CSE446: Perceptron
Spring 2017

Ali Farhadi

Hessam Bagherinezhad

Slides adapted from Dan Klein, Luke Zettlemoyer

Who needs probabilities?

- Previously: model data with distributions
- Joint: P(X,Y)
  - e.g. Naïve Bayes
- Conditional: P(Y|X)
  - e.g. Logistic Regression
- But wait, why probabilities?
- Lets try to be error-driven!

mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	make
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	ameri
bad	4	121	110	2600	12.8	77	europ
bad	8	350	175	4100	13	73	ameri
bad	6	198	95	3102	16.5	74	ameri
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad	8	302	139	3570	12.8	78	ameri
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
good	4	120	79	2625	18.6	82	ameri
bad	8	455	225	4425	10	70	ameri
good	4	107	86	2464	15.5	76	europ
bad	5	131	103	2830	15.9	78	europ

Generative vs. Discriminative

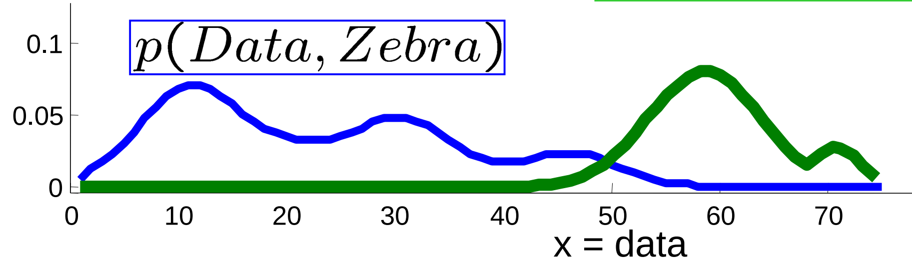
- Generative classifiers:
  - E.g. naïve Bayes
  - A joint probability model with evidence variables
  - Query model for causes given evidence

- Discriminative classifiers:
  - No generative model, no Bayes rule, maybe no probabilities at all!
  - Try to predict the label Y directly from X
  - Robust, accurate with varied features
  - Loosely: mistake driven rather than model driven

Discriminative vs. generative

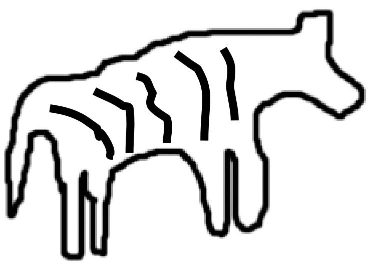
- Generative model

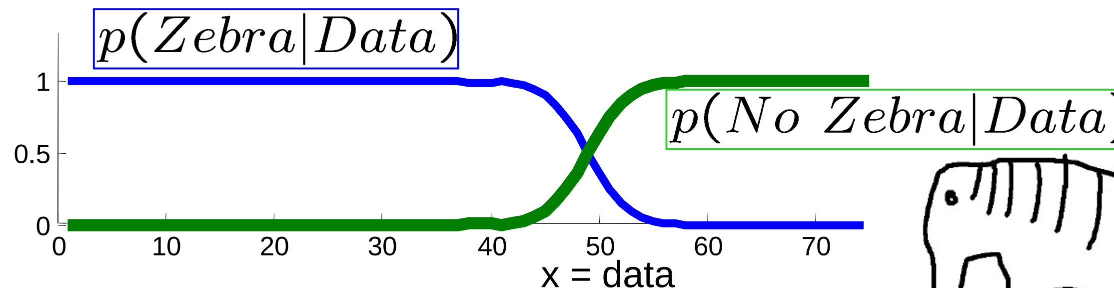
  (*The artist*)



