

MACHINE LEARNING

INTRODUCTION TO DATA SCIENCE

TIM KRASKA



CELEBRATION OF KNOWLEDGE - TOPICS

1) ER - diagrams

- What is an entity, relationship, attribute, and a key. How to use them
- What is an n-ary relationship
- How to translate an ER diagram into a relations /tables
- Difference star- vs snow-flake schema.

3) Relational Algebra

- What is the difference between the SQL semantics and relational algebra (sets vs multiset)
- Projection, selection, and joins in relational algebra and how to write queries in it.

2) SQL

Selections, Group-By, Where, Having, Joins, Sub-queries, Different types of joins

3) Data Integration

- What are different problems in data integration
- Edit, Jaccard, and weighted Jaccard distance
- Bag of words
- What is tokenization and what are common problems

4) Data wrangling / Regular expression

- Common syntax/operations
 - matching a single character “.”
 - Or “|”
 - Character classes “[]”
 - Negation “^”
 - Ranges
 - Anchor “^” “\$”
 - Repetition Ranges

5) Statistics

- Basic statistic (e.g., sample mean, variance) and how to implement them at scale
- Random variables
- Bayes rule
- Central limit theorem:
- How to read a normal distribution, t-distribution, etc. Table
- p-test (especially t-test)
- What are problems with p values

6) Visualization

- Different types of visualizations
- What makes a good visualization
- What are potential problems with visualization

MONTY HALL PROBLEM

Let's Make a Deal

1. One door hides a car, two hide goats.
2. You choose door.
3. Monty (who knows where the car is) opens a different door with a goat.
4. You can switch doors or keep your original choice.

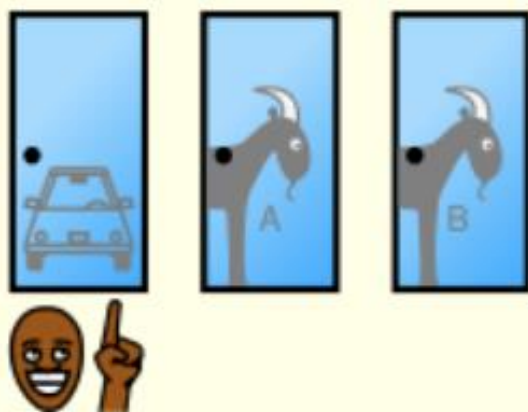


What is the best strategy for winning a car?

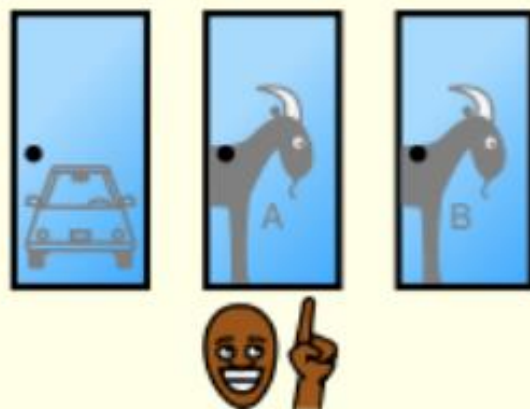
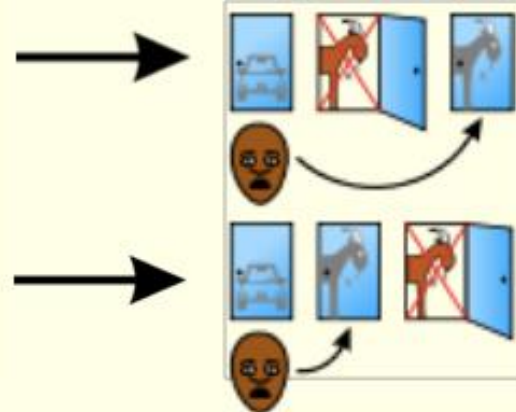
- A) Switch
- B) Don't switch
- C) It doesn't matter

You can play it here:

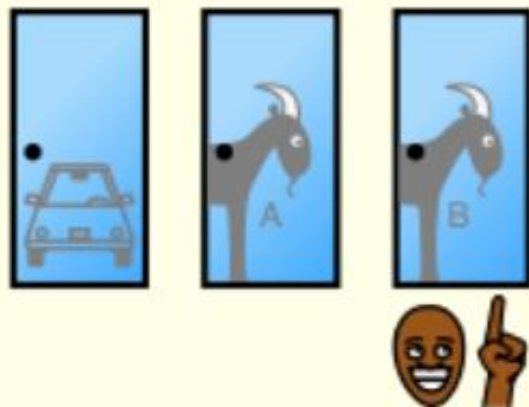
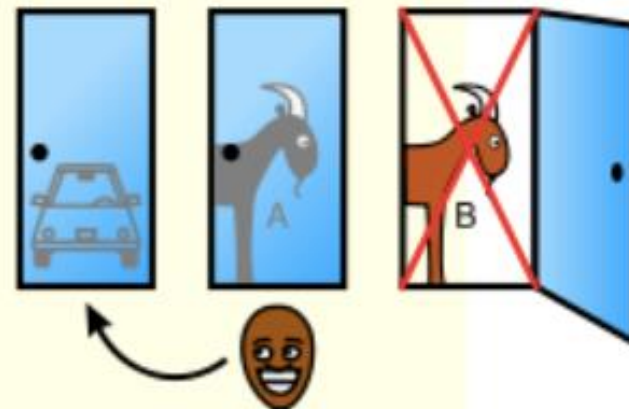
<http://math.ucsd.edu/~crypto/Monty/monty.html>



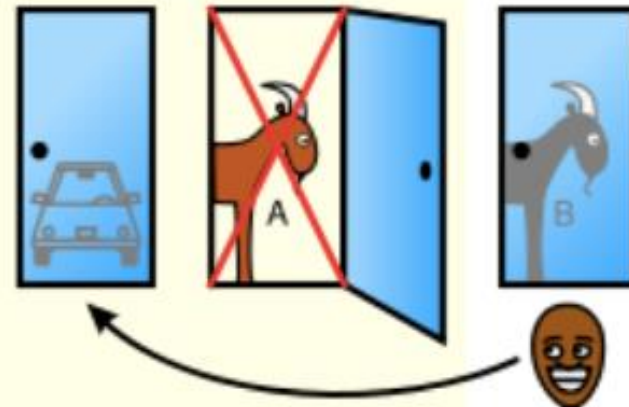
*Host reveals
Goat A
or
Host reveals
Goat B*



*Host must
reveal Goat B*



*Host must
reveal Goat A*



MONTY HALL PROBLEM: BAYES RULE

You pick door A.

Data D = Monty opened door B

Hypothesis space H :

h_1 = Car is behind door A

h_2 = Car is behind door C

h_3 = Car is behind door B

what is $P(h_1 | D)$?

what is $P(h_2 | D)$?

what is $P(h_3 | D)$?

Prior probability:

$$P(h_1) = 1/3 \quad P(h_2) = 1/3 \quad P(h_3) = 1/3$$

Likelihood:

$$P(D | h_1) = 1/2$$

$$P(D | h_2) = 1$$

$$P(D | h_3) = 0$$

$$P(D) = p(D|h_1)p(h_1) + p(D|h_2)p(h_2) + p(D|h_3)p(h_3) = 1/6 + 1/3 + 0 = 1/2$$

By Bayes rule:

$$P(h_1|D) = P(D|h_1)p(h_1) / P(D) = \frac{1}{2} \cdot \frac{1}{3} / \frac{1}{2} = \frac{1}{3}$$

$$P(h_2|D) = P(D|h_2)p(h_2) / P(D) = \frac{1}{2} \cdot \frac{1}{3} / \frac{1}{2} = \frac{2}{3}$$

$$P(X | Y) = \frac{P(Y | X)P(X)}{P(Y)}$$

Assume that we have two classes

$c_1 = \text{male}$, and $c_2 = \text{female}$.

We have a person whose sex we do not know, say “*drew*” or *d*.

Classifying *drew* as male or female is equivalent to asking is it more probable that *drew* is **male** or **female**, I.e which is greater $p(\text{male} | \text{drew})$ or $p(\text{female} | \text{drew})$

(Note: “Drew can be a male or female name”)



Drew Barrymore



Drew Carey

What is the probability of being called “*drew*” given that you are a **male**?

What is the probability of being a **male**?

$$p(\text{male} | \text{drew}) = \frac{p(\text{drew} | \text{male}) p(\text{male})}{p(\text{drew})}$$

$p(\text{drew})$

What is the probability of being named “*drew*”?

(actually irrelevant, since it is that same for all classes)

Is Officer Drew a Male or Female?

Luckily, we have a small database with names and sex.

We can use it to apply Bayes rule...

Name	Sex
Drew	Male
Claudia	Female
Drew	Female
Drew	Female
Alberto	Male
Karin	Female
Nina	Female
Sergio	Male

Officer Drew

- A) $P(\text{male} | \text{drew}) = 0.25 / (3/8)$
 $P(\text{female} | \text{drew}) = 0.25 / (3/8)$
- B) $P(\text{male} | \text{drew}) = 0.125 / (3/8)$
 $P(\text{female} | \text{drew}) = 0.250 / (3/8)$
- C) $P(\text{male} | \text{drew}) = 0.5$
 $P(\text{female} | \text{drew}) = 0.5$
- D) $P(\text{male} | \text{drew}) = 1$
 $P(\text{female} | \text{drew}) = 0.5$



Officer Drew

$$p(c_j | d) = \frac{p(d | c_j) p(c_j)}{p(d)}$$

Name	Sex
Drew	Male
Claudia	Female
Drew	Female
Drew	Female
Alberto	Male
Karin	Female
Nina	Female
Sergio	Male

$$p(\text{male} | \text{drew}) = \frac{1/3 * 3/8}{3/8} = \frac{0.125}{3/8}$$

$$p(\text{female} | \text{drew}) = \frac{2/5 * 5/8}{3/8} = \frac{0.250}{3/8}$$

Officer Drew is more likely to be a **Female**.



Officer Drew IS a female!

