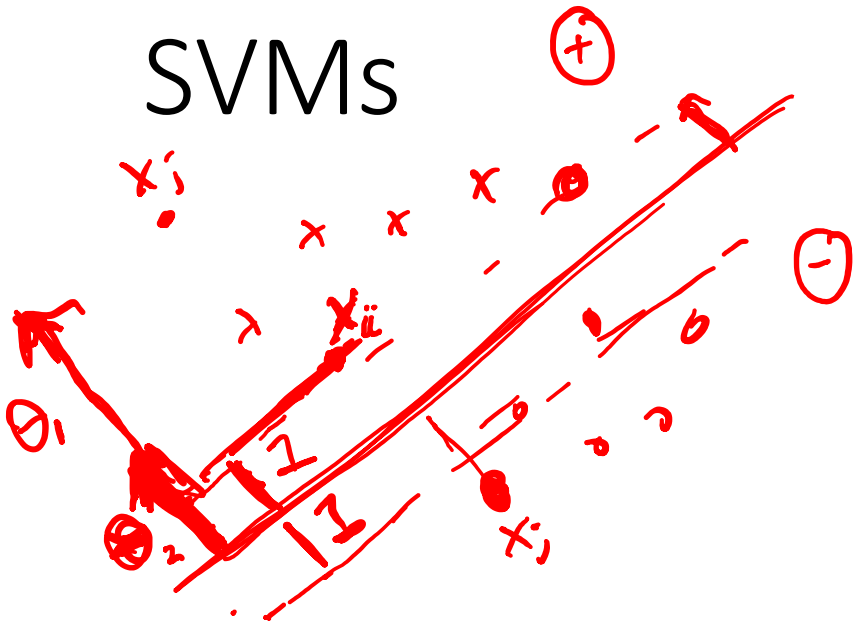
A hyperplane in $\mathbb{R}^2$ is a line



A hyperplane in $\mathbb{R}^3$ is a plane



SVMs



$y_i \in 0, 1$

$y_i \in -1, 1$

$$\frac{\theta^T x_i}{\|\theta\|}$$

$\theta^T x > 0 \quad y_i = 1$

$\theta^T x < 0 \quad y_i = -1$

$$\underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} y_i \theta^T x_i$$

$\theta_1^T x_i$, $\theta_2^T x_i$ — sign remains same

$\theta_1^T x_j$, $\theta_2^T x_j$ — distance relative

Case 1

$y_i * \theta^T x_i$

$y_i \theta^T x_i > 0$

$y_i = 1$

$\theta^T x_i$ is (+)

correct

Case 2

$y_j * \theta^T x_j$

$y_j = -1$

$\theta^T x_j$ is (−)

$y_i \theta^T x_i > 0$

Case 3

if $y_i = -1$

$\theta^T x_i$ (+)

$y_i \theta^T x_i < 0$

Case 4 $y_i = +1$

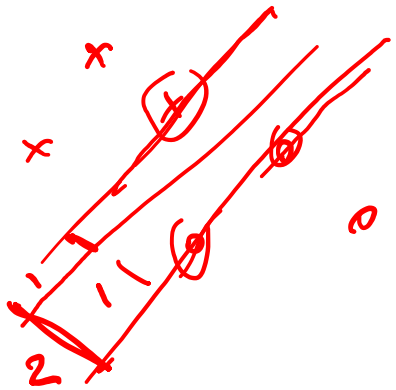
negative $\theta^T x_i$ (−)

