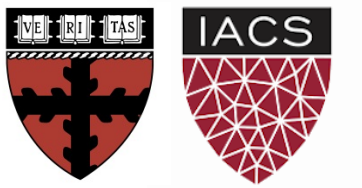
Practical Machine Learning

Verena Kaynig-Fittkau (vkaynig@seas.Harvard.edu)

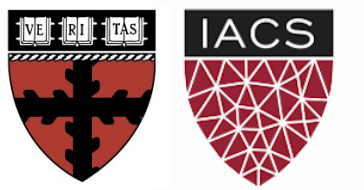
We are drowning in information and starving for knowledge

John Naisbitt



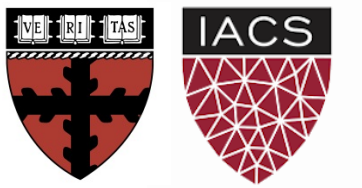
Machine Learning

- Analyze training data
- Make predictions for new unseen data:
  - supervised learning
- Find patterns:
  - unsupervised learning

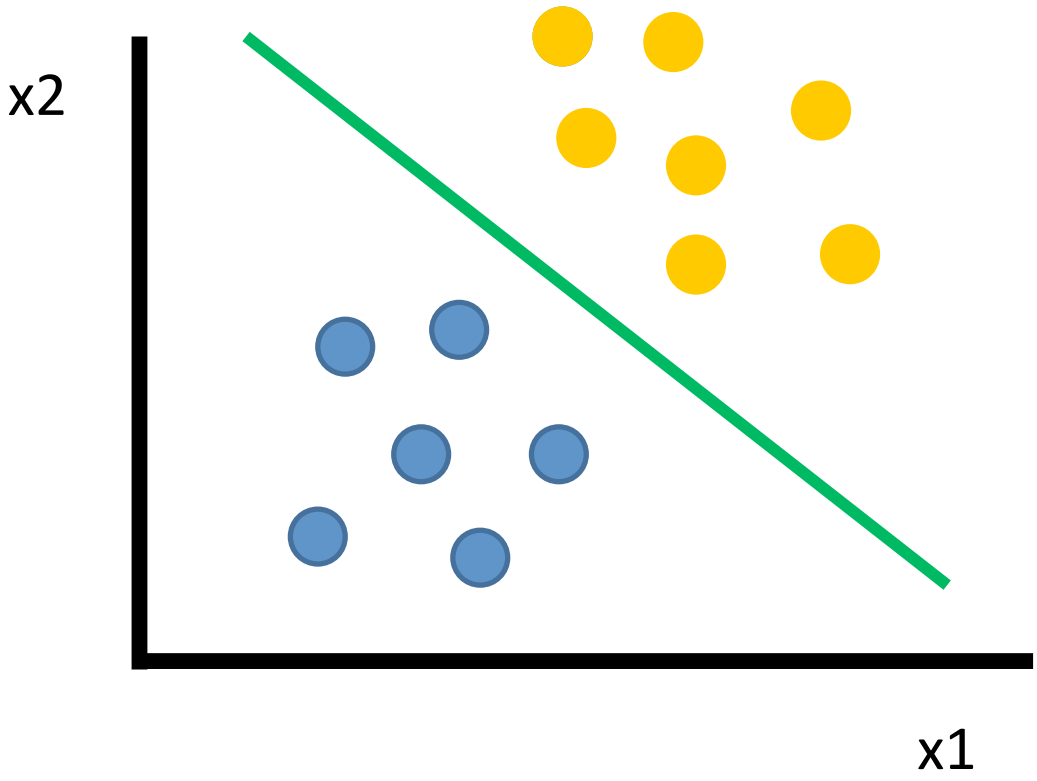


Machine Learning

- Supervised Learning
  - SVM, Decision Tree, Boosting, Random Forest
- Unsupervised Learning
  - K-means, mean shift