Officer Drew

$$p(\text{male} | drew) = \frac{1/3 * 3/8}{3/8} = \frac{0.125}{3/8}$$

$$p(\text{female} | drew) = \frac{2/5 * 5/8}{3/8} = \frac{0.250}{3/8}$$

MACHINE LEARNING PROBLEMS

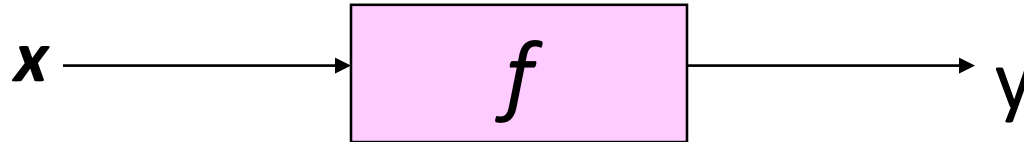
	Supervised Learning	Unsupervised Learning
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Many classifiers to choose from

- Decision Trees
- K-nearest neighbor
- Support Vector Machines
- Logistic Regression
- Naïve Bayes
- Random Forrest
- Bayesian network
- Randomized Forests
- Boosted Decision Trees
- RBMs
-

LINEAR CLASSIFIER

Find a function to classify High Value Customers



High Value Customers

Salary	Nb Orders
150	70
300	100
200	80
120	100

Low Value Customers

Salary	Nb Orders
40	80
220	20
100	20
175	10

Task: Find a a_1, a_2, a_3 :

High value customer $a_1 \square salary + a_2 \square orders - a_3 > 0$

Low value customer $a_1 \square salary + a_2 \square orders - a_3 < 0$

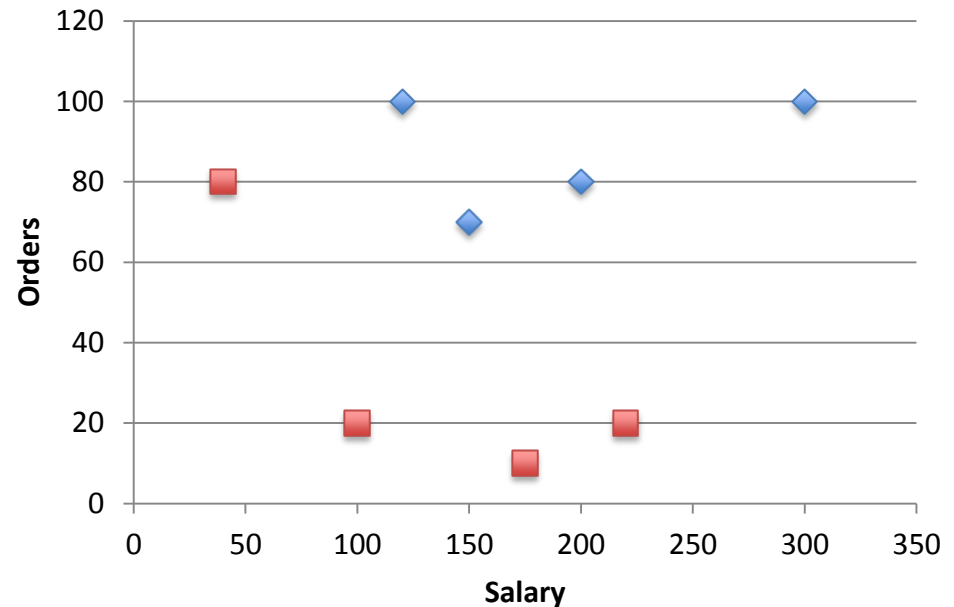
Find a function to classify High Value Customers

High Value
Customers

Salary	Orders
150	70
300	100
200	80
120	100

Low Value
Customers

Salary	Orders
40	80
220	20
100	20
175	10



Task: Find a a_1, a_2, a_3 :

High value customer $a_1 \square salary + a_2 \square orders - a_3 > 0$

Low value customer $a_1 \square salary + a_2 \square orders - a_3 < 0$

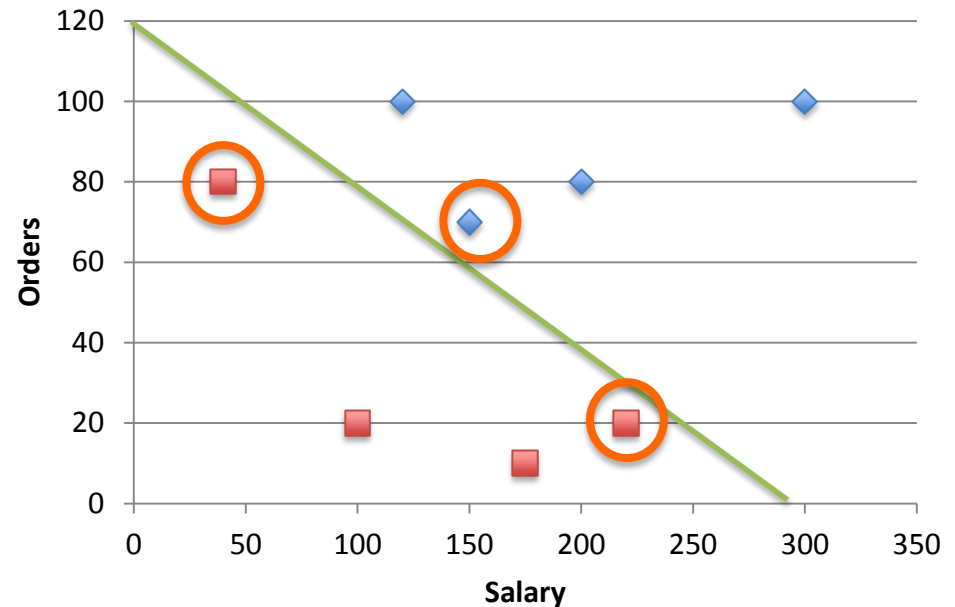
Find a function to classify High Value Customers

High Value
Customers

Salary	Orders
150	70
300	100
200	80
120	100

Low Value
Customers

Salary	Orders
40	80
220	20
100	20
175	10

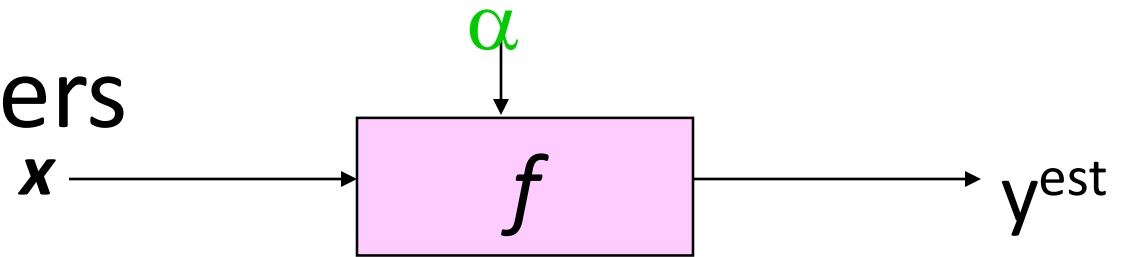


Task: Find a a_1, a_2, a_3 :

High value customer $a_1 \square salary + a_2 \square orders - a_3 > 0$

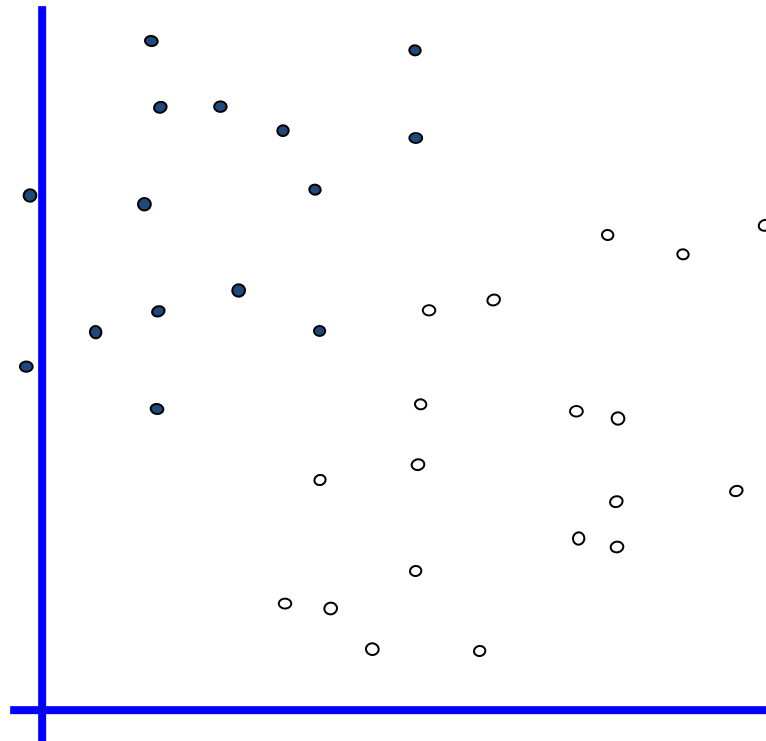
Low value customer $a_1 \square salary + a_2 \square orders - a_3 < 0$

Linear Classifiers



$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

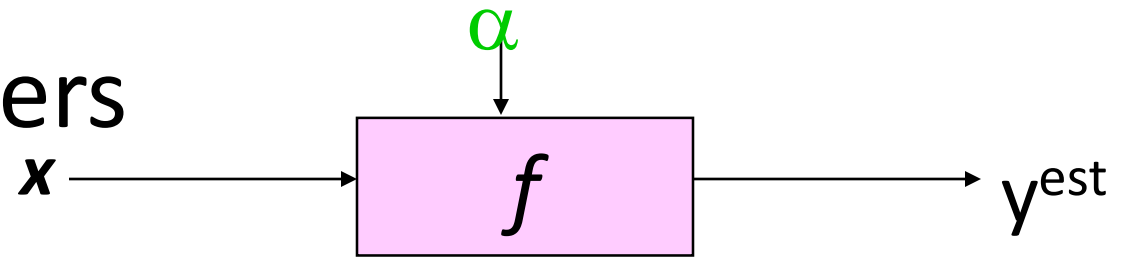
- denotes +1
- denotes -1



How would you classify this data?

