CSE546: Perceptron
Winter 2012

Luke Zettlemoyer

Slides adapted from Dan Klein

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

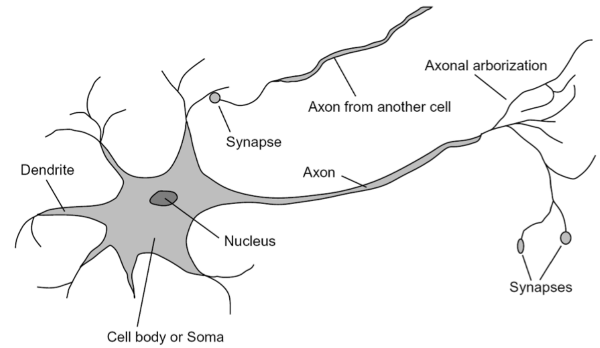
Generative vs. Discriminative

- 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, often 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

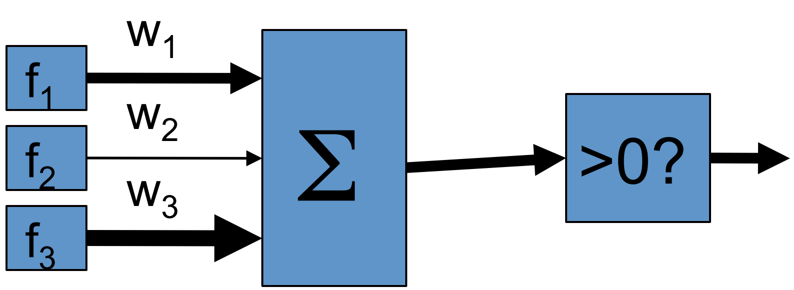
Linear Classifiers

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



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

- 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)

$$w \cdot f(x)$$

$$\sum_i w_i \cdot f_i(x)$$

x	$f(x)$	w
"free money"	BIAS : 1 free : 1 money : 1 ...	BIAS : -3 free : 4 money : 2 ...

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

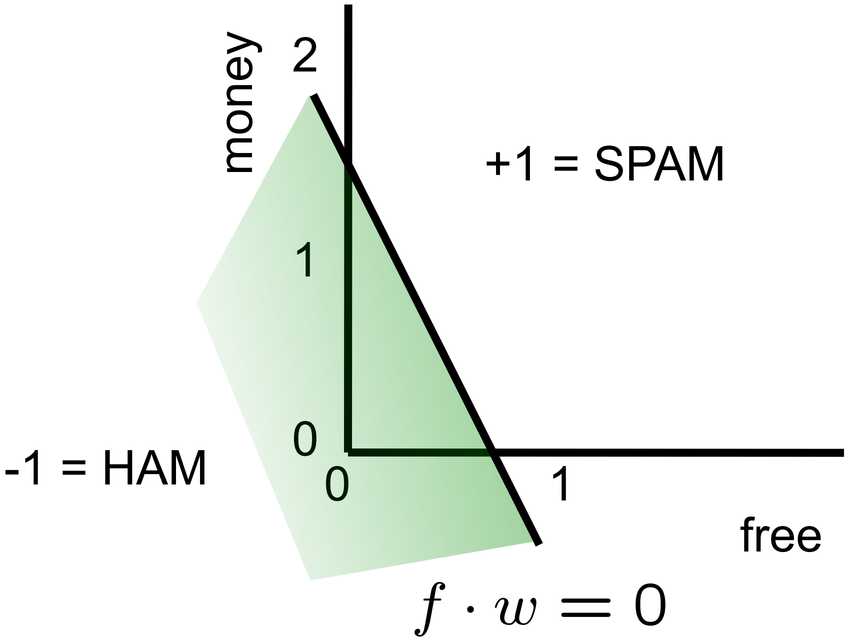
w.f(x) > 0 ➔ SPAM!!!

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
...		



