CS 360: Machine Learning

Prof. Sara Mathieson

Fall 2019

HAVERFORD COLLEGE

Admin

- **Midterm 2 TODAY!**

- No office hours Friday

- **Final project presentations:**
  - Wednesday Dec 18: 1-4pm (block out the entire time, but we may not need all of it)
  - Option to present last day of class (email me)

Outline for November 21

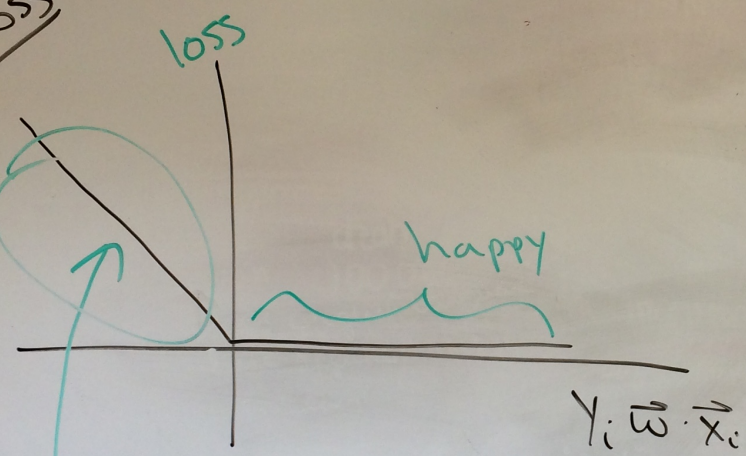
- Support Vector Machines Review
- Likelihood functions (Bernoulli and Logistic Regression)
- Finish practice problems (Q2-Q4)

Outline for November 21

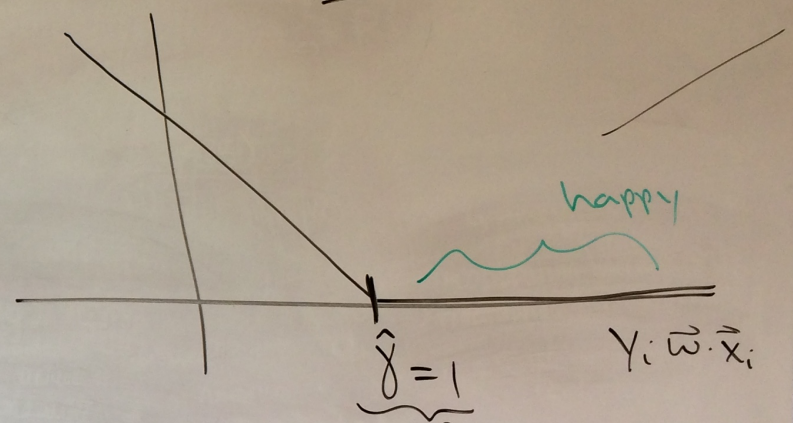
- Support Vector Machines Review
- Likelihood functions (Bernoulli and Logistic Regression)
- Finish practice problems (Q2-Q4)

Perceptron & SVM: Same goal: separating hyperplane

hinge loss



SVM



Perceptron: $J_i(\vec{w}) = \max(0, -y_i \vec{w} \cdot \vec{x}_i)$

if incorrect: $\boxed{\vec{w} \leftarrow \vec{w} + y_i \vec{x}_i}$ SGD

$$J_i(\vec{w}) = \max(0, 1 - y_i \vec{w} \cdot x_i)$$

SVM #1 Goal

maximize geometric margin

$$\max \gamma = \frac{\hat{\gamma}}{\|\vec{w}\|}$$

$\hat{\gamma} \to 1$

$\|\vec{w}\| \leftarrow$ min

func

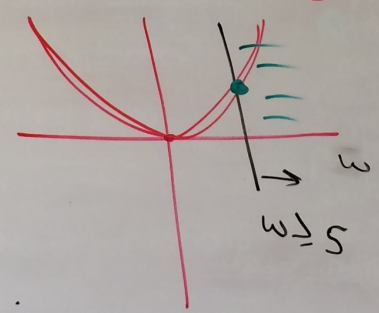
$$\min_{\vec{w}, b} \frac{1}{2}\|\vec{w}\|^2$$

Constraint

$$\text{s.t.} \quad \underbrace{y_i}_{\text{truth}} \underbrace{(\vec{w} \cdot \vec{x}_i + b)}_{\text{prediction}} \geq 1$$

pt's. functional margin.