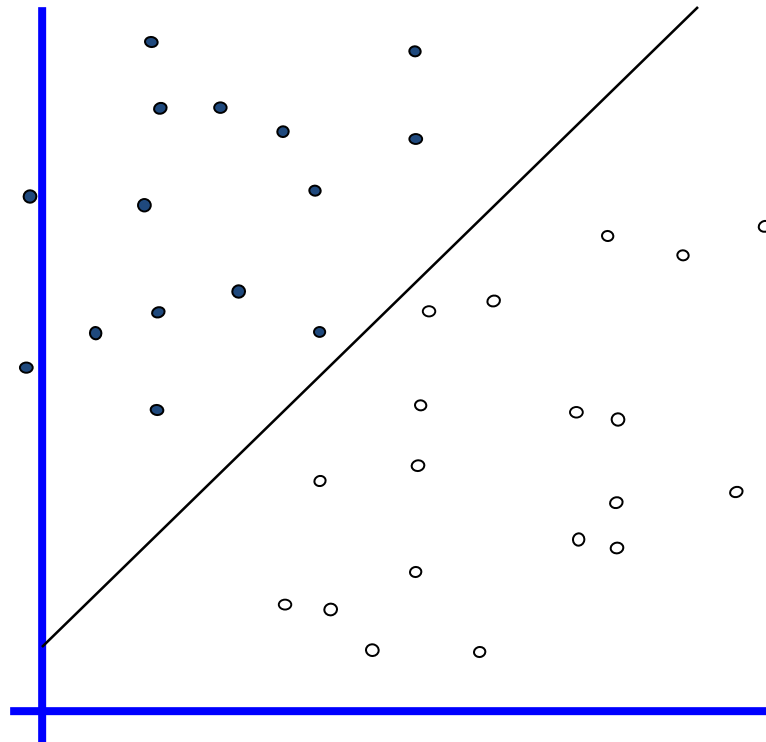
\mathbf{x} and \mathbf{w} are vectors

Linear Classifiers



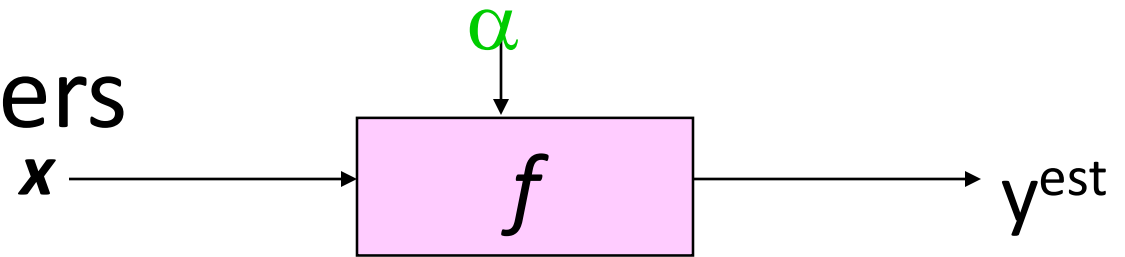
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

- denotes +1
- denotes -1



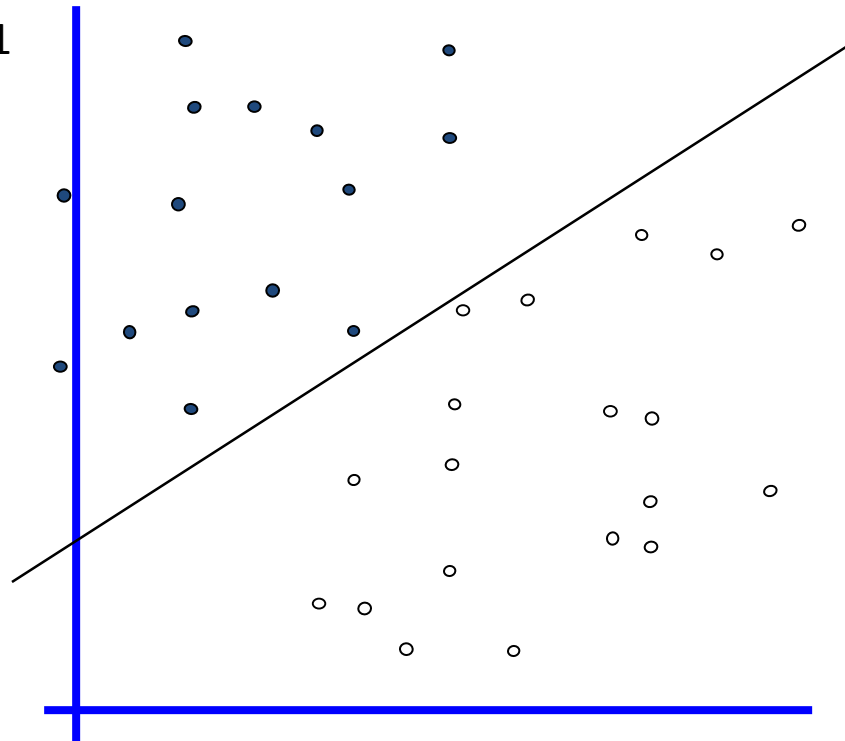
How would you classify this data?

Linear Classifiers



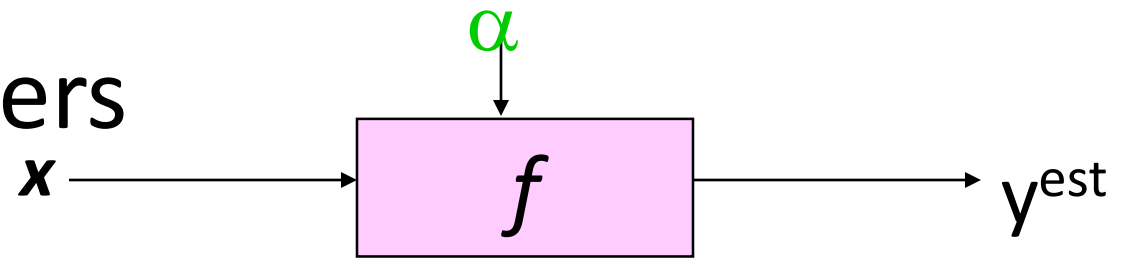
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

- denotes +1
- denotes -1

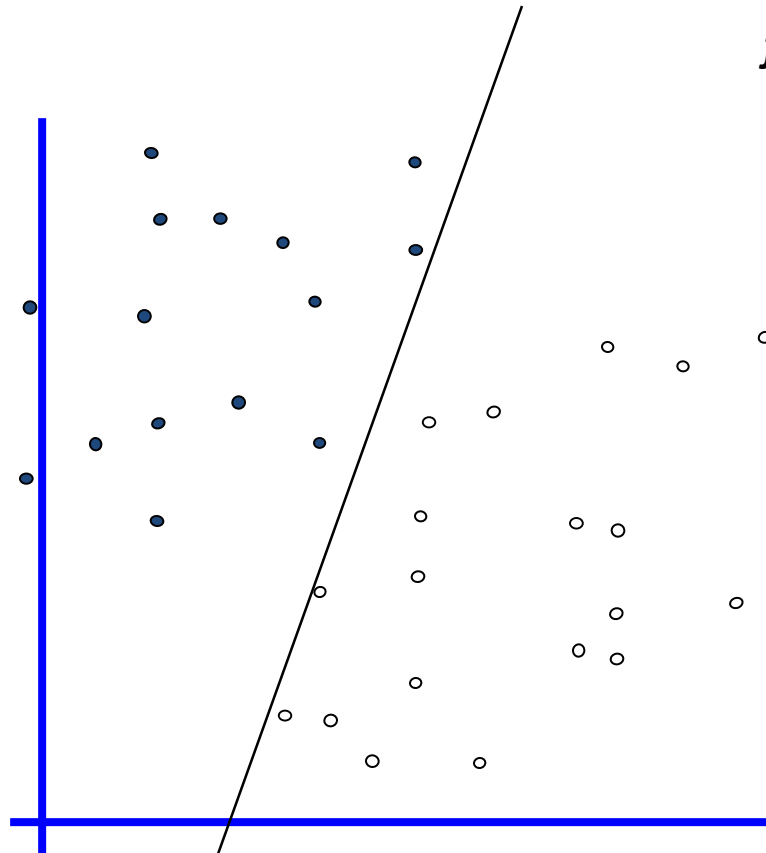


How would you classify this data?

Linear Classifiers



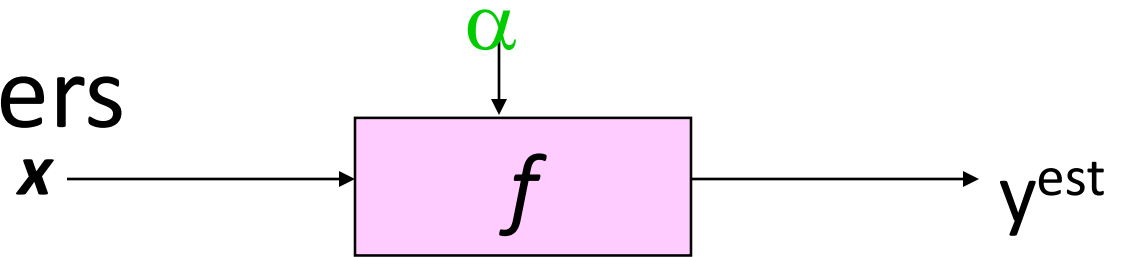
- denotes +1
- denotes -1



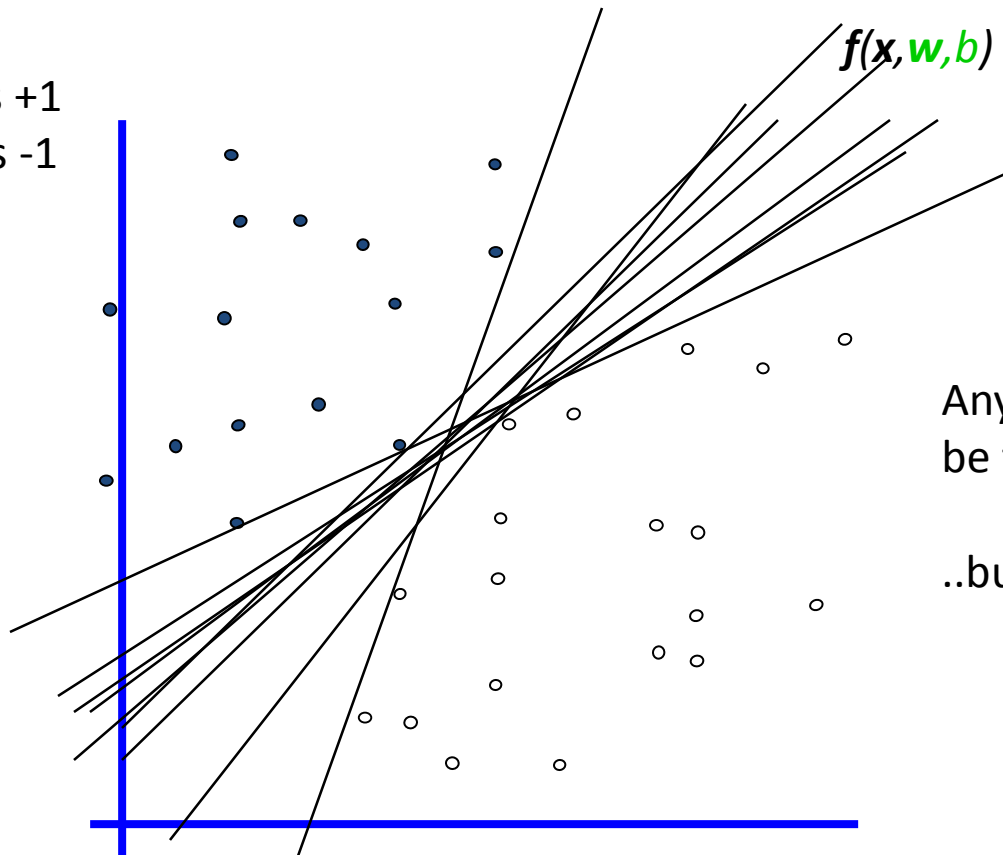
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

How would you classify this data?