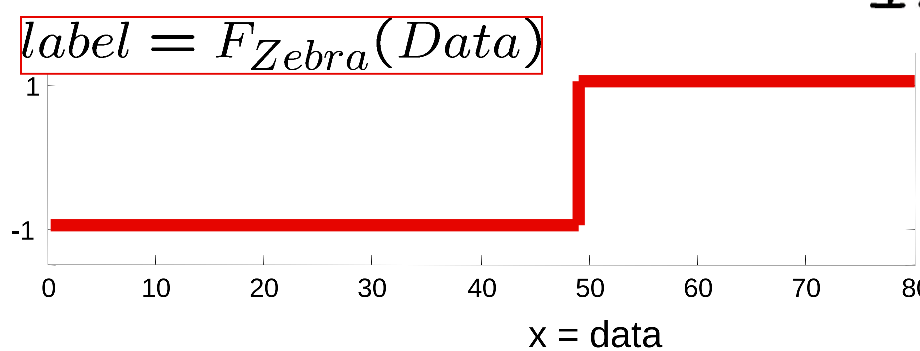
- Discriminative model

  (The lousy painter)



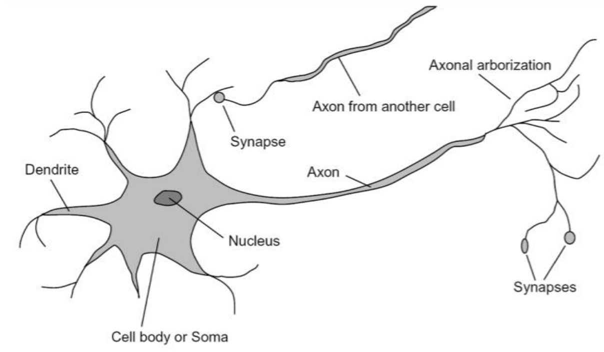


- Classification function



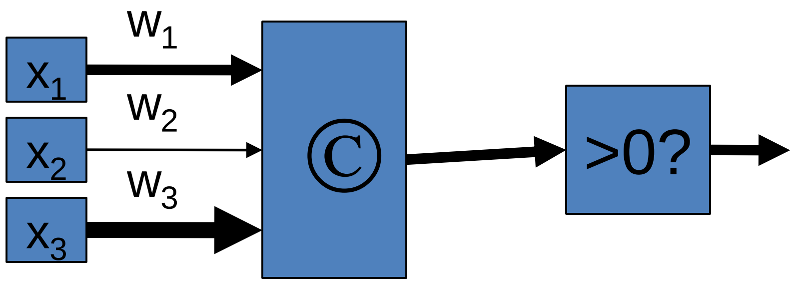
Linear Classifiers

- Inputs are feature values
- Each feature has a weight
- Sum is the activation



$$\text{activation}_w(x) = \sum_i w_i x_i = w \cdot x$$

- If the activation is:
  - Positive, output *class 1*
  - Negative, output *class 2*



Example: Spam

- Imagine 3 features (spam is "positive" class):
  - free (number of occurrences of "free")
  - money (occurrences of "money")
  - BIAS (intercept, always has value 1)

"free money"

x

```
BIAS  :  1
free  :  1
money :  1
...
```

w

```
BIAS  : -3
free  :  4
money :  2
...
```

$$
\begin{array}{ll}
(1)(-3) & + \\
(1)(4) & + \\
(1)(2) & + \\
\ldots & \\
= 3 &
\end{array}
$$

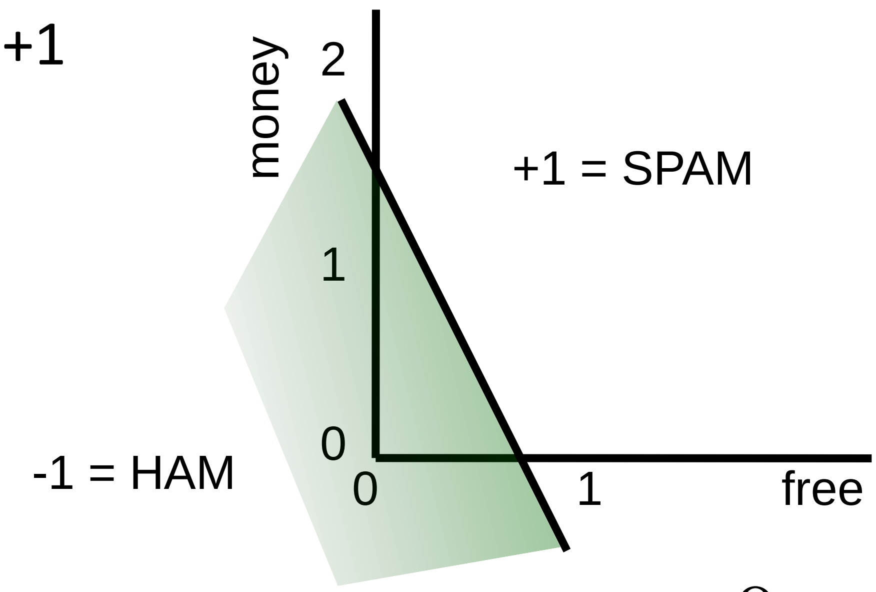
w•x > 0 ➔ SPAM!!!