Supervised Learning

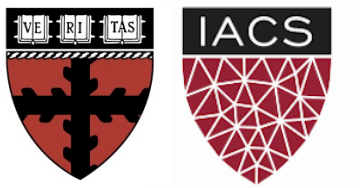


data points

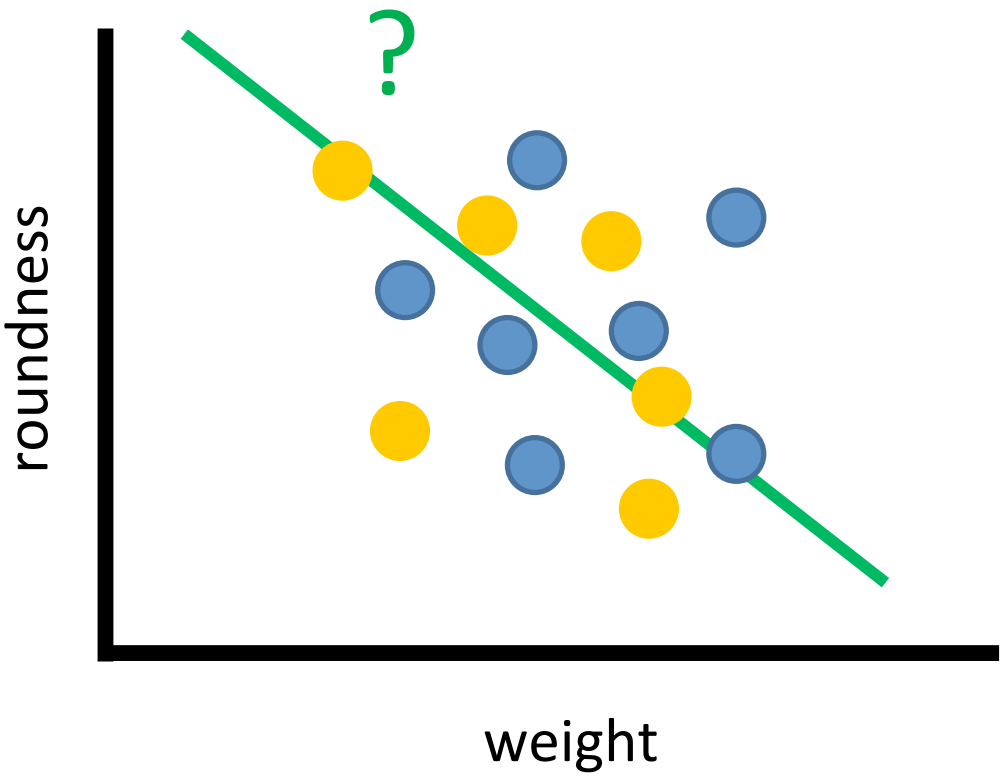
labels

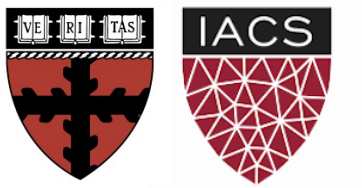
features

separating hyper plane

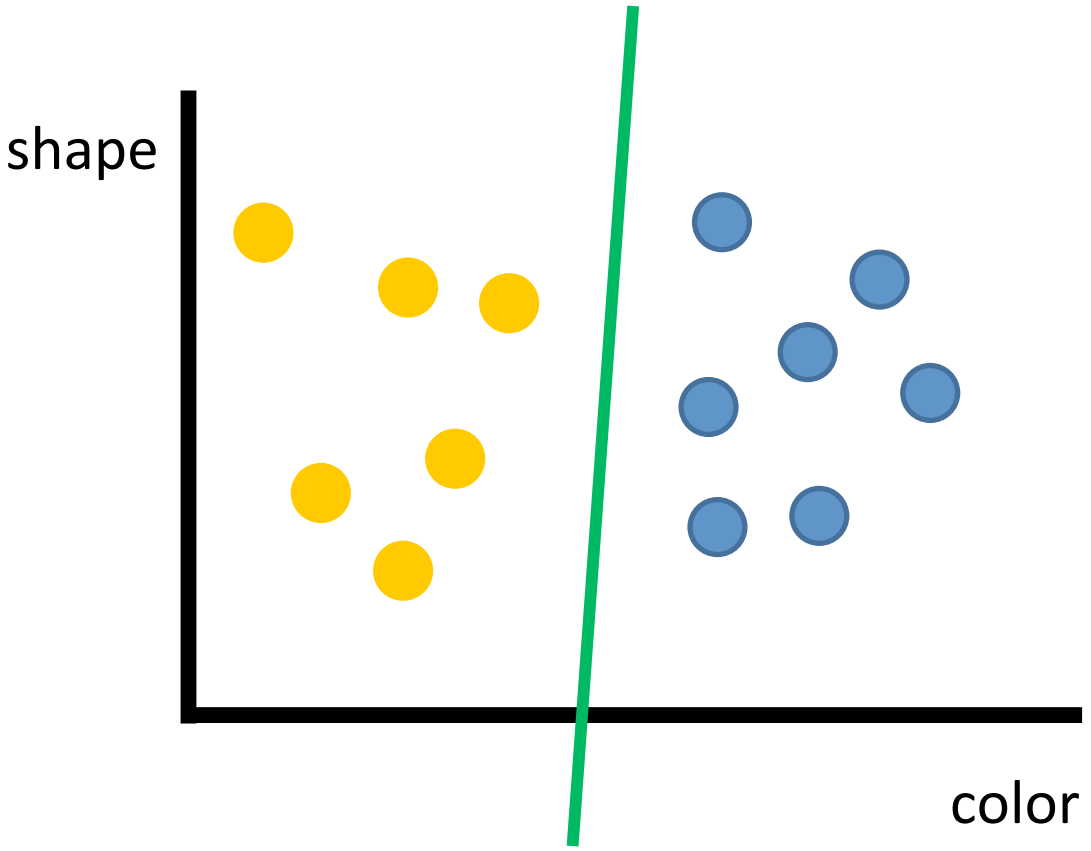


Features are important

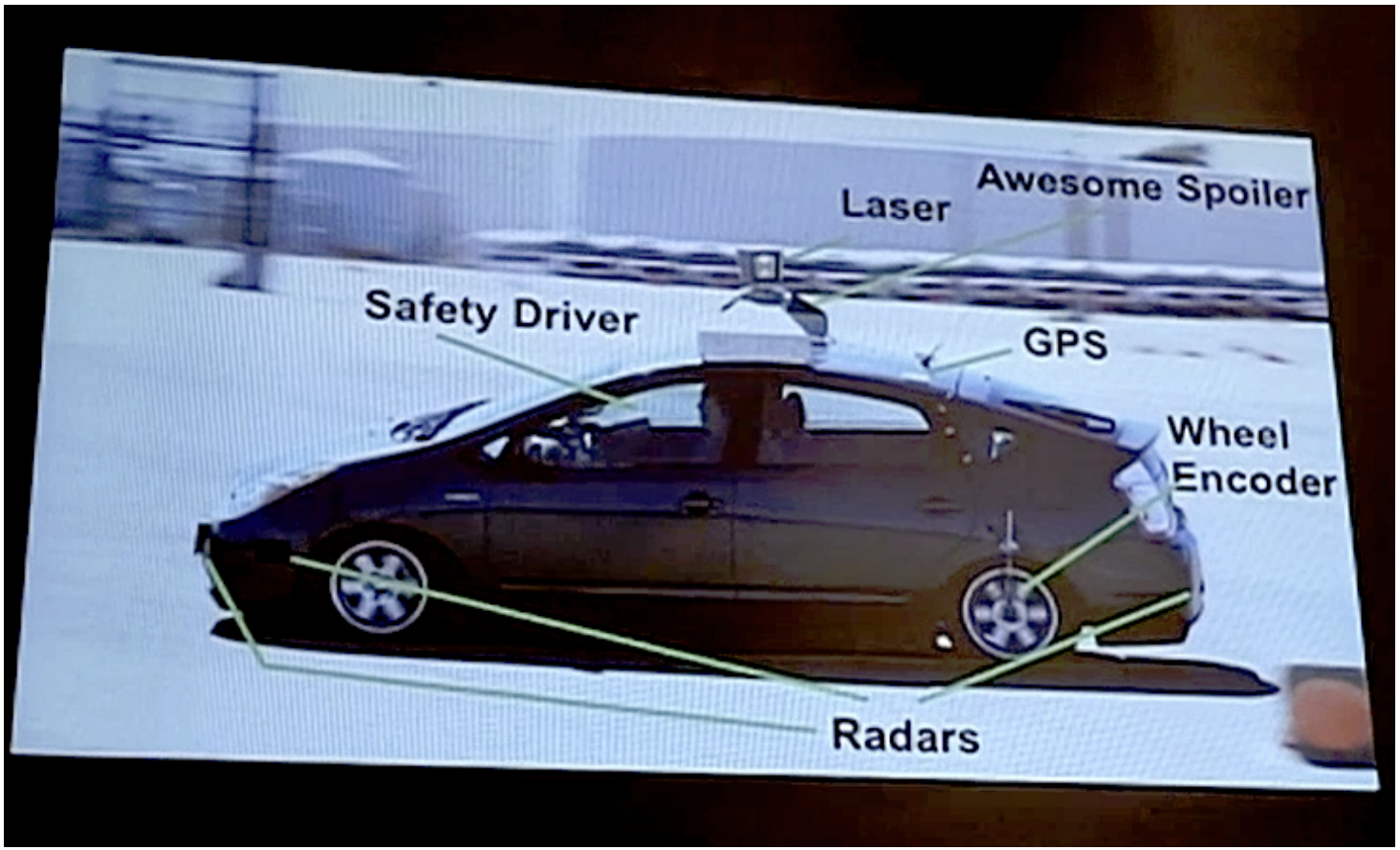




Features are important

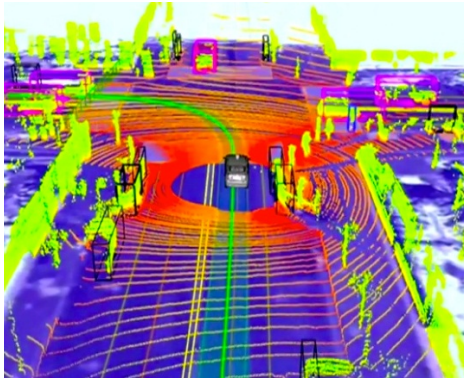


Google's Self-Driving Car



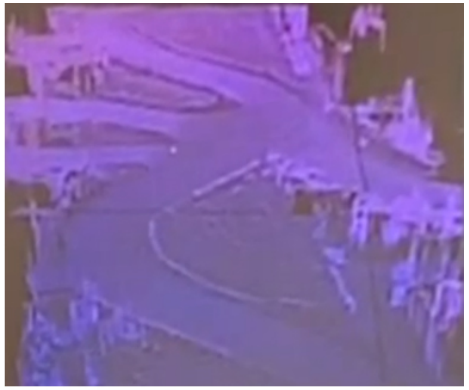
Car Features

Laser scan

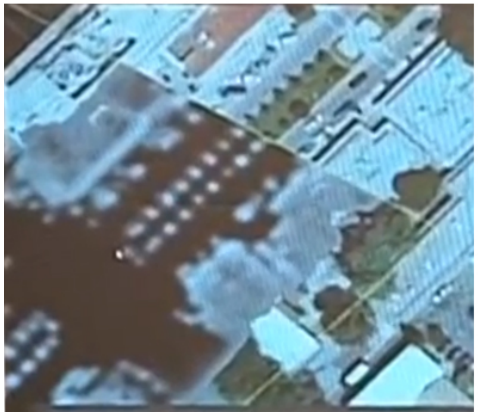


Intensity model

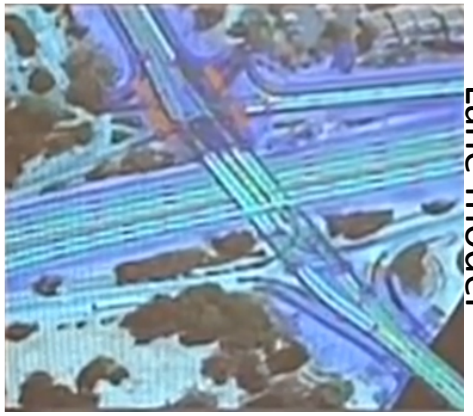
Elevation model



Camera vision



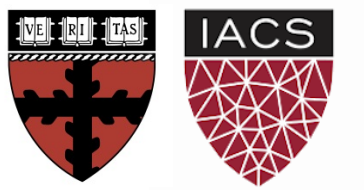
2D stationary map



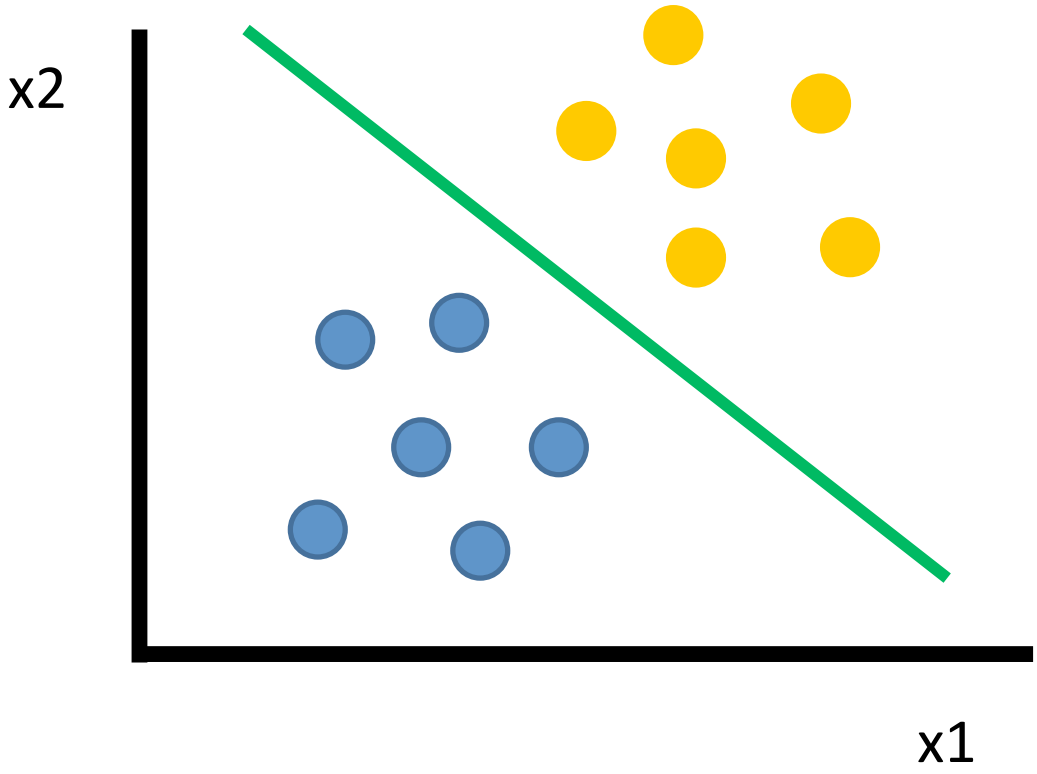
Lane model

So just measure everything?