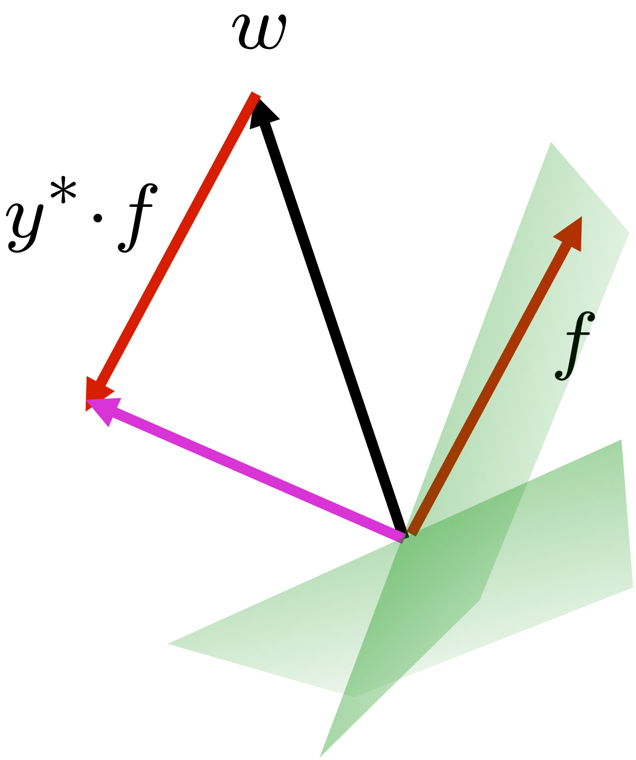
Binary Perceptron Algorithm

- Start with zero weights
- For each training instance (x,y*):
  - Classify with current weights

$$y = \begin{cases} +1 & \text{if } \; w \cdot f(x) \geq 0 \\ -1 & \text{if } \; w \cdot f(x) < 0 \end{cases}$$

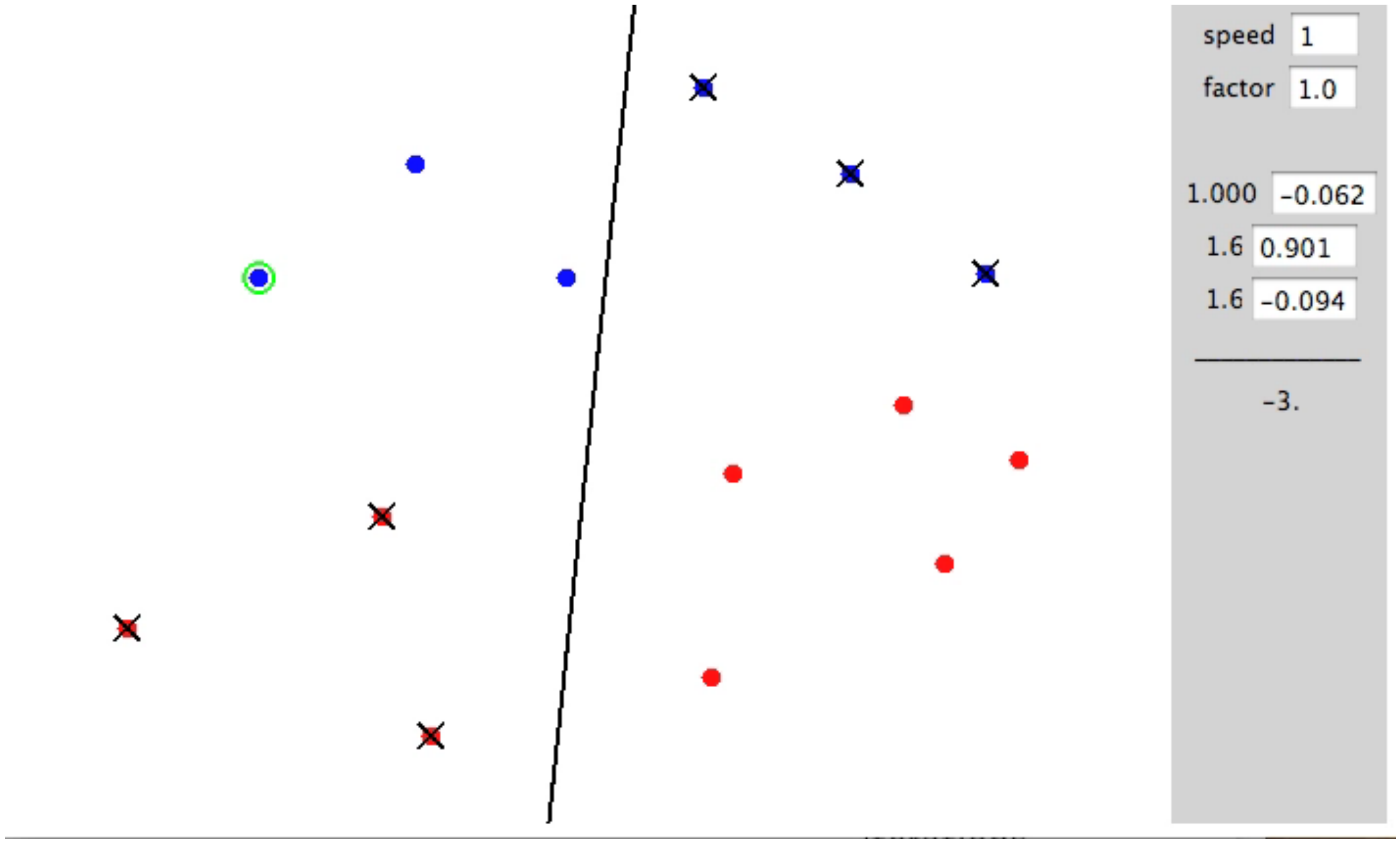
  - If correct (i.e., y=y*), no change!
  - If wrong: update