$i = 1 \cdots n$



Lagrangian

"primal"

$$L(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(\vec{w} \cdot \vec{x}_i + b) - 1 \right]$$

$\Rightarrow 0$

Lagrange multipliers

$\alpha_i = 0 \Rightarrow$ constraint not used

$\alpha_i > 0 \Rightarrow y_i(\vec{w} \cdot \vec{x}_i + b) - 1 = 0$

support vectors

$$L(x, y, \lambda) = \underbrace{f(x, y)}_{\text{min or max}} - \lambda \underbrace{g(x, y)}_{\text{constraint}}$$

$\Rightarrow g(x, y) = 0$

take derivative wrt $\vec{w}$ & b

"dual"

$$W(\vec{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j$$

$\alpha_i \geq 0 \quad \forall i, \quad \sum_{i=1}^{n} y_i \alpha_i = 0$

$\boxed{\text{gradient method}}$

$\Rightarrow$ $\boxed{\text{output: } \vec{\alpha}}$

$$\boxed{\vec{w}^* = \sum_{i=1}^{n} \alpha_i^* y_i \vec{x}_i}$$ weight vector

$\Rightarrow$ prediction: $h(\vec{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \vec{x}_i \cdot \vec{x} + b^* \right)$

replace with any kernel I want

measure of similarity

$K_{dot}(\vec{x}, \vec{z}) = \vec{x} \cdot \vec{z}$

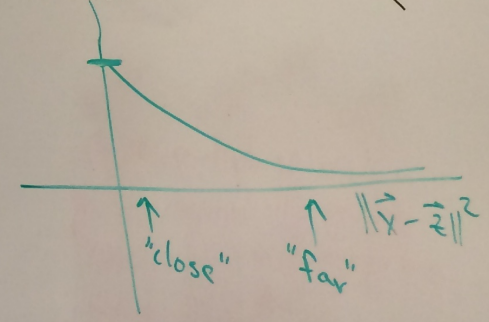
$\vec{x} =$ A G T A C G

$\vec{z} =$ A G A A T G

$K_{dna}(\vec{x} \cdot \vec{z}) = 4$

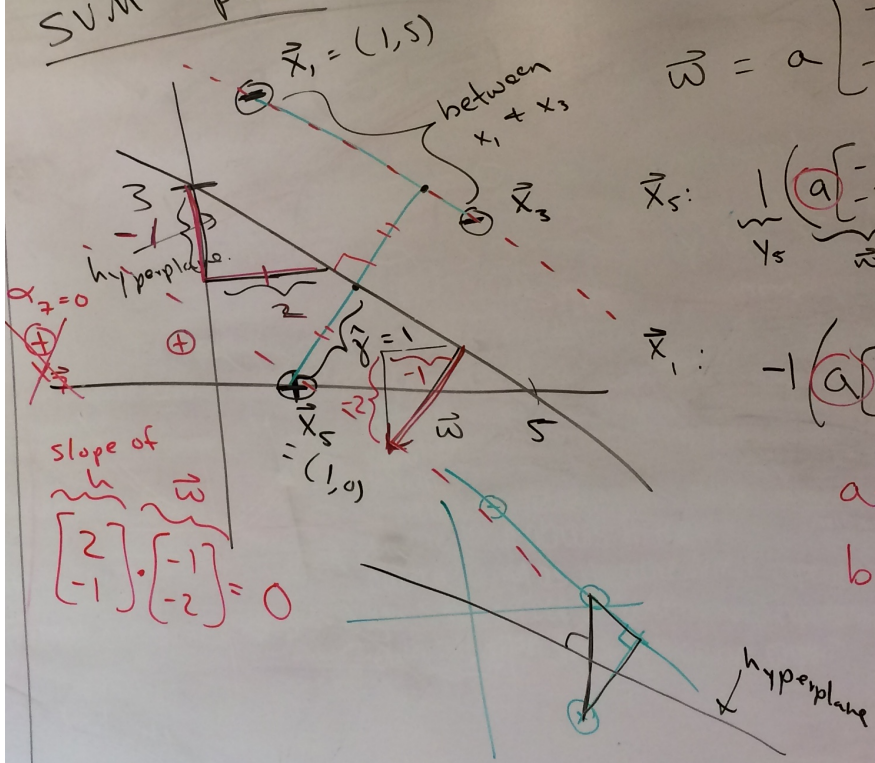
not the margin!

$K_{rbf}(\vec{x}, \vec{z}) = \exp\left(-\gamma \cdot \|\vec{x} - \vec{z}\|^2\right)$



$\vec{w}$
hyperplane

SVM pset



$$\vec{w} = a \begin{bmatrix} -1 \\ -2 \end{bmatrix}$$

$$\vec{x}_5: \quad \underbrace{1}_{y_5} \Big(a \underbrace{\begin{bmatrix} -1 \\ -2 \end{bmatrix}}_{\vec{w}} \cdot \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{\vec{x}_5} + b \Big) = 1$$

$$\vec{x}_1: \quad -1 \Big(a \begin{bmatrix} -1 \\ -2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 5 \end{bmatrix} + b \Big) = 1$$

