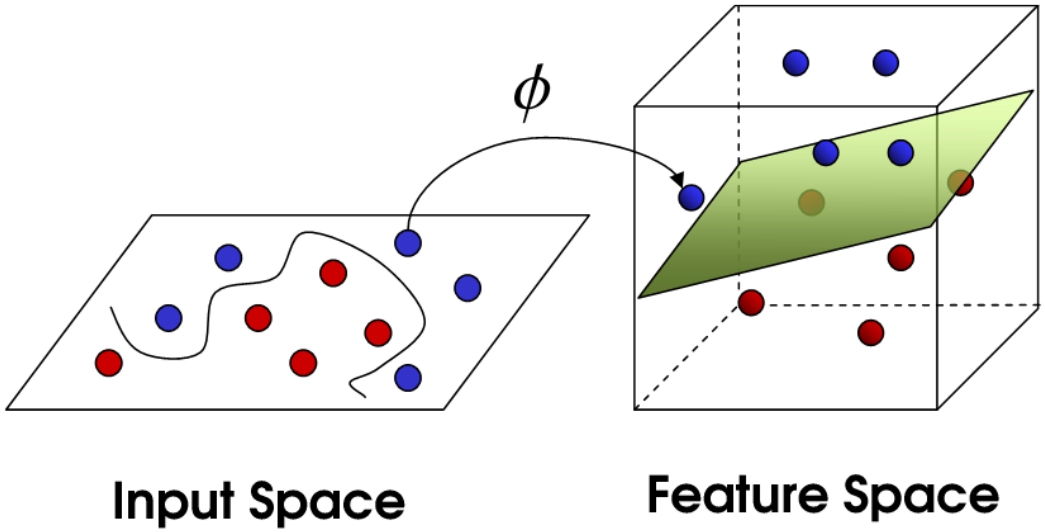
CS109 – Data Science
SVM, Performance evaluation

Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau



http://i.stack.imgur.com/1gvce.png

Announcements

- HW1 grades went out yesterday
- They are looking really good, well done everyone!

- HW2 is due this Thursday!

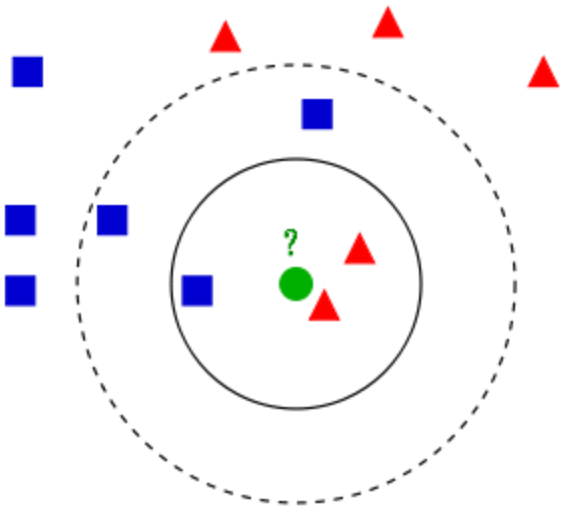
- You should submit an executed notebook
- But please without pages of test output

Recap K-NN

- Keeps all training data
- Training is fast
- Prediction is slow

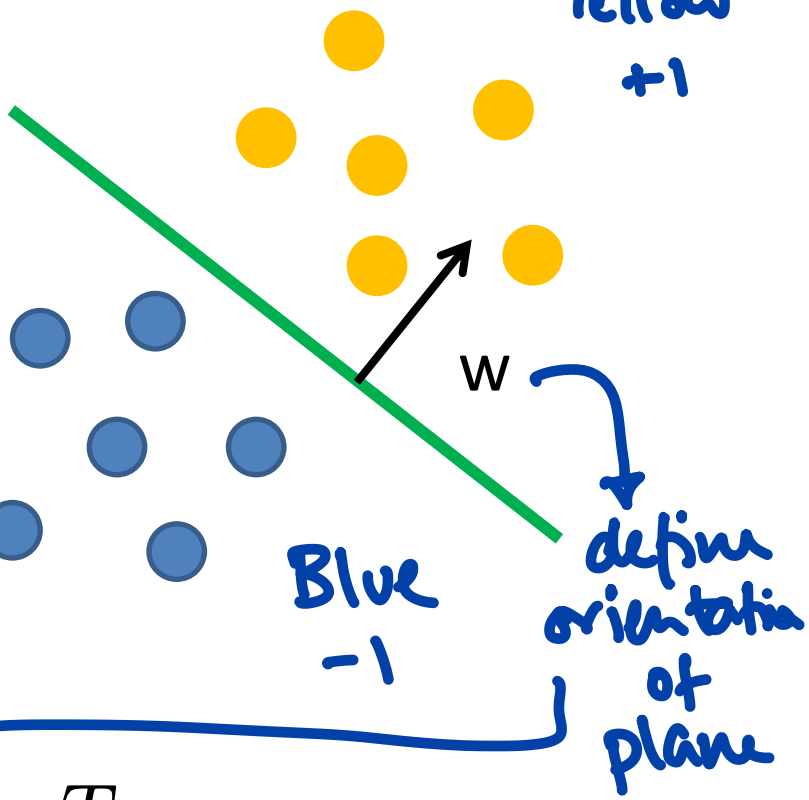
related.