- More features = better classification?
- Practical issues:
  - Data volume, computation overhead
- Theoretical issues:
  - Generalization performance
  - Curse of dimensionality



Supervised Learning

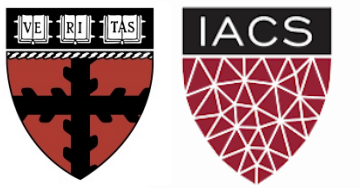


data points

labels

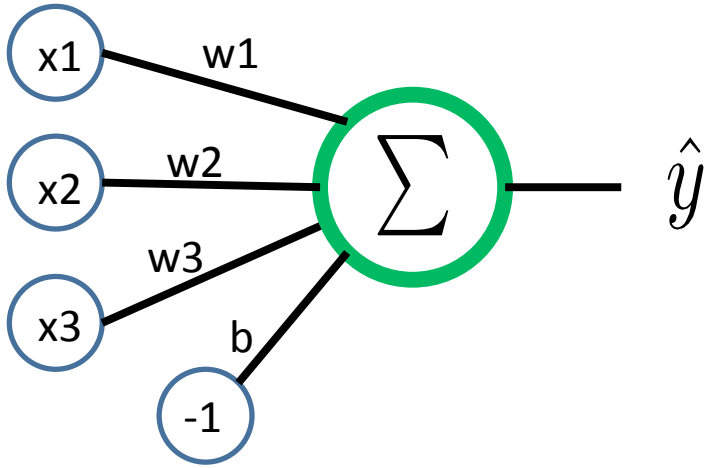
features

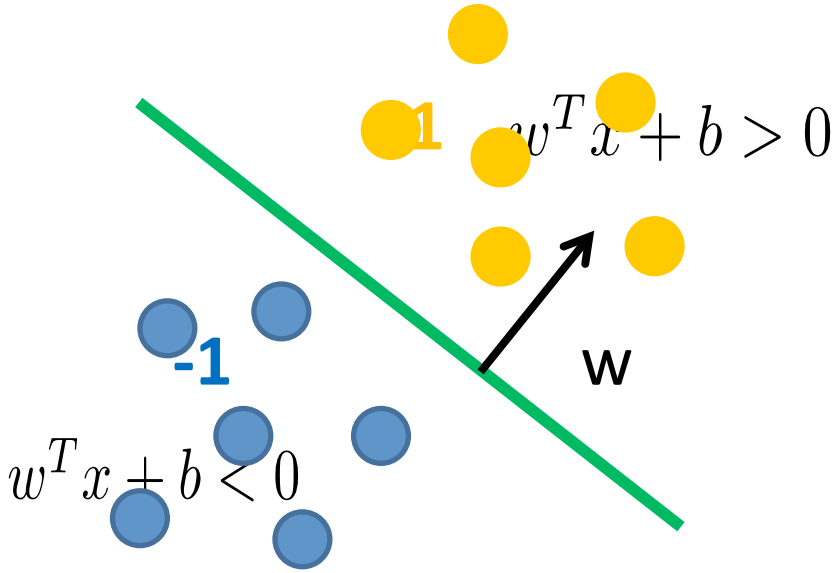
separating hyper plane



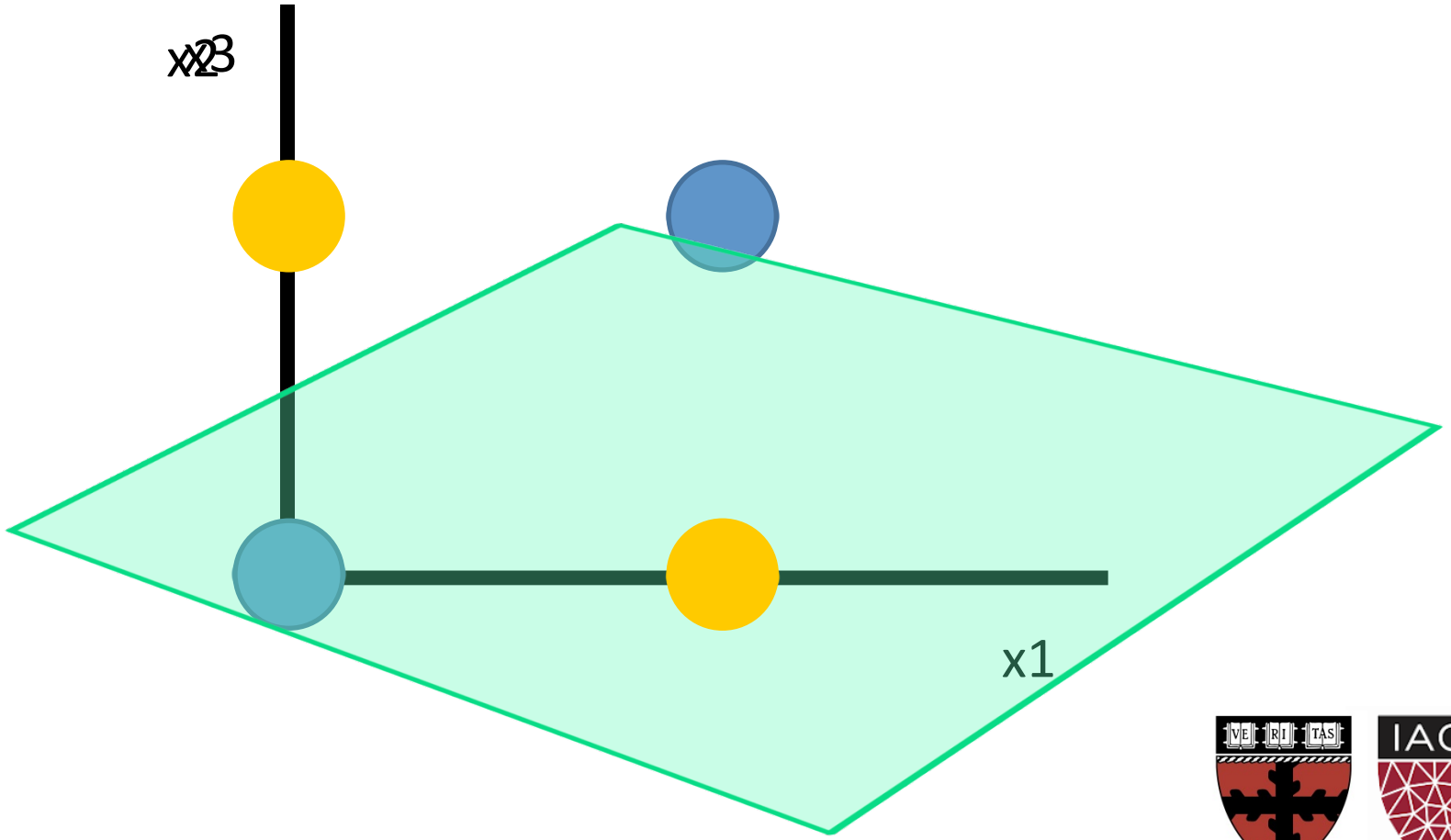
Perceptron

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





The XOR Problem

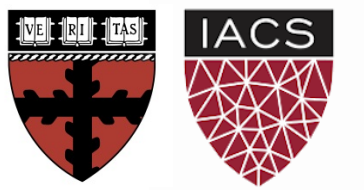


Support Vector Machine

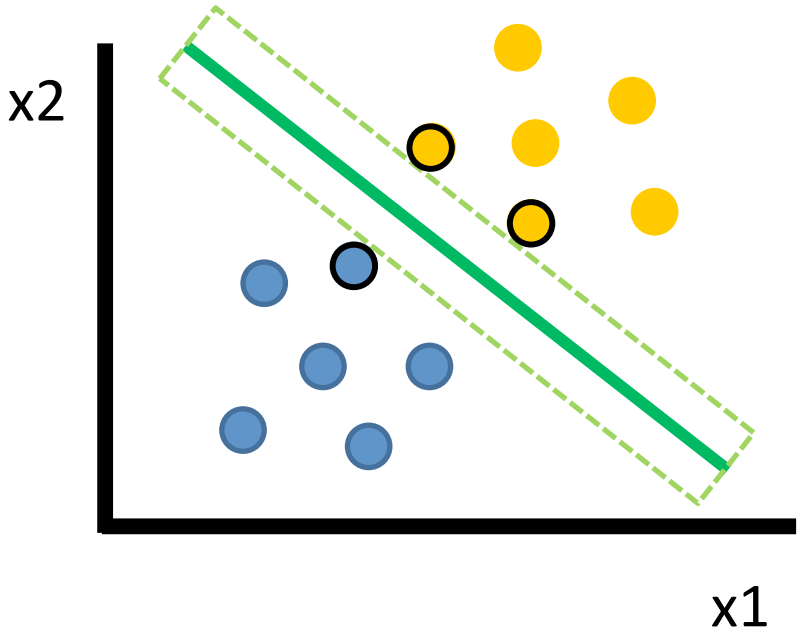
- Widely used for all sorts of classification problems

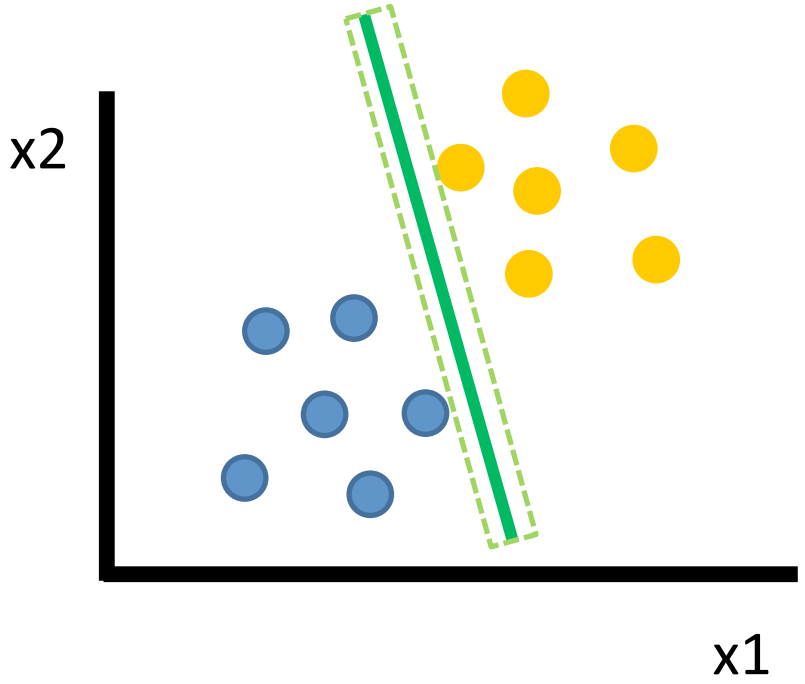
www.clopinet.com/isabelle/Projects/SVM/applist.html