$$a = \frac{1}{5}$$

$$b = \frac{6}{5}$$

$$\vec{w} = \begin{bmatrix} -1/5 \\ -2/5 \end{bmatrix}$$

Perceptron Recap

$$\alpha = 0.2$$

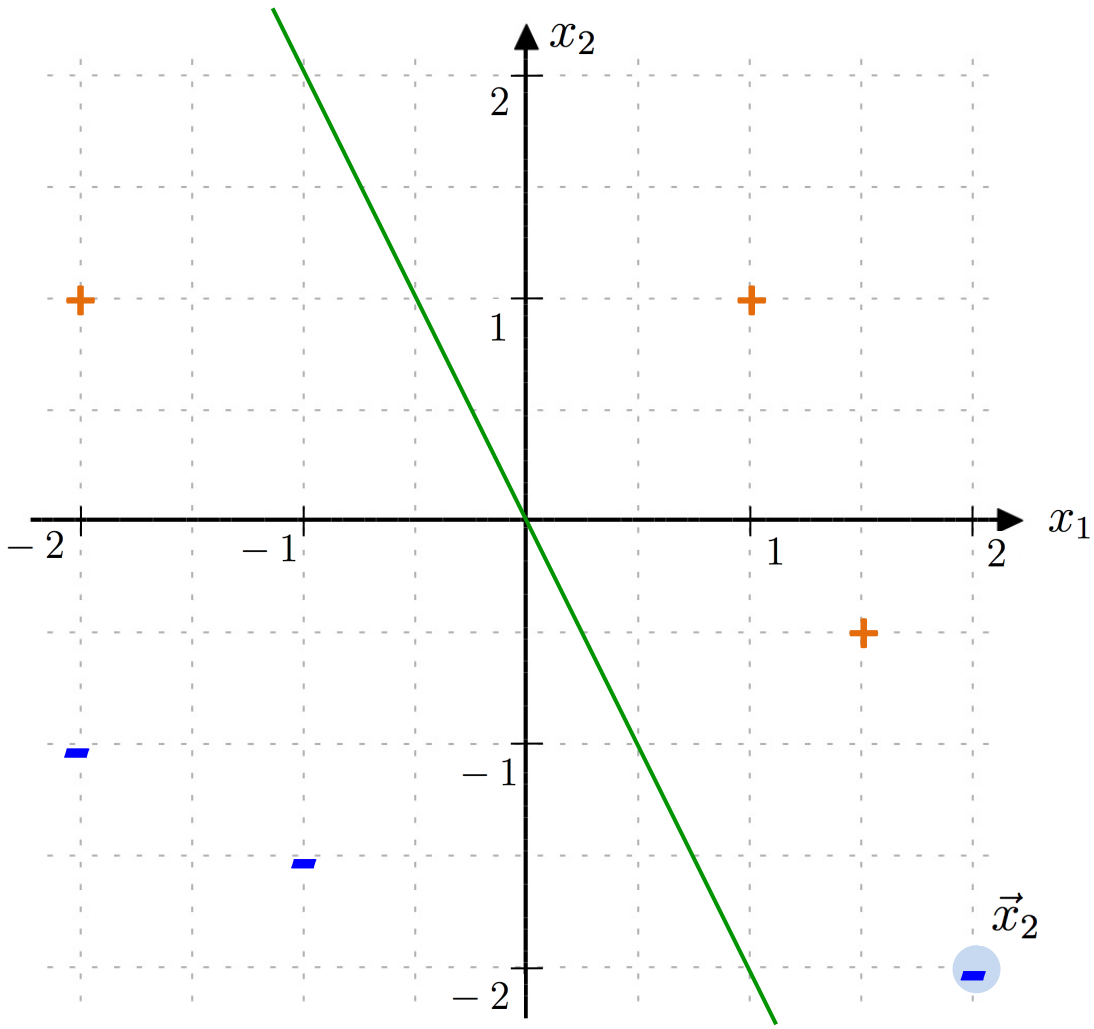
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Perceptron Recap