$$w = w + y^* f(x)$$



Examples: Perceptron

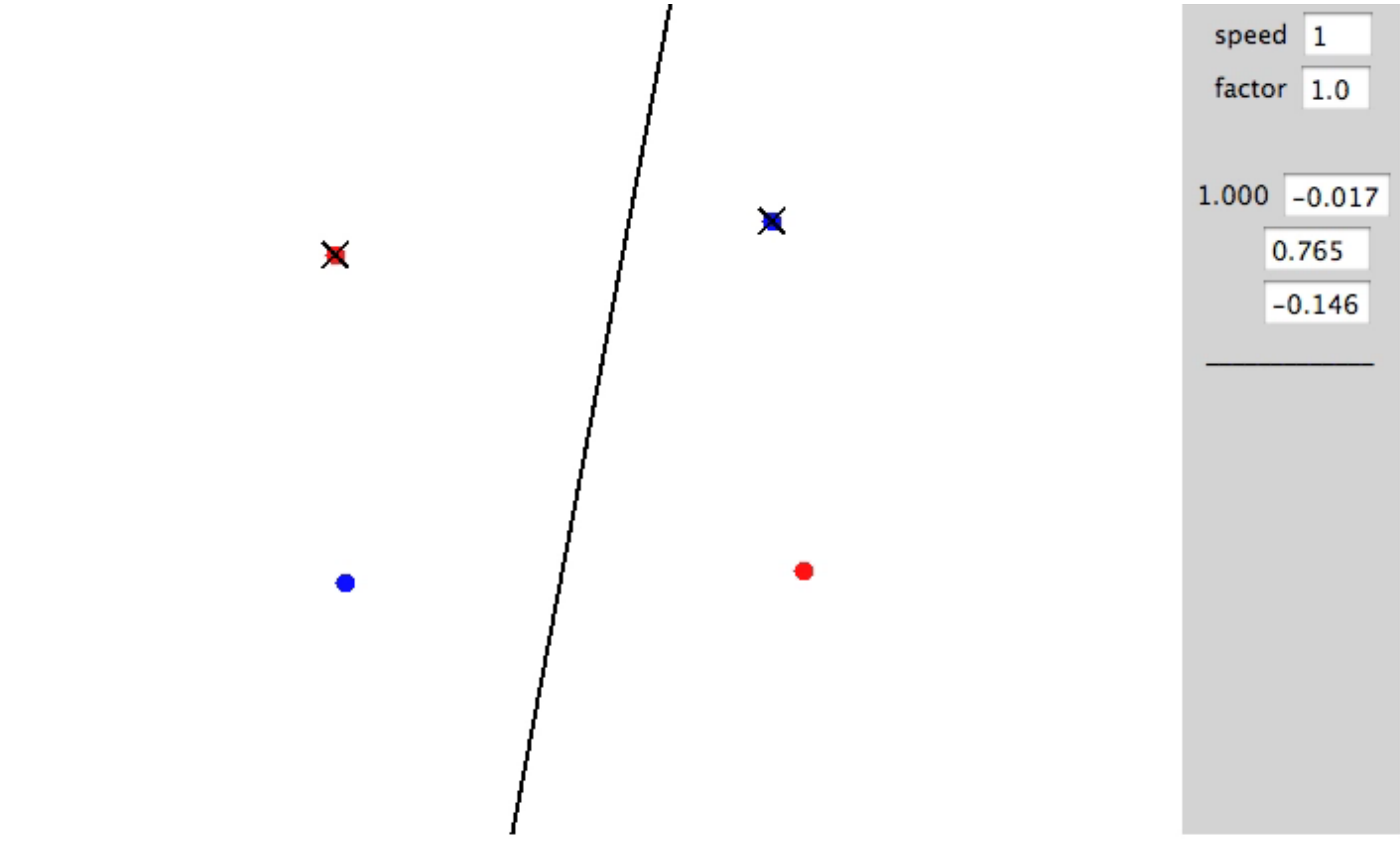
- Separable Case



http://isl.ira.uka.de/neuralNetCourse/2004/VL_11_5/Perceptron.html

Examples: Perceptron

- Inseparable Case



http://isl.ira.uka.de/neuralNetCourse/2004/VL_11_5/Perceptron.html

From Logistic Regression to the Perceptron: 2 easy steps!

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

$$w = w + \eta \sum_j [y_j^* - p(y_j^* | x_j, w)] f(x_j)$$

- Perceptron: y is {0,1}, y(x;w) is prediction given w

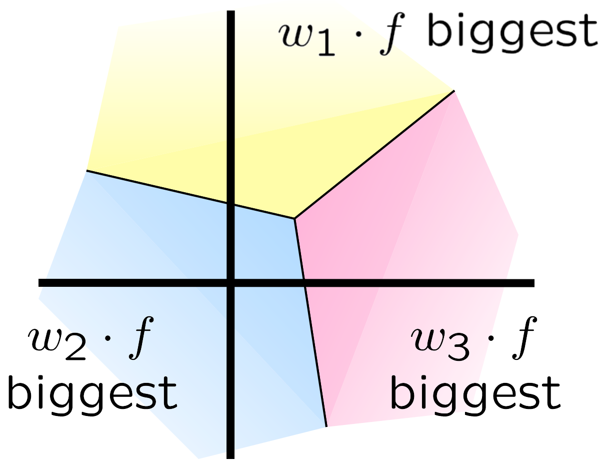
$$w = w + [y^* - y(x;w)] f(x)$$

Differences?

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