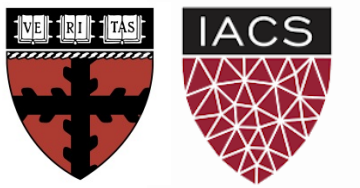
- Some people say it is the best of the shelf classifier out there



Maximum Margin Classification

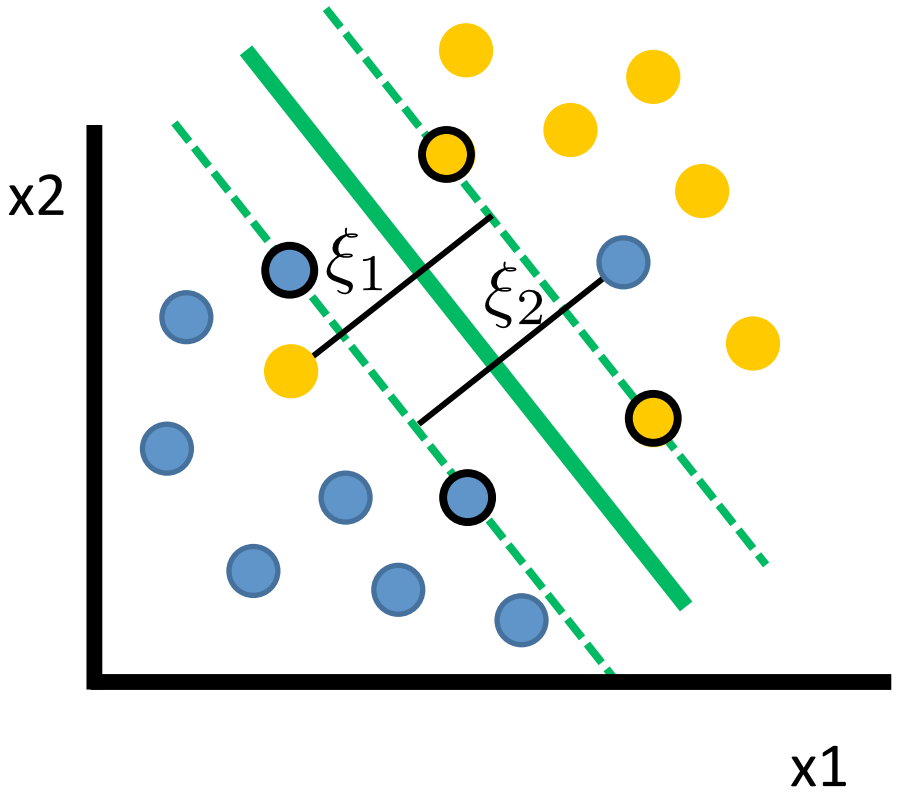


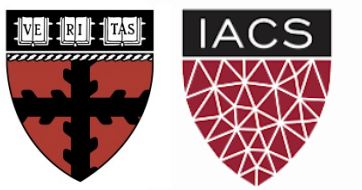




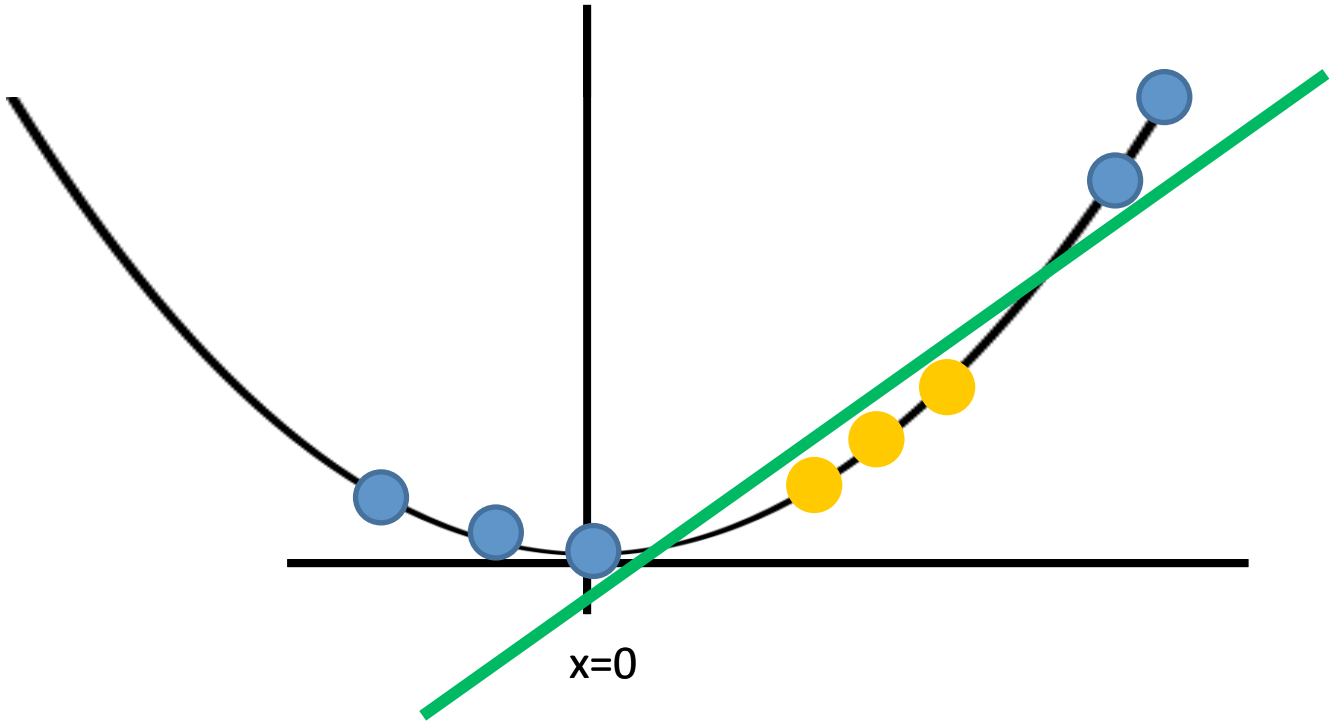
What about outliers?

ξ_i: slack variables

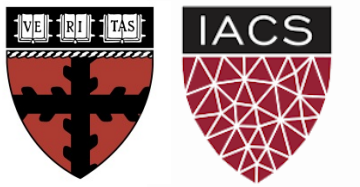




XOR problem revised



Did we add information to make the problem seperable?



SVM with a polynomial Kernel visualization

Created by:
Udi Aharoni

Quadratic Kernel

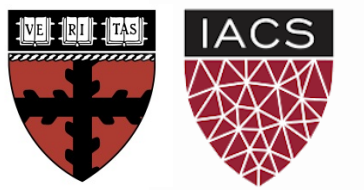
$$x = (x_1, x_2)$$

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

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

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

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



Kernel Functions

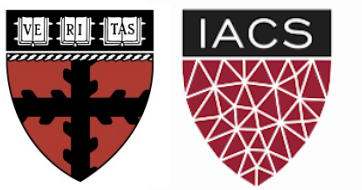
$$K(x, z) = \Phi(x) \cdot \Phi(z)$$

- Polynomial:

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

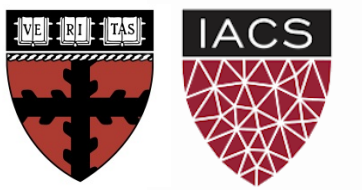
- Radial basis function (RBF):

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



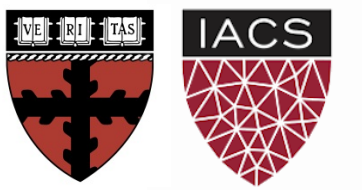
Kernel Trick for SVMs

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



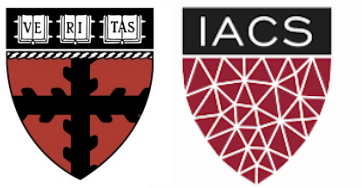
SVM Applet

http://www.ml.inf.ethz.ch/education/lectures_and_seminars/annex_estat/Classifier/JSupportVectorApplet.html



Tips and Tricks

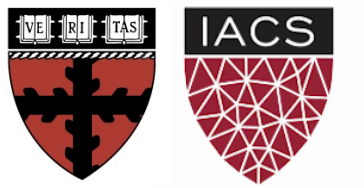
- SVMs are not scale invariant
- Check if your library normalizes by default
- Normalize your data
  - mean: 0 , std: 1
  - map to [0,1] or [-1,1]
- Normalize test set in same way!