Go through all data points → slow prediction



Separating Hyperplane

- x: data point
- y: label $\in \{-1, +1\}$
- w: weight vector

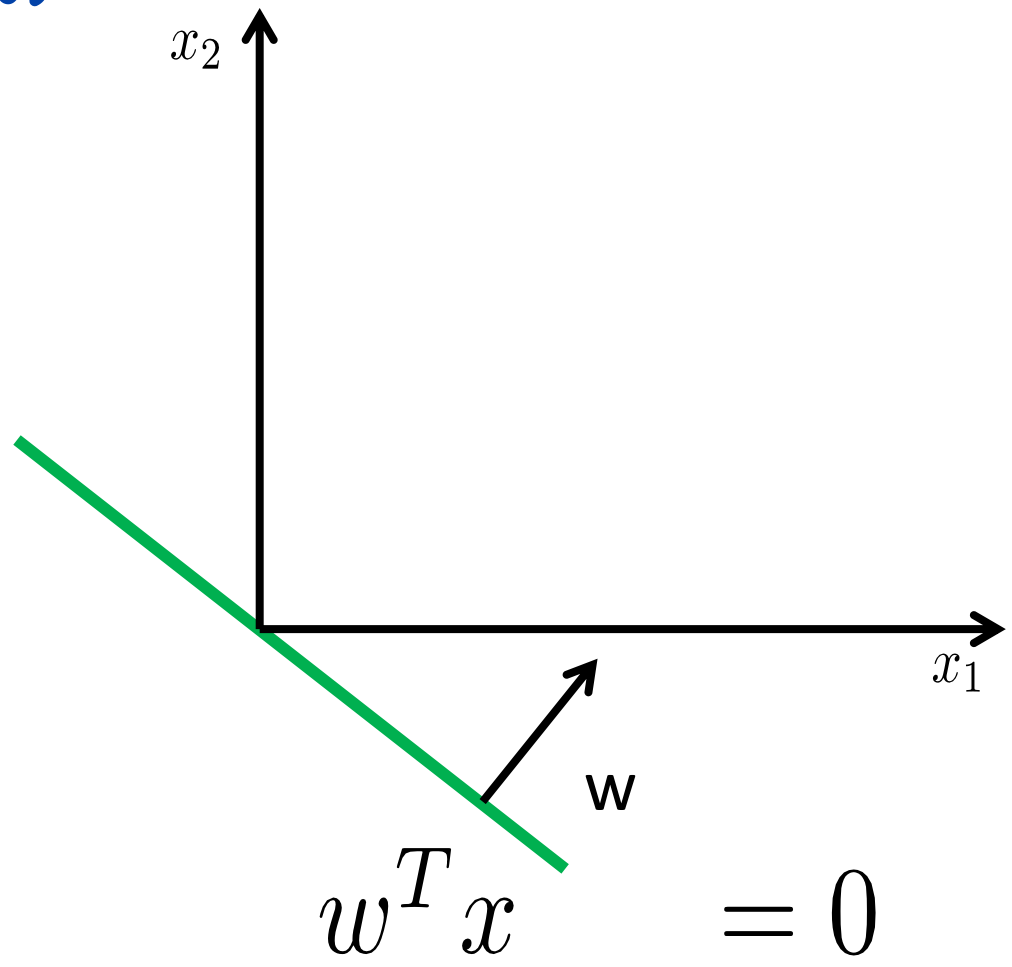


$$w^T x = 0$$

Separating Hyperplane

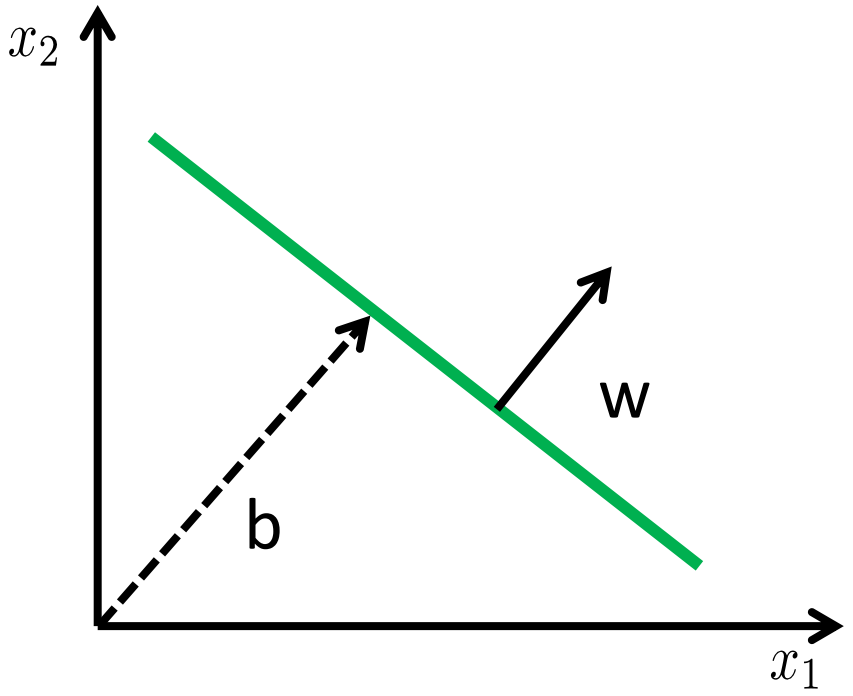
has 2 dimensions

- x: data point
- y: label $\in \{-1, +1\}$
- w: weight vector



Separating Hyperplane

- x: data point
- y: label $\in \{-1, +1\}$
- w: weight vector
- b: bias

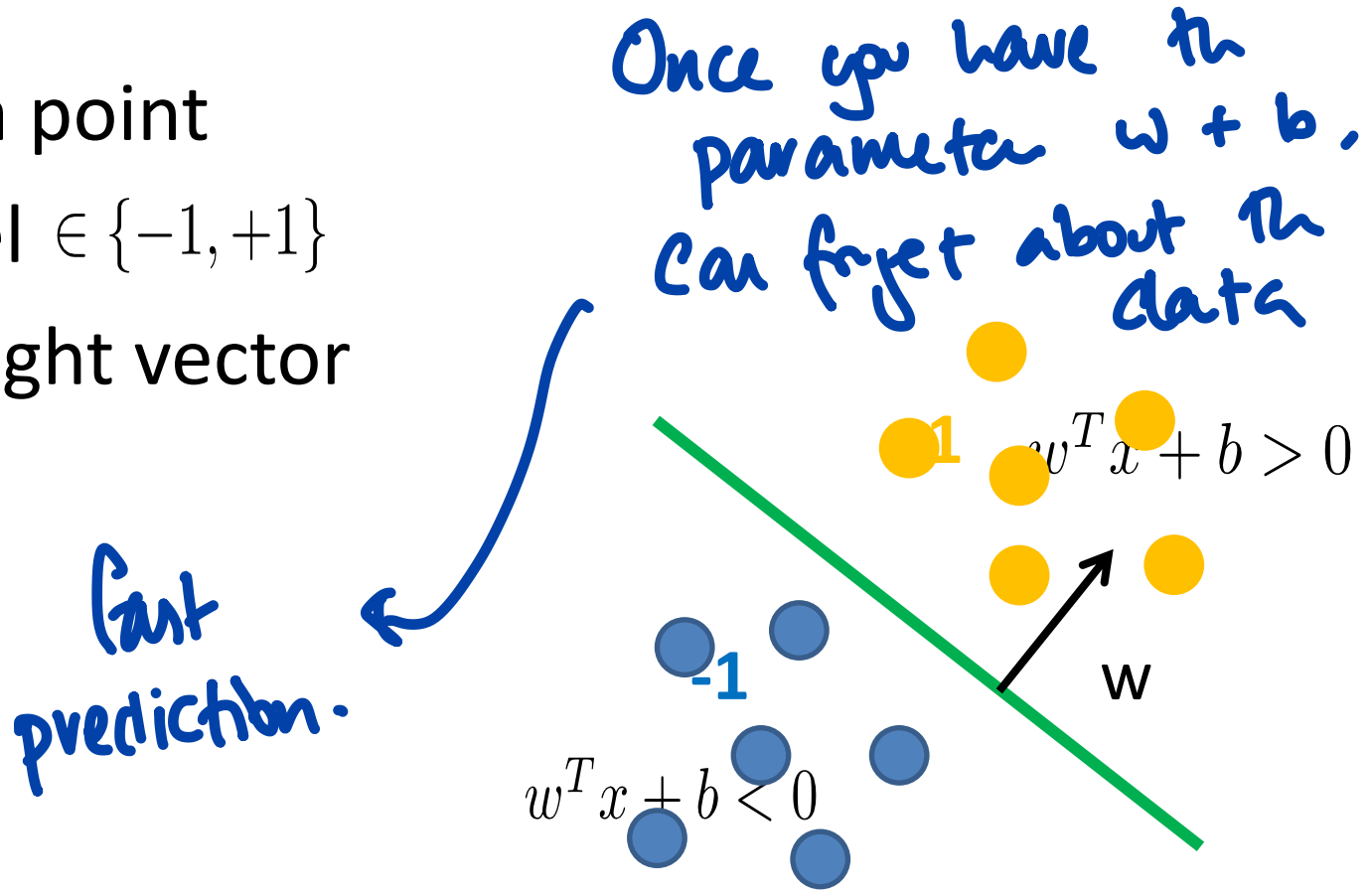


$$w^T x + b = 0$$

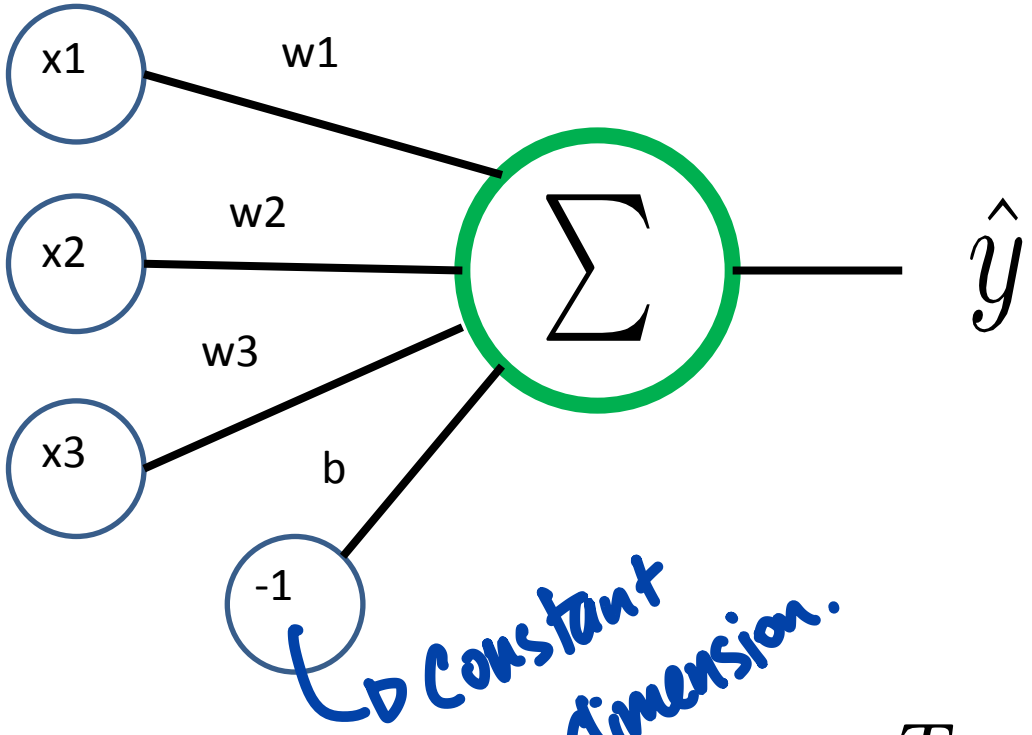
orient shift

Separating Hyperplane

- x: data point
- y: label $\in \{-1, +1\}$
- w: weight vector
- b: bias

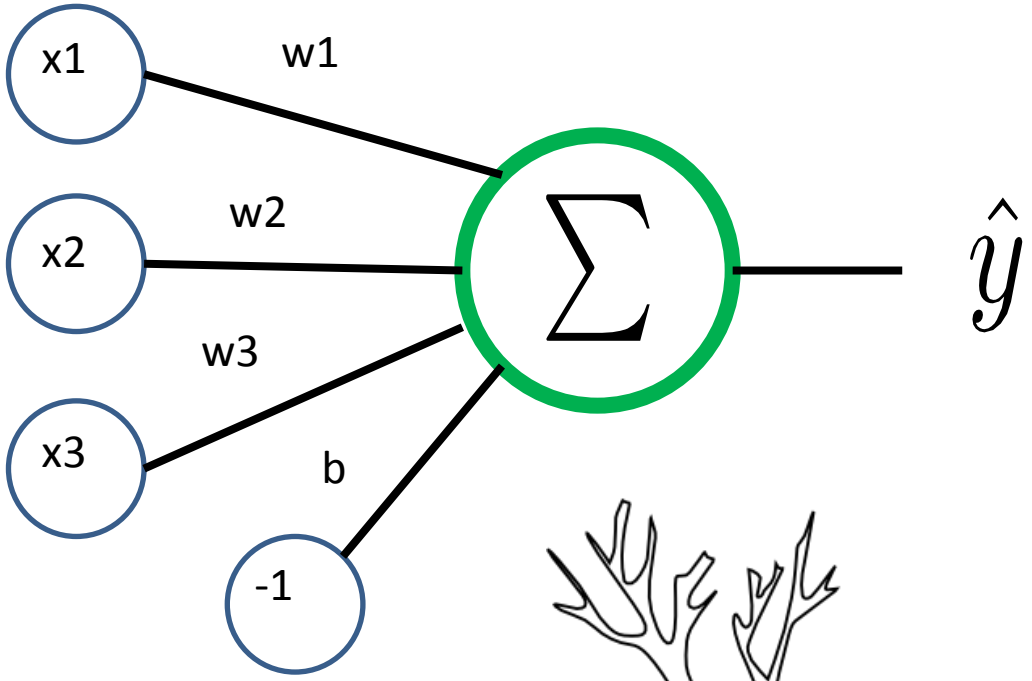


Perceptron



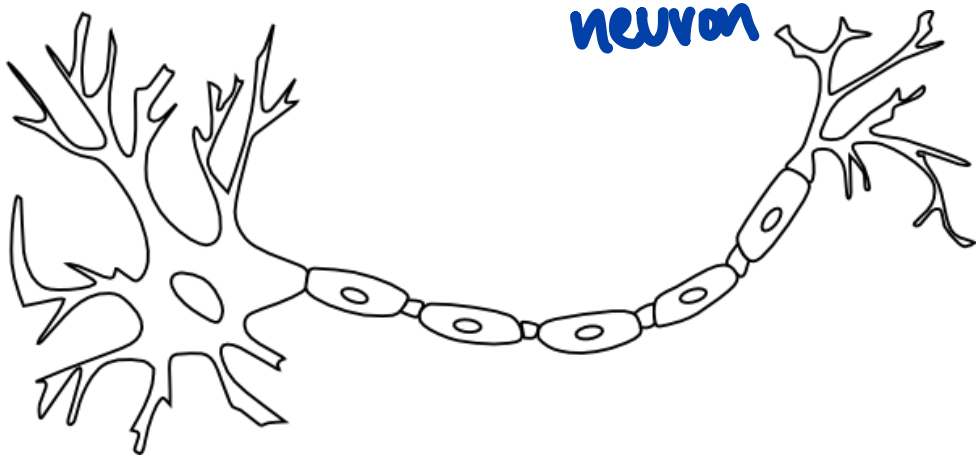
$$w^T x + b = 0$$

Perceptron



- Simple model of a neuron

Mathematical description of a neuron



Perceptron History

- invented 1957
- by Frank Rosenblatt

- the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence. (NYT 1958)

(http://en.wikipedia.org/wiki/Perceptron

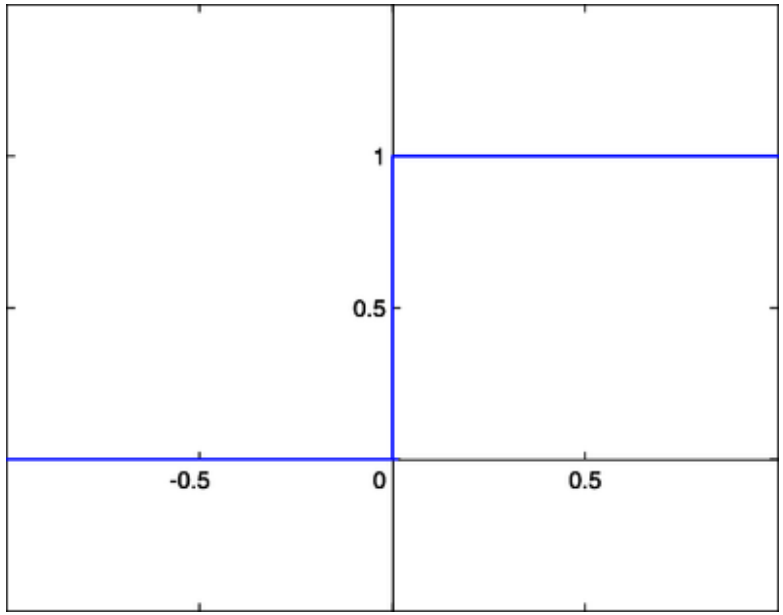
Thought they found a model of the brain

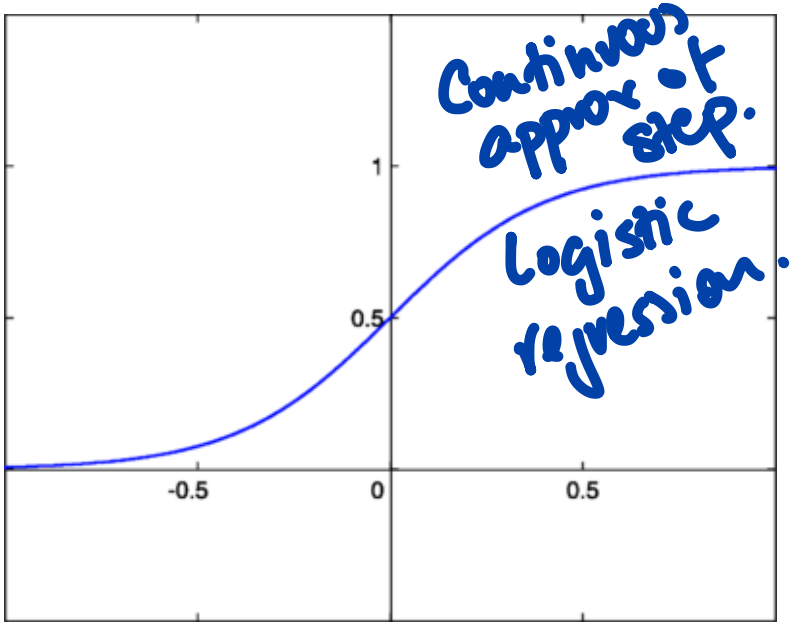
Are now the basis of deep learning.

Perceptron.mp4

https://www.youtube.com/watch?v=cNxadbrN_aI&list=PLdVOMWcqwwIlaygvb9ZteZ1r4Br6kRuBO

Side Note: Step vs Sigmoid Activation





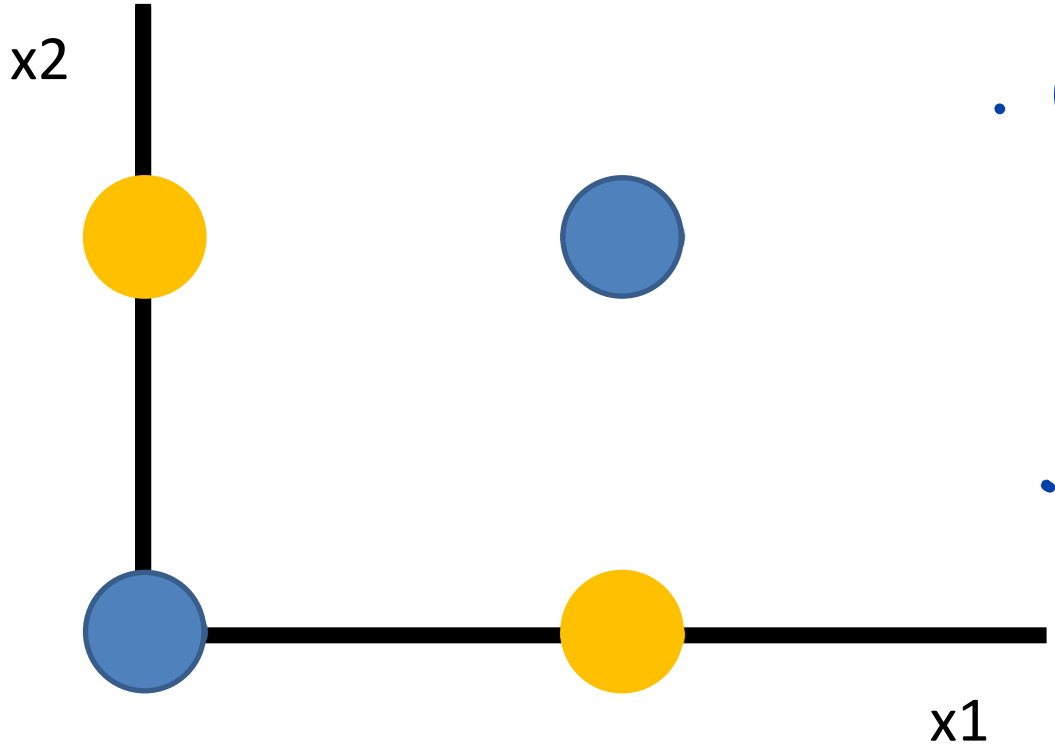
$$s(x) = \frac{1}{1 + e^{-cx}}$$

The Critics

- 1969: Minsky and Papert publish their book "Perceptrons"
  - It included weaknesses of the method.
- Very controversial book, some blame the book for causing the whole research area to stagnate.

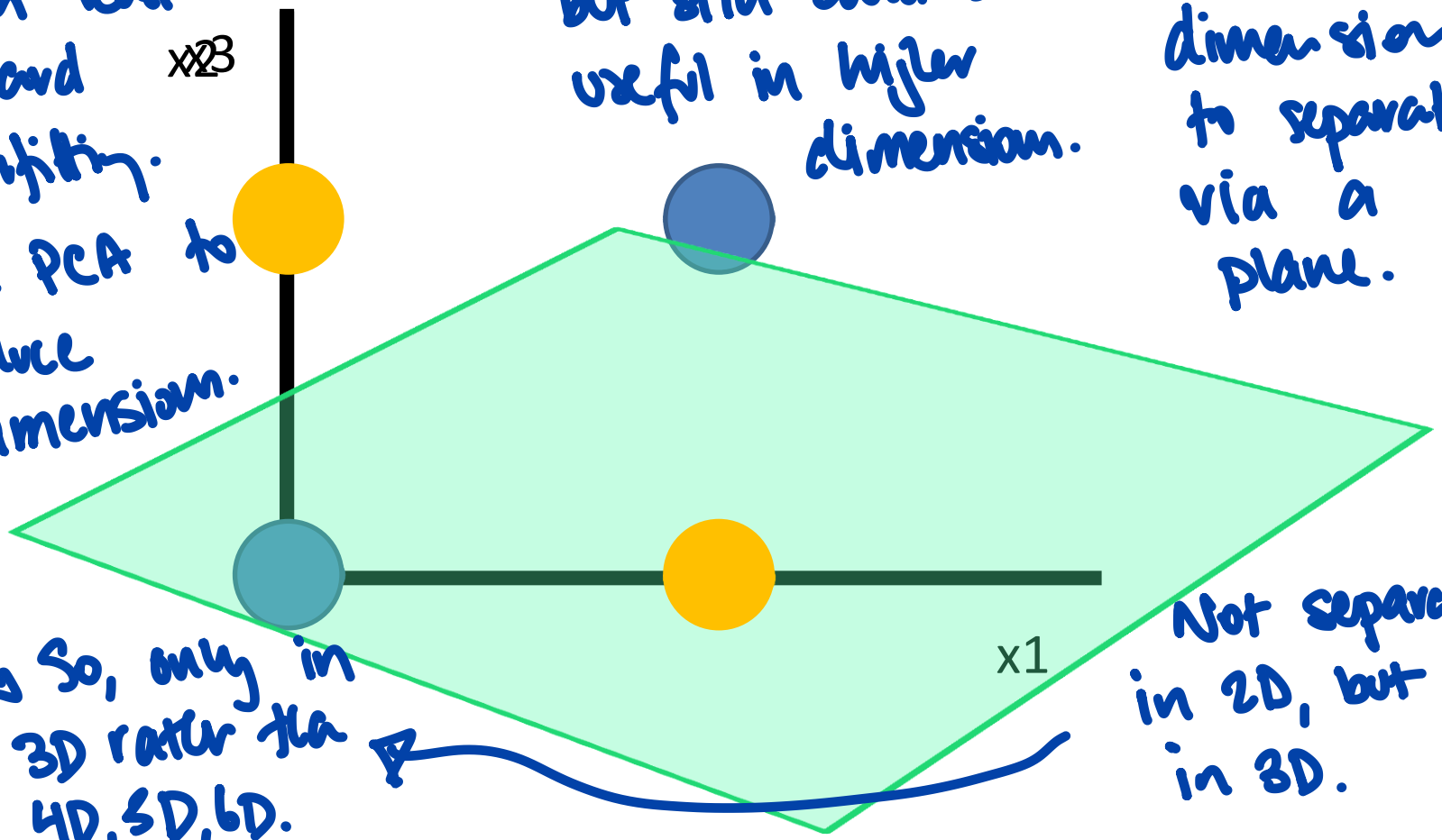
https://en.wikipedia.org/wiki/Perceptrons_(book)

The XOR Problem



- Can't separate w/ a hyperplane
- Fundamental limitation.