$$\alpha = 0.2$$

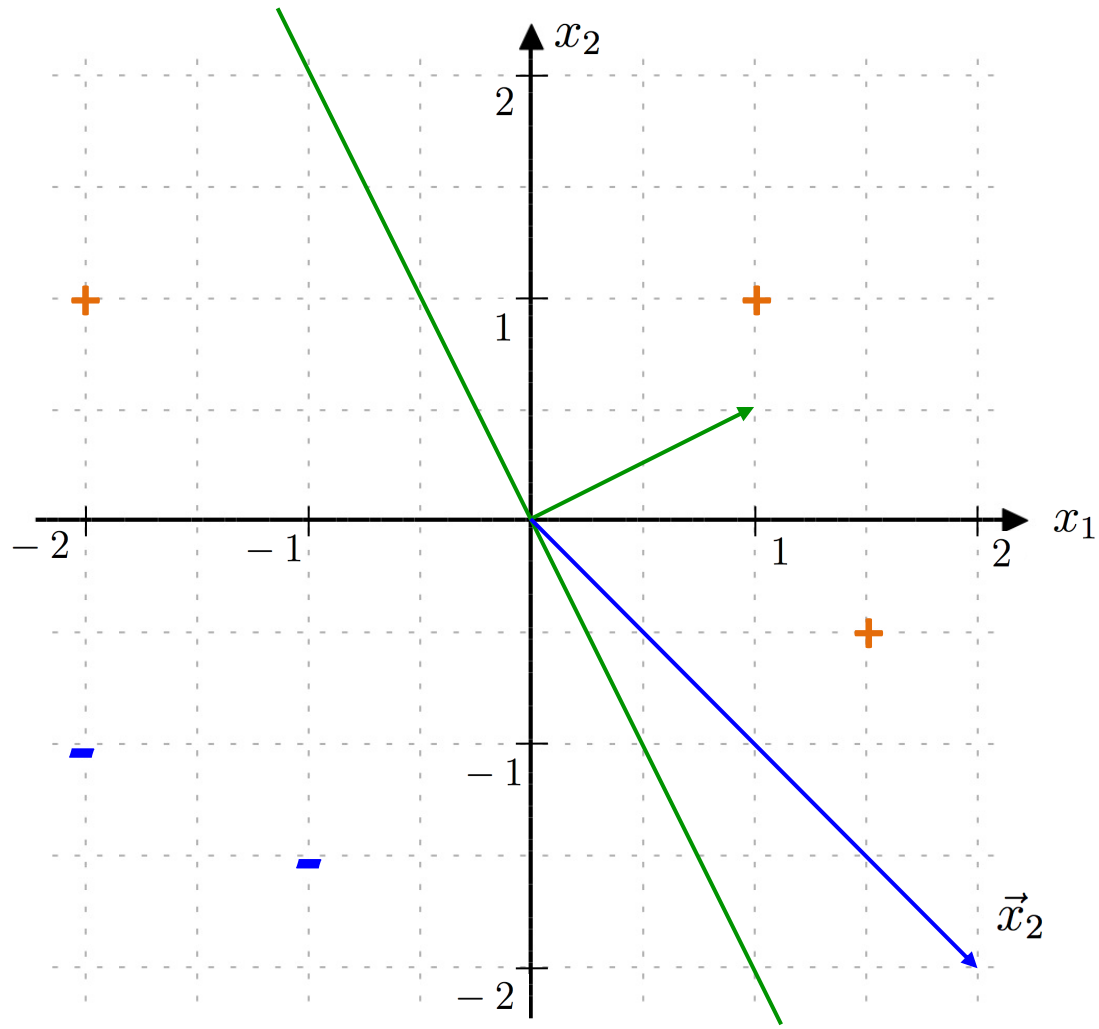
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Perceptron Recap

$$\alpha = 0.2$$

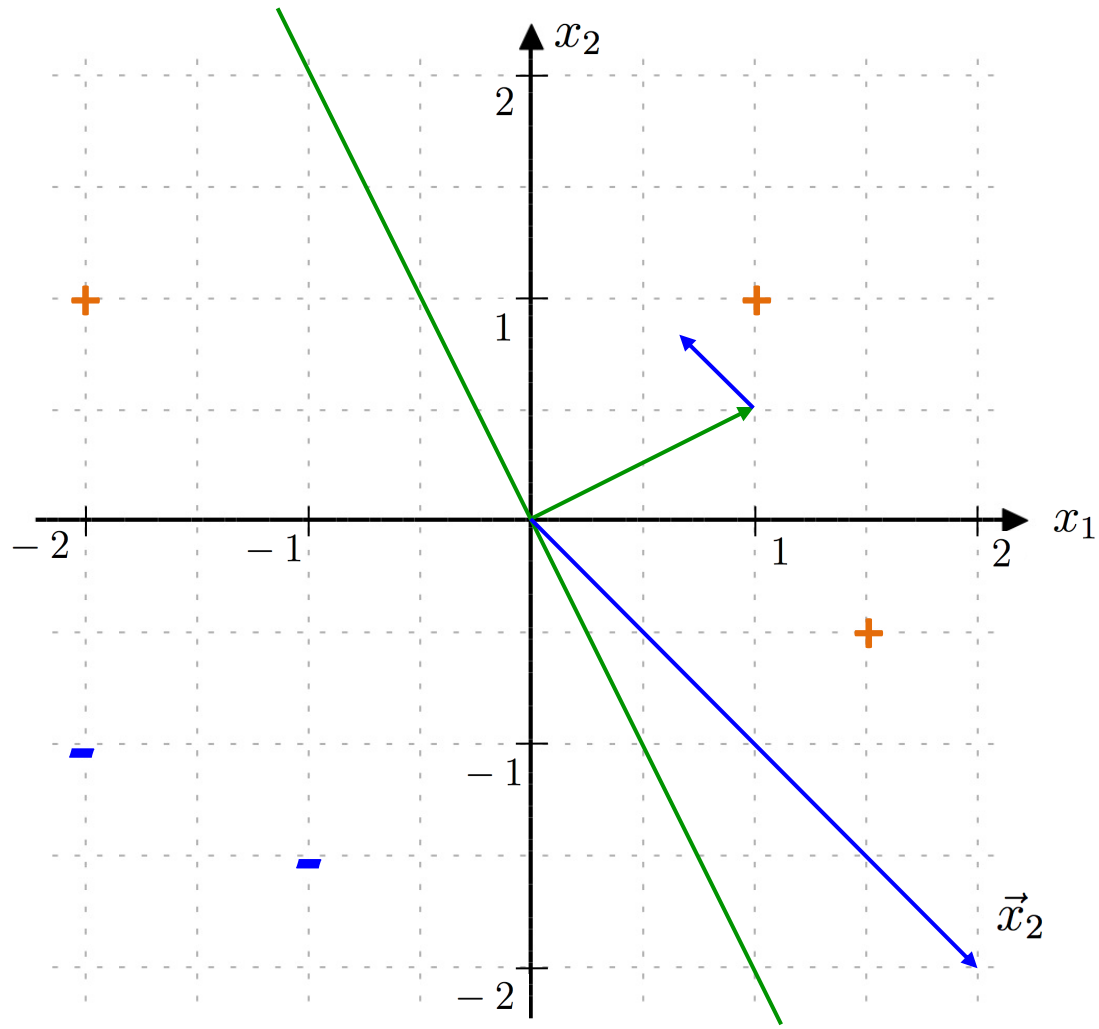
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification
"Push" ***w*** away from negative point