Binary Decision Rule

- In the space of feature vectors
  - Examples are points
  - Any weight vector is a hyperplane
  - One side corresponds to y=+1
  - Other corresponds to y=-1

w

```
BIAS  : -3
free  :  4
money :  2
...
```



$$w \cdot x = 0$$

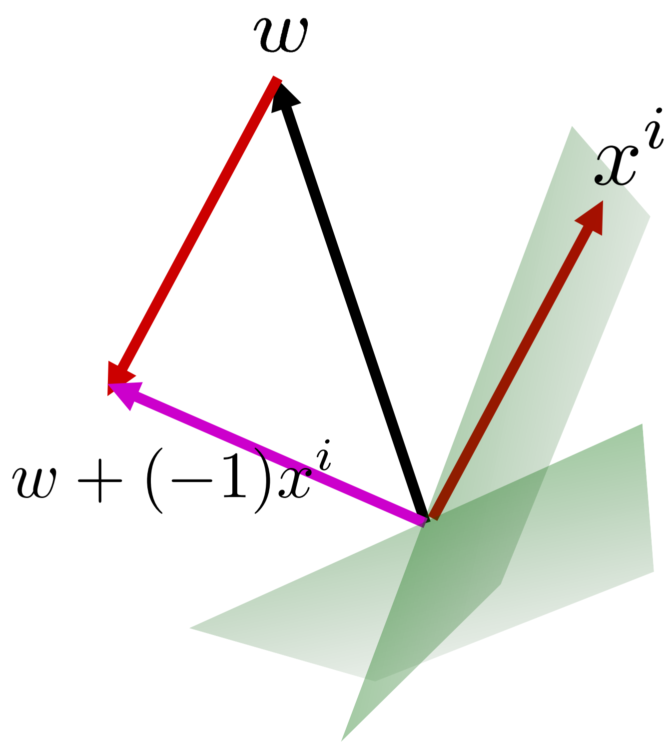
Binary Perceptron Algorithm

- Start with zero weights: w=0
- For t=1..T (T passes over data)
  - For i=1..n: (each training example)
    - Classify with current weights

      $$y = sign(w \cdot x^i)$$

      - sign(x) is +1 if x>0, else -1
    - If correct (i.e., $y=y^i$), no change!
    - If wrong: update

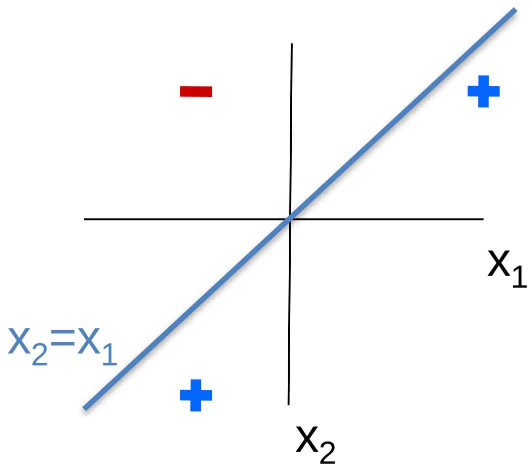
      $$w = w + y^i x^i$$



- For t=1..T, i=1..n:
  - $y = sign(w \cdot x^i)$
  - if $y \neq y^i$

    $w = w + y^i x^i$

x_1	x_2	y
3	2	1
-2	2	-1
-2	-3	1



Initial:

- w = [0,0]

t=1,i=1

- [0,0]•[3,2] = 0, sign(0)=-1
- w = [0,0] + [3,2] = [3,2]

t=1,i=2

- [3,2]•[-2,2]=-2, sign(-2)=-1

t=1,i=3

- [3,2]•[-2,-3]=-12, sign(-12)=-1
- w = [3,2] + [-2,-3] = [1,-1]

t=2,i=1

- [1,-1]•[3,2]=1, sign(1)=1

t=2,i=2

- [1,-1]•[-2,2]=-4, sign(-4)=-1

t=2,i=3

- [1,-1]•[-2,-3]=1, sign(1)=1

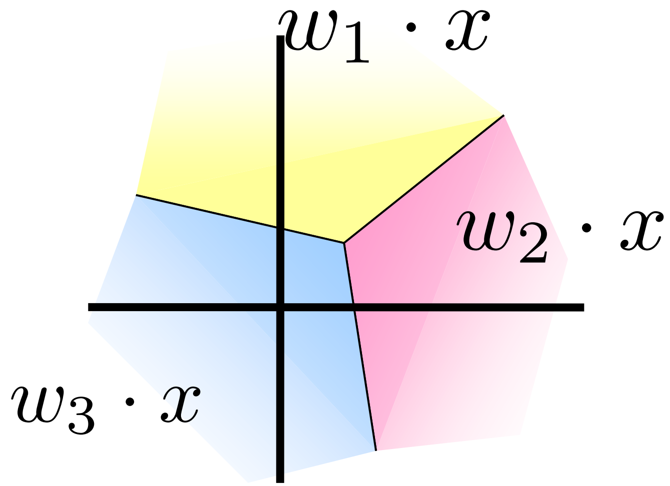
Converged!!!

- $y=w_1x_1+w_2x_2 \rightarrow y=x_1+-x_2$
- So, at $y=0 \rightarrow x_2=x_1$

Multiclass Decision Rule

Example: y is {1,2,3}

- We are fitting three planes: w_1, w_2, w_3
- Predict i when $w_i \bullet x$ is highest



- If we have more than two classes:
  - Have a weight vector for each class: w_y
  - Calculate an activation for each class

$$\text{activation}_w(x, y) = w_y \cdot x$$

  - Highest activation wins

$$y^* = \arg\max_y(\text{activation}_w(x, y))$$

Example

“win the vote” ⇒

x

BIAS	:	1
win	:	1
game	:	0
vote	:	1
the	:	1
...		

w_{SPORTS}

BIAS	:	-2
win	:	4
game	:	4
vote	:	0
the	:	0
...		

$w_{POLITICS}$

BIAS	:	1
win	:	2
game	:	0
vote	:	4
the	:	0
...		

w_{TECH}

BIAS	:	2
win	:	0
game	:	2
vote	:	0
the	:	0
...		

$x \cdot w_{SPORTS} = 2$

$x \cdot w_{POLITICS} = 7$

$x \cdot w_{TECH} = 2$

POLITICS wins!!!

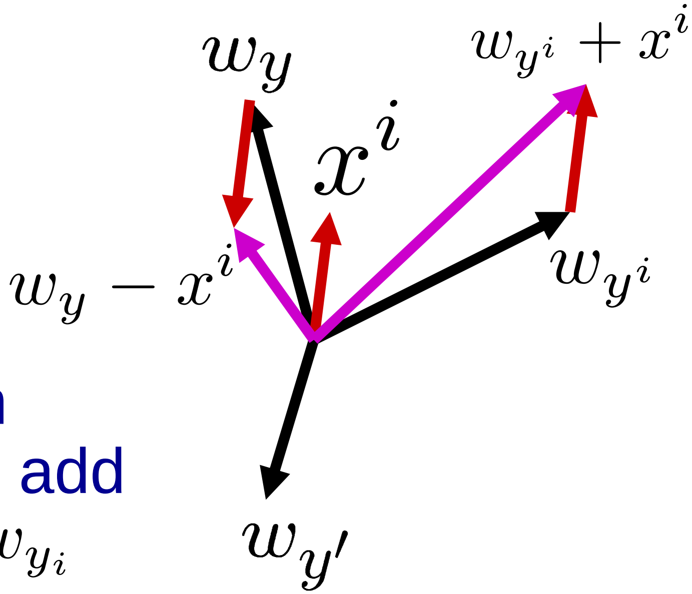
The Multi-class Perceptron Alg.