Tips and Tricks

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

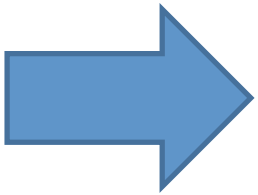
  Chih-Wei Hsu et al., "A Practical Guide to Support Vector Classification", Bioinformatics (2010)



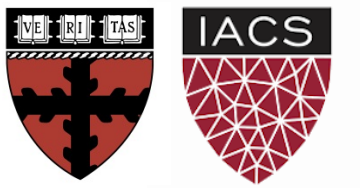
Parameter Tuning

- Given a classification task

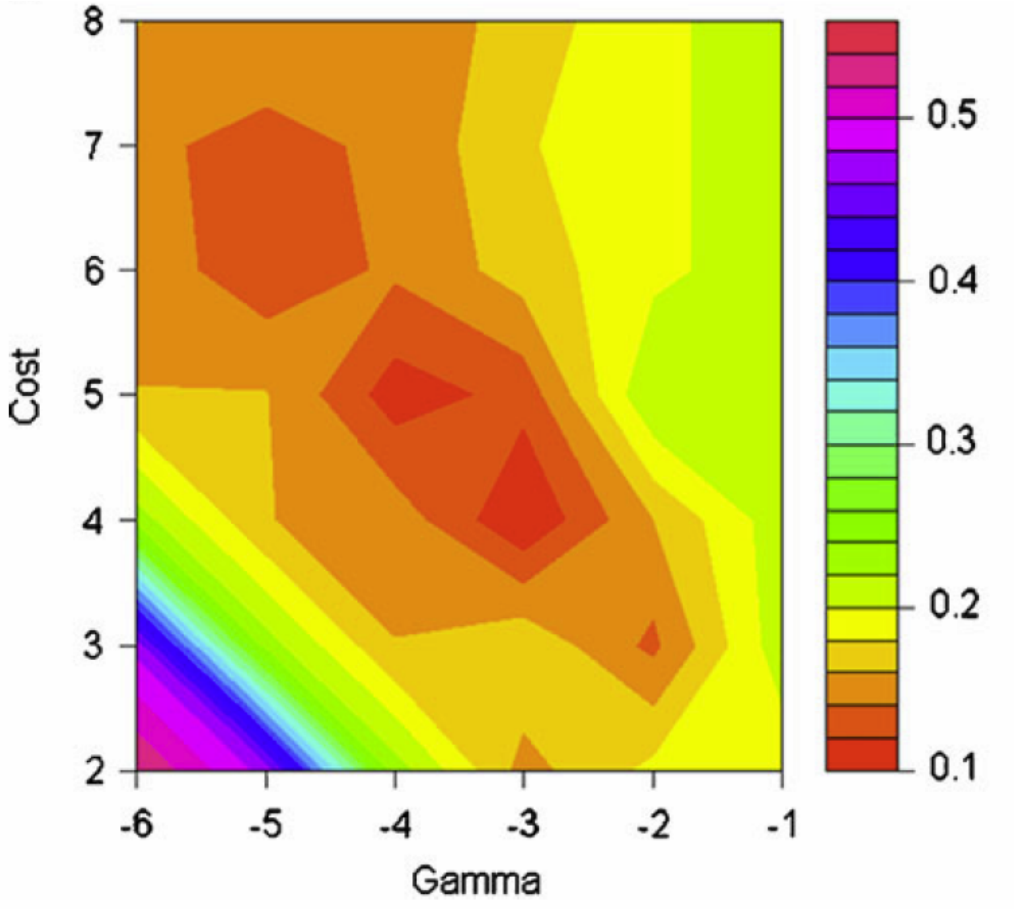
- Which kernel ?
- Which kernel parameter values?
- Which value for C?



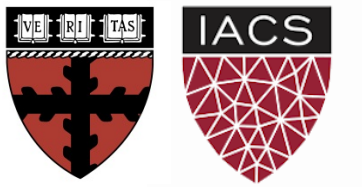
Try different combinations and take the best.



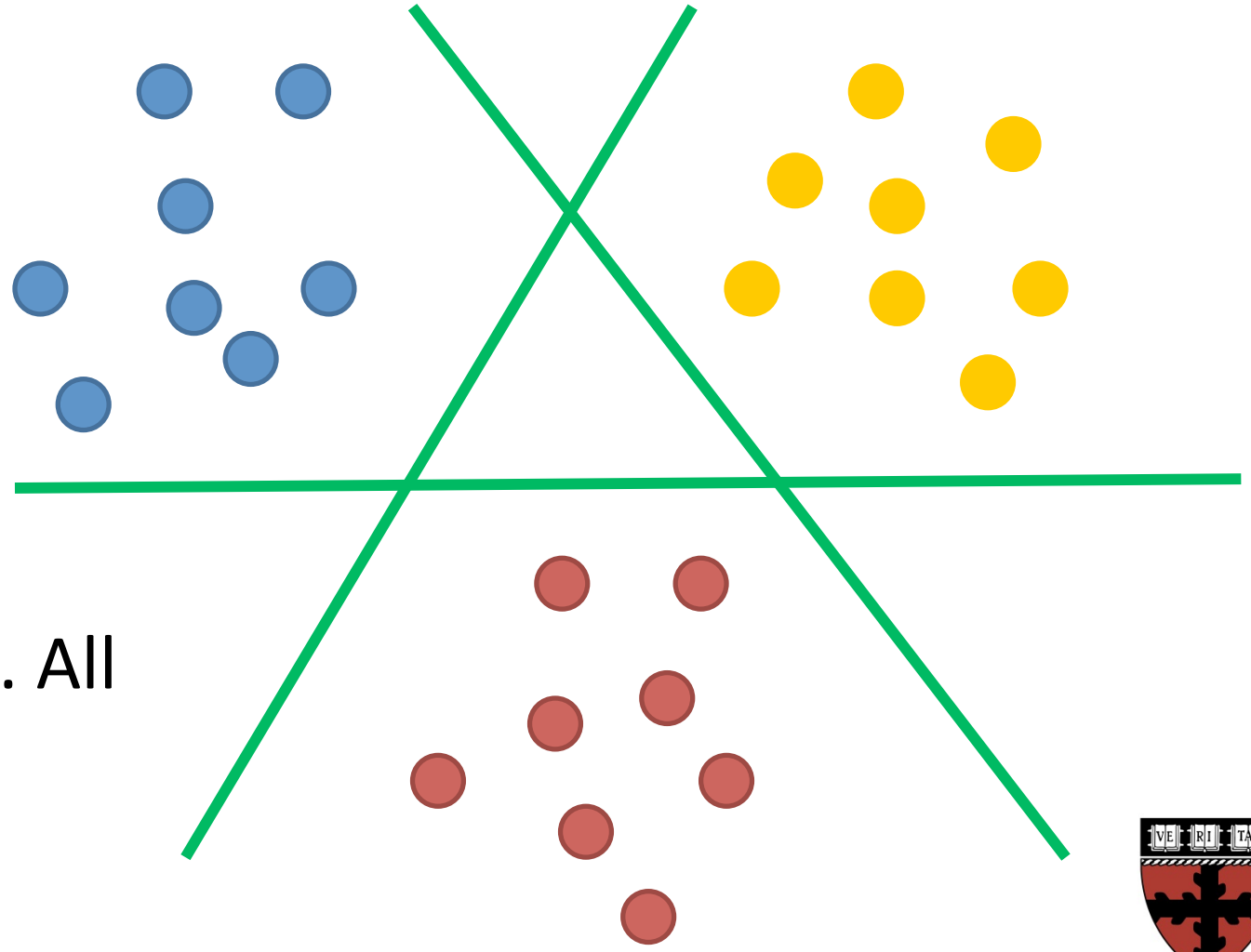
Grid Search



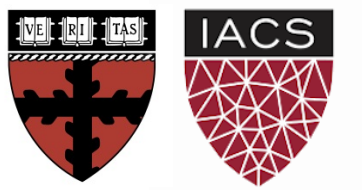
Zang et al., "Identification of heparin samples that contain impurities or contaminants by chemometric pattern recognition analysis of proton NMR spectral data", Anal Bioanal Chem (2011)