The XOR Problem



- Adding dimension can lead toward overfitting.
- Use PCA to reduce dimension.

So, only in 3D rather than 4D, 5D, 6D.

PCA wouldn't work, but still could be useful in higher dimension.

Add a dimension to separate via a plane.

Not separable in 2D, but in 3D.

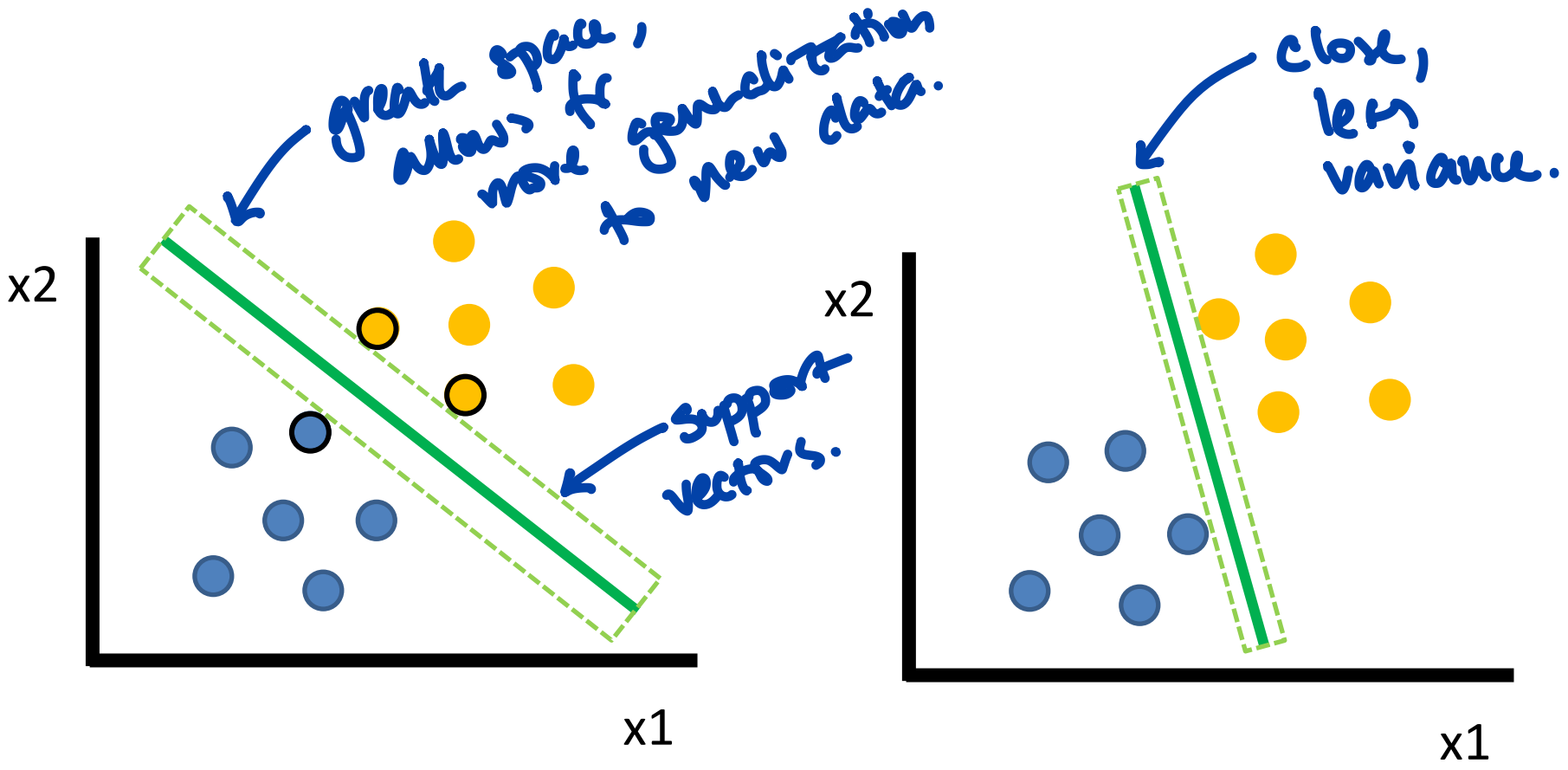
Support Vector Machine

- Widely used for all sorts of classification problems

- Some people say it is the best off the shelf classifier out there

Do not really require parameter tuning

Maximum Margin Classification



Solution depends only on the support vectors!

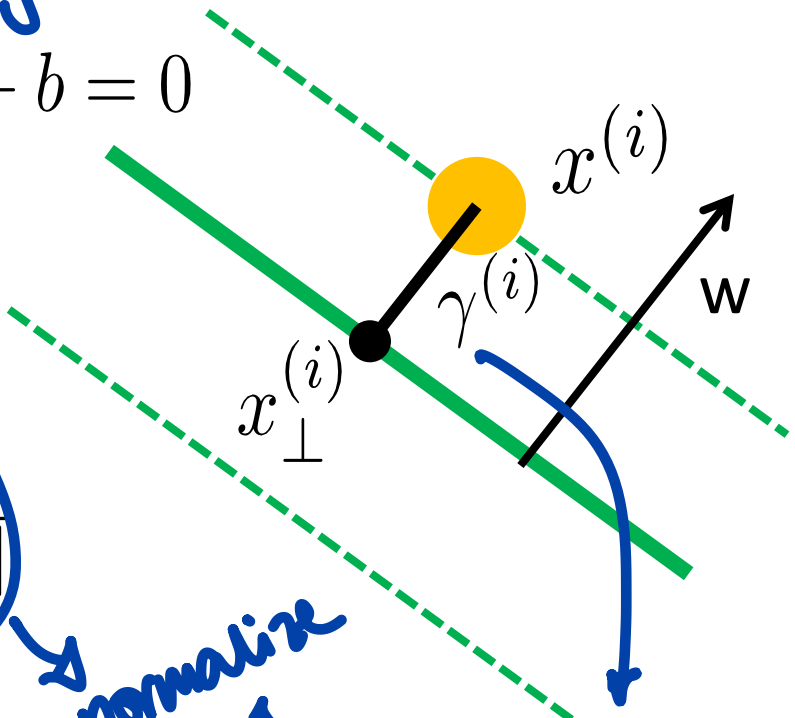
Loonly need these, no data.

Maximum Margin Classification

sci-kit learn have "C" and "gamma"

want this b/c know this

$$w^T x + b = 0$$



margin:

$$x_\perp^{(i)} = x^{(i)} - \gamma^{(i)} \cdot \frac{w}{||w||}$$

normalize to 1

$$w^T x_\perp^{(i)} + b = 0$$

gamma = width of max margin

$$\gamma^{(i)} = \left(\frac{w^T x^{(i)} + b}{||w||}\right)$$

Combo of eqns above.

Maximum Margin Classification

tells pos or neg away from hyperplane.

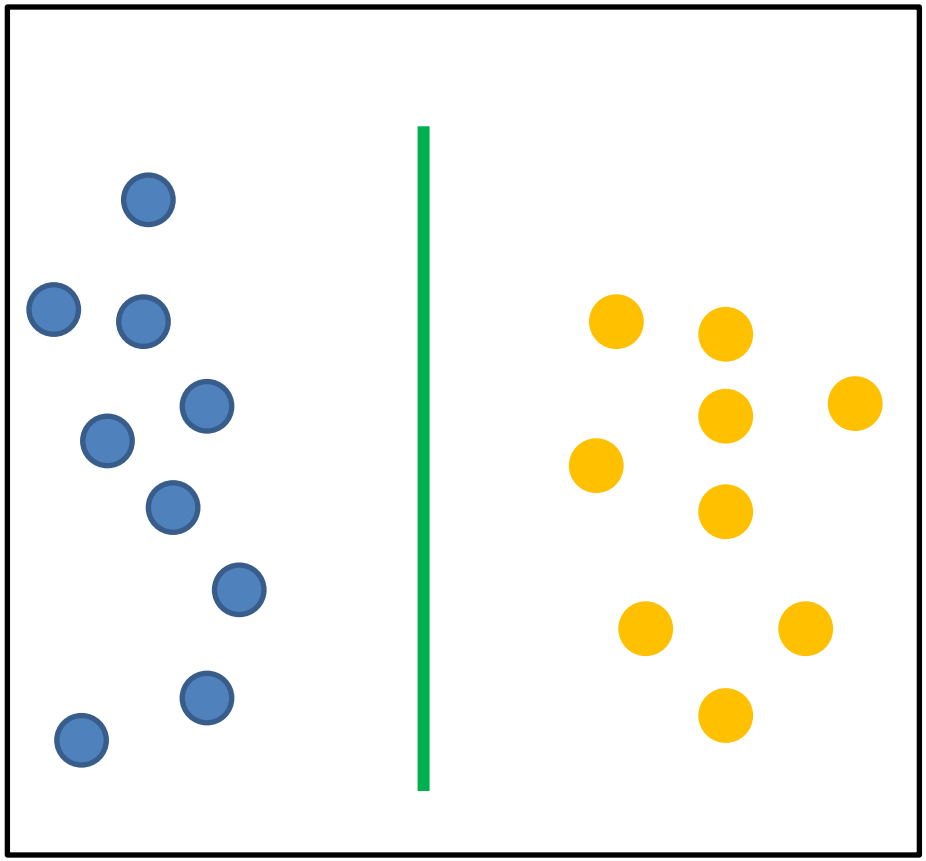
$$\gamma^{(i)} = y^{(i)}(w^T x + b)$$

Software does this; we don't write our own optimizer.

$$\begin{aligned} \max_{\gamma,w,b} \quad & \gamma \\ \text{s.t.} \quad & y^{(i)}(w^T x^{(i)} + b) \geq \gamma, \quad i = 1, \ldots, m \\ & ||w|| = 1. \end{aligned}$$

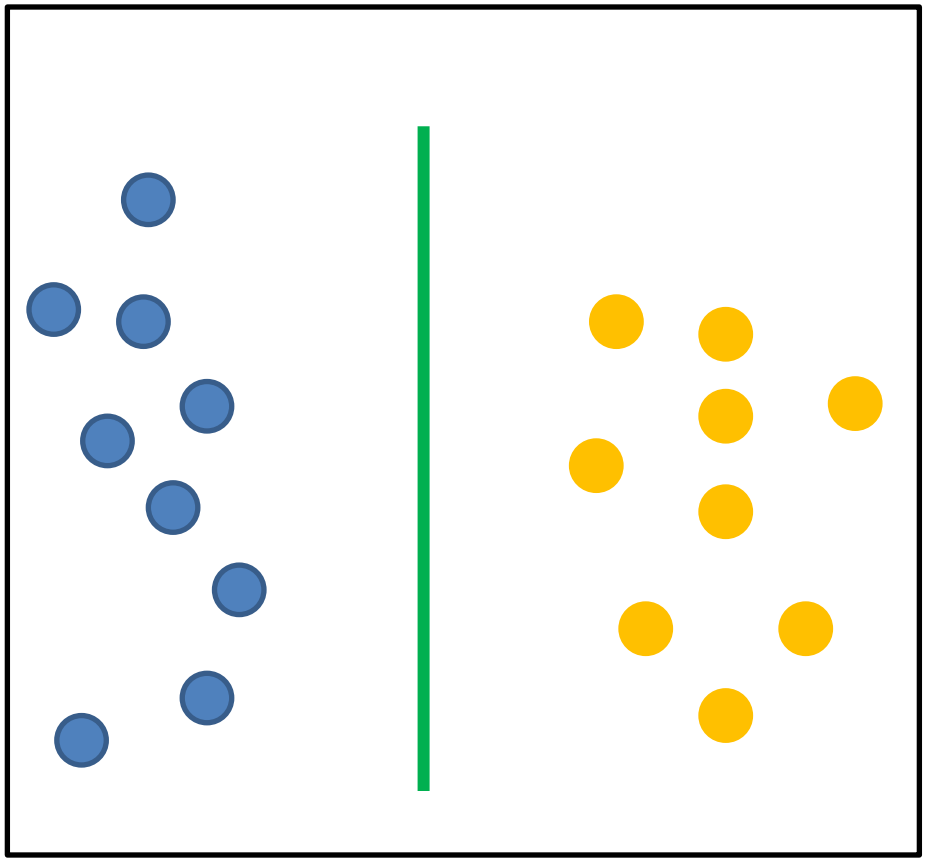
non-convex

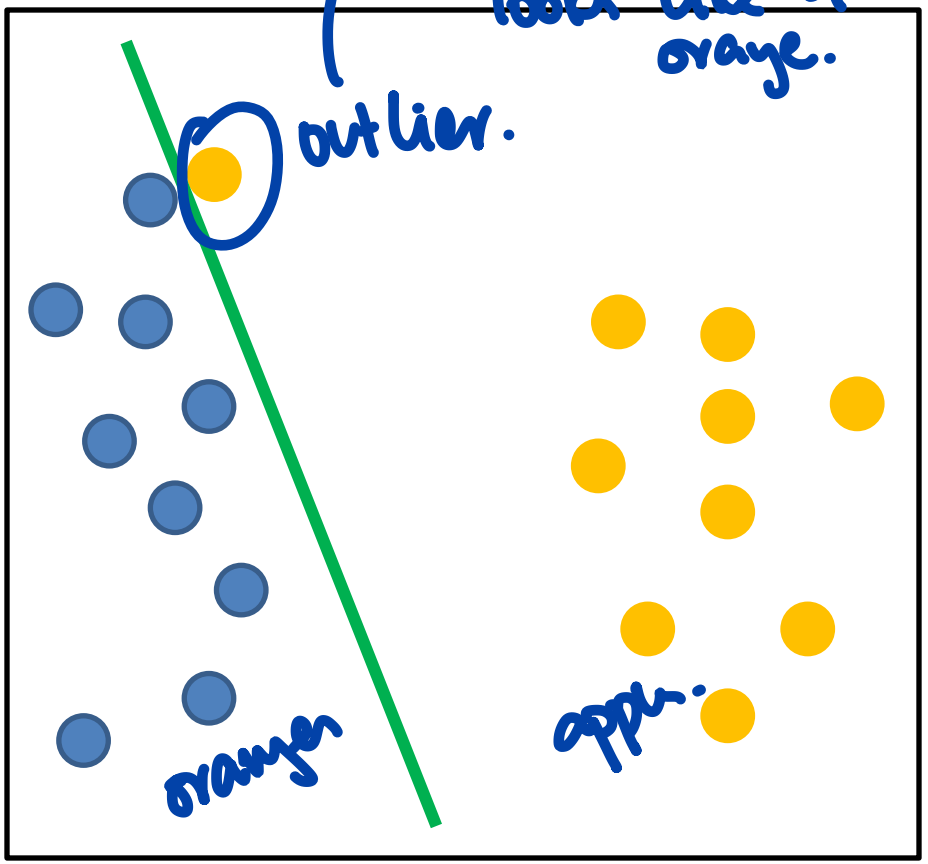
This Is Kind of Odd



- Which data points do we care the most about? Rare
- What would those samples look like?