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Perceptron Recap

$$\alpha = 0.2$$

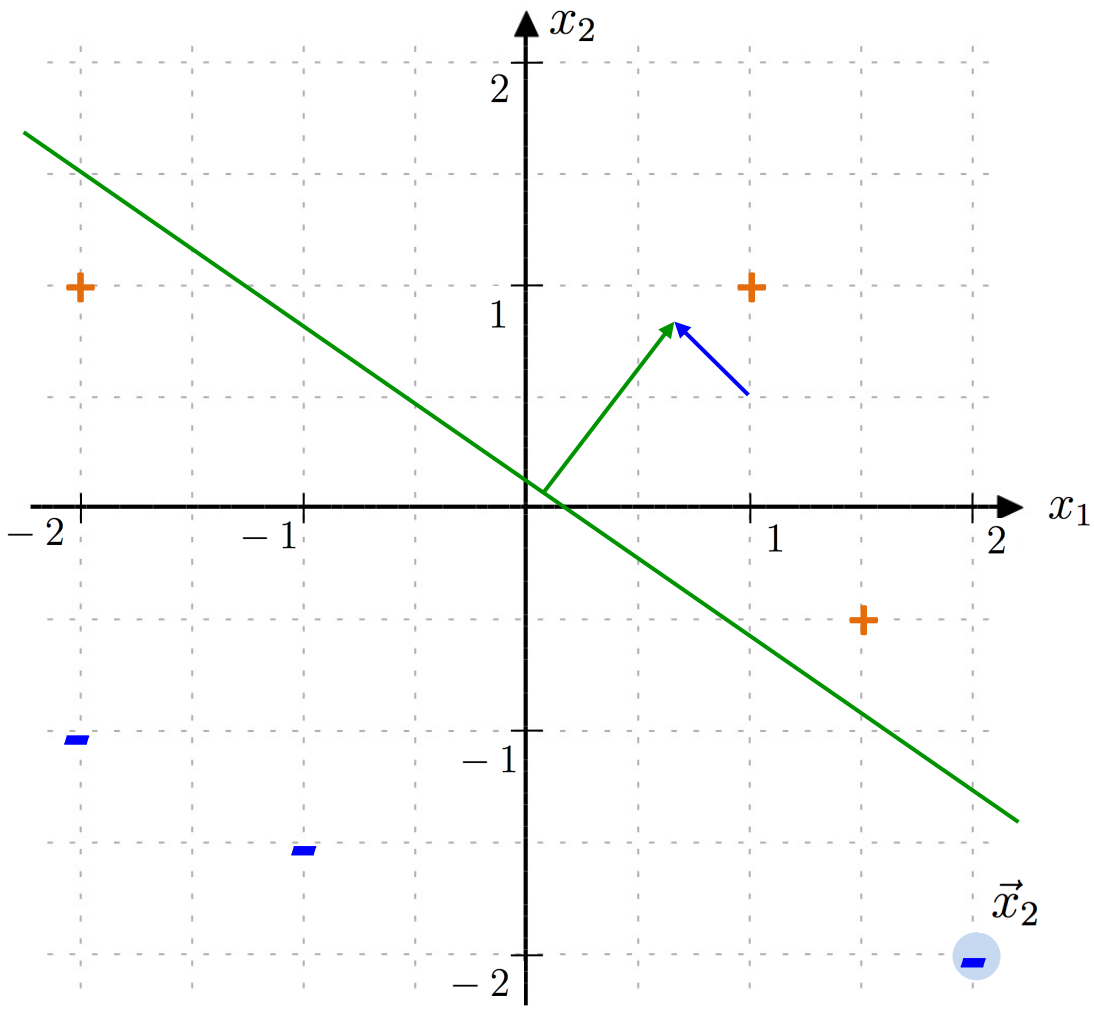
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification
"Push" ***w*** away from negative point