Multi Class

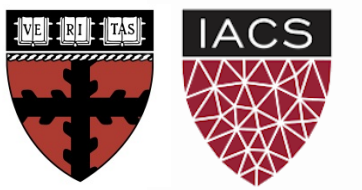


One vs. All

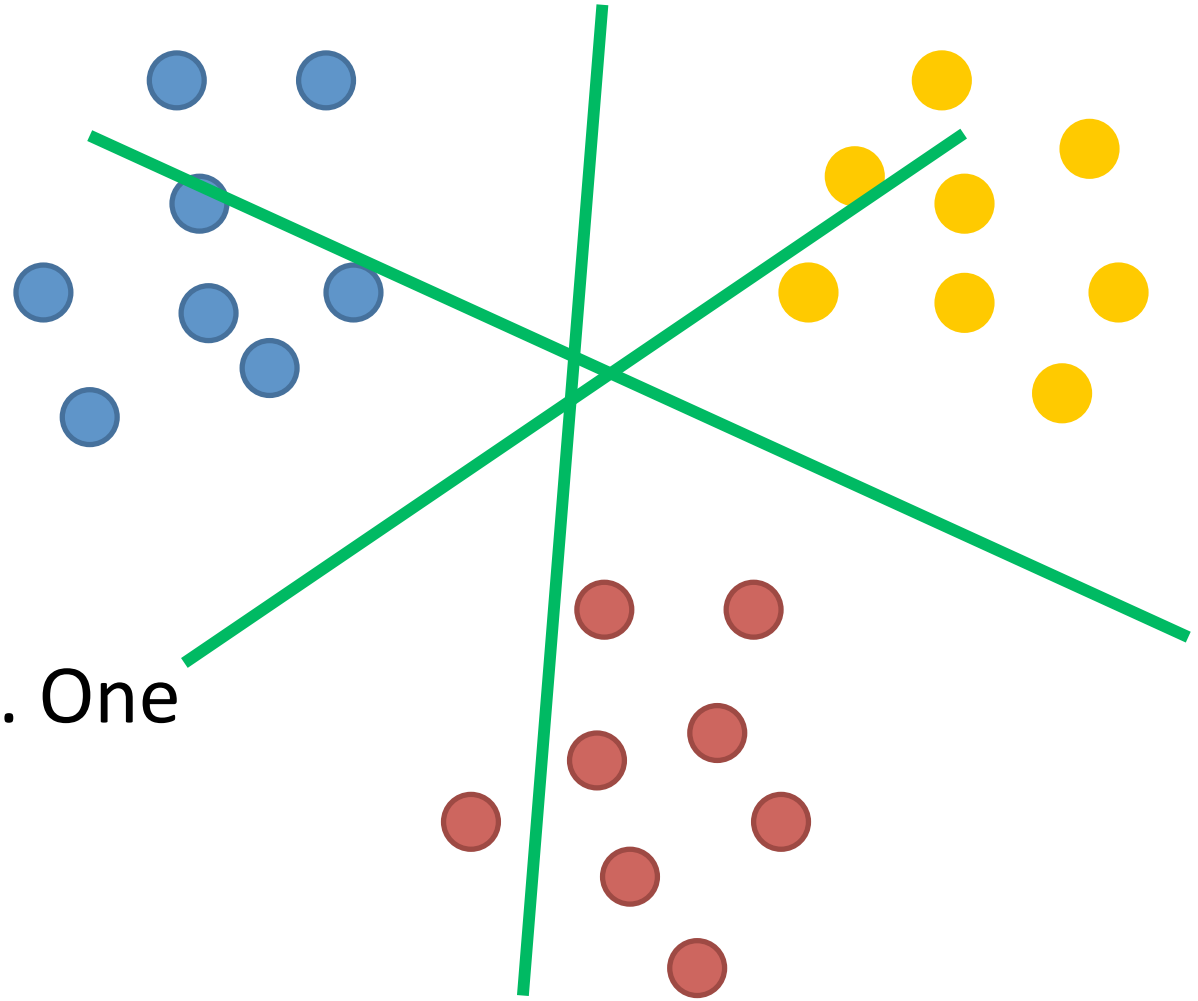


One vs All

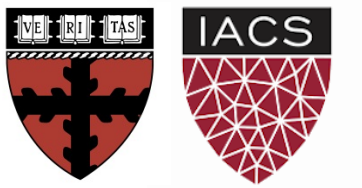
- Train n classifier for n classes
- Take classification with greatest positive margin
- Slow training