Linear Classifiers



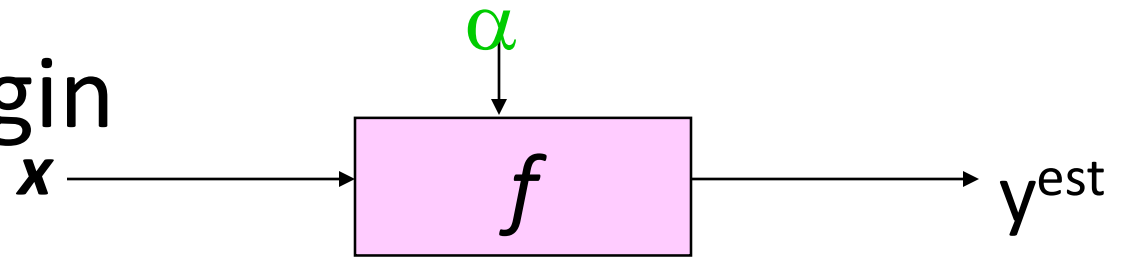
- denotes +1
- denotes -1



Any of these would
be fine..

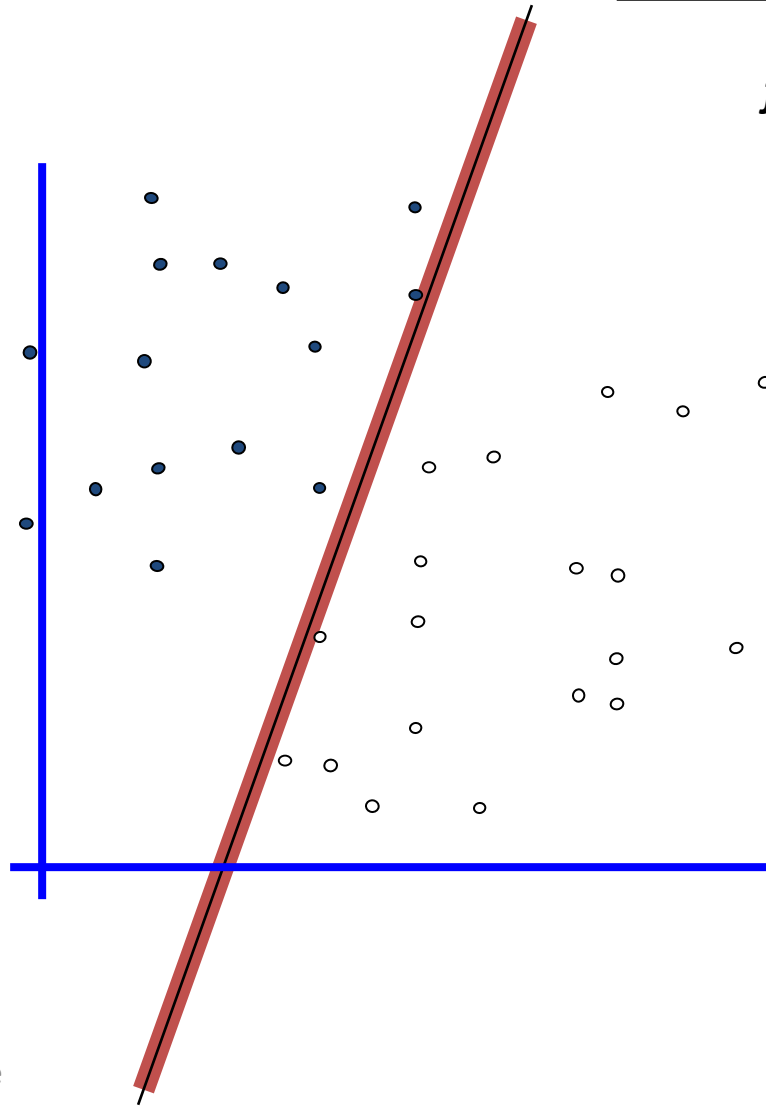
..but which is best?

Classifier Margin



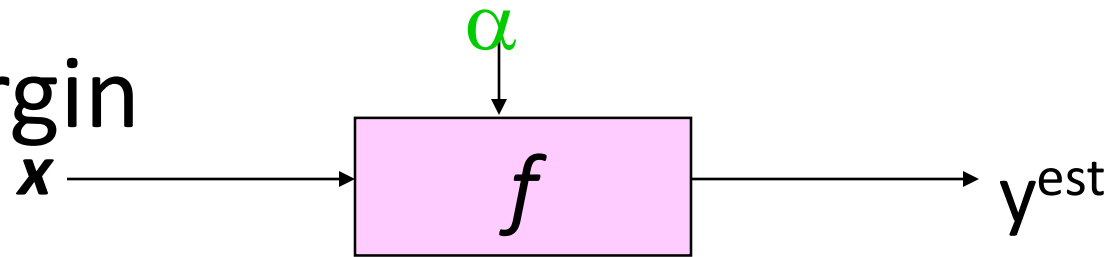
$$f(x, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot x - b)$$

- denotes +1
- denotes -1

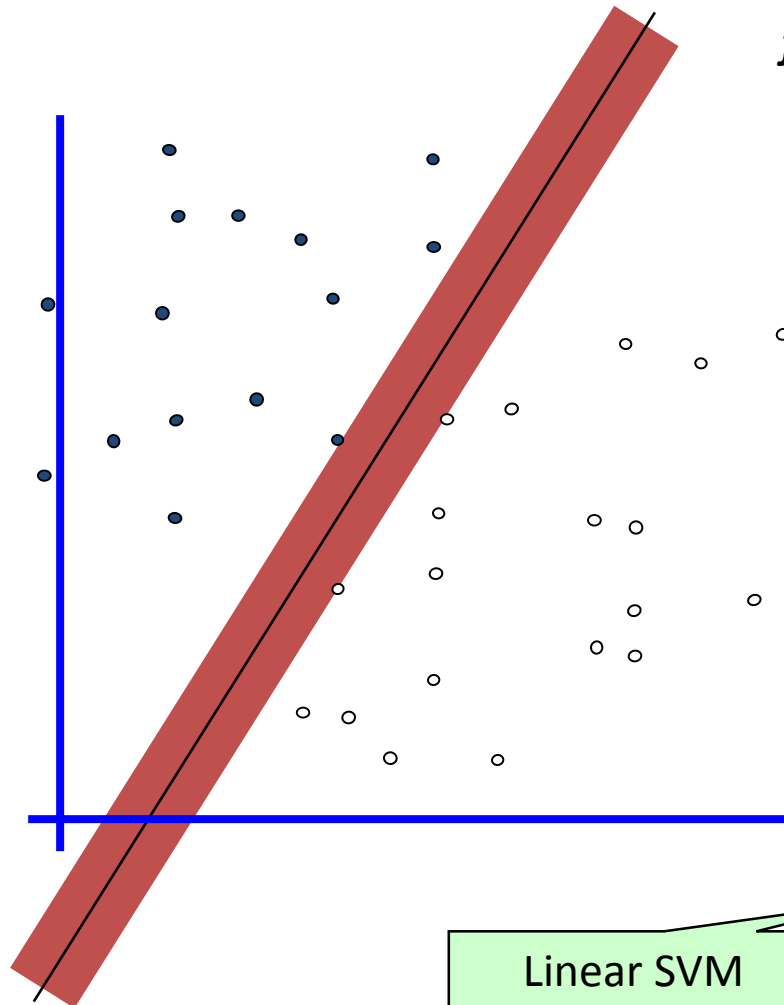


Define the **margin** of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Maximum Margin



- denotes +1
- denotes -1



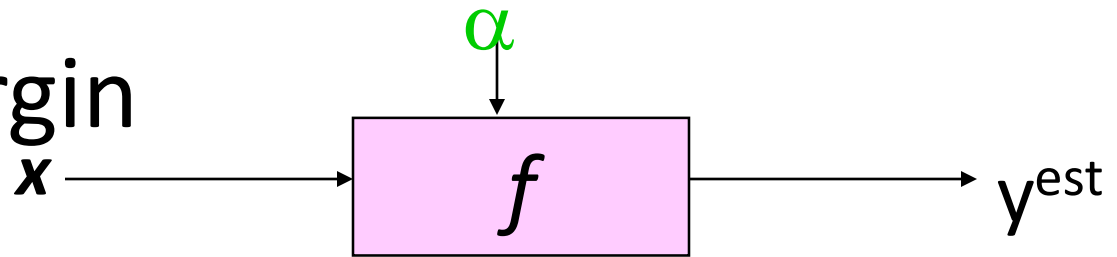
$$f(x, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

The **maximum margin linear classifier** is the linear classifier with the, um, maximum margin.

This is the simplest kind of SVM (Called an LSVM)