Multiclass Decision Rule

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



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

  - Highest activation wins

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

Example

"win the vote"

"win the election"

"win the game"

w_{SPORTS}

```
BIAS  :
win   :
game  :
vote  :
the   :
...
```

$w_{POLITICS}$

```
BIAS  :
win   :
game  :
vote  :
the   :
...
```

w_{TECH}

```
BIAS  :
win   :
game  :
vote  :
the   :
...
```

Example

"win the vote"

```
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
...
```

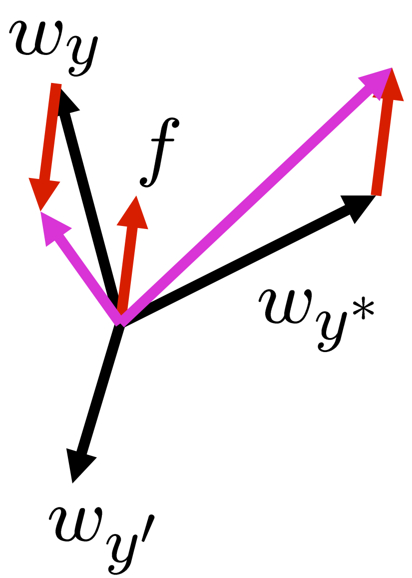
The Multi-class Perceptron Alg.