Multi Class

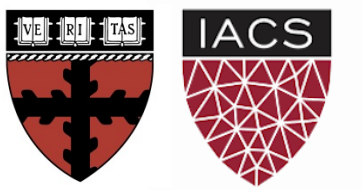


One vs. One

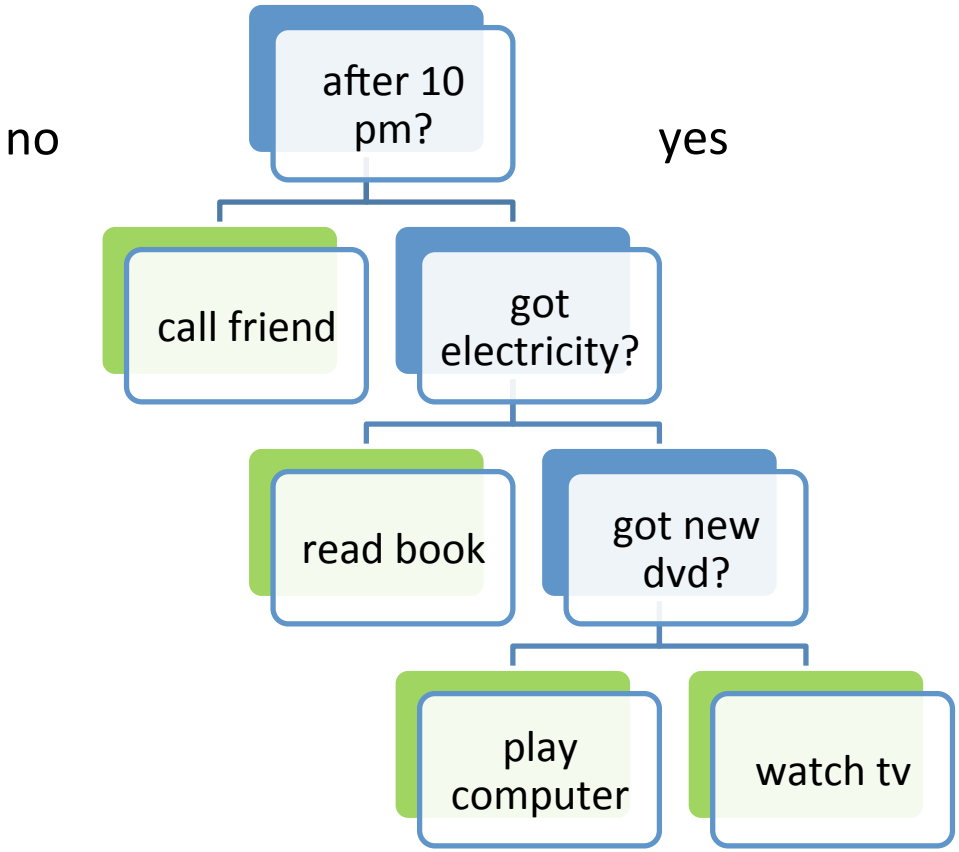


One vs One

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

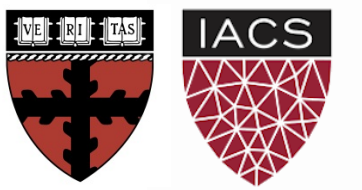


Decision Tree

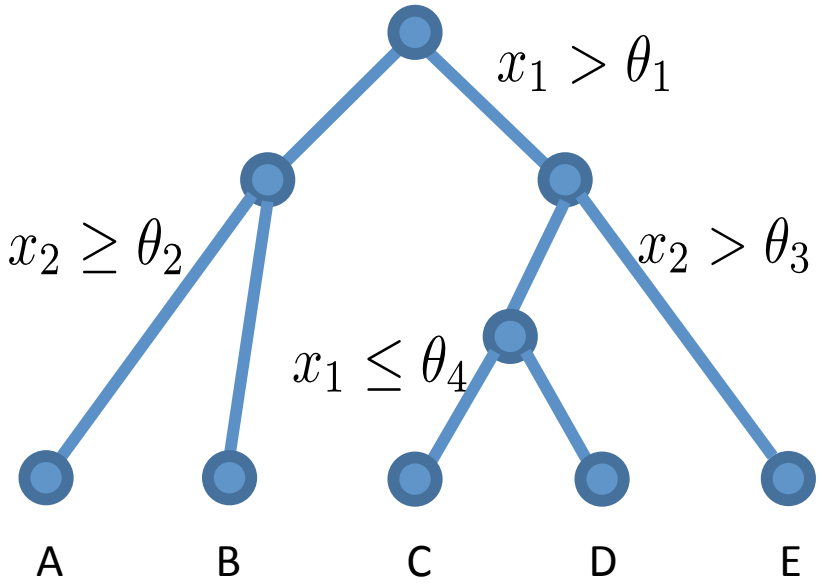


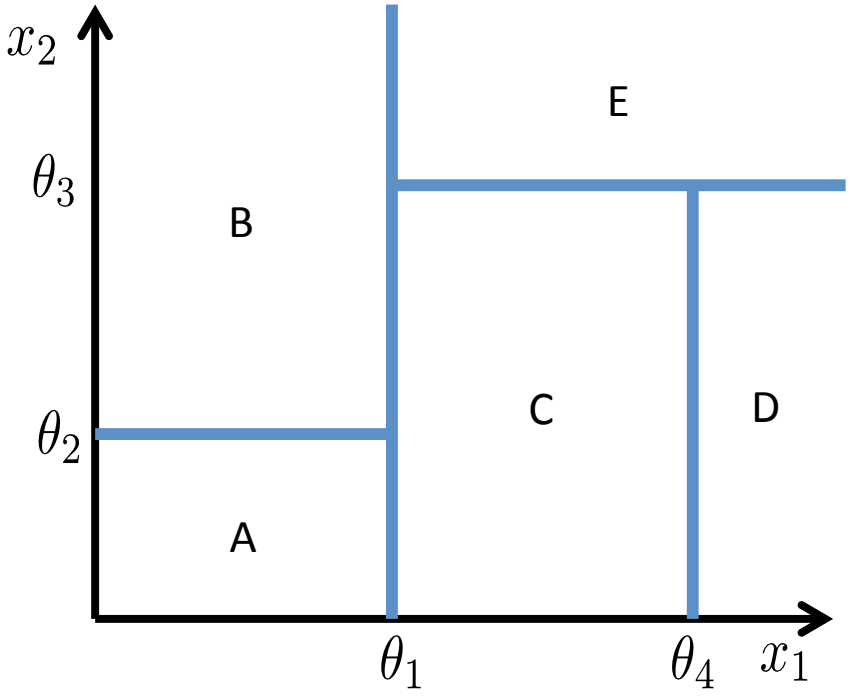
Decision Trees

- Fast training
- Fast prediciton
- Easy to understand
- Easy to interpret



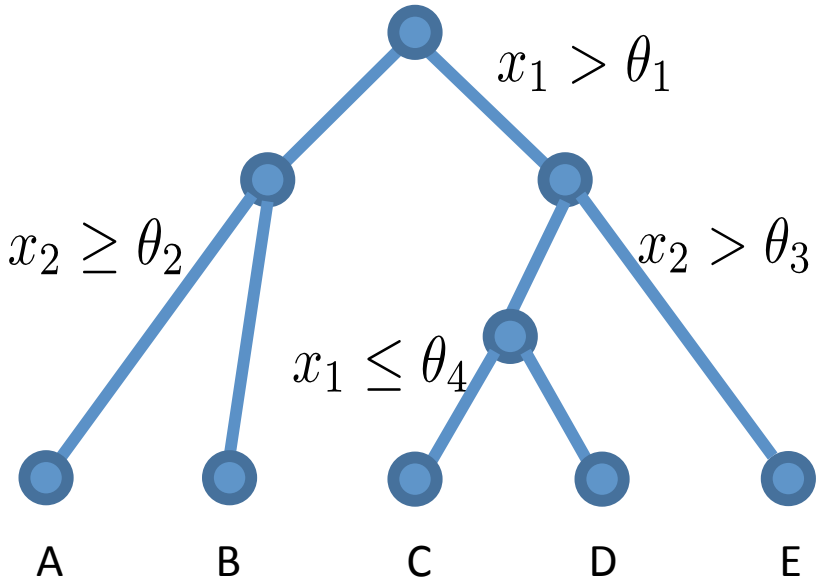
Decision Tree - Idea

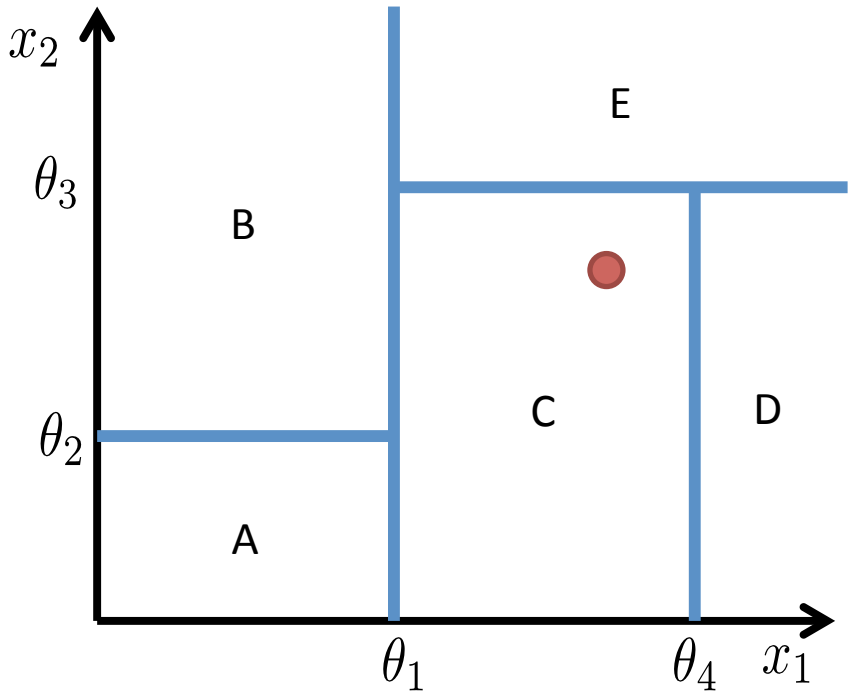




Bishop, "Pattern Recognition and Machine Learning", Springer, 2006

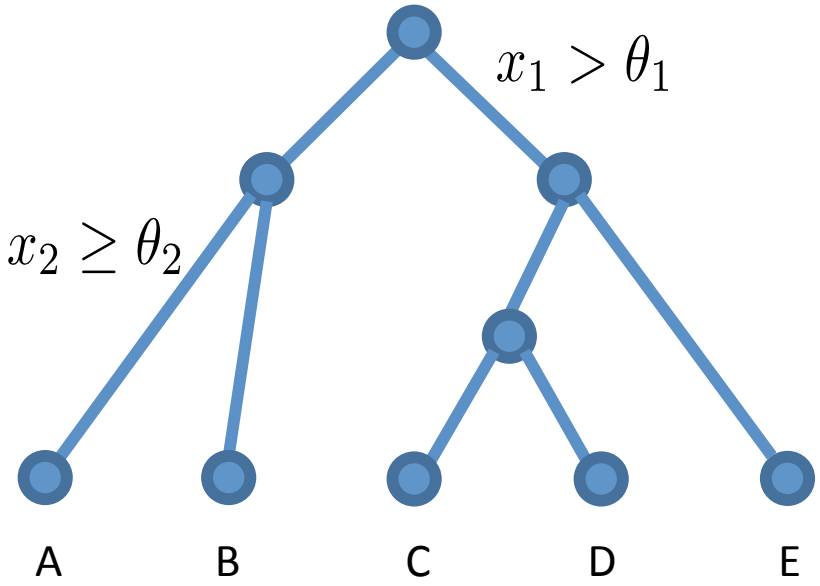
Decision Tree - Prediction



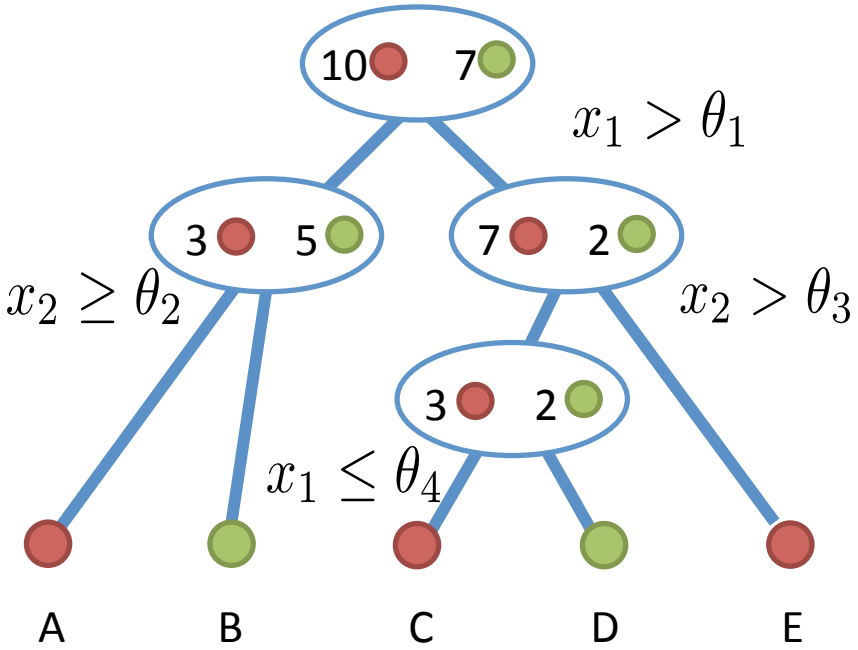


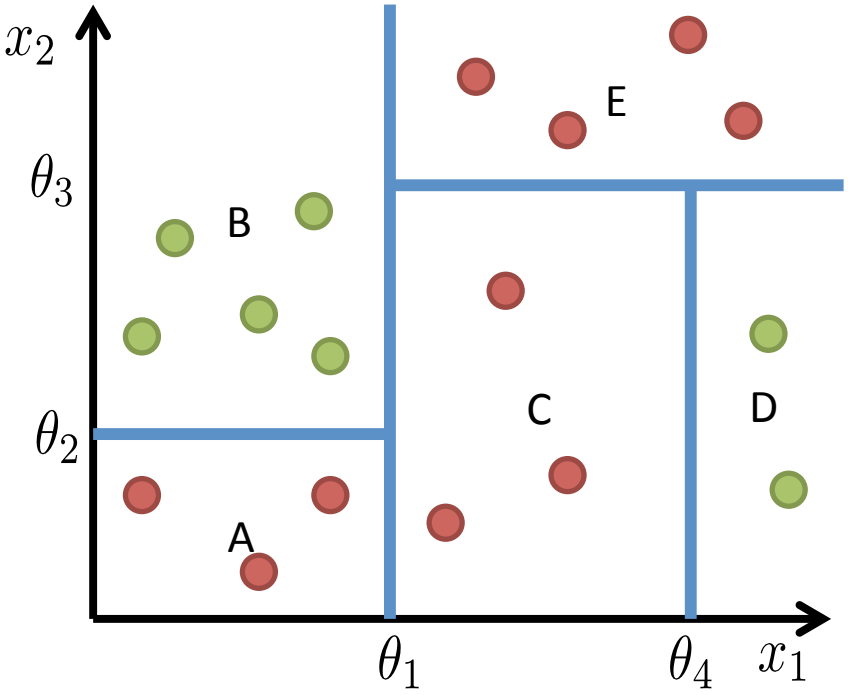
Decision Tree -Training

- Learn the tree structure:
  - which feature to query
  - which threshold to choose



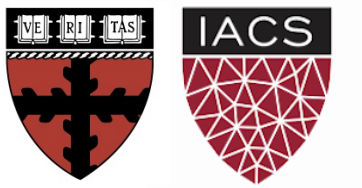
Node Purity


10
7
$x_1 > \theta_1$
3
5
7
2
$x_2 \geq \theta_2$
$x_2 > \theta_3$
3
2
$x_1 \leq \theta_4$
A
B
C
D
E


x_2
θ_3
θ_2
A
B
C
D
E
θ_1
θ_4
x_1

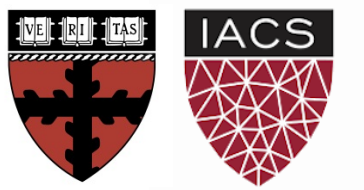
When to Stop

- node contains only one class
- node contains less than x data points
- max depth is reached
- node purity is sufficient
- you start to overfit => cross-validation



Decision Trees - Disadvantages

- Sensitive to small changes in the data
- Overfitting
- Only axis aligned splits



Decision Trees vs SVM

Characteristic	SVM	Trees
Natural handling of data of "mixed" type	▼	▲
Handling of missing values	▼	▲
Robustness to outliers in input space	▼	▲
Insensitive to monotone transformations of inputs	▼	▲
Computational scalability (large N)	▼	▲
Ability to deal with irrelevant inputs	▼	▲
Ability to extract linear combinations of features	▲	▼
Interpretability	▼	◆
Predictive power	▲	▼

Hastie et al.,"The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer (2009)

Wisdom of Crowds

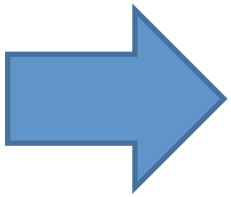
The collective knowledge of a diverse and independent body of people typically exceeds the knowledge of any single individual, and can be harnessed by voting.