Two Very Similar Problems





What about outliers?

Don't want a perfect classification on training data.

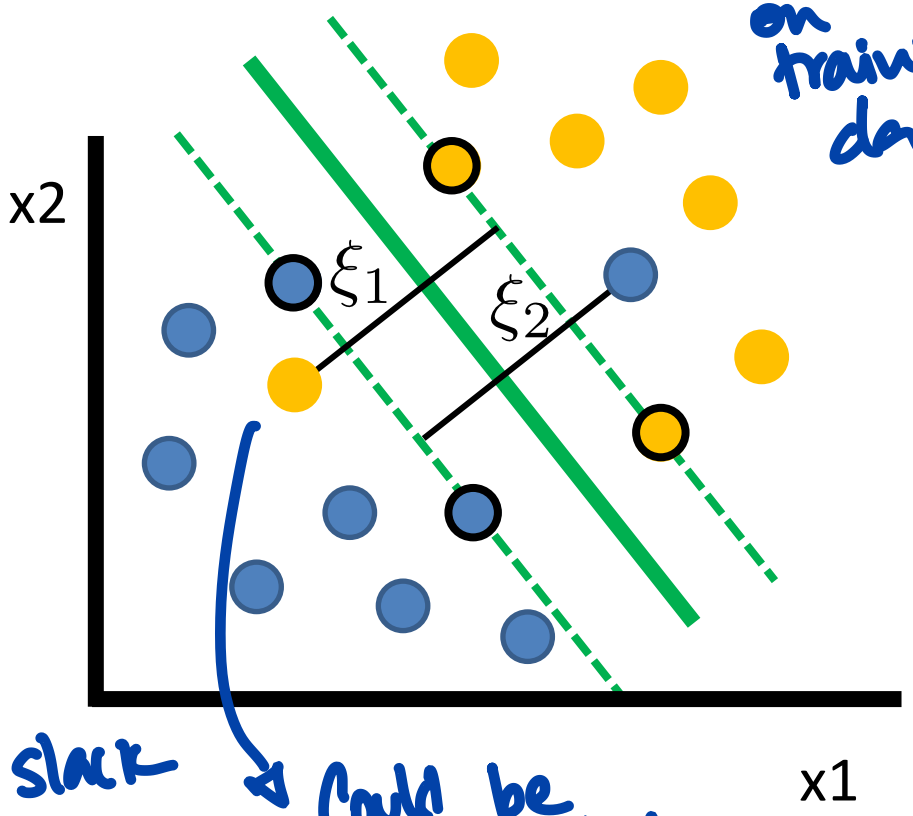
ξ_i: slack variables

$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C\sum_i \xi_i$

subject to:

$y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i$

$(i = 1, \ldots, n)$

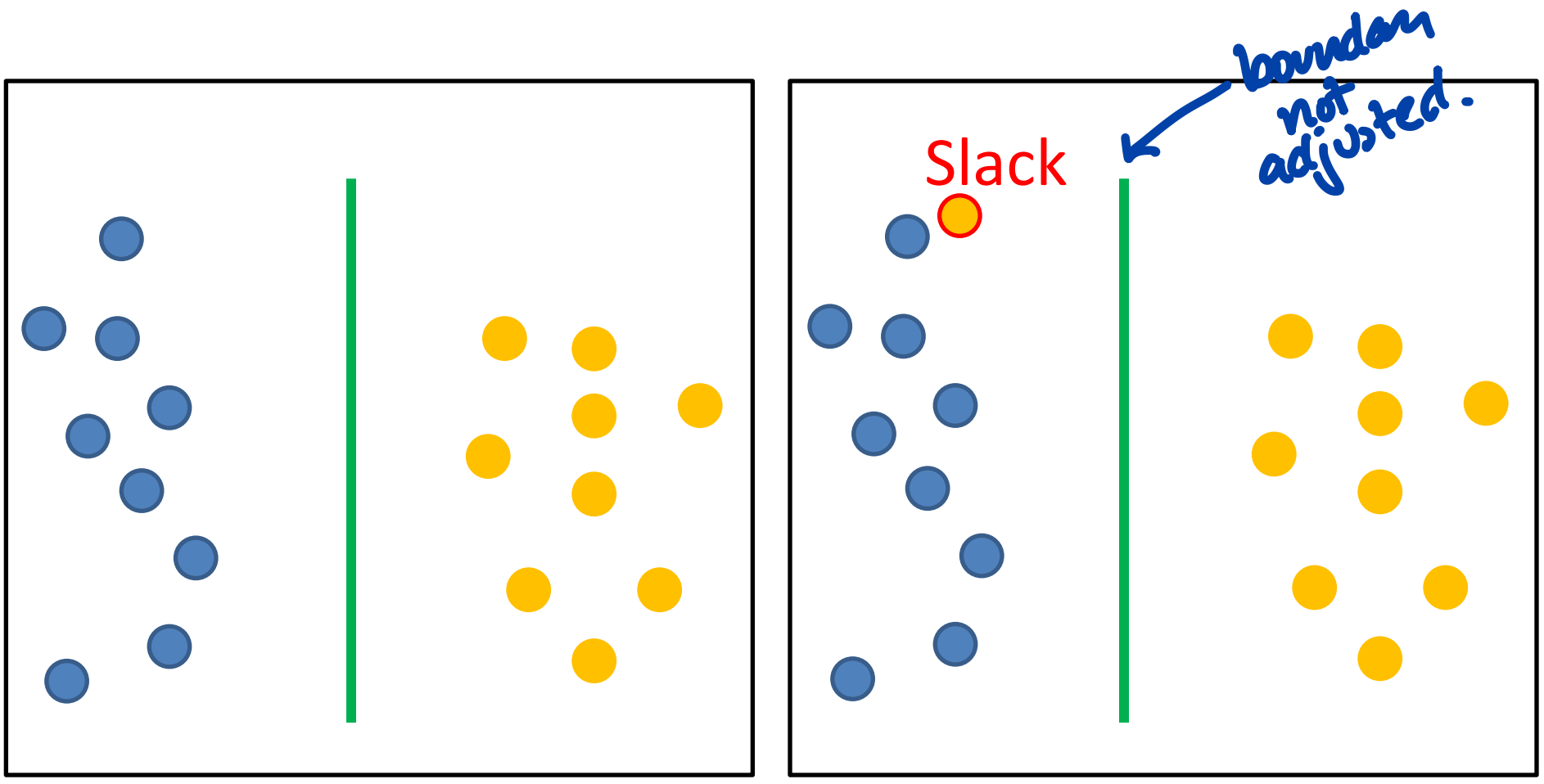


Could be mislabeled.

large C → not much slack

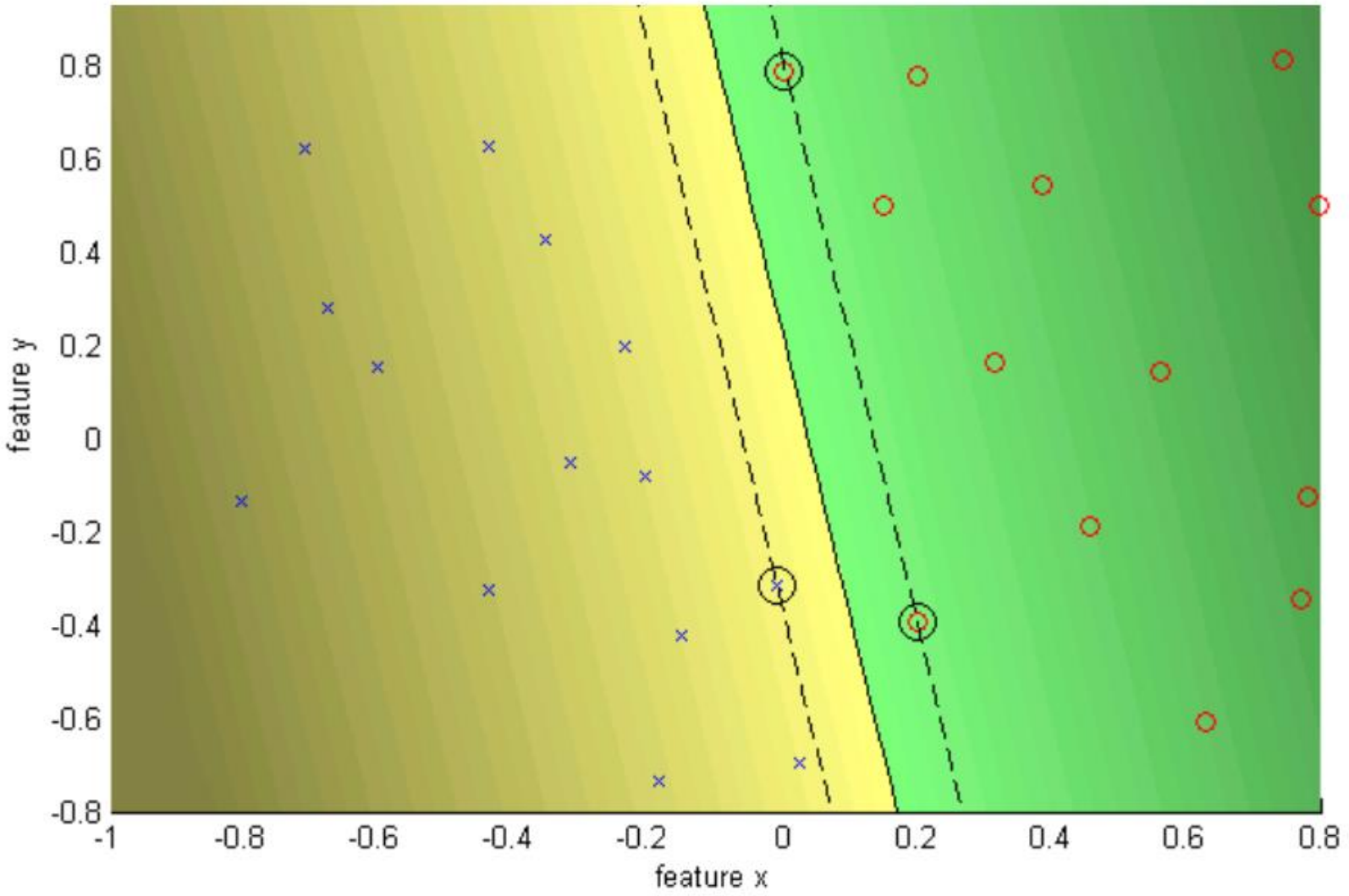
Small C → more room/slack

Two Very Similar Problems



Hard Margin (C = Infinity)

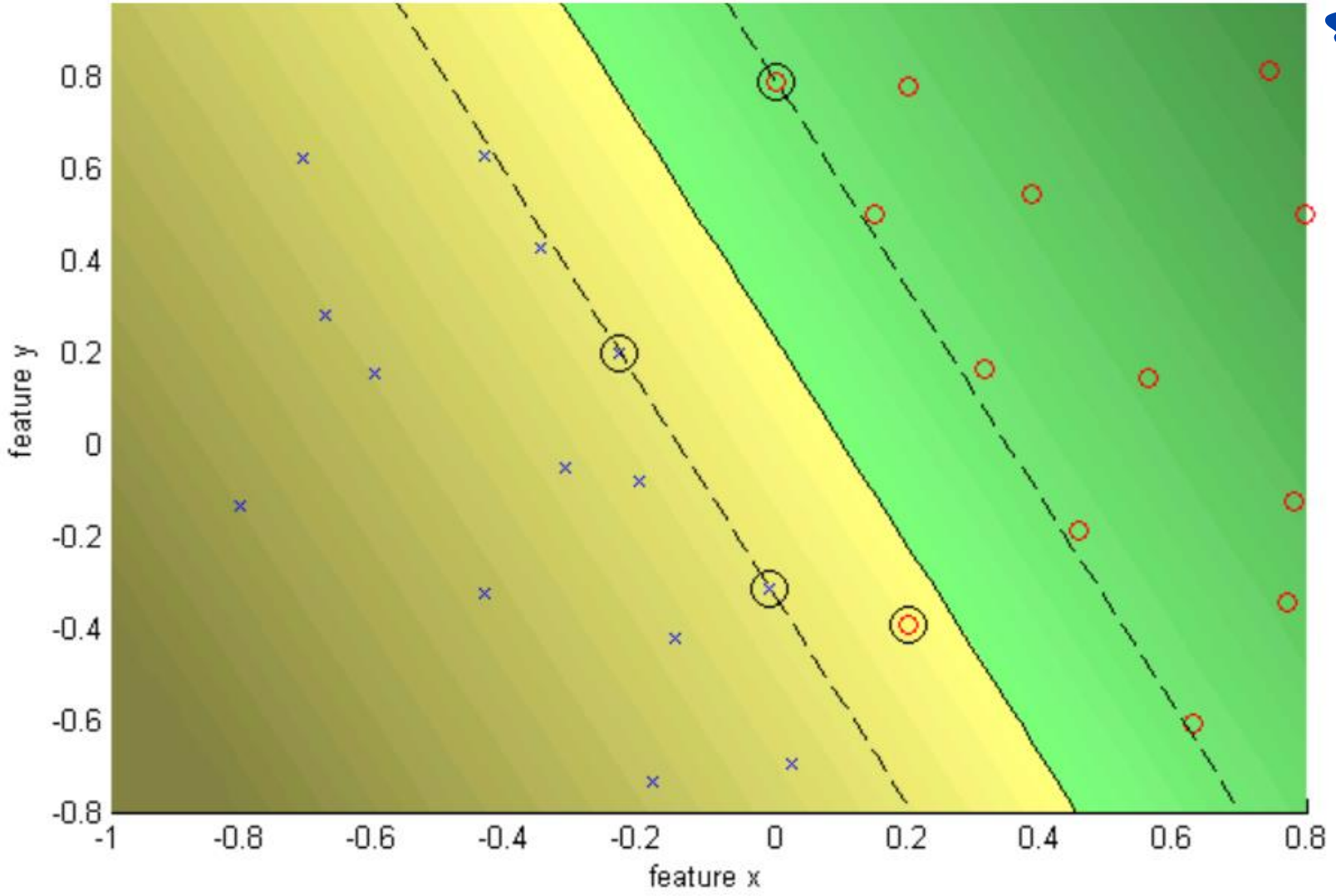
no mistakes; margin small.