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Perceptron Recap

$$\alpha = 0.2$$

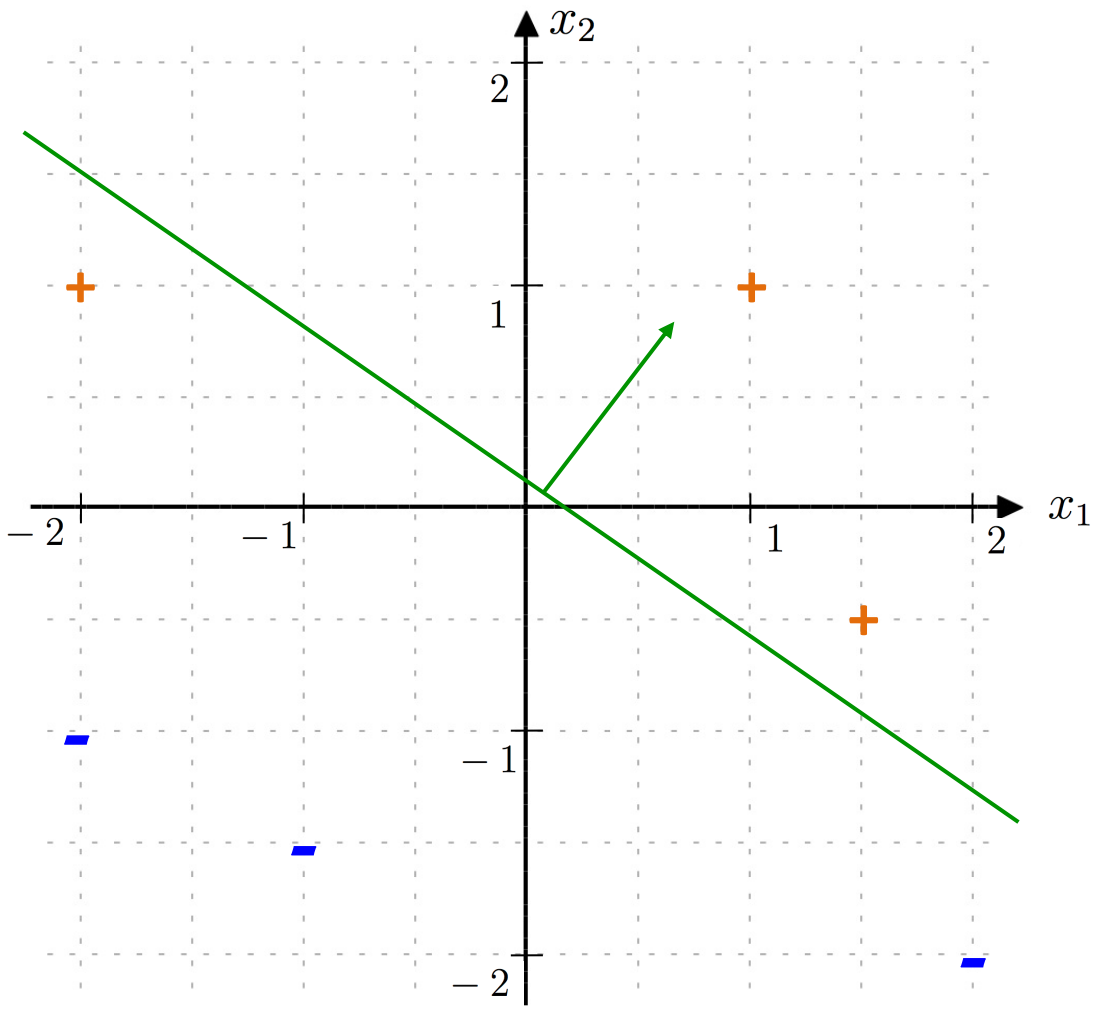
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

What is the new weight vector?



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

SVM flowchart

Goal: maximize the separation between positive and negative examples

SVM flowchart

Goal: maximize the separation between positive and negative examples

Create optimization problem: “maximizing the margin”

SVM flowchart

Goal: maximize the separation between positive and negative examples

→

Create optimization problem: "maximizing the margin"

→

Convert this "primal" optimization problem into the "dual"

SVM flowchart

Goal: maximize the separation between positive and negative examples

Create optimization problem: "maximizing the margin"

Convert this "primal" optimization problem into the "dual"

Training: solve dual optimization problem to find alpha value for each training example

SVM flowchart

Goal: maximize the separation between positive and negative examples

Create optimization problem: "maximizing the margin"

Convert this "primal" optimization problem into the "dual"

Training: solve dual optimization problem to find alpha value for each training example

Testing: use alpha values of support vectors to classify new points (don't explicitly calculate w)

SVM flowchart

Goal: maximize the separation between positive and negative examples

Create optimization problem: "maximizing the margin"

- Use relationship between functional & geo margin
- Put arbitrary constraint on functional margin
- Take multiplicative inverse to make max into min

Convert this "primal" optimization problem into the "dual"

Training: solve dual optimization problem to find alpha value for each training example