Linear SVM

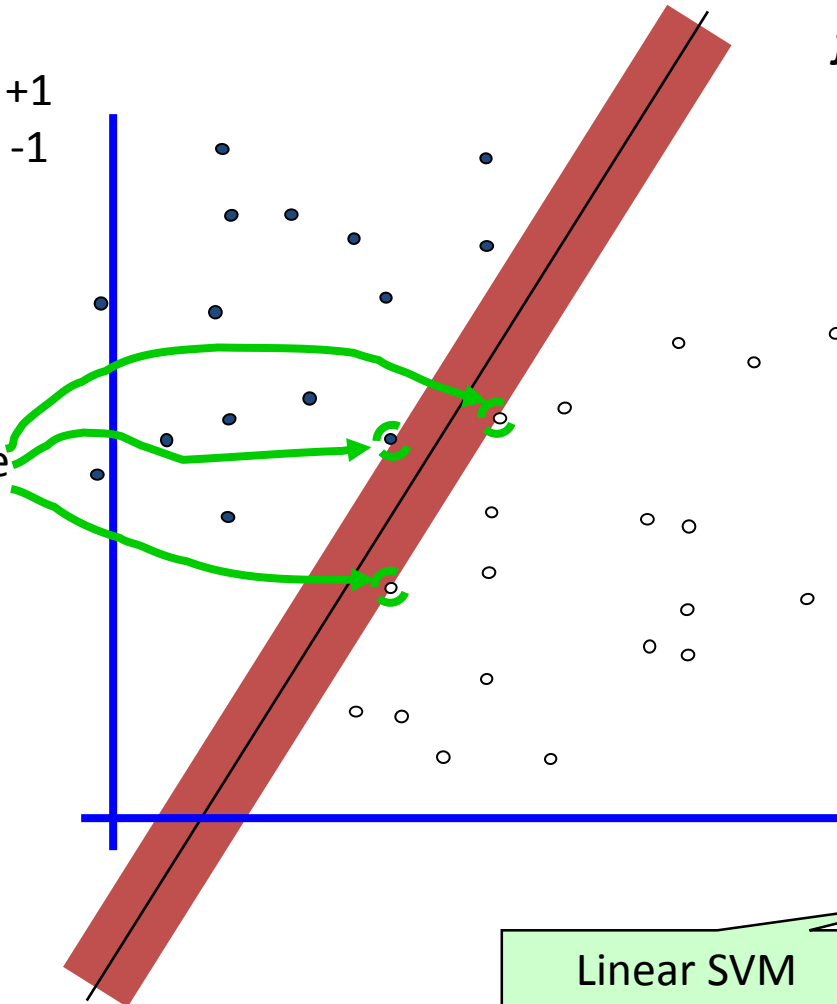
Maximum Margin



$$f(x, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

- denotes +1
- denotes -1

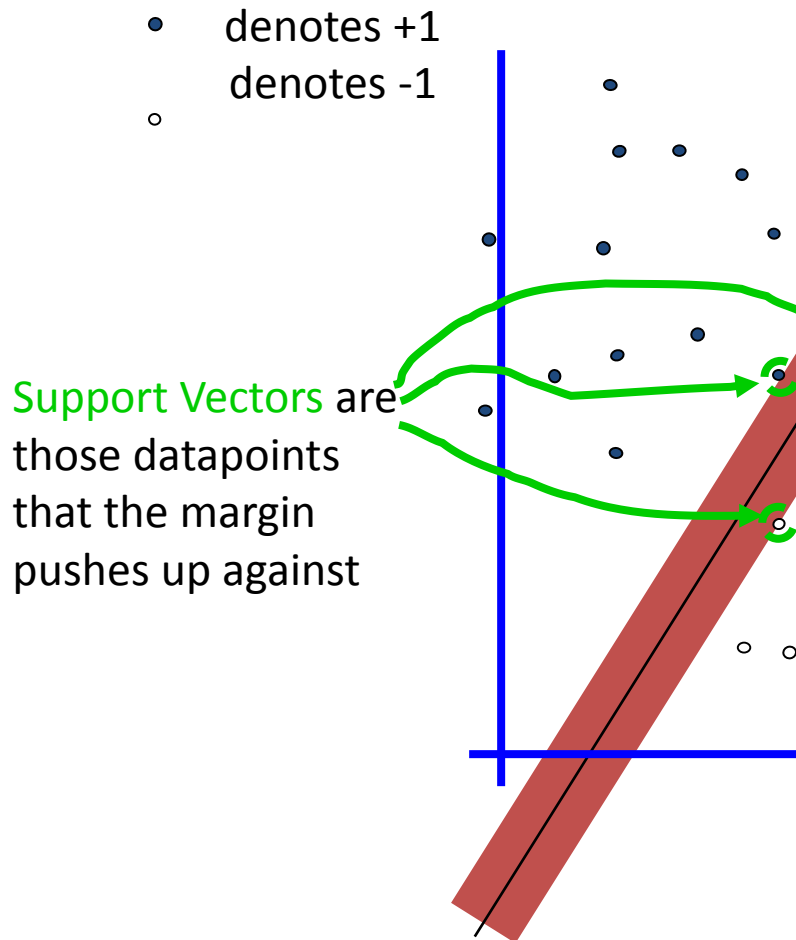
Support Vectors are those datapoints that the margin pushes up against



The **maximum margin linear classifier** is the linear classifier with the, um, maximum margin.
This is the simplest kind of SVM (Called an LSVM)

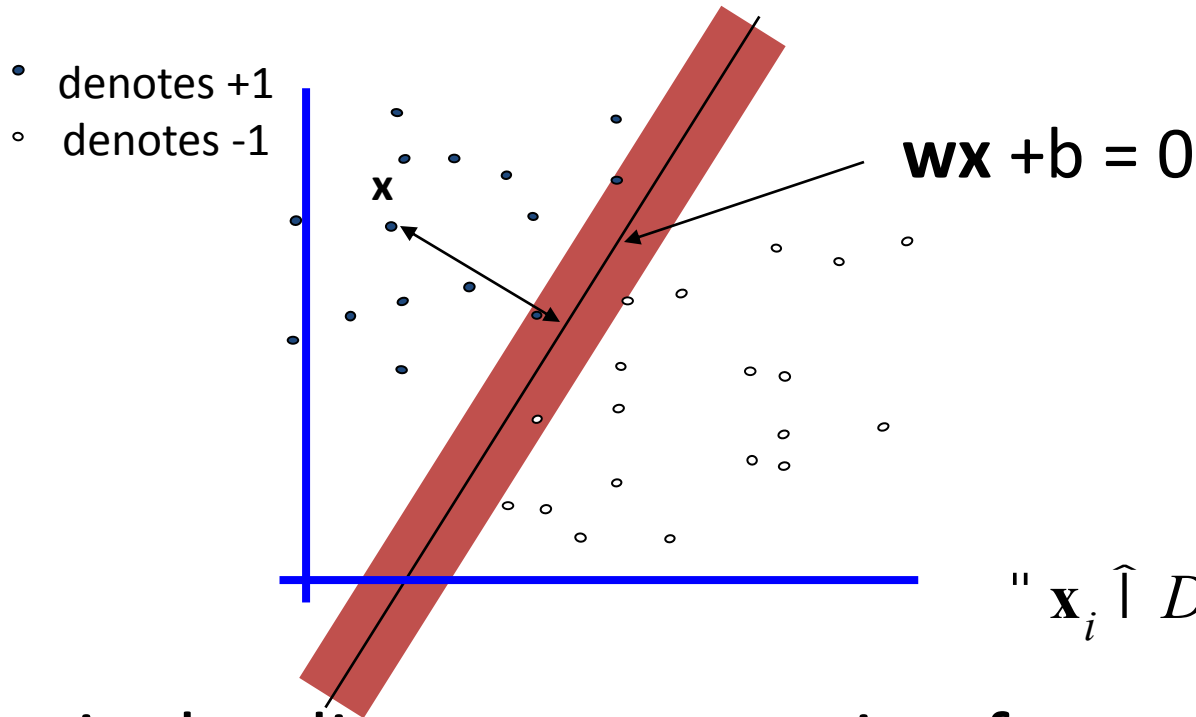
Linear SVM

Why Maximum Margin?



1. Intuitively this feels safest.
2. If we've made a small error in the location of the boundary (it's been jolted in its perpendicular direction) this gives us least chance of causing a misclassification.
3. Leave-one-out-cross-validation (LOOCV) is easy since the model is immune to removal of any non-support-vector datapoints.
4. There's some theory (using VC dimension) that is related to (but not the same as) the proposition that this is a good thing.
5. Empirically it works very very well.

Estimate the Margin

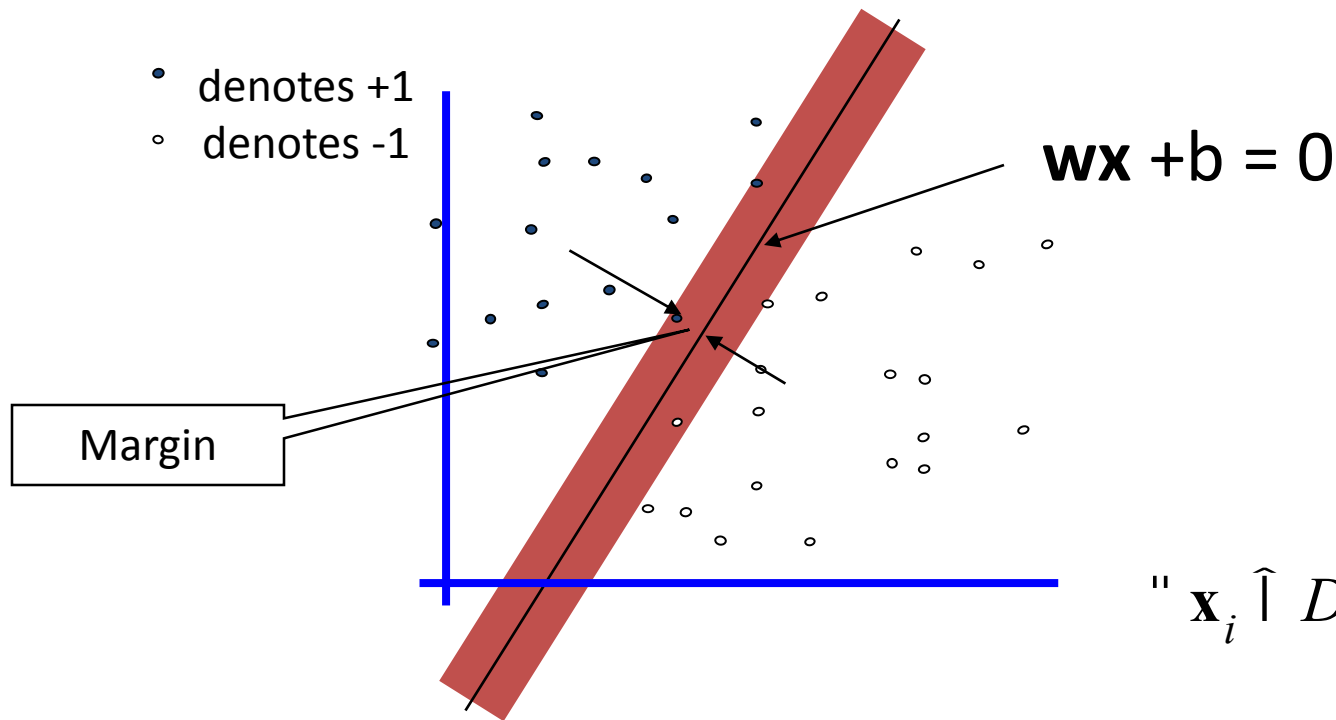


$$\mathbf{x}_i \in D: y_i (\mathbf{x}_i \times \mathbf{w} + b) > 0$$

- What is the distance expression for a point \mathbf{x} to a line $\mathbf{w}\mathbf{x} + b = 0$?

$$d(\mathbf{x}) = \frac{|\mathbf{x} \times \mathbf{w} + b|}{\|\mathbf{w}\|_2} = \frac{|\mathbf{x} \times \mathbf{w} + b|}{\sqrt{\sum_{i=1}^d w_i^2}}$$