- Start with zero weights
- Iterate training examples
  - Classify with current weights

$$y = \arg\max_y \; w_y \cdot f(x)$$
$$= \arg\max_y \; \textstyle\sum_i w_{y,i} \cdot f_i(x)$$

  - If correct, no change!
  - If wrong: lower score of wrong answer, raise score of right answer

$$w_y = w_y - f(x)$$
$$w_{y^*} = w_{y^*} + f(x)$$



Properties of Perceptrons

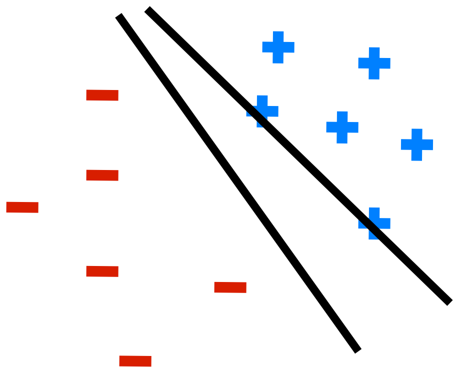
- Separability: some parameters get the training set perfectly correct

- Convergence: if the training is separable, perceptron will eventually converge (binary case)

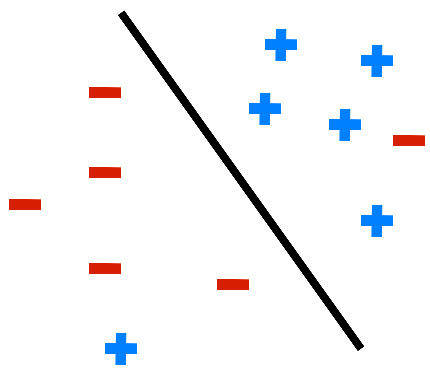
- Mistake Bound: the maximum number of mistakes (binary case) related to the margin or degree of separability

$$\text{mistakes} < \frac{k}{\delta^2}$$

Separable

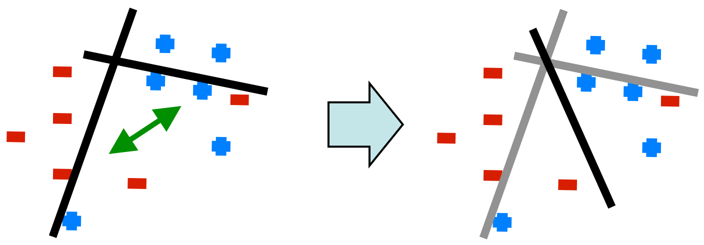


Non-Separable

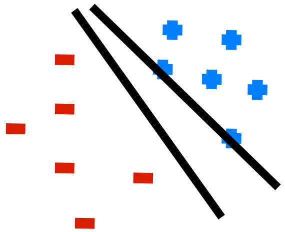


Problems with the Perceptron

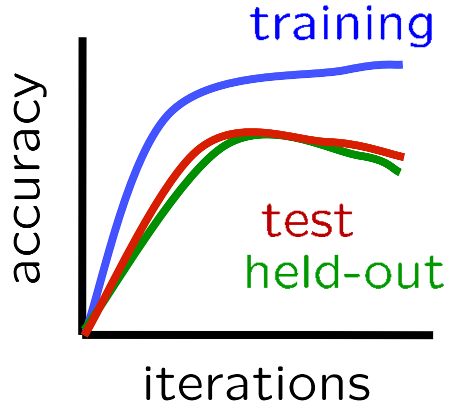
- Noise: if the data isn't separable, weights might thrash
  - Averaging weight vectors over time can help (averaged perceptron)