- Start with zero weights
- For t=1..T, i=1..n (T times over data)
  - Classify with current weights

$$y = \arg\max_y w_y \cdot x^i$$

  - If correct ($y=y_i$), no change!
  - If wrong: subtract features x^i from weights for predicted class w_y and add them to weights for correct class w_{y_i}

$$w_y = w_y - x^i$$
$$w_{y^i} = w_{y^i} + x^i$$



Perceptron vs. logistic regression

Update rule

- Logistic regression:

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \widehat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})]$$

  - Need all of training data

- Perceptron:

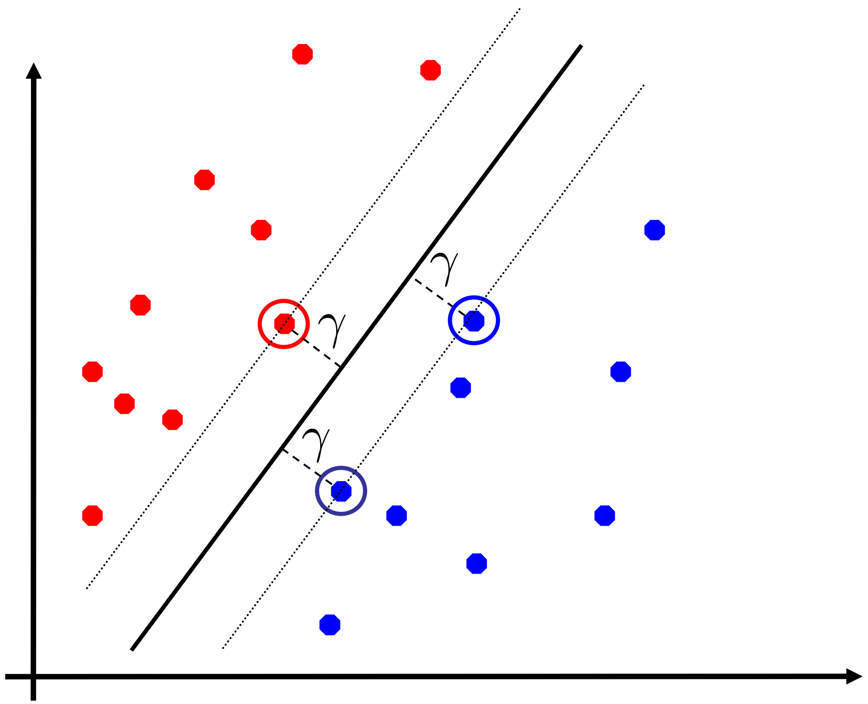
if mistake, $w = w + y^i x^i$

  - Update only with one example.
  - Ideal for online algorithms.

Linearly Separable (binary case)

- The data is linearly separable with *margin* γ, if:

$$\exists w. \forall t. y^t (w \cdot x^t) \geq \gamma > 0$$



- For $y^t=1$

$$w \cdot x^t \geq \gamma$$

- For $y^t=-1$

$$w \cdot x^t \leq -\gamma$$

Mistake Bound for Perceptron

$$\|x\|_2 = \sqrt{\sum_i x_i^2}$$

- Assume data is separable with margin γ:

$$\exists w^* \text{ s.t. } \|w^*\|_2 = 1 \text{ and } \forall t. y^t (w^* \cdot x^t) \geq \gamma$$

- Also assume there is a number R such that:

$$\forall t. \|x^t\|_2 \leq R$$

- Theorem: The number of mistakes (parameter updates) made by the perceptron is bounded:

$$mistakes \leq \frac{R^2}{\gamma^2}$$

Constant with respect to # of examples!