Estimate the Margin

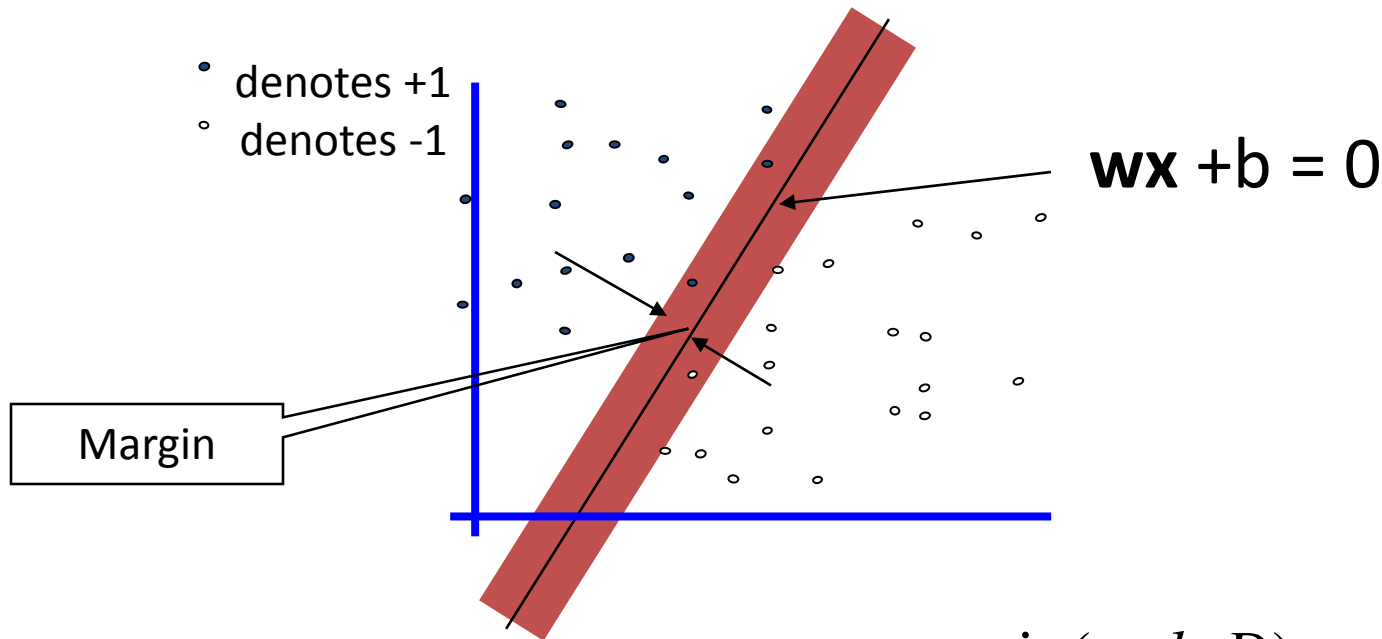


$$\mathbf{x}_i \in D : y_i (\mathbf{x}_i \cdot \mathbf{w} + b) > 0$$

- What is the expression for margin?

$$\text{margin} \equiv \min_{\mathbf{x} \in D} d(\mathbf{x}) = \min_{\mathbf{x} \in D} \frac{|\mathbf{x} \cdot \mathbf{w} + b|}{\sqrt{\sum_{i=1}^d w_i^2}}$$

Maximize Margin

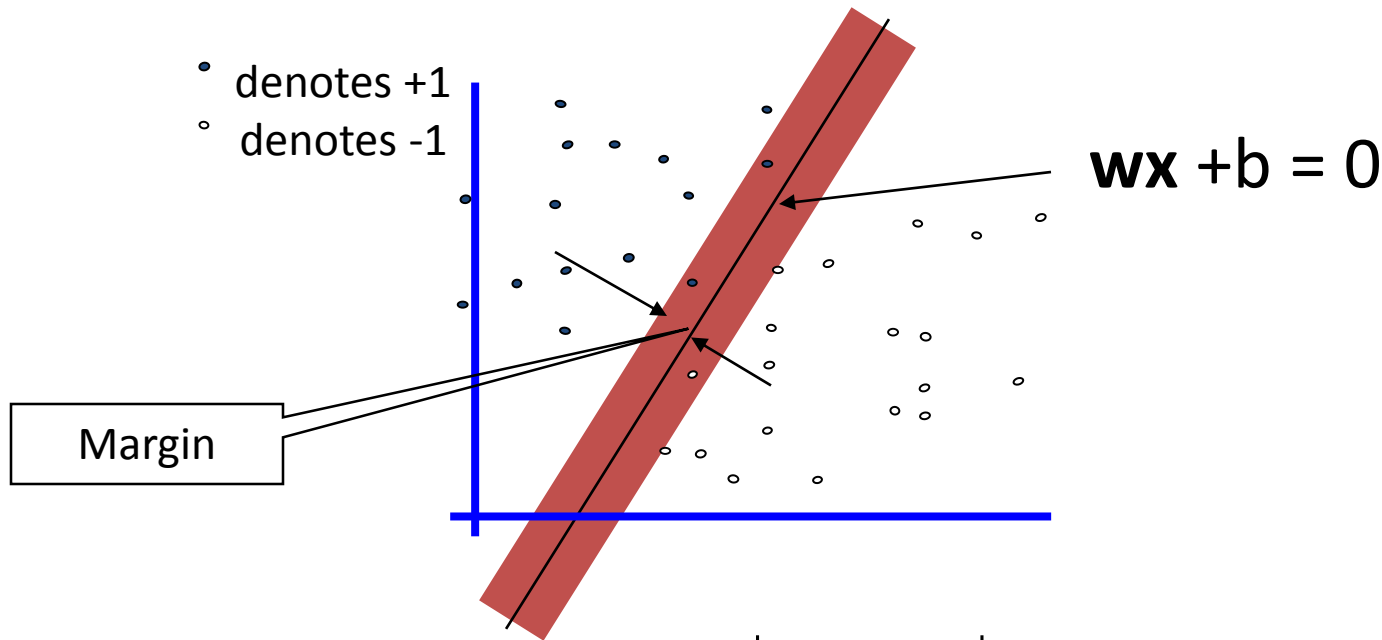


$$\operatorname{argmax}_{\mathbf{w}, b} \operatorname{margin}(\mathbf{w}, b, D)$$

$$= \operatorname{argmax}_{\mathbf{w}, b} \min_{\mathbf{x}_i \in D} d(\mathbf{x}_i)$$

$$= \operatorname{argmax}_{\mathbf{w}, b} \min_{\mathbf{x}_i \in D} \frac{|b + \mathbf{x}_i \times \mathbf{w}|}{\sqrt{\sum_{i=1}^d w_i^2}}$$

Maximize Margin

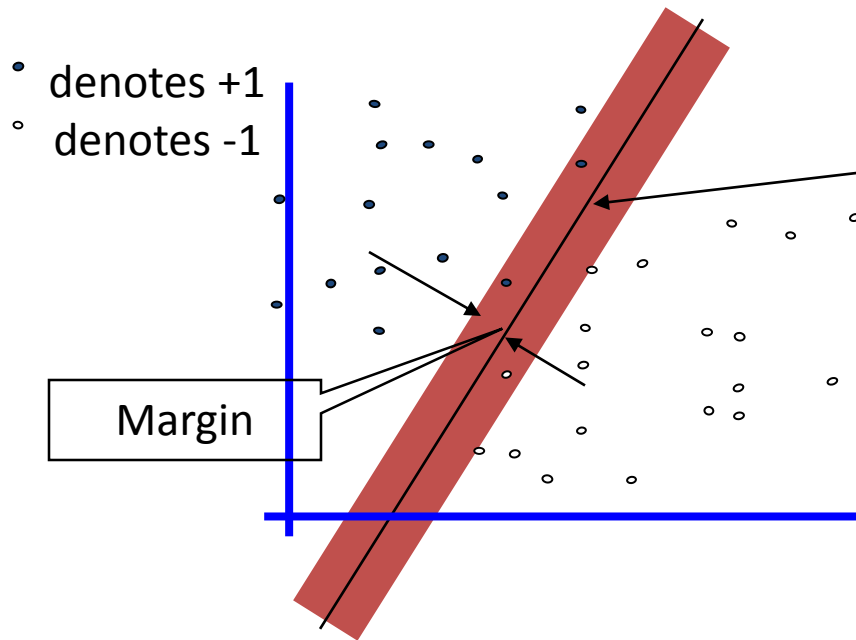


$$\operatorname{argmax}_{\mathbf{w}, b} \min_{\mathbf{x}_i \in D} \frac{|b + \mathbf{x}_i \cdot \mathbf{w}|}{\sqrt{\sum_{i=1}^d w_i^2}}$$

$$\text{subject to } \forall \mathbf{x}_i \in D : y_i (\mathbf{x}_i \cdot \mathbf{w} + b) > 0$$

- Min-max problem \rightarrow game problem

Maximize Margin



$$\mathbf{w}\mathbf{x} + b = 0$$

$$\operatorname{argmax}_{\mathbf{w}, b} \min_{\mathbf{x}_i \in D} \frac{|b + \mathbf{x}_i \cdot \mathbf{w}|}{\sqrt{\sum_{i=1}^d w_i^2}}$$

$$\text{subject to } \forall \mathbf{x}_i \in D : y_i (\mathbf{x}_i \cdot \mathbf{w} + b) \geq 0$$



$$\operatorname{argmin}_{\mathbf{w}, b} \sum_{i=1}^d w_i^2$$

$$\text{subject to } \forall \mathbf{x}_i \in D : y_i (\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1$$

Strategy:

$$\forall \mathbf{x}_i \in D : |b + \mathbf{x}_i \cdot \mathbf{w}| \geq 1$$

If you want to learn why this holds I highly recommend the following video

https://www.youtube.com/watch?v=_PwhiWxHK8o (not necessary to know for the exam 2)