http://www.robots.ox.ac.uk/~az/lectures/ml/lect2.pdf

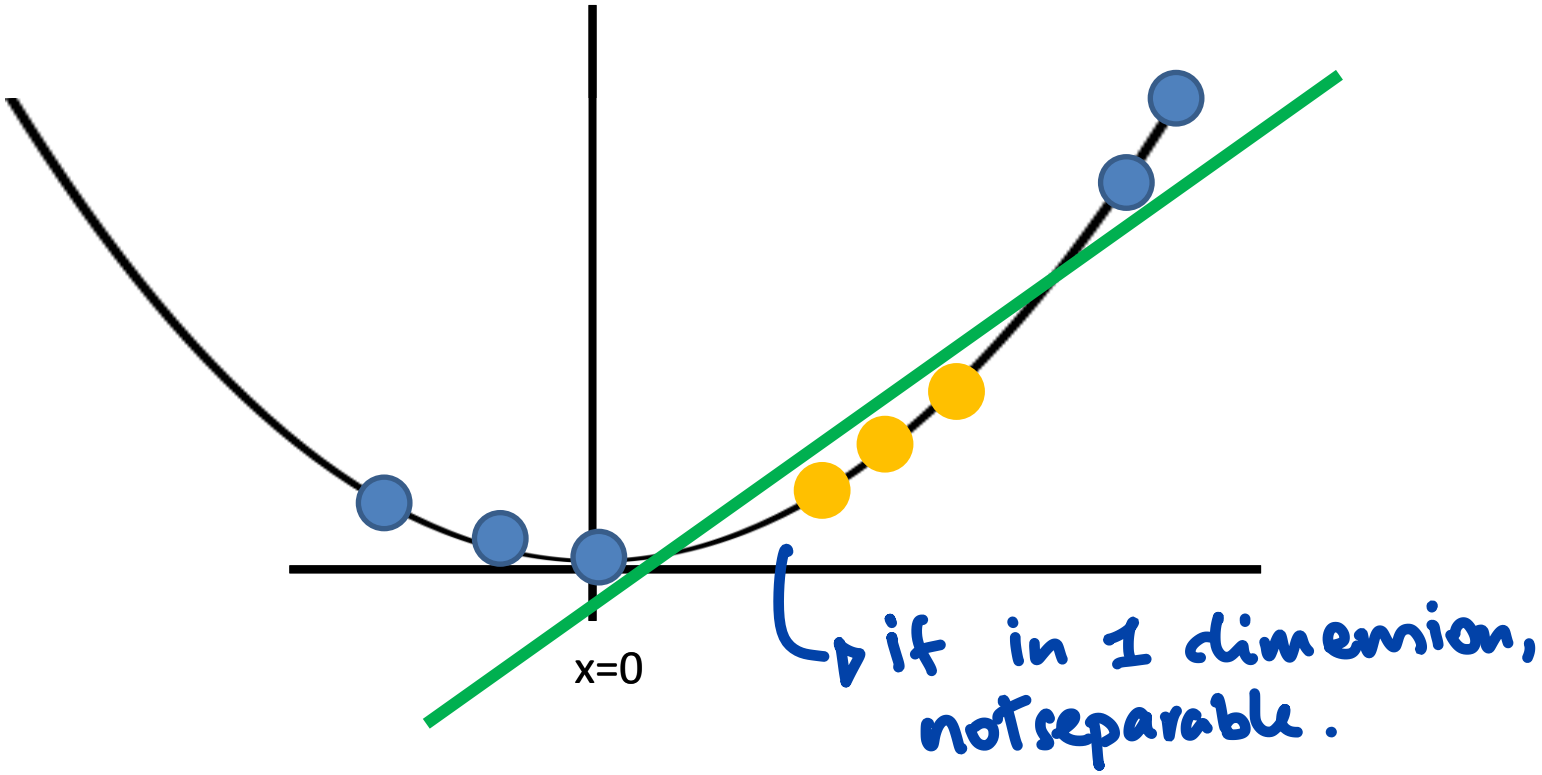
Soft Margin (C = 10)

Not perfect but great slack.



http://www.robots.ox.ac.uk/~az/lectures/ml/lect2.pdf

XOR problem revised



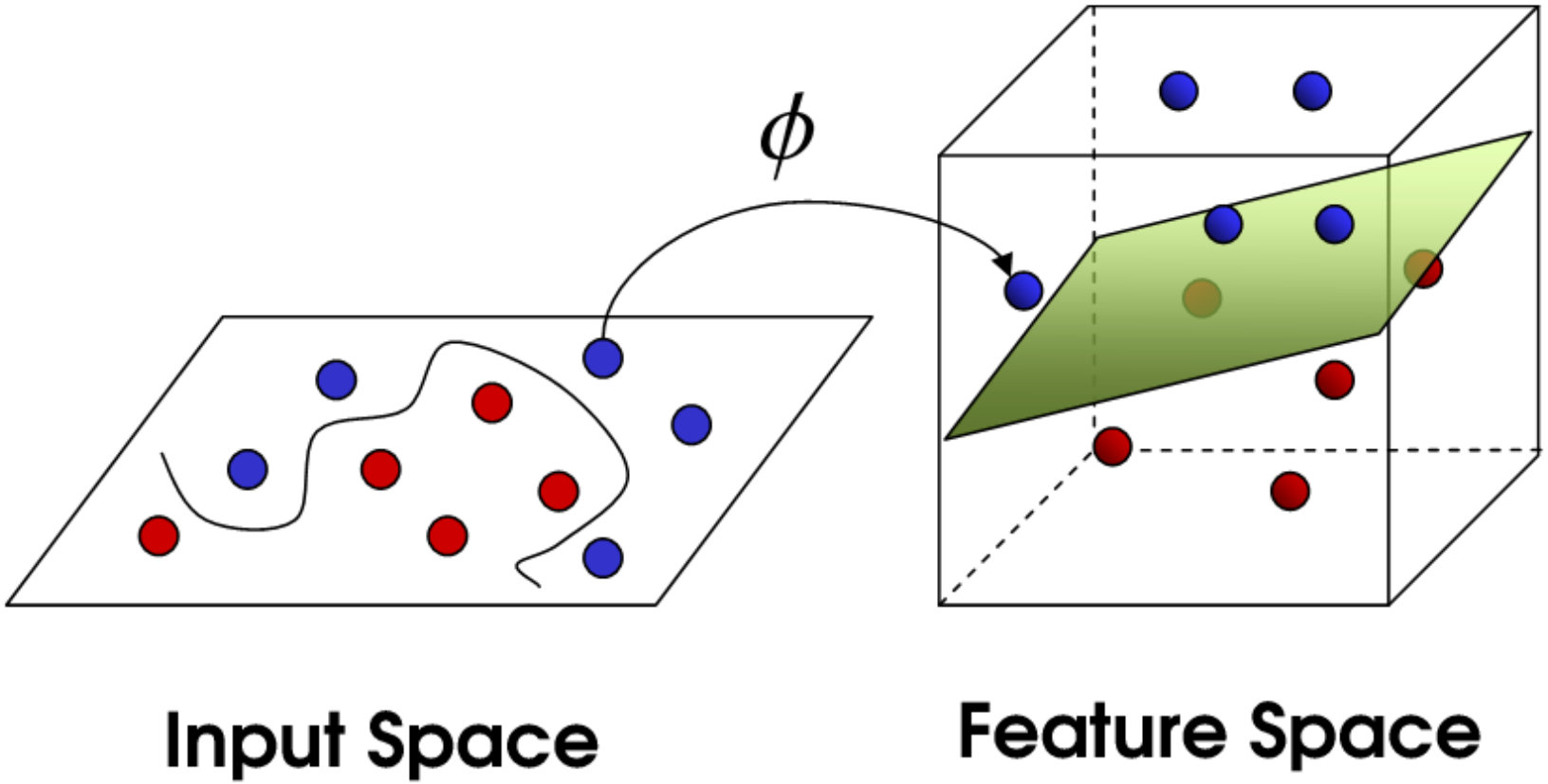
Did we add information to make the problem seperable?

No. Just dimension 2.

Cross-validation to determine the # of dimension.

Non-Linear Decision Boundary

Adding dimension make the hyperplane non-linear.



SVM with a polynomial Kernel visualization

Created by:
Udi Aharoni

Quadratic Kernel

quadratic expansion of 2D point

$$x = (x_1, x_2)$$

$$\Phi(x) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

$$\Phi(x) \cdot \Phi(z) = 1 + 2\sum_{i=1}^{d} x_i z_i$$

need to dot product; high computational cost.

$$+ \sum_{i=1}^{d} x_i^2 z_i^2 + 2\sum_{i=1}^{d}\sum_{j=i+1}^{d} x_i x_j z_i z_j$$

$$= (1 + x \cdot z)^2$$

kernel trick → Just use this; no Φ

Kernel Functions

$$K(x, z) = \Phi(x) \cdot \Phi(z)$$

- Polynomial:

$$K(x, z) = (1 + x \cdot z)^s$$

can do any dimension 2+

I need to tune; might be "d" in sklearn

- Radial basis function (RBF):

(very popular)

$$K(x, z) = \exp(-\gamma(x - z)^2)$$

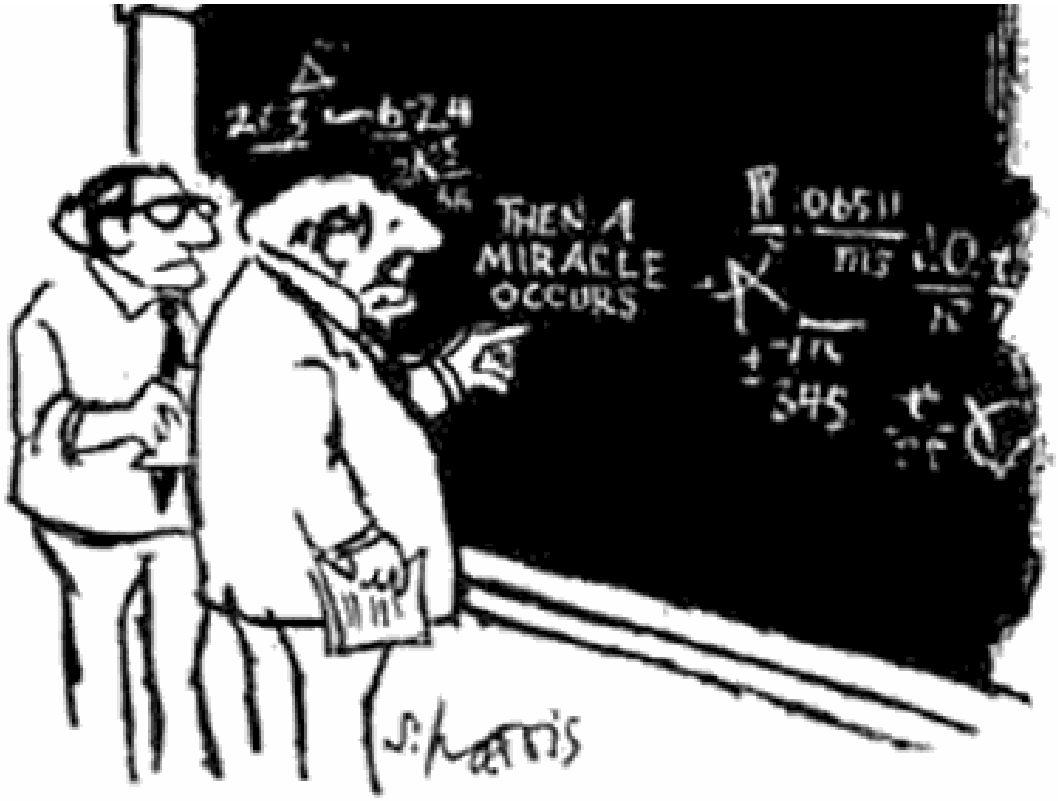
Works b/c don't need the actual point, just the dot product

So what is the excitement?

$\max_\alpha \sum$

s.t. α_i

$\sum$



$(i)^T x^{(j)}$

Andrew Ng for this explanation

arg r

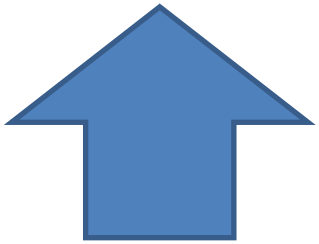
s.t. y

So what is the excitement?

$$\max_\alpha \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j x^{(i)^T} x^{(j)}$$

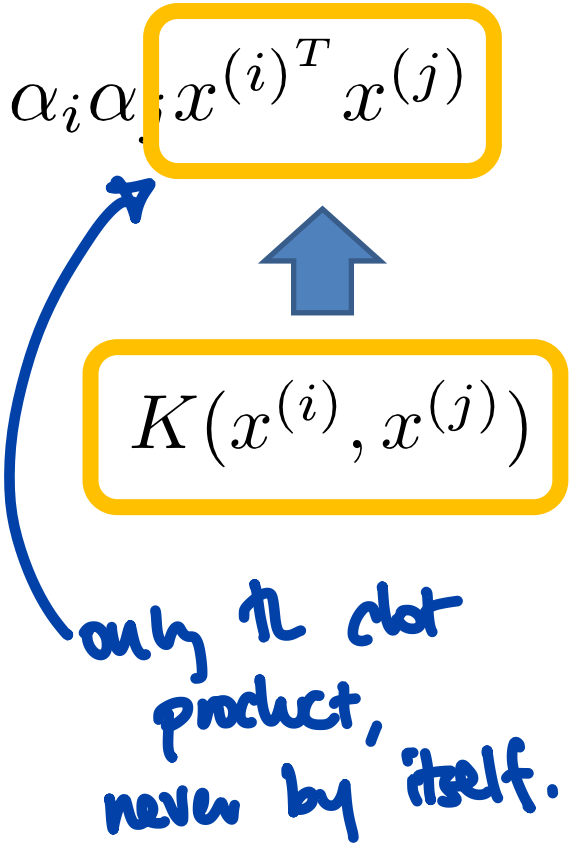
$$\text{s.t. } \alpha_i \geq 0, \ i = 1, \ldots, m$$

$$\sum_{i=1}^m \alpha_i y^{(i)} = 0$$



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

$$\text{s.t. } y^{(i)}(w^T x^{(i)} + b) \geq 1$$



Prediction

$$w^T x + b = \sum_{i=1}^{m} \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b.$$

- Again we can use the kernel trick!
- Prediction speed depends on number of support vectors

The Miracle Explained

- Andrew Ng does this really well

- http://cs229.stanford.edu/notes/cs229-notes3.pdf
- Course is also on Youtube, ItunesU, etc.