Perceptron Convergence (by Induction)

- Let w^k be the weights after the k-th update (mistake), we will show that:

$$k^2\gamma^2 \leq \|w^k\|_2^2 \leq kR^2$$

- Therefore:

$$k \leq \frac{R^2}{\gamma^2}$$

- Because R and γ are fixed constants that do not change as you learn, there are a finite number of updates!
- Proof does each bound separately (next two slides)

Lower bound

Perceptron update:

$$w = w + y^t x^t$$

- Remember our margin assumption:

$$\exists w^* \text{ s.t. } \|w^*\|_2 = 1 \text{ and } \forall t. y^t(w^* \cdot x^t) \geq \gamma$$

- Now, by the definition of the perceptron update, for k-th mistake on t-th training example:

$$\begin{aligned} w^{k+1} \cdot w^* &= (w^k + y^t x^t) \cdot w^* \\ &= w^k \cdot w^* + y^t(w^* \cdot x^t) \\ &\geq w^k \cdot w^* + \gamma \end{aligned}$$

- So, by induction with $w^0=0$, for all k:

$$\begin{aligned} k\gamma &\leq w^k \cdot w^* \\ &\leq \|w^k\|_2 \end{aligned}$$

$$k^2\gamma^2 \leq \|w^k\|_2^2$$

Because:

$$w^k \cdot w^* \leq \|w^k\|_2 \times \|w^*\|_2$$

and $\|w^*\|_2 = 1$

Upper Bound

Perceptron update:

$$w = w + y^t x^t$$

Data Assumption:

$$\forall t. \|x^t\|_2 \leq R$$

- By the definition of the Perceptron update, for k-th mistake on t-th training example:

$$\|w^{k+1}\|_2^2 = \|w^k + y^t x^t\|_2^2$$

$$= \|w^k\|_2^2 + (y^t)^2 \|x^t\|_2^2 + 2y^t x^t \cdot w^k$$

$$\leq \|w^k\|_2^2 + R^2$$

$\leq R^2$ because $(y^t)^2 = 1$ and $\|x^t\|_2 \leq R$

< 0 because Perceptron made error (y^t has different sign than $x^t \bullet w^t$)

- So, by induction with w_0=0 have, for all k:

$$\|w_k\|_2^2 \leq kR^2$$

Perceptron Convergence (by Induction)

- Let w^k be the weights after the k-th update (mistake), we will show that:

$$k^2\gamma^2 \leq \|w^k\|_2^2 \leq kR^2$$

- Therefore:

$$k \leq \frac{R^2}{\gamma^2}$$

- Because R and γ are fixed constants that do not change as you learn, there are a finite number of updates!
- If there is a linear separator, Perceptron will find it!!!

From Logistic Regression to the Perceptron: 2 easy steps!

Perceptron update when y is {-1,1}:

$$w = w + y^j x^j$$

- Logistic Regression: (in vector notation): y is {0,1}

$$w = w + \eta \sum_j [y^j - P(y^j | x^j, w)] x^j$$

- Perceptron: when y is {0,1}:

$$w = w + [y^j - sign^0(w \cdot x^j)] x^j$$

  - $\text{sign}^0(x)$ = +1 if x>0 and 0 otherwise

Differences?

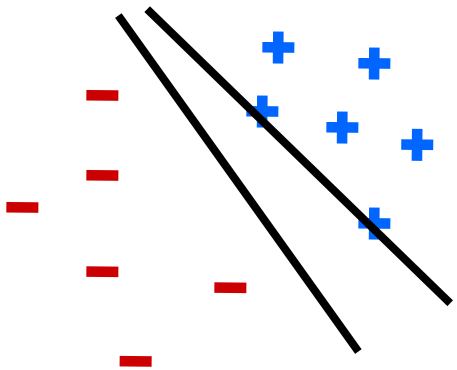
- Drop the Σ_j over training examples: online vs. batch learning
- Drop the dist'n: probabilistic vs. error driven learning