Testing: use alpha values of support vectors to classify new points (don't explicitly calculate w)

SVM flowchart

Goal: maximize the separation between positive and negative examples

Create optimization problem: "maximizing the margin"

- Use relationship between functional & geo margin
- Put arbitrary constraint on functional margin
- Take multiplicative inverse to make max into min

Convert this "primal" optimization problem into the "dual"

- Use Lagrange multipliers to incorporate constraints
- Minimize with respect to w and b

Training: solve dual optimization problem to find alpha value for each training example

Testing: use alpha values of support vectors to classify new points (don't explicitly calculate w)

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\left(\vec{w} \cdot \vec{x} + b\right)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\left(\vec{w} \cdot \vec{x} + b\right)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\left(\vec{w} \cdot \vec{x} + b\right)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Geometric Margin:
(distance between example and hyperplane)

$$\gamma_i = y_i \left(\frac{\vec{w}}{\|\vec{w}\|} \cdot \vec{x}_i + \frac{b}{\|\vec{w}\|} \right)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\left(\vec{w} \cdot \vec{x} + b\right)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Geometric Margin:
(distance between example and hyperplane)

$$\gamma_i = y_i \left(\frac{\vec{w}}{\|\vec{w}\|} \cdot \vec{x}_i + \frac{b}{\|\vec{w}\|} \right)$$

Note:

$$\boxed{\gamma_i = \frac{\hat{\gamma}_i}{\|\vec{w}\|}}$$

Optimization Problem: try 1

Goal: maximize the minimum distance between example and hyperplane

$$\gamma = \min_{i=1,\cdots,n} \gamma_i$$

Optimization Problem: try 1

Goal: maximize the minimum distance between example and hyperplane

$$\gamma = \min_{i=1,\cdots,n} \gamma_i$$

Formulation: optimize a function with respect to a constraint

$$\begin{aligned} \max_{\gamma,\vec{w},b} \quad & \gamma \\ \text{s.t.} \quad & y_i(\vec{w}\cdot\vec{x}_i + b) \geq \gamma, \quad i = 1,\cdots,n \\ \text{and} \quad & \|\vec{w}\| = 1 \end{aligned}$$

(force functional and geometric margin to be equal)

Optimization Problem: try 2

Idea: substitute functional margin
divided by magnitude of weight vector

$$\max_{\hat{\gamma},\vec{w},b} \quad \frac{\hat{\gamma}}{\|\vec{w}\|}$$
$$\text{s.t.} \quad y_i(\vec{w}\cdot\vec{x}_i + b) \geq \hat{\gamma}, \quad i = 1, \cdots, n$$

(gets rid of non-convex constraint)

Optimization Problem: try 3

Idea: put arbitrary constraint on functional margin

$$\boxed{\hat{\gamma} = 1}$$

$$\min_{\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2$$

$$\text{s.t.} \quad y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1, \quad i = 1, \cdots, n$$

Optimization Problem: try 3

Idea: put arbitrary constraint on functional margin

$$\boxed{\hat{\gamma} = 1}$$

$$\min_{\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2$$
$$\text{s.t.} \quad y_i(\vec{w}\cdot\vec{x}_i + b) \geq 1, \quad i = 1, \cdots, n$$

$$\boxed{\begin{aligned} &\min_{\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2 \\ &\text{s.t.} \quad -y_i(\vec{w}\cdot\vec{x}_i + b) + 1 \leq 0, \quad i = 1, \cdots, n \end{aligned}}$$

Lagrangian

- The alpha values are our Lagrange multipliers
- We don't care about our constraint if it is not *active*

$$\mathcal{L}(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(\vec{w} \cdot \vec{x}_i + b) - 1 \right]$$

Lagrangian

- The alpha values are our Lagrange multipliers
- We don't care about our constraint if it is not *active*

$$\mathcal{L}(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(\vec{w} \cdot \vec{x}_i + b) - 1 \right]$$

- First minimize with respect to w & b, becomes W(alpha)

$$\mathcal{L}(\vec{w}, b, \vec{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j$$

Lagrangian

- The alpha values are our Lagrange multipliers
- We don't care about our constraint if it is not *active*

$$\mathcal{L}(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(\vec{w} \cdot \vec{x}_i + b) - 1 \right]$$

- First minimize with respect to w & b, becomes W(alpha)

$$\mathcal{L}(\vec{w}, b, \vec{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \boxed{\vec{x}_i \cdot \vec{x}_j}$$

Kernel Trick

- Now we can replace dot products with any kernel!

$$W(\vec{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j K(\vec{x}_i, \vec{x}_j)$$

Final Goal: classification

- After using Kernel Trick with dual optimization problem, we have:

$$\vec{w}^* = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$$

- To classify, we use:

$$h(\vec{x}) = \text{sign}\left(\vec{w}^* \cdot \vec{x} + b^*\right)$$

Final Goal: classification

- After using Kernel Trick with dual optimization problem, we have:

$$\vec{w}^* = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$$

- To classify, we use:

$$h(\vec{x}) = \text{sign}\left(\vec{w}^* \cdot \vec{x} + b^*\right)$$

$$h(\vec{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \vec{x}_i \cdot \vec{x} + b^*\right)$$