Kernel Trick for SVMs

- Arbitrary many dimensions
- Little computational cost
- Maximal margin helps with curse of dimensionality

PCA → SVM.

Face Recognition

Face Recognition

- Load image data
- Put your test data aside
- Extract Eigenfaces (PCA on images).
- Train SVM
- Evaluate performance

Cross validation & Do these steps for all

- Red are cross validation steps

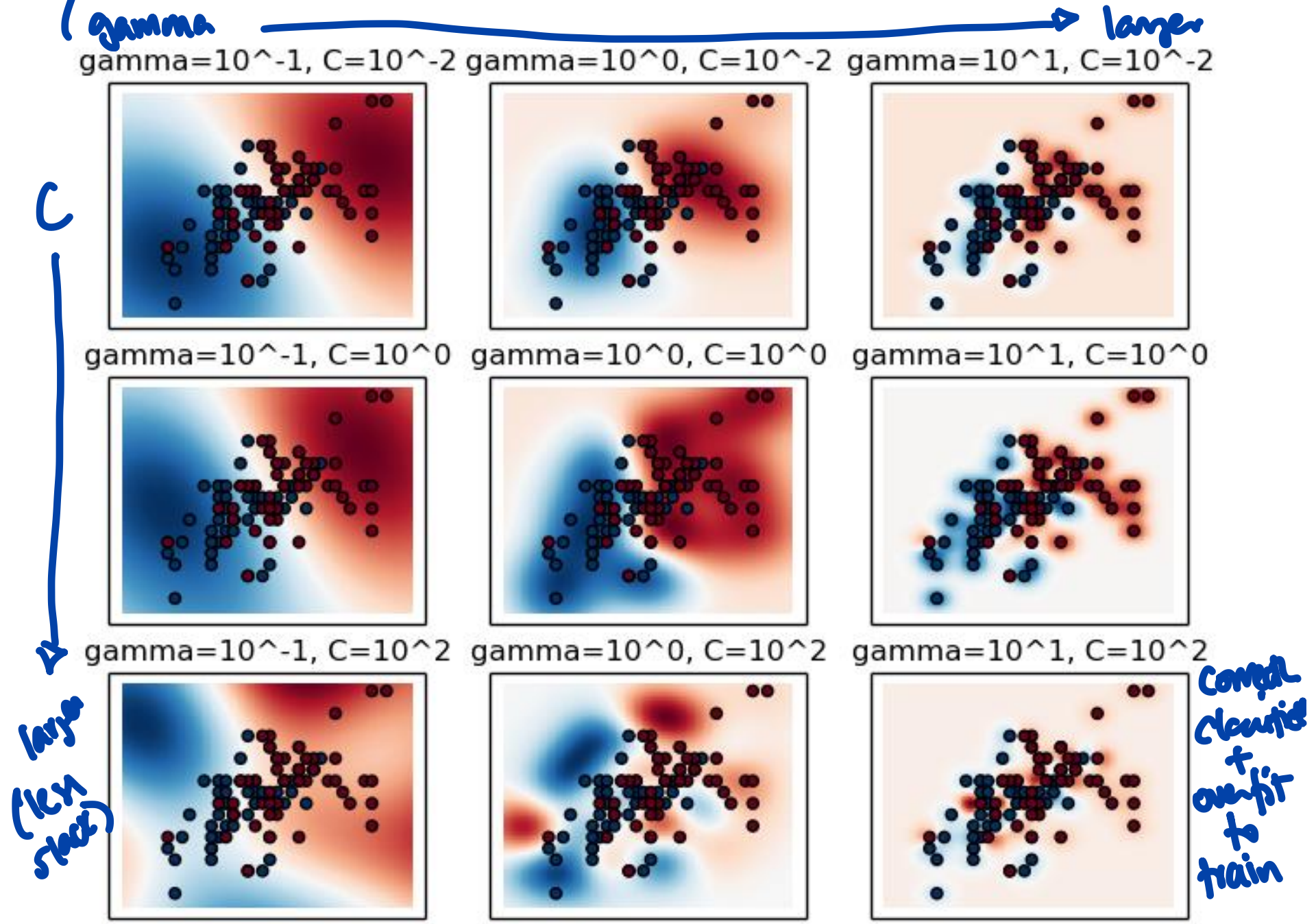
http://scikit-learn.org/stable/auto_examples/applications/face_recognition.html#example-applications-face-recognition-py

SVM_sign_language.mp4

Jhon Gonzalez

https://www.youtube.com/watch?v=cxHMgl2_5zg

→ tune parameters in exponential steps.



http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

Tips and Tricks

- SVMs are not scale invariant (milk vs. nanometers).
- Check if your library normalizes by default
  - likely not.
- Normalize your data
  - mean: 0 , std: 1
  - map to [0,1] or [-1,1]
- Normalize test set in same way!

Tips and Tricks

- RBF kernel is a good default
- For parameters try exponential sequences
- Read:

  Chih-Wei Hsu et al., "A Practical Guide to Support Vector Classification", Bioinformatics (2010)

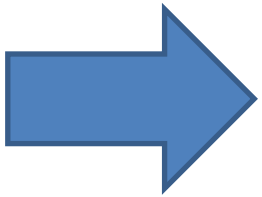
SVM vs KNN

- What are the main key differences?

Parameter Tuning

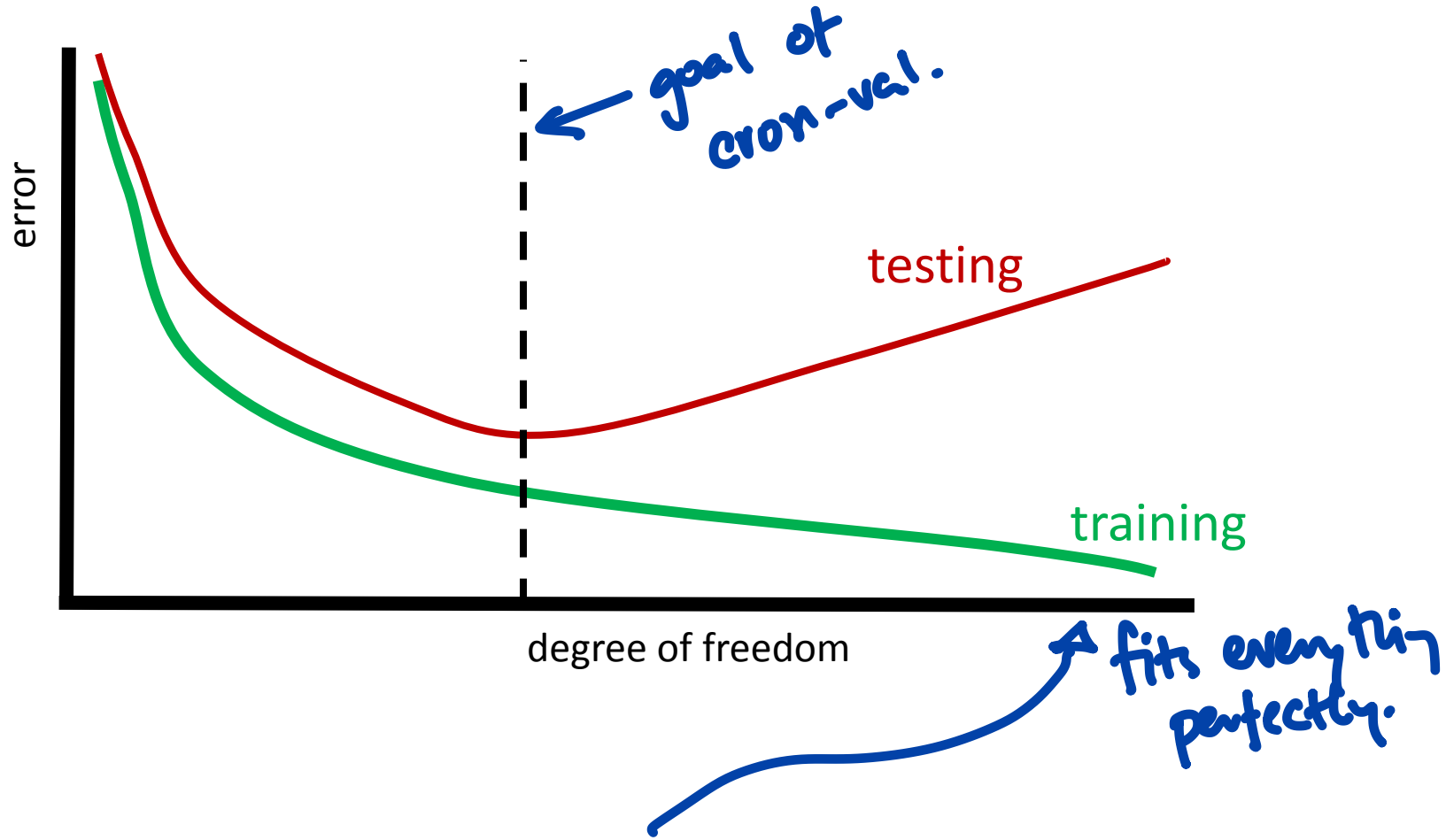
- Given a classification task

- Which kernel ?
- Which kernel parameter values?
- Which value for C?



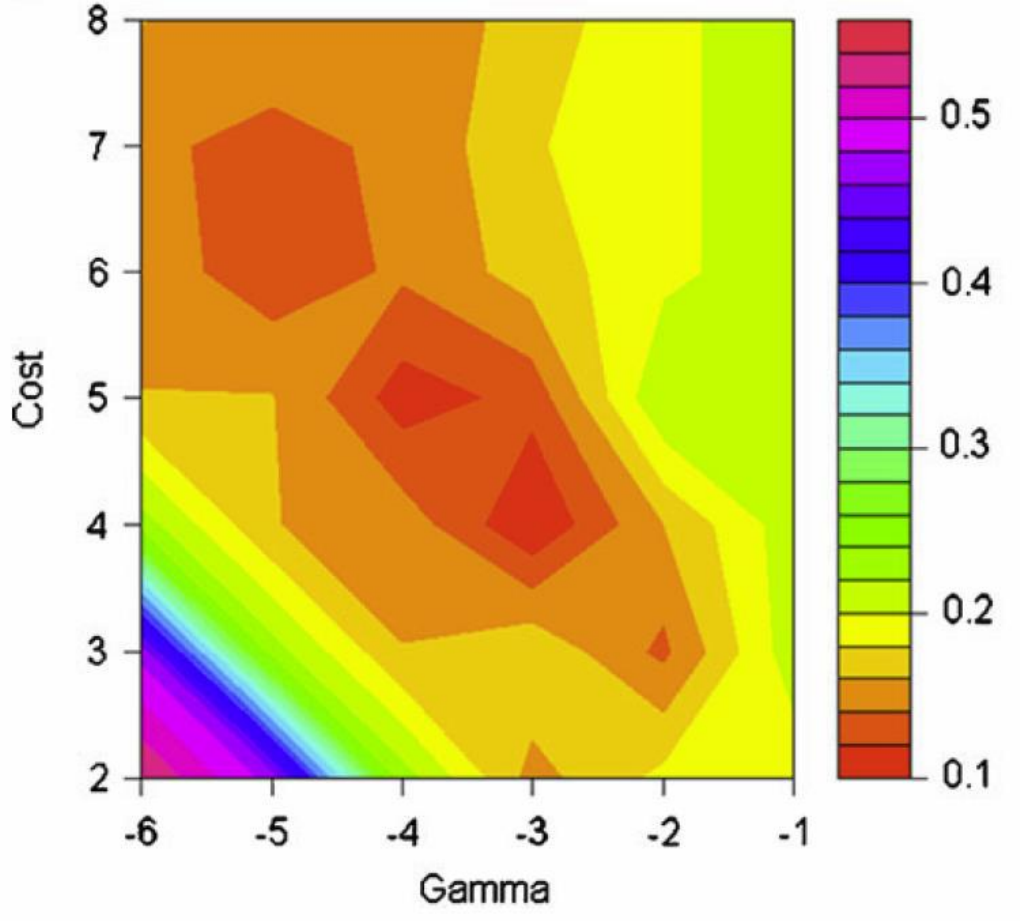
Try different combinations
and take the best.

Train vs. Test Error



Where is KNN on this graph for K=1, or for K=Inf?

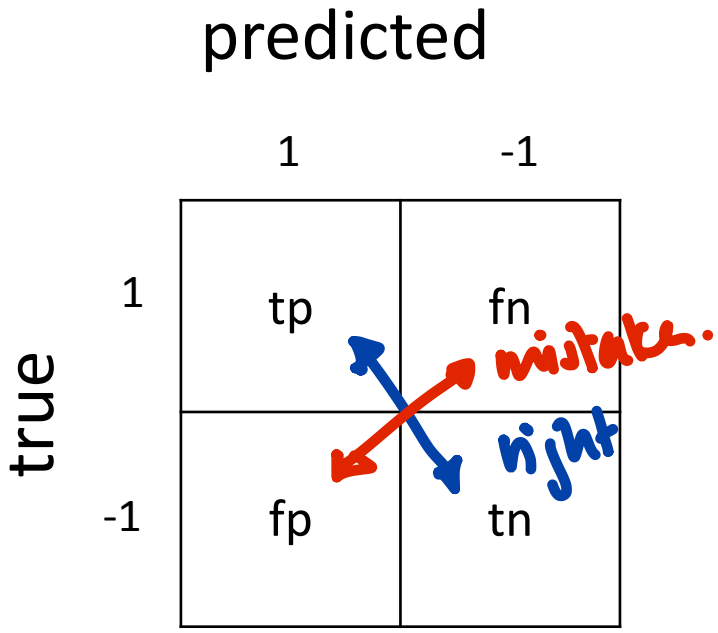
Grid Search



Zang et al., “Identification of heparin samples that contain impurities or contaminants by chemometric pattern recognition analysis of proton NMR spectral data”, Anal Bioanal Chem (2011)

Error Measures

- True positive (tp)
- True negative (tn)
- False positive (fp)
- False negative (fn)



TPR and FPR

- True Positive Rate:

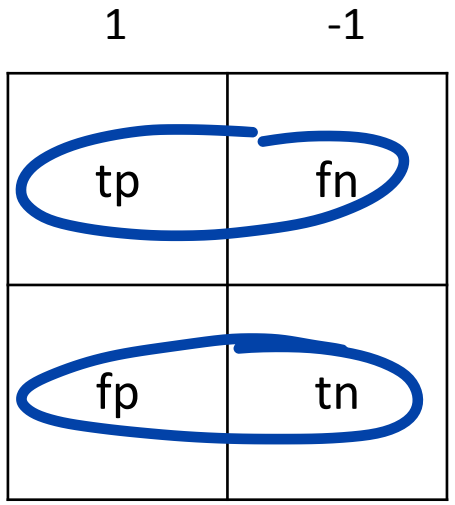
$$\frac{tp}{tp+fn}$$

↳ all positive examples.