Properties of Perceptrons

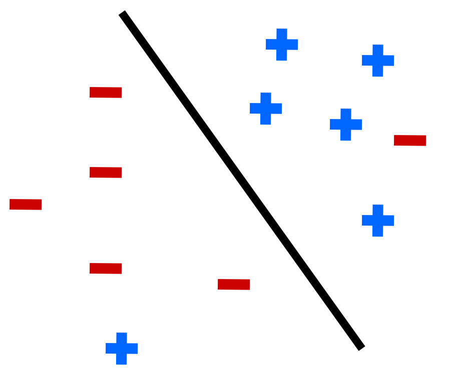
- Separability: some parameters get the training set perfectly correct
- Convergence: if the training is separable, perceptron will eventually converge (binary case)
- Mistake Bound: the maximum number of mistakes (binary case) related to the *margin* or degree of separability

$$mistakes \leq \frac{R^2}{\gamma^2}$$

Separable

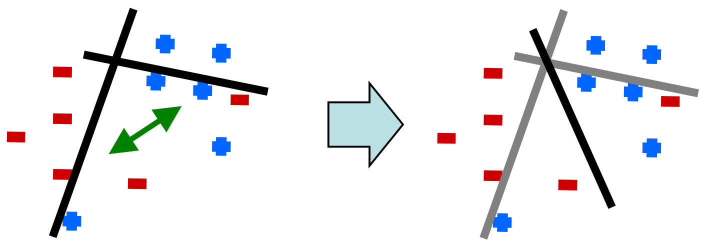


Non-Separable

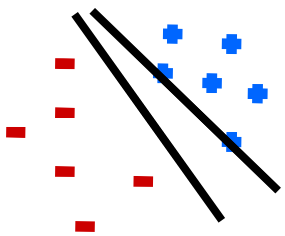


Problems with the Perceptron

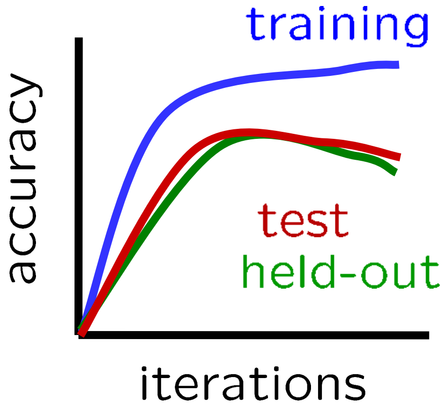
- Noise: if the data isn't separable, weights might thrash
  - Averaging weight vectors over time can help (averaged perceptron)



- Mediocre generalization: finds a "barely" separating solution

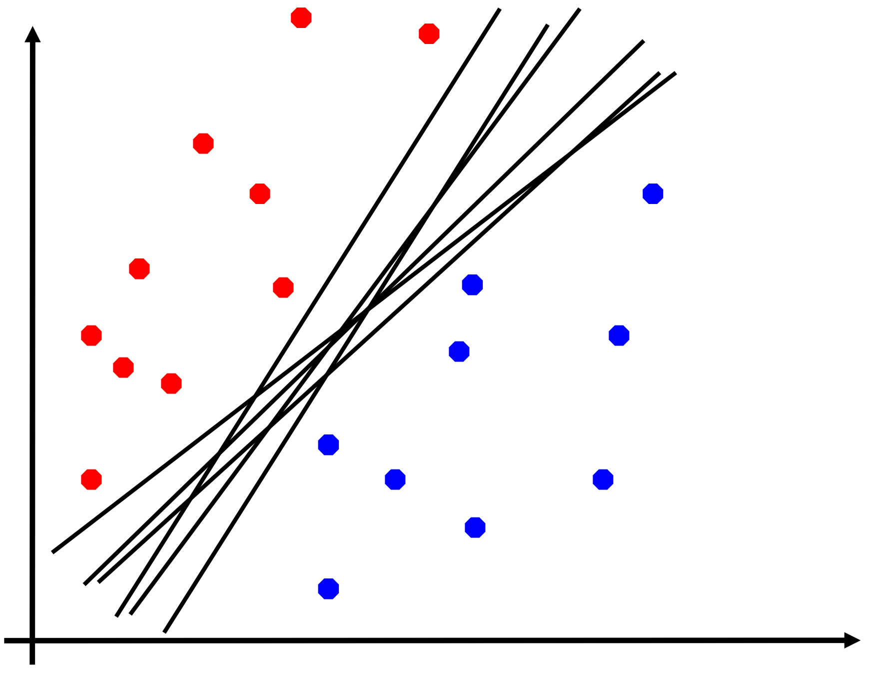


- Overtraining: test / held-out accuracy usually rises, then falls
  - Overtraining is a kind of overfitting