Final Goal: classification

- After using Kernel Trick with dual optimization problem, we have:

$$\vec{w}^* = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$$

- To classify, we use:

$$h(\vec{x}) = \text{sign}\left(\vec{w}^* \cdot \vec{x} + b^*\right)$$

$$h(\vec{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \vec{x}_i \cdot \vec{x} + b^*\right)$$

$$\boxed{h(\vec{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(\vec{x}_i, \vec{x}) + b^*\right)}$$

Outline for November 21

- Support Vector Machines Review
- Likelihood functions (Bernoulli and Logistic Regression)
- Finish practice problems (Q2-Q4)

$$\vec{Y} = \underbrace{\overset{1}{H}}_{p}\ \underbrace{\overset{0}{T}}_{1-p}\ \underbrace{\overset{0}{T}}_{1-p}\ \underbrace{\overset{1}{H}}_{p} \leftarrow \text{given; want } p$$

$$L(p \mid \vec{Y}) = p^{\#\text{heads}} (1-p)^{\#\text{tails}}$$

$$L(p \mid \vec{Y}) = \prod_{i=1}^{n} \underbrace{p^{y_i}}_{y_i=1} \underbrace{(1-p)^{1-y_i}}_{y_i=0}$$

MLE
- log
- derivative wrt p
- set to 0
- solve for $\hat{p}$

log reg

$$L(\vec{w} \mid \vec{x}_i, y_i \ i=1 \cdots n)$$

$$= \prod_{i=1}^{n} \underbrace{h_{\vec{w}}(\vec{x}_i)^{y_i}}_{\text{prob } (+)} \underbrace{(1-h_{\vec{w}}(\vec{x}_i))^{1-y_i}}_{\text{prob } (-)}$$

Outline for November 21

- Support Vector Machines Review
- Likelihood functions (Bernoulli and Logistic Regression)
- Finish practice problems (Q2-Q4)

Follow ups on questions from class

- **ANOVA**: considered a special case of linear regression

- **AdaBoost**: why ½ in front of the score?
  - Comes out in the derivation
  - Main idea: solve for the classifier scores that minimize exponential loss

Handout 19, Question 2

- r = 1/3, probability of one classifier being wrong
- T = 5, number of classifiers
- R = number of votes for the wrong class
- If R=3,4,5 then we will vote for the wrong class overall

Q2

$T = 5$ # classifiers

R = # wrong

r = prob of one being wrong. $\Big\} \frac{1}{3}$

$R > \underbrace{\frac{T}{2}}_{2.5}$

	prob
$R = 5 \rightarrow$	$\binom{5}{5}\left(\frac{1}{3}\right)^5$
$R = 4 \rightarrow$	$\binom{5}{4}\left(\frac{1}{3}\right)^4\left(\frac{2}{3}\right)$
$R = 3 \rightarrow$	$\binom{5}{3}\left(\frac{1}{3}\right)^3\left(\frac{2}{3}\right)^2$

$\oplus$ $\boxed{\approx 0.21}$ prob that ensemble is wrong

Q3

$n = 2$

3 unique datasets $\left\{ \begin{array}{l} \{x_1, x_1\} \\ \{x_1, x_2\} \\ \{x_2, x_2\} \end{array} \right.$

$n = 3$

$\Rightarrow$ 10 datasets

$3 + 3 \cdot 2 + 1$

Handout 19, Question 2

- This analysis assumed classifiers were independent!

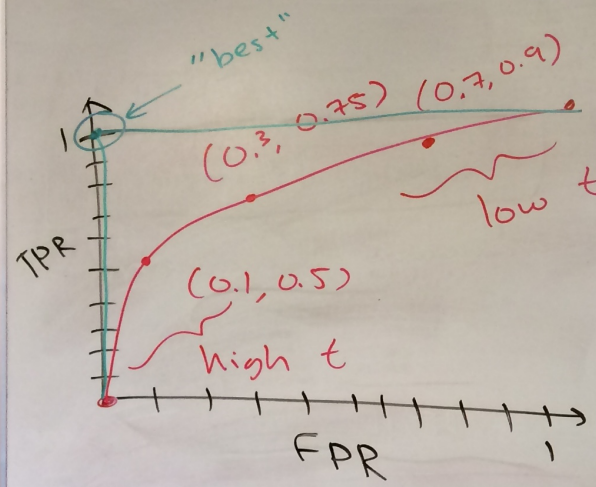
- What if they are not? How did Random Forests help us decorrelate classifiers?

Handout 19, Question 2

- This analysis assumed classifiers were independent!

- What if they are not? How did Random Forests help us decorrelate classifiers?

- Note about Bagging: choosing *n* with resampling actually does produce a very different dataset
  - As *n* increases, roughly 0.37 not chosen each time



ideal conf matrix

10	0
0	20

Q4