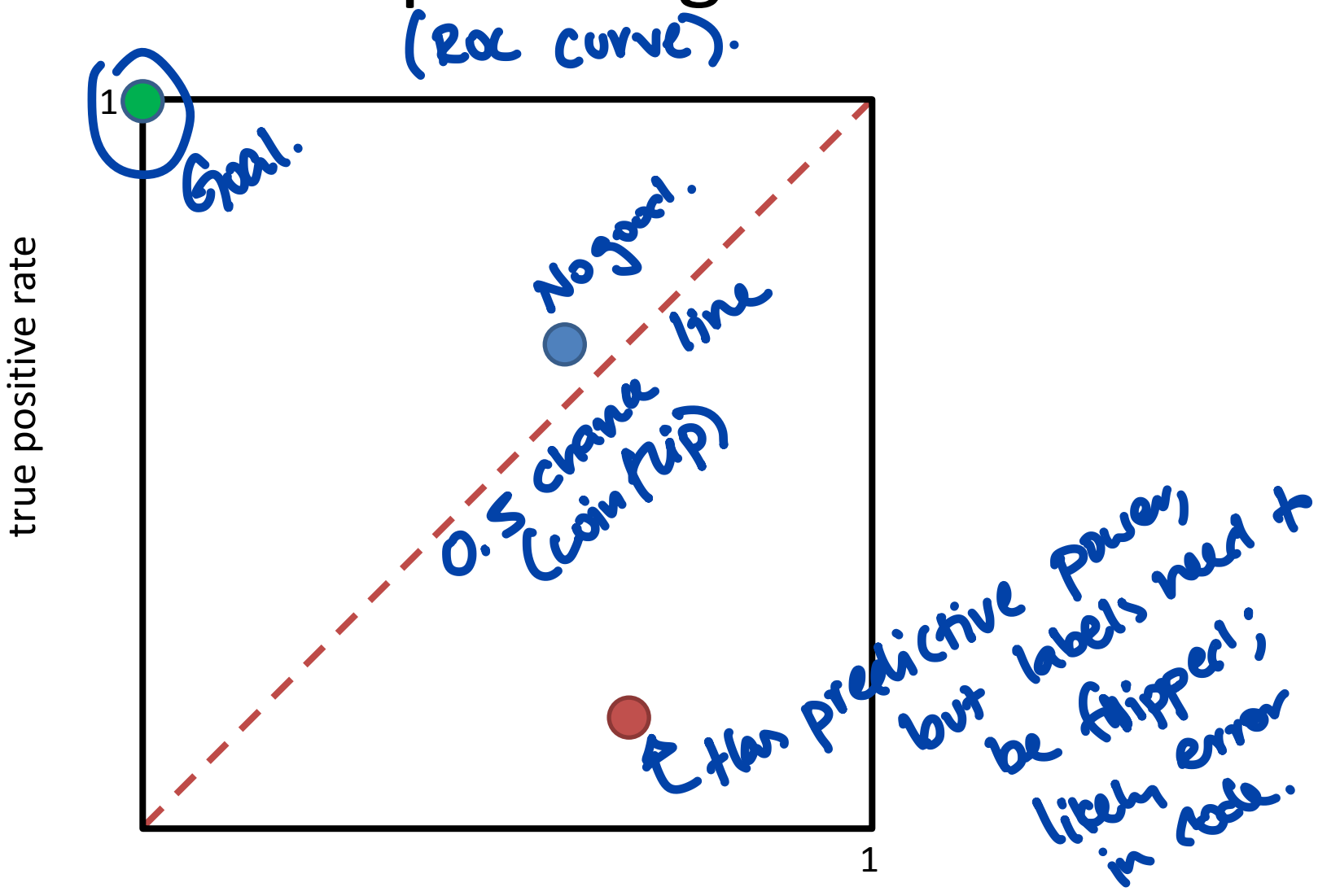
- False Positive Rate:

$$\frac{fp}{fp+tn}$$

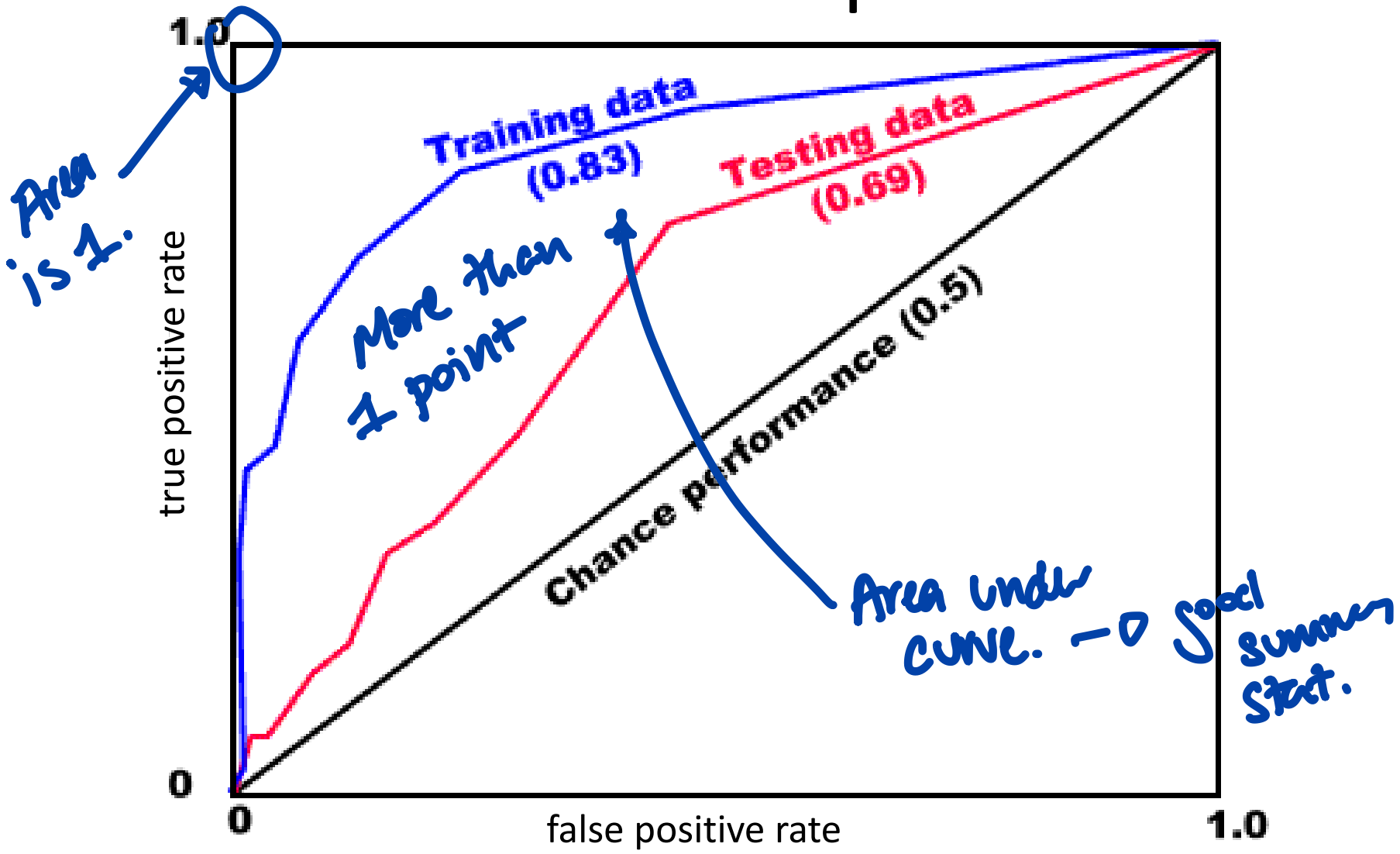
↳ all negative examples.



Reciever Operating Characteristic



ROC Example

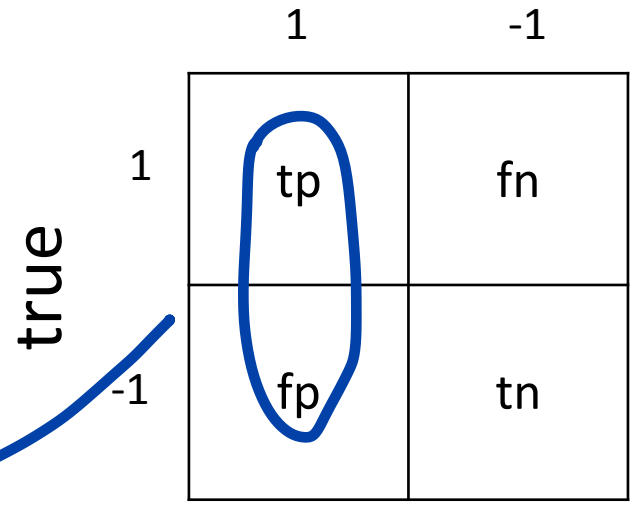


https://inclass.kaggle.com/c/ca-2015/details/evaluation

Precision Recall

Good for imbalanced datasets, like churn.

- Recall: $\frac{tp}{tp+fn}$ (TPR) (same; diff name)
- Precision: $\frac{tp}{tp+fp}$ ↳ not the same as FPR

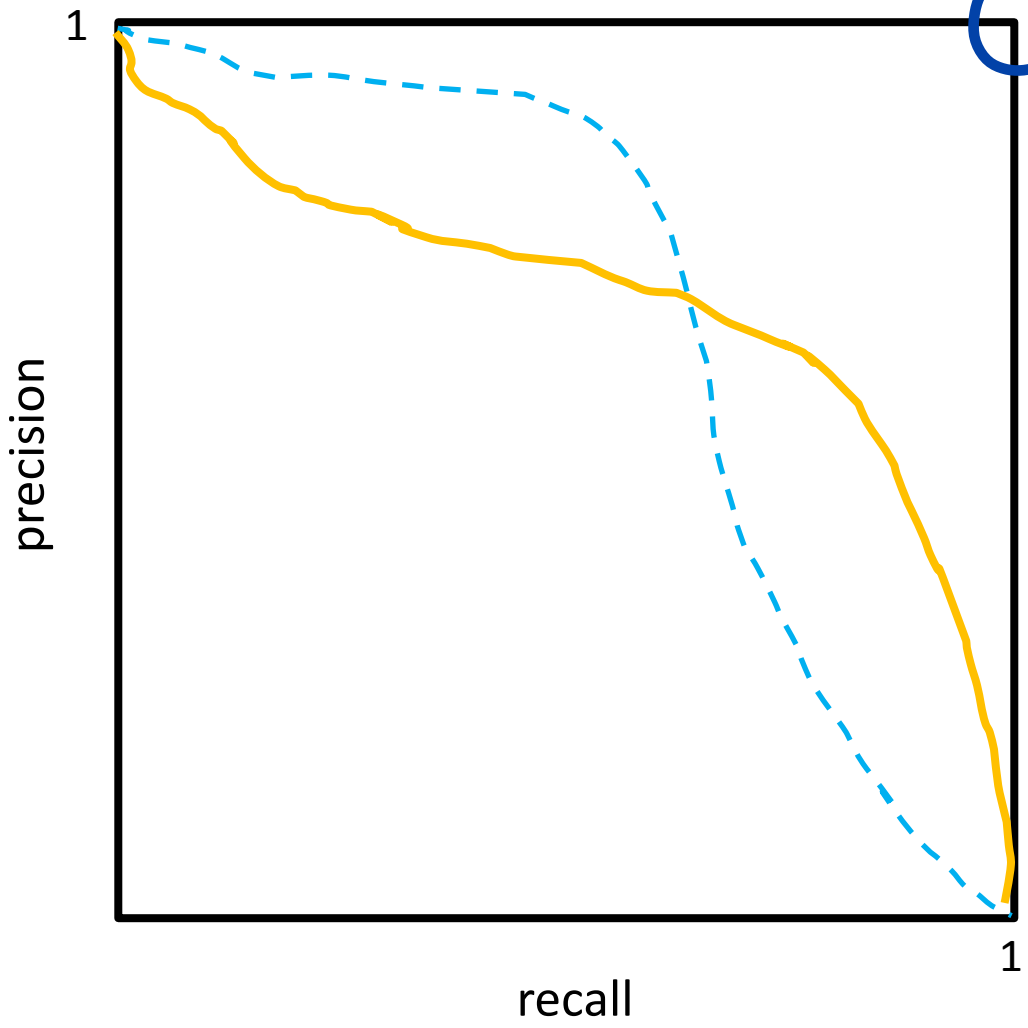


Precision Recall

- **Recall**: If I pick a random positive example, what is the probability of making the right prediction?

- **Precision**: If I take a positive prediction example, what is the probability that it is indeed a positive example?