Maximum Margin Linear Classifier

$$\{\vec{w}^*, b^*\} = \operatorname{argmin}_{\vec{w}, b} \sum_{k=1}^d w_k^2$$

subject to

$$y_1 (\vec{w} \cdot \vec{x}_1 + b) \geq 1$$

$$y_2 (\vec{w} \cdot \vec{x}_2 + b) \geq 1$$

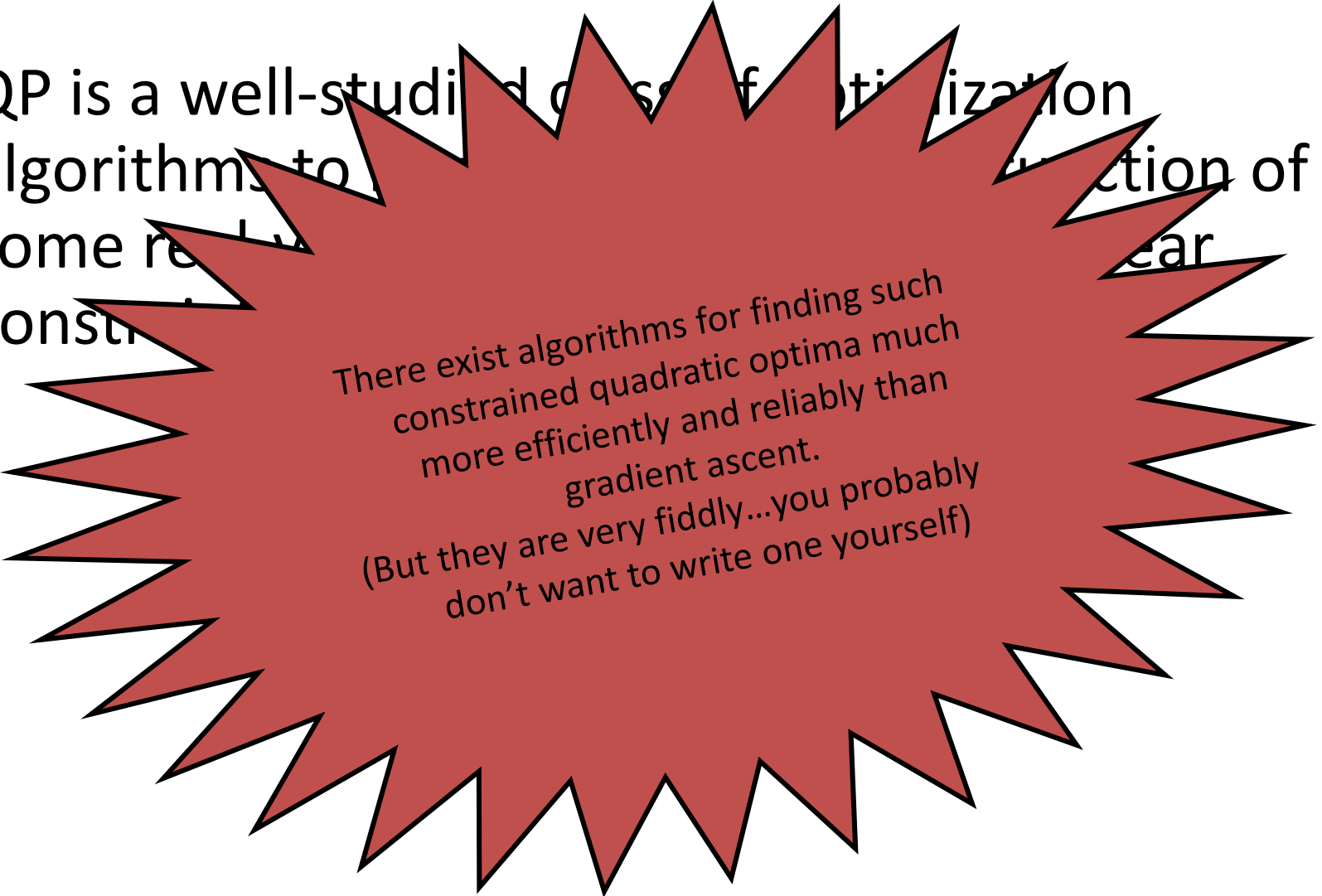
....

$$y_N (\vec{w} \cdot \vec{x}_N + b) \geq 1$$

- How to solve it?

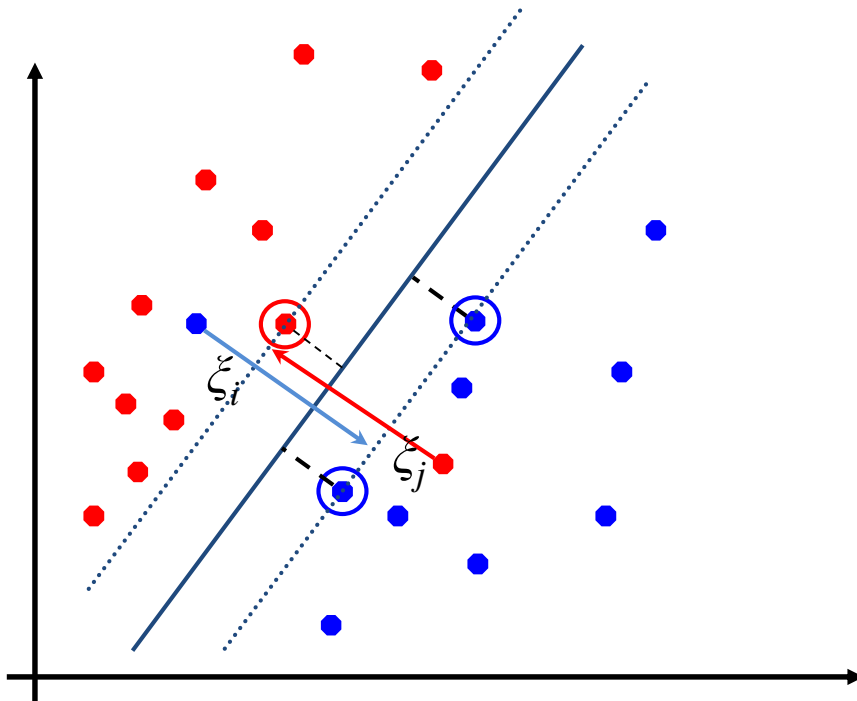
Learning via Quadratic Programming

QP is a well-studied class of optimization algorithms to find the minimum of some real-valued function of linear constraints.



There exist algorithms for finding such constrained quadratic optima much more efficiently and reliably than gradient ascent.
(But they are very fiddly...you probably don't want to write one yourself)

Soft Margin Classification



- If the training data is not linearly separable, *slack variables* ξ_i can be added to allow misclassification of difficult or noisy examples.
- Allow some errors
 - Let some points be moved to where they belong, at a cost
- Still, try to minimize training set errors, and to place hyperplane “far” from each class (large margin)

Regularization

Soft Margin Classification Mathematically

- The old formulation:

Find \mathbf{w} and b such that

$\Phi(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w}$ is minimized and for all $\{(\mathbf{x}_i, y_i)\}$

$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$$

- The new formulation incorporating slack variables:

Find \mathbf{w} and b such that

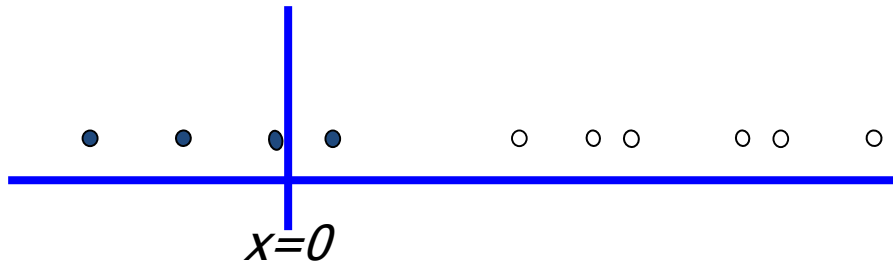
$\Phi(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum \xi_i$ is minimized and for all $\{(\mathbf{x}_i, y_i)\}$

$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0 \text{ for all } i$$

- Parameter C can be viewed as a way to control overfitting
 - A regularization term

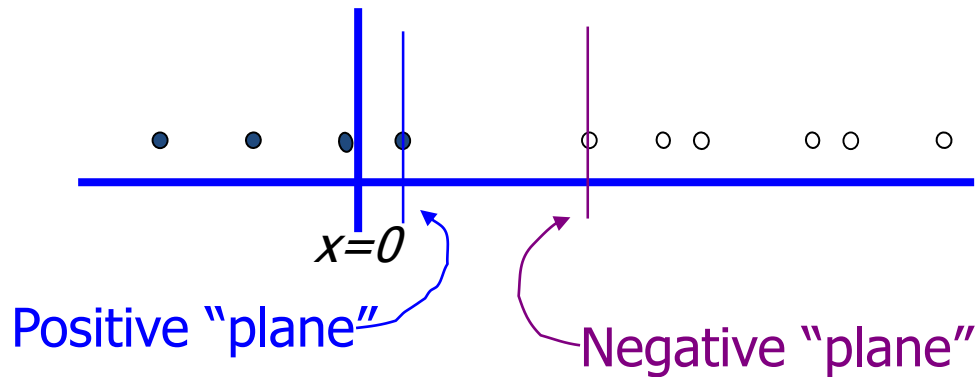
Suppose we're in 1-dimension

What would
SVMs do with
this data?



Suppose we're in 1-dimension

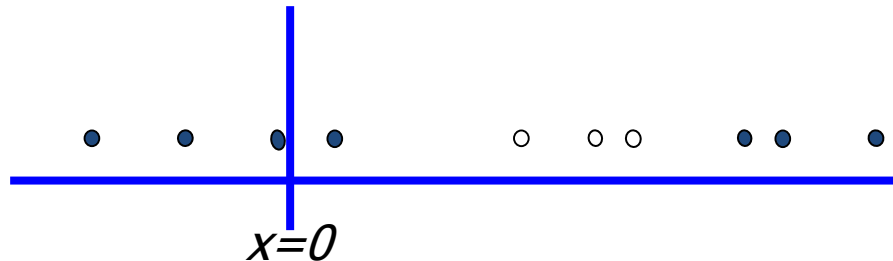
Not a big surprise



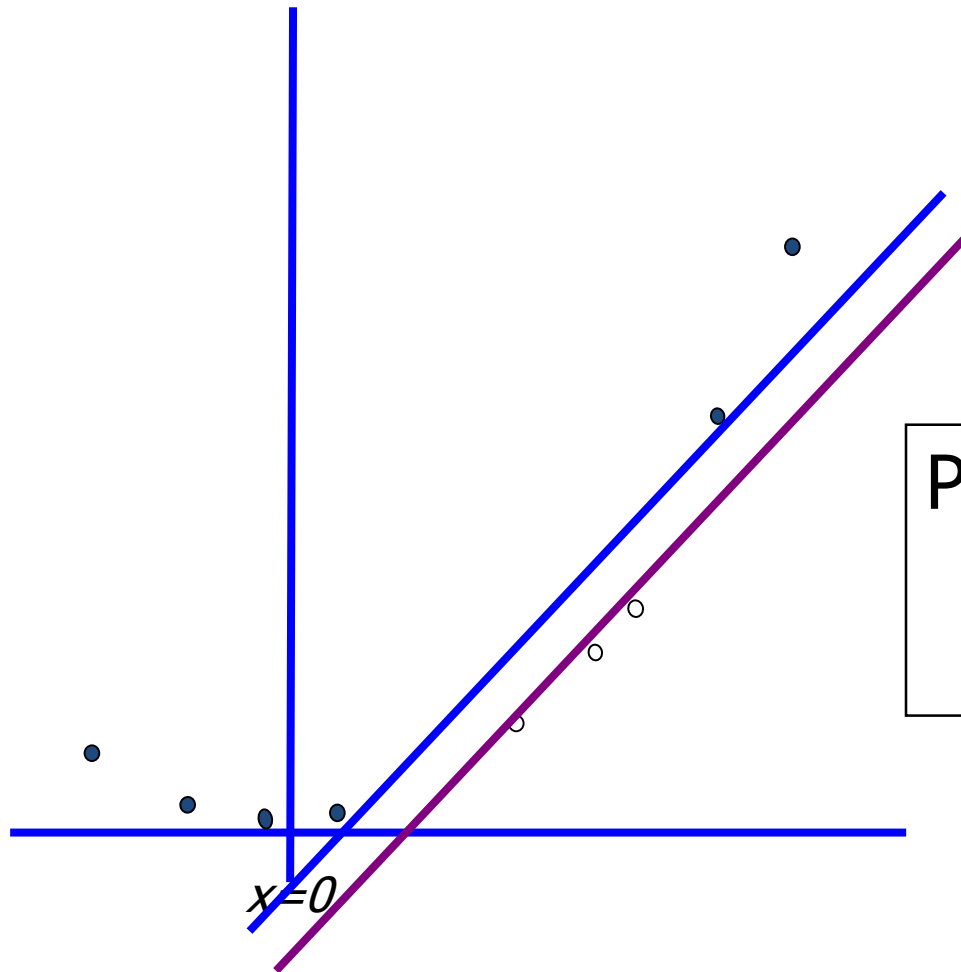
Harder 1-dimensional dataset

That's wiped the smirk off SVM's face.