- Mediocre generalization: finds a "barely" separating solution

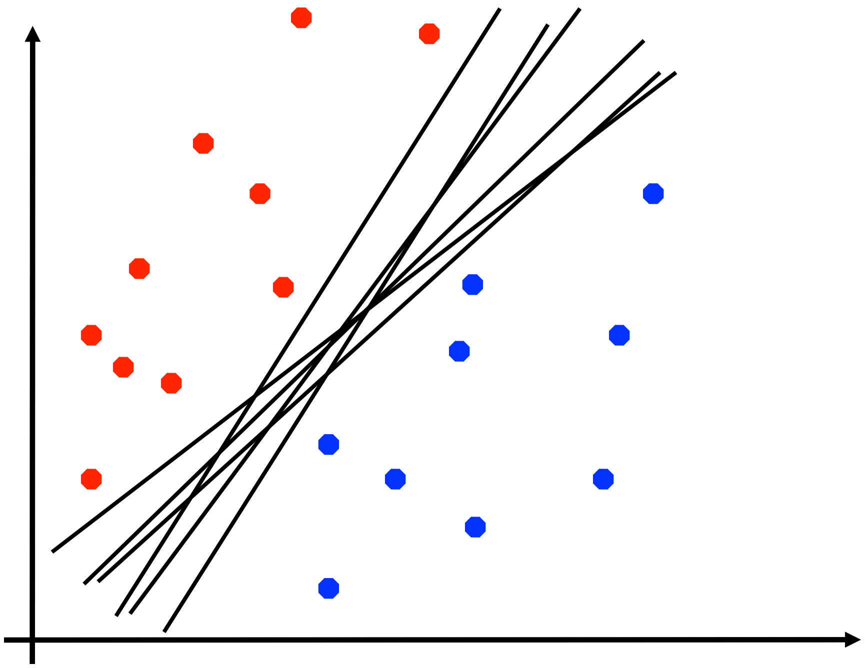


- Overtraining: test / held-out accuracy usually rises, then falls
  - Overtraining is a kind of overfitting



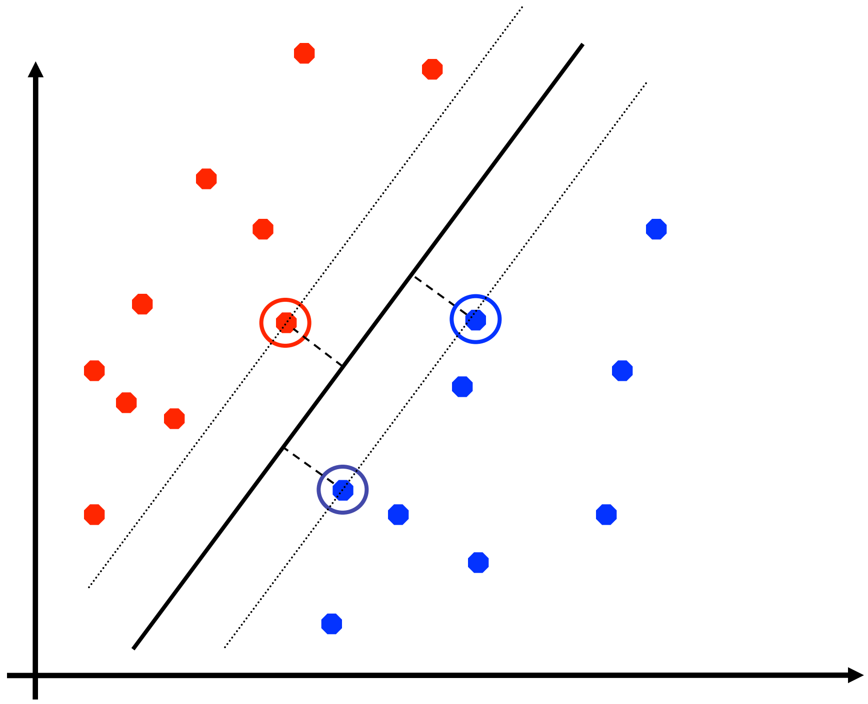
Linear Separators

- 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

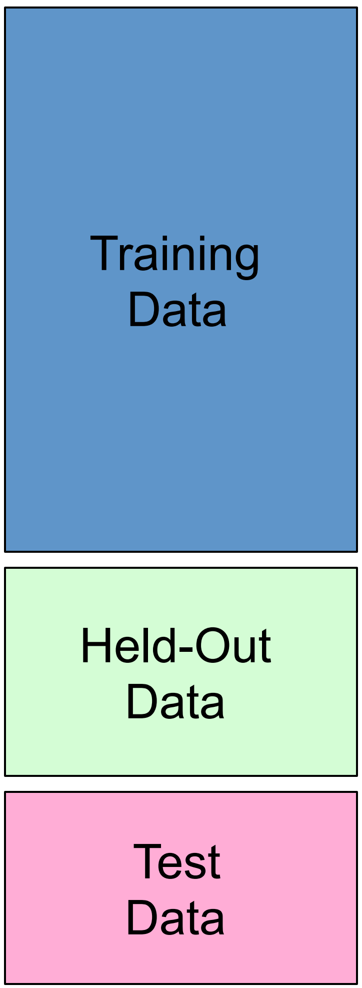


SVM

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

$$\forall i, y \; w_{y^*} \cdot f(x_i) \geq w_y \cdot f(x_i) + 1$$

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: one pass through the data per gradient step, use validation to stop
- The perceptron:
  - Parameters from reactions to mistakes
  - Parameters: discriminative interpretation
  - Training: go through the data until held-out accuracy maxes out