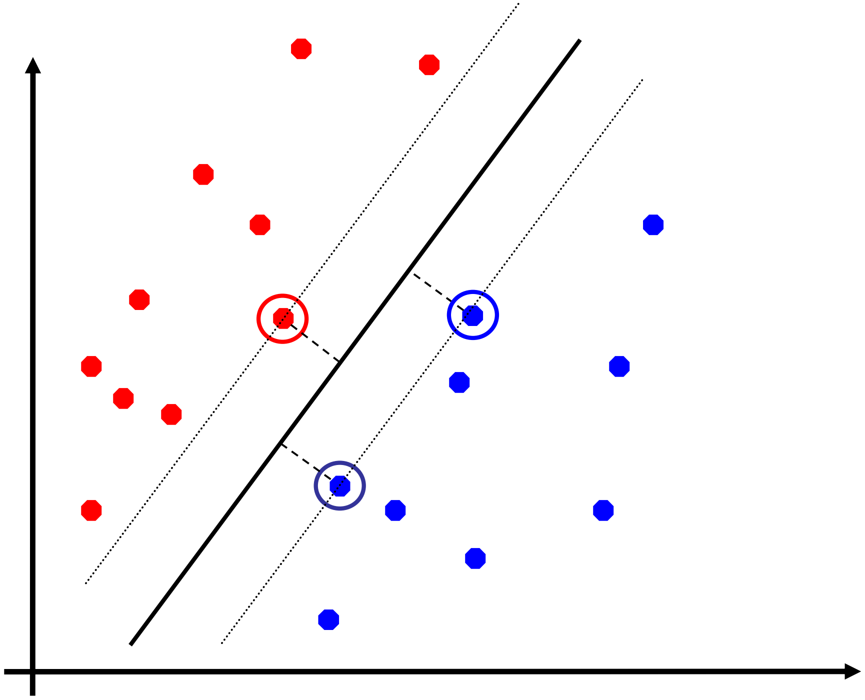
Linear Separators

- Which of these linear separators is optimal?



Support Vector Machines

- Maximizing the margin: good according to intuition, theory, practice
- Support vector machines (SVMs) find the separator with max margin



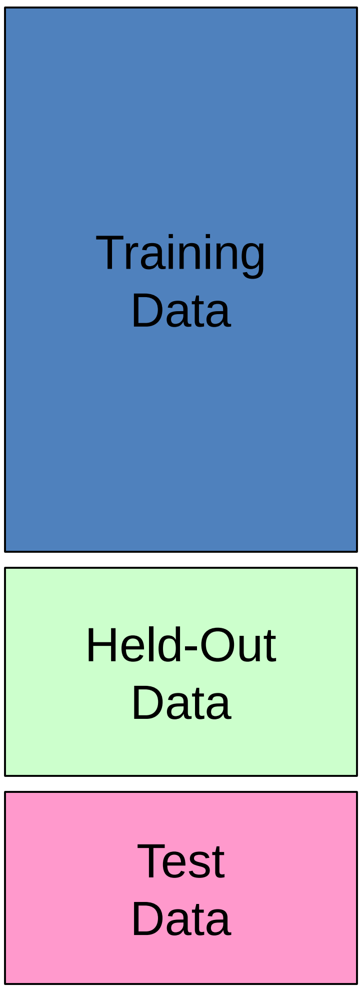
SVM

$$\min_w \frac{1}{2}||w||^2$$

$$\forall i, y \quad w_{y^*} \cdot x^i \geq w_y \cdot x^i + 1$$

Three Views of Classification

(more to come later in course!)



- Naïve Bayes:
  - Parameters from data statistics
  - Parameters: probabilistic interpretation
  - Training: one pass through the data
- Logistic Regression:
  - Parameters from gradient ascent
  - Parameters: linear, probabilistic model, and discriminative
  - Training: gradient ascent (usually batch), regularize to stop overfitting
- The perceptron:
  - Parameters from reactions to mistakes
  - Parameters: discriminative interpretation
  - Training: go through the data until held-out accuracy maxes out