What can be done about this?



Harder 1-dimensional dataset



Permitting non-linear basis functions

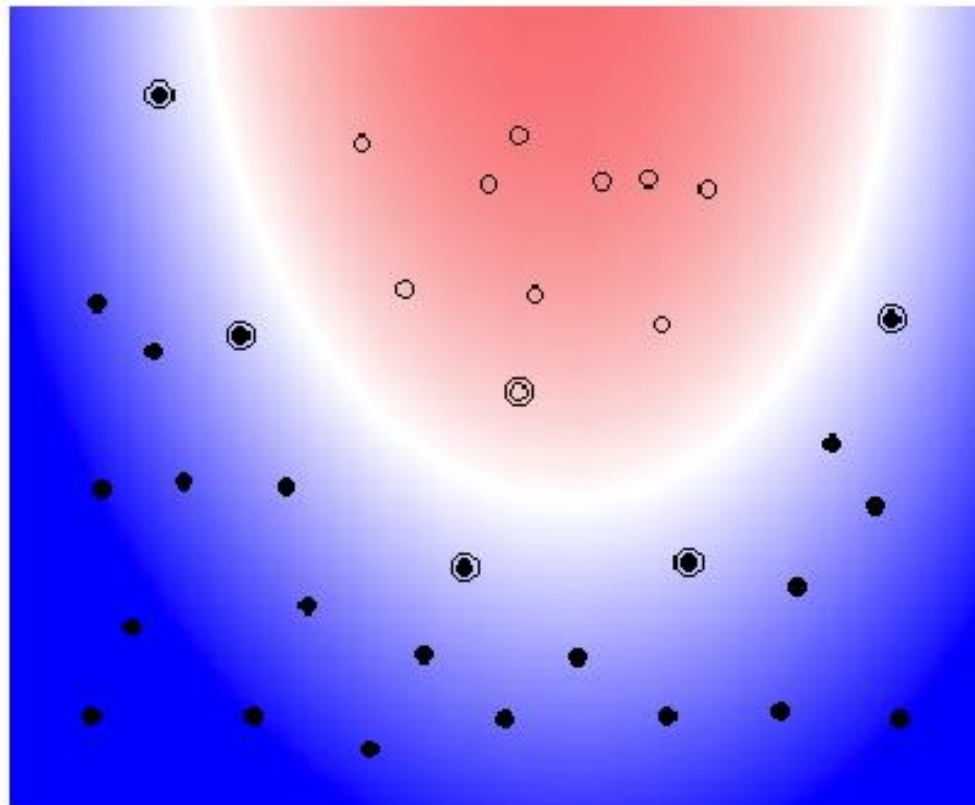
$$\mathbf{z}_k = (x_k, x_k^2)$$

Nonlinear Kernel (I)

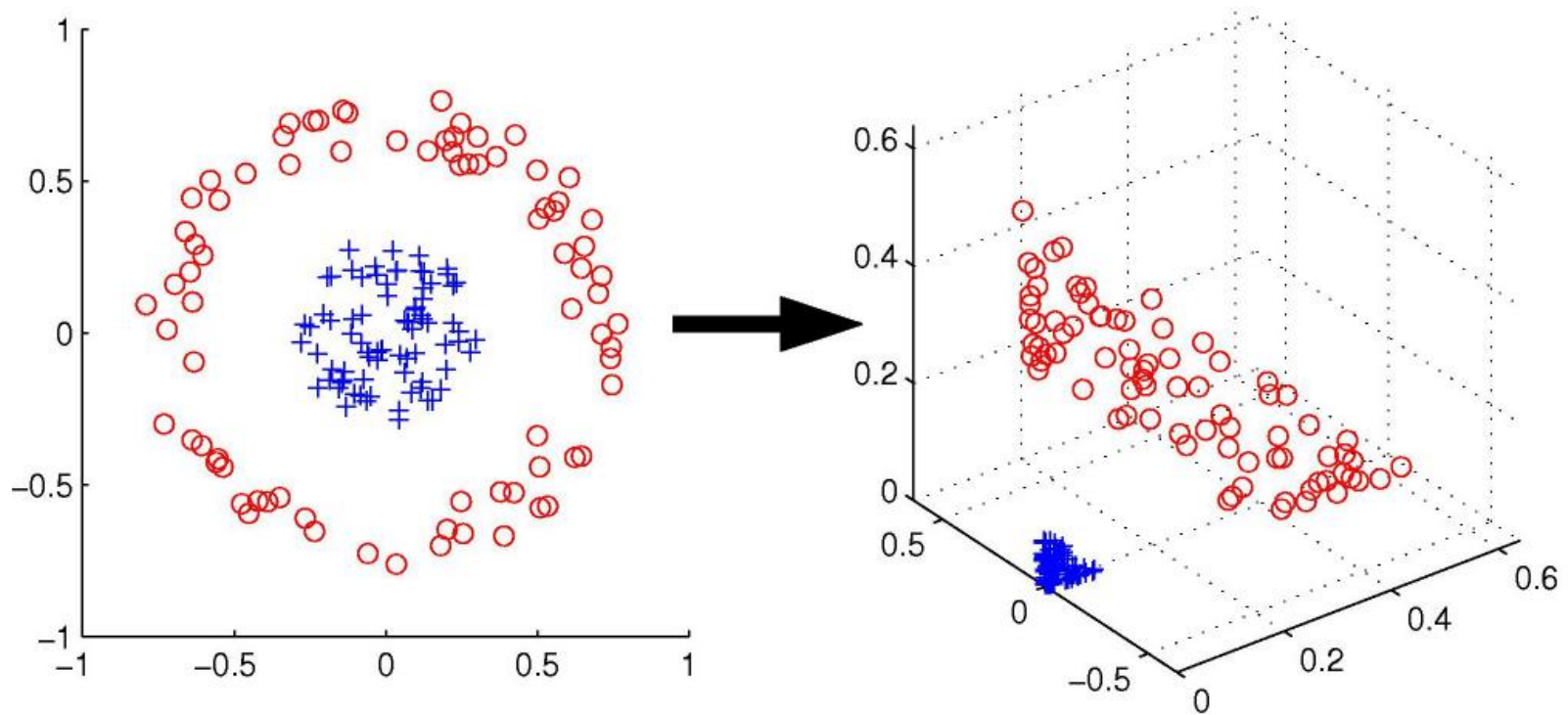
Example: SVM with Polynomial of Degree 2

Kernel: $K(\vec{x}_i, \vec{x}_j) = [\vec{x}_i \cdot \vec{x}_j + 1]^2$

plot by Bell SVM applet



Non-Linear Kernel



$$\begin{aligned}\phi : \quad \mathbb{R}^2 &\longrightarrow \mathbb{R}^3 \\ (x_1, x_2) &\longmapsto (z_1, z_2, z_3) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)\end{aligned}$$

SVM with a polynomial Kernel visualization

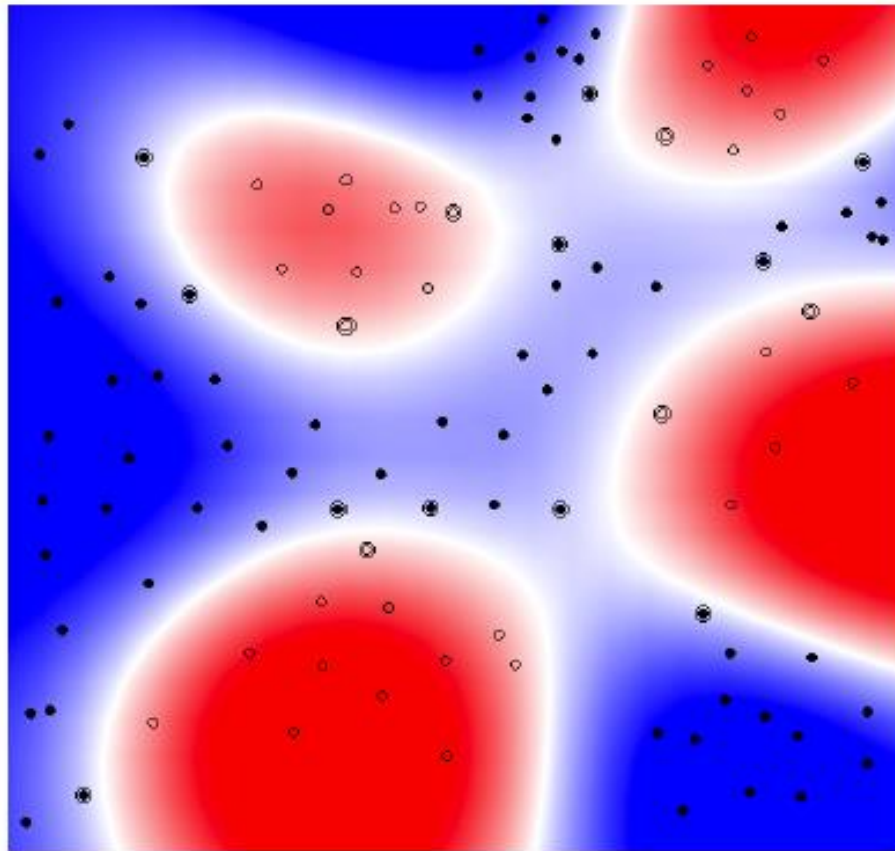
Created by:
Udi Aharoni

Nonlinear Kernel (II)

Example: SVM with RBF-Kernel

Kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-|\mathbf{x}_i - \mathbf{x}_j|^2 / \sigma^2)$

plot by Bell SVM applet



Demo

<http://cs.stanford.edu/people/karpathy/svmjs/demo/>

Kernel Tricks

- Pro
 - Introducing nonlinearity into the model
 - Computational cheap
- Con
 - Still have potential overfitting problems

SVM Performance

- Anecdotally they work very very well.
- Example: They are currently the best-known classifier on a well-studied hand-written-character recognition benchmark
- Anecdotally reliable people doing practical real-world work who claim that SVMs have saved them when their other favorite classifiers did poorly.
- There is a lot of excitement and **religious fervor** about SVMs
- Despite this, some practitioners are a little skeptical.

References

- An excellent tutorial on VC-dimension and Support Vector Machines:
C.J.C. Burges. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2(2):955-974, 1998.
- The VC/SRM/SVM Bible:
Statistical Learning Theory by Vladimir Vapnik, Wiley-Interscience; 1998
- Software: SVM-light,
<http://svmlight.joachims.org/>, free download