$a_1x_1 + a_2x_2 \cdots a_nx_n$

$x_1a_1 + x_2a_2 \cdots x_na_n$) scale support vectors

SVMs

distance 1 from boundary



$\theta^T x_+ + b \geq 1$

$\theta^T x_- + b \leq -1$

$\theta^T x_+ + b = 1$ support

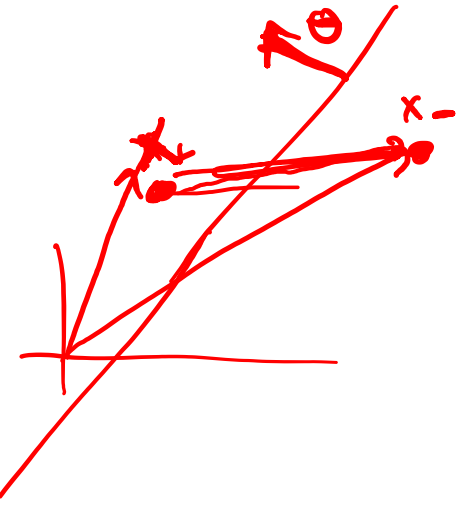
$\theta^T x_- + b = -1$

$y_i \theta^T x_i + b \geq 1$

$y_i \theta^T x_i + b - 1 \geq 0$

$\theta^T x_+ = 1 - b$

$\theta^T x_- = -1 - b$



$(x_+ - x_-) \cdot \frac{\theta}{\|\theta\|} = w$

$\frac{x_+ \cdot \theta}{\|\theta\|} - \frac{x_- \cdot \theta}{\|\theta\|} = w$

$\frac{1 - b - (-1 - b)}{\|\theta\|} = \frac{2}{\|\theta\|}$

max

$\frac{\|\theta\|}{2}$

min

$\frac{1}{2}\|\theta\|^2$

Margin Formula

$$\boxed{g(x) = xw}$$

- Let x_+^* and x_-^* be the samples closest to the hyperplane (which we call **support vectors**) , and designate that:

$$g(x_+^*) = 1$$
$$g(x_-^*) = -1$$

$$width = x_+^* w - x_-^* w$$

- The distance of x from the hyperplane is:

$$d(x|w) = \frac{xw}{||w||}$$

- Therefore

$$d(x_+^*) = \frac{1}{||w||}$$

- And the margin will be:

$$margin = J(w) = d(x_+^*) - d(x_-^*) = \frac{2}{||w||}$$

- Do we want to maximize or minimize this?

Margin Formula

$$J(w) = \frac{2}{||w||}$$

- Do we want to maximize or minimize this?
- Or conversely minimize:

$$J(w) = \frac{||w||}{2} = \frac{1}{2} w^T w$$

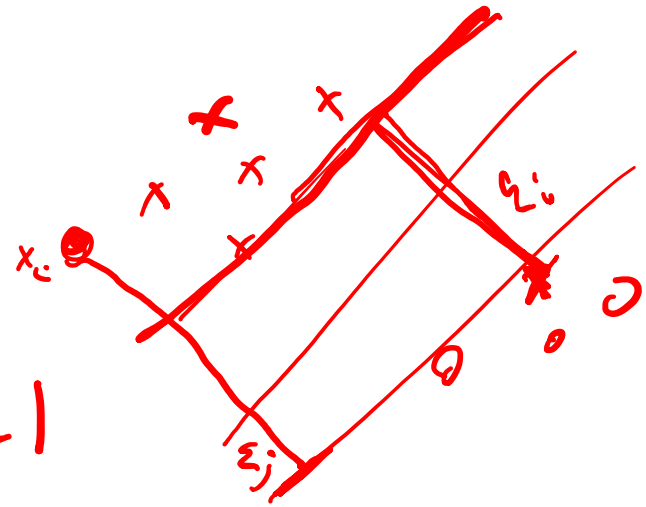
- However
  - Why not just choose $w = [\infty, \infty, \infty, ..., \infty]$?
  - And what about wanting $x_+ w \geq 1, x_- w \leq -1$?