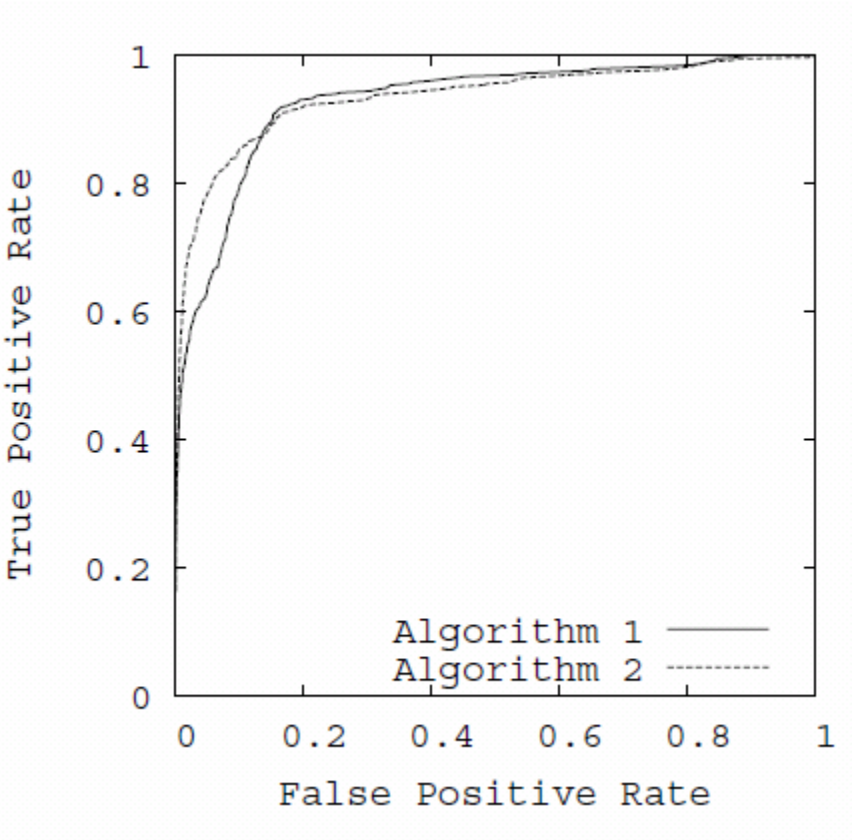
Precision Recall Curve

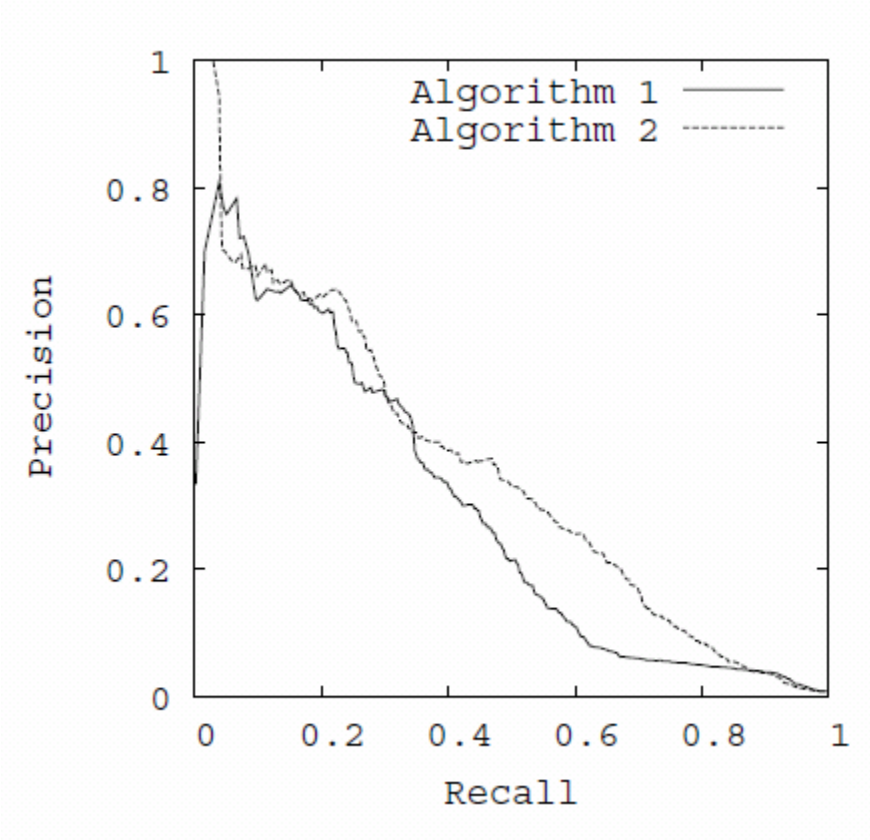


goal.
↳ Precision & recall to be 1.

Which is more important depends on the application

Comparison





J. Davis & M. Goadrich,
"The Relationship Between Precision-Recall and ROC Curves.",
ICML (2006)

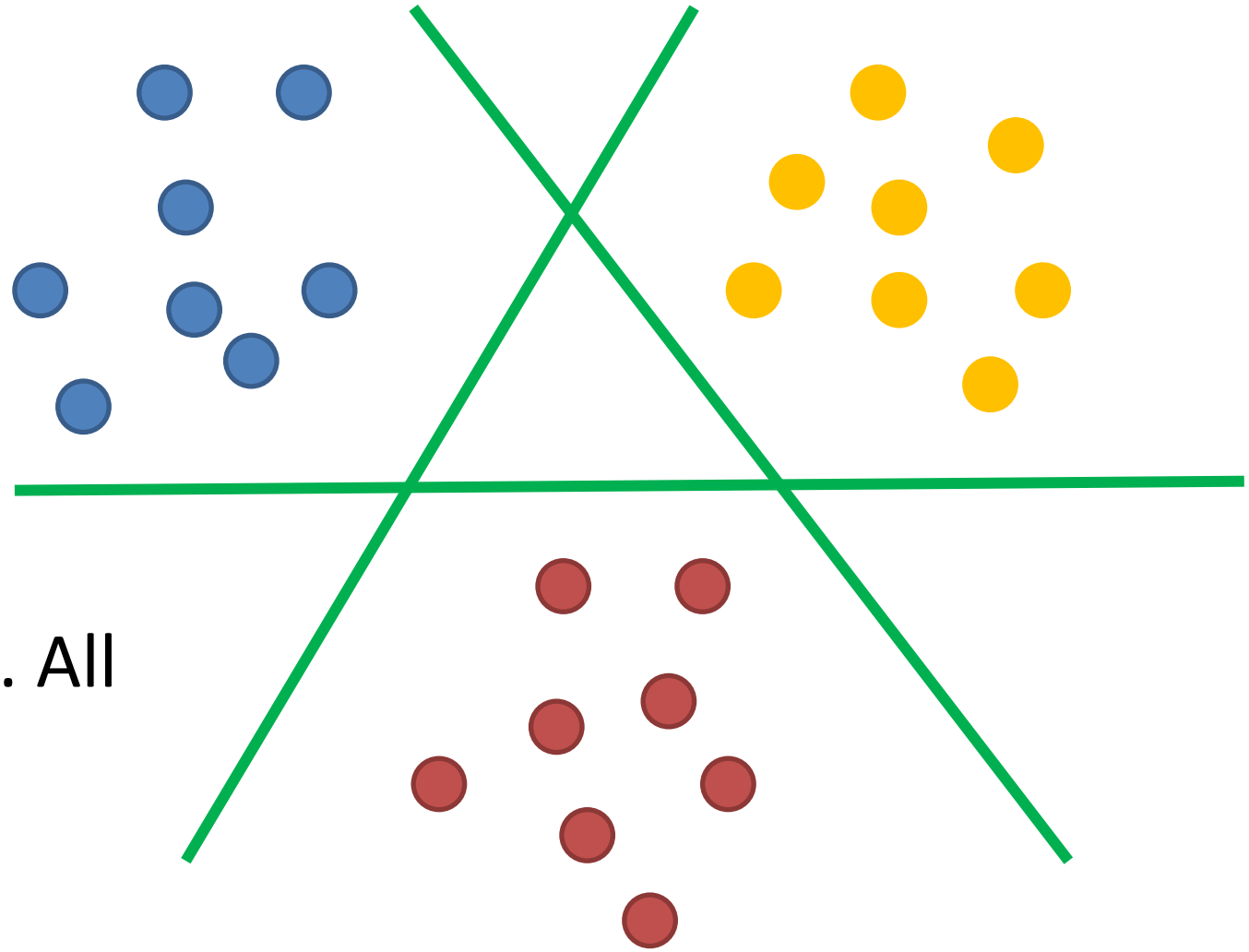
F-measure

- Weighted average of precision and recall

$$F_\beta = \frac{(\beta^2+1)\cdot P \cdot R}{\beta^2 \cdot P + R}$$

- Usual case: $\beta = 1$
- Increasing β allocates weight to recall

Multi Class

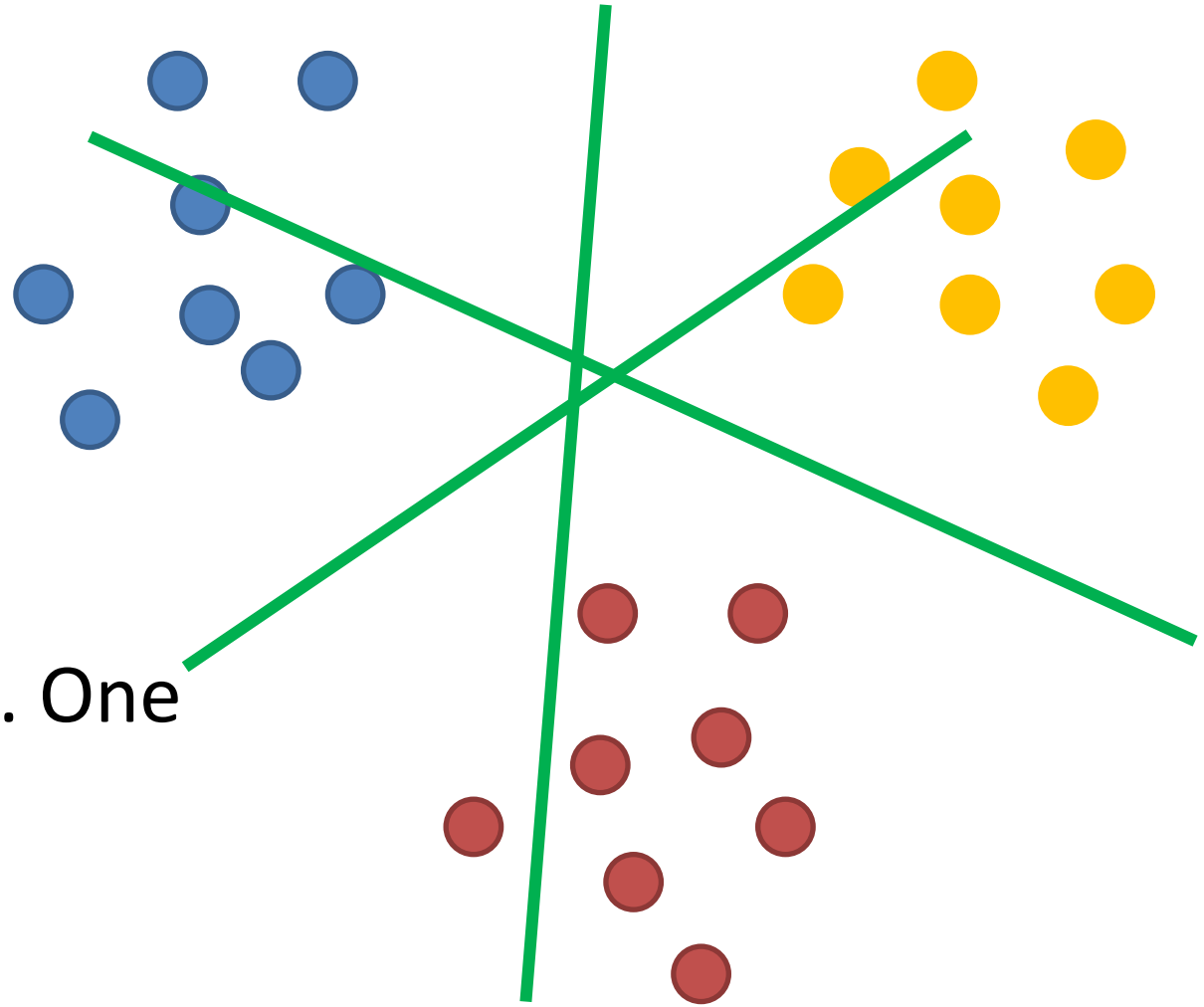


One vs. All

One vs All

- Train n classifier for n classes
- Take classification with greatest margin
- Slow training

Multi Class

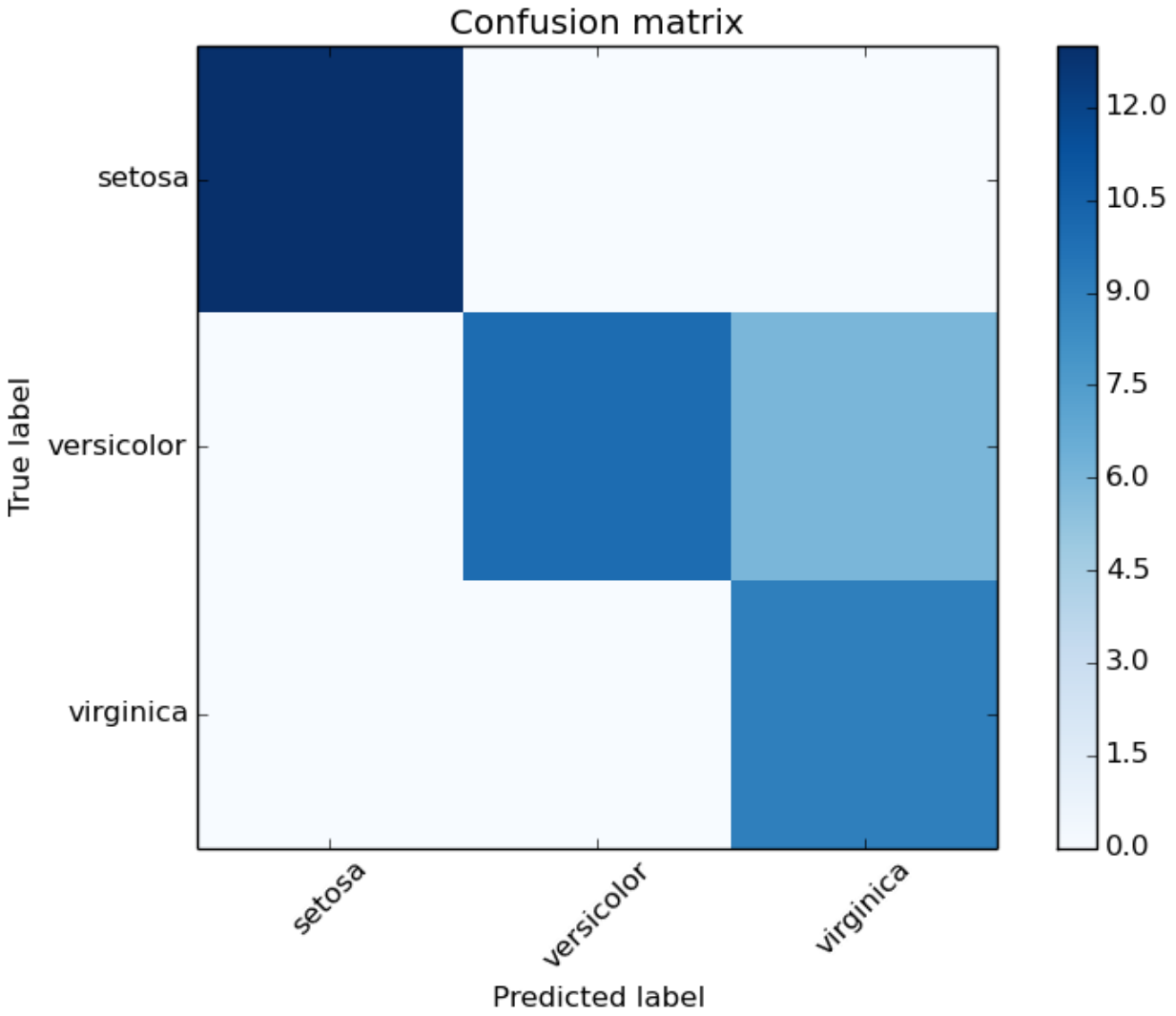


One vs. One

One vs One

- Train n(n-1)/2 classifiers
- Take majority vote
- Fast training

Confusion Matrix



http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html

Recap

- Perceptrons are great
- But really just a separating hyperplane
- So is SVM
- Kernels are neat
- Evaluation metrics are important