hyper parameter

Slack variables

$$\min \frac{1}{2}\|\theta\|^2$$

$$\text{s.t. } \forall i,\ y_i \theta^T x_i + b \geq 1$$



margin — mistakes

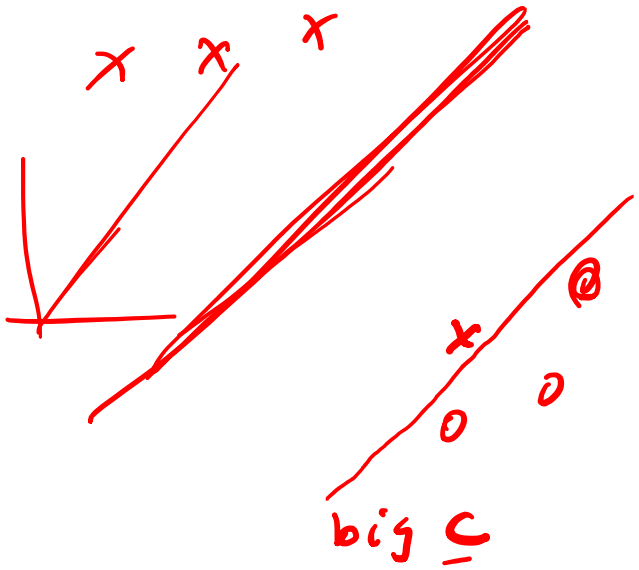
$$\frac{1}{2}\|\theta\|^2 + C\,(\#\text{ of mistakes})$$

$$\text{s.t. } \forall i,\ y_i \theta^T x_i + b \geq 1$$

$$\frac{1}{2}\|\theta\|^2 + C\sum_{i=1}^{N} \xi_i$$

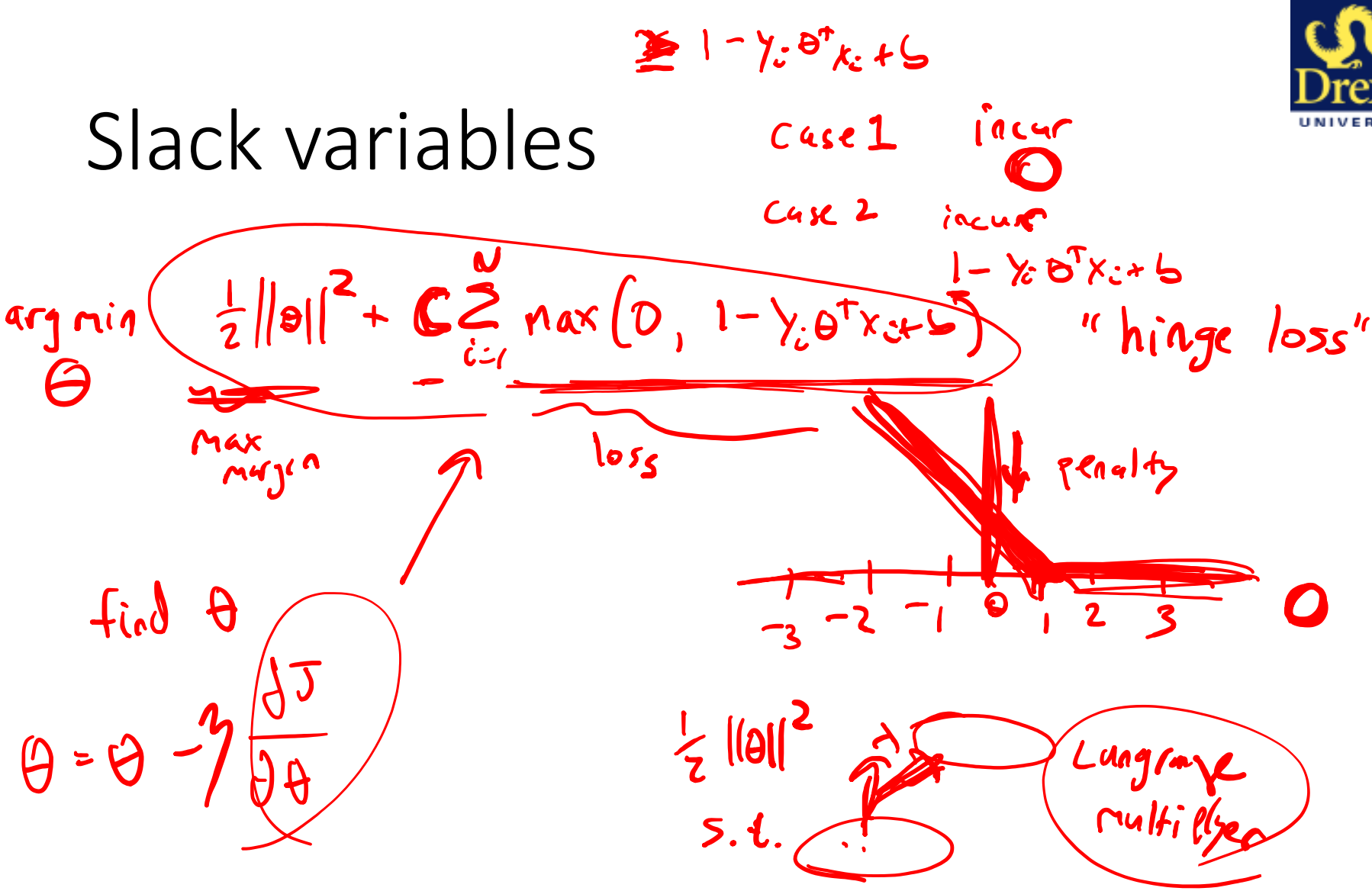
$$\text{s.t. } \forall i,\ y_i \theta^T x_i + b \geq 1 - \xi_i$$

$$\xi_i \geq 1 - y_i \theta^T x_i + b$$



big C

Slack variables



Margin Formula

$$J(w) = \frac{1}{2} w^T w$$

- The *loss* (error) attributed to a positive sample can be written as:
$$\max(0, 1 - x_+ w)$$
- And the loss attributed to a negative sample can be written as:
$$\max(0, 1 + x_- w)$$
- If we let $y_+ = 1$ and $y_- = -1$ then we can generalize this as:
$$\max(0, 1 - y x w)$$
- And our overall objective function is to minimize:
$$J(w) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \max(0, 1 - Y_i X_i w)$$
  - Where C is a blending hyperparameter.

Finding w

$$J(w) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \max(0, 1 - Y_i X_i w)$$

- How do we find w?
- Let's do gradient descent!
- Let's first look at online gradient descent:

$$J(w) = \frac{1}{2} w^T w + C \max(0, 1 - Y_i X_i w)$$

Finding w

$$J(w) = \frac{1}{2} w^T w + C \max(0, 1 - Y_i X_i w)$$

- So what are the gradients?

$$\frac{d}{dw}\left(\frac{1}{2} w^T w\right) = w$$

$$\frac{d}{dw}(C \max(0, 1 - Y_i X_i w)) = \begin{cases} 0 & if \ Y_i X_i w \geq 1 \\ -C Y_i X_i^T & otherwise \end{cases}$$

- So

$$\frac{dJ}{dw} = \begin{cases} w & if \ Y_i X_i w = 0 \\ w - C Y_i X_i^T & otherwise \end{cases}$$

- And of course we could do something similar for batch, mini-batch, or stochastic gradient descent.

Simple Example

- Given data

$$X = \begin{bmatrix} -1 & -1 \\ -1 & 1 \\ 1 & -1 \\ 1 & 1 \end{bmatrix} \qquad Y = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix}$$

- So that we don't need to worry about w_0, lets add a bias feature so:

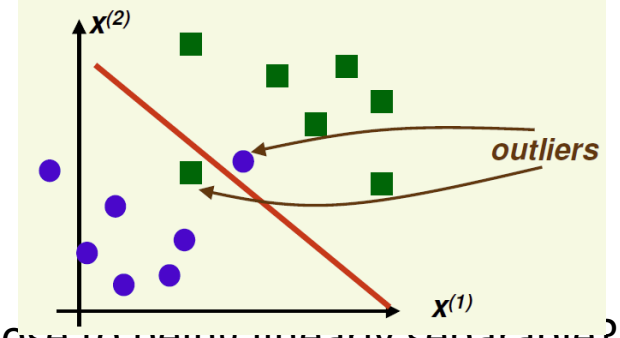
$$X = \begin{bmatrix} 1 & -1 & -1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \\ 1 & 1 & 1 \end{bmatrix}$$

- Choosing $C = 1, \eta = 0.1$ and full batch GD, the solution converges after only a few iterations to:

$$w = [1\ 0\ 0]^T$$

Non-Separable Situations

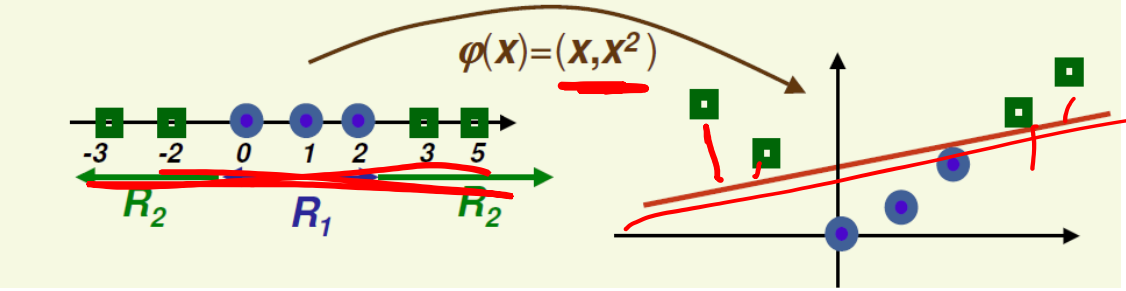
- In the previous example, the data was perfectly linearly separable.
- But in most cases it is not ☹
- Still, the gradient descent approach can work well (varying C accordingly) if our data just has a few outliers.



- But what if it's not even close to being linearly separable?

Mapping Functions

- A *mapping function*, $\phi(x)$, takes use from the feature space of x to some higher dimensional feature space
- A higher dimensional space is more easily linearly separable.
- For instance, imagine the mapping function $\phi(x) = (x, x^2)$ on the data below:



Non-Linear Situations

- So we could explicitly first move our data to a higher dimensional space, and then create a SVM there:

$$\frac{dJ}{dw} = \begin{cases} w & if\ Y_i\phi(X_i)w = 0 \\ w - CY_i\phi\left(X_i^T\right) & otherwise \end{cases}$$

- Then determine class with:

$$\widehat{Y}_i = \phi(X_i)w$$

(Very) Non-Linear Situations

- The mapping function technique works ok some mapping functions, but it's often unclear what good mapping functions are.
  - Wasn't that mapping function for the simple xor problem confusing?
  - Furthermore, going to higher dimensional space explicitly can result in ill-effects due to the curse of dimensionality.
- Other solutions (in particular the dual solution) can leverage something called the *kernel trick* in order to overcome these limitations.
  - But for the sake of time and the dual solution's complexity, we'll save this for your own reading.

(Very) Non-Linear Situations

- That being said…..
- At least be aware of your options!
- Most SVM implementations include the dual-form and therefore allow you to specify a kernel.
- Typical choices are:
  - Linear – Basically working in the original feature space.
  - Polynomial
    - Quadratic being a special case of this.
  - Gaussian Radial Basis Function (RBF)
    - Simulates working in an infinitely high dimensional space.

Kernel Choice

- Which kernel should we use?
  - Depends on how many features we have compared to how many observations we have.
  - Working in too low of a space (as a rule of thumb, where we have more observations than features) we may *underfit*, so perhaps we can benefit in going to a higher space.
  - Conversely, working into too high of a space can result in overfitting.
    - Although intrinsically SVMs look to avoid this.

Multiple Classes

- Prior to Support Vector machines, all of our classification algorithms allowed for multi-class classification directly
  - Inference
  - Naïve Bayes
  - Decision Trees
  - K-Nearest Neighbors
- But some classifiers, like SVMs, only support binary classification “out of the box”
- How can we then do multi-class classification given just binary classifiers?

Multiple Classes

- There are two relatively straightforward ways to do multi-class classification from a set of binary classifiers:
  1. One vs All (need K classifiers) – often imbalanced, ambiguous regions
     - Choose the one that does best
  2. One vs one ($\frac{K(K-1)}{2}$ classifiers) – lots of classifiers
     - Fewer ambiguous regions
     - Choose the one that gets the most votes.

Resources

- https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
- https://www.saedsayad.com/support_vector_machine.htm
- https://svivek.com/teaching/lectures/slides/svm/svm-sgd.pdf