James Surowiecki

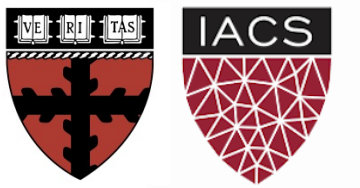
http://socialmedia4srm.wordpress.com/

Ensemble Methods

- A single decision tree does not perform well
- But, it is super fast
- What if we learn multiple trees?



For multiple trees we need even more data!

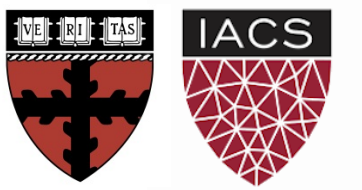


Bootstrap

- Resampling method from statistics
- Useful to get error bars on estimates

- Take N data points
- Draw N times with replacement

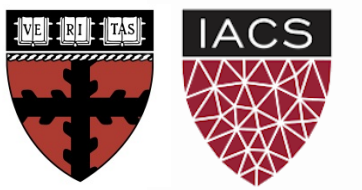
- Get estimate from each bootstrapped sample



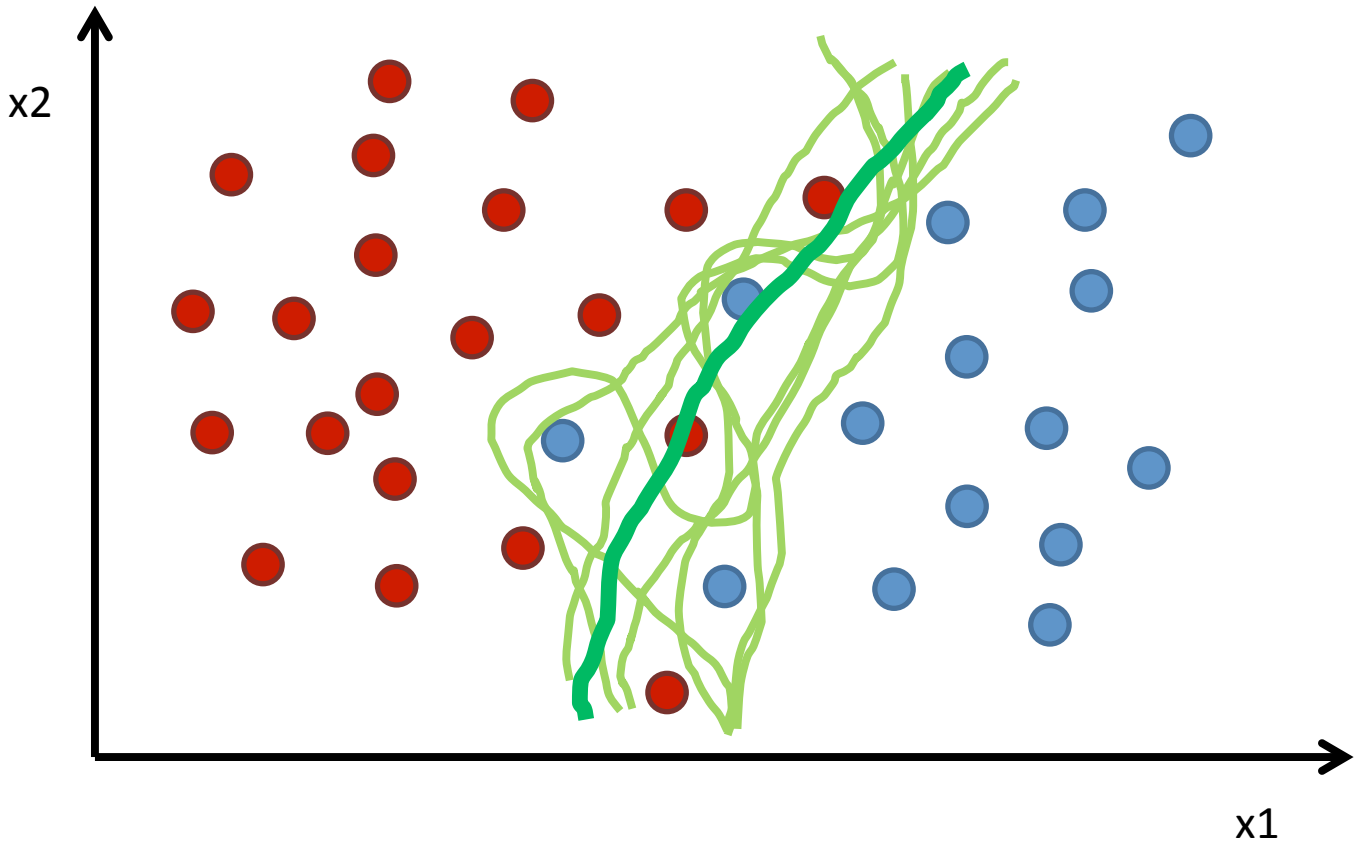
Bagging

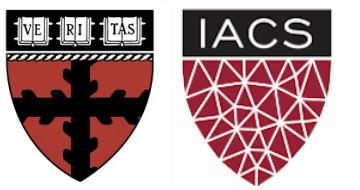
- Bootstrap aggregating

- Sample with replacement from your data set
- Learn a classifier for each bootstrap sample
- Average the results



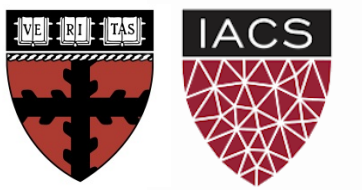
Bagging Example





Bagging

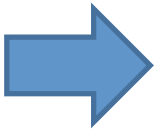
- Reduces overfitting (variance)
- Normally uses one type of classifier
- Decision trees are popular
- Easy to parallelize



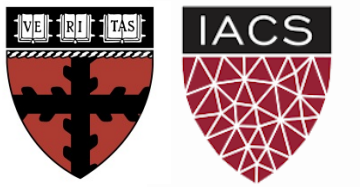
Boosting

- Also ensemble method like Bagging
- But:
  - weak learners evolve over time
  - votes are weighted

- Better than Bagging for many applications



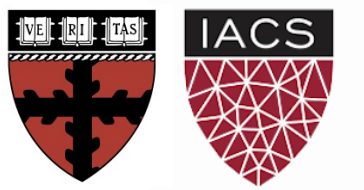
Very popular method



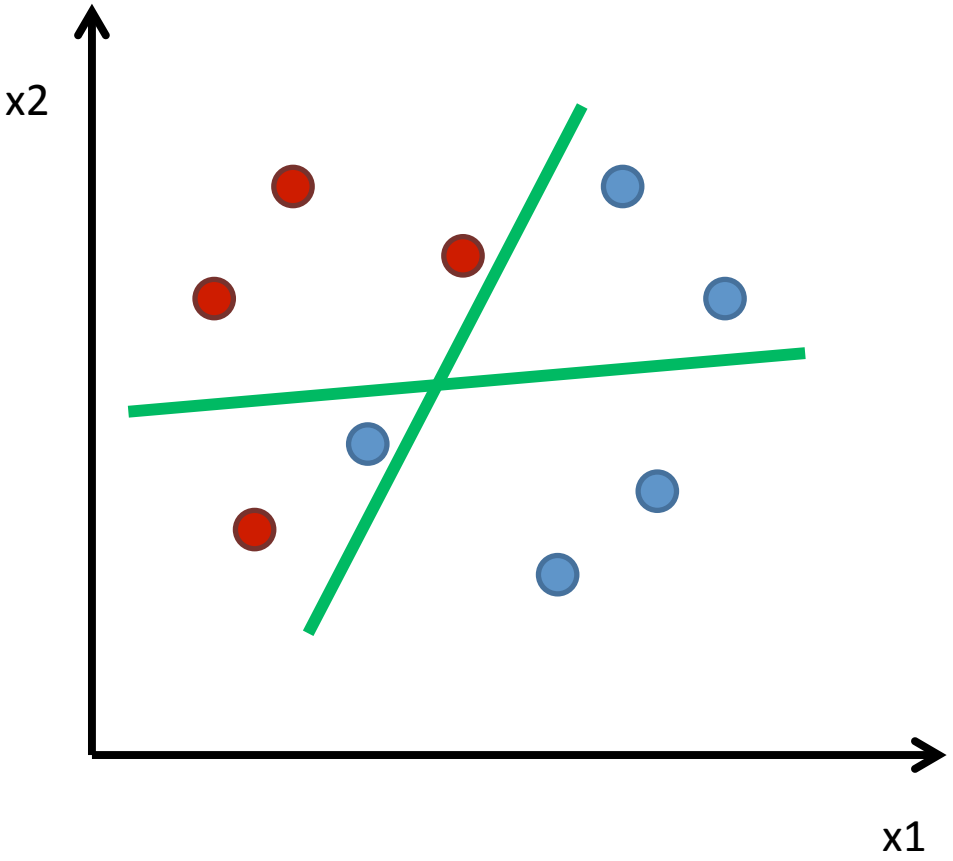
Boosting

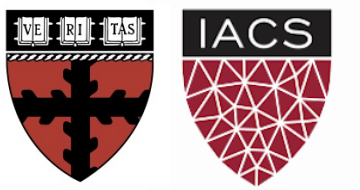
"Boosting is one of the most powerful learning ideas introduced in the last twenty years."

Hastie et al.,"The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer (2009)



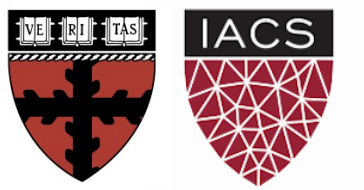
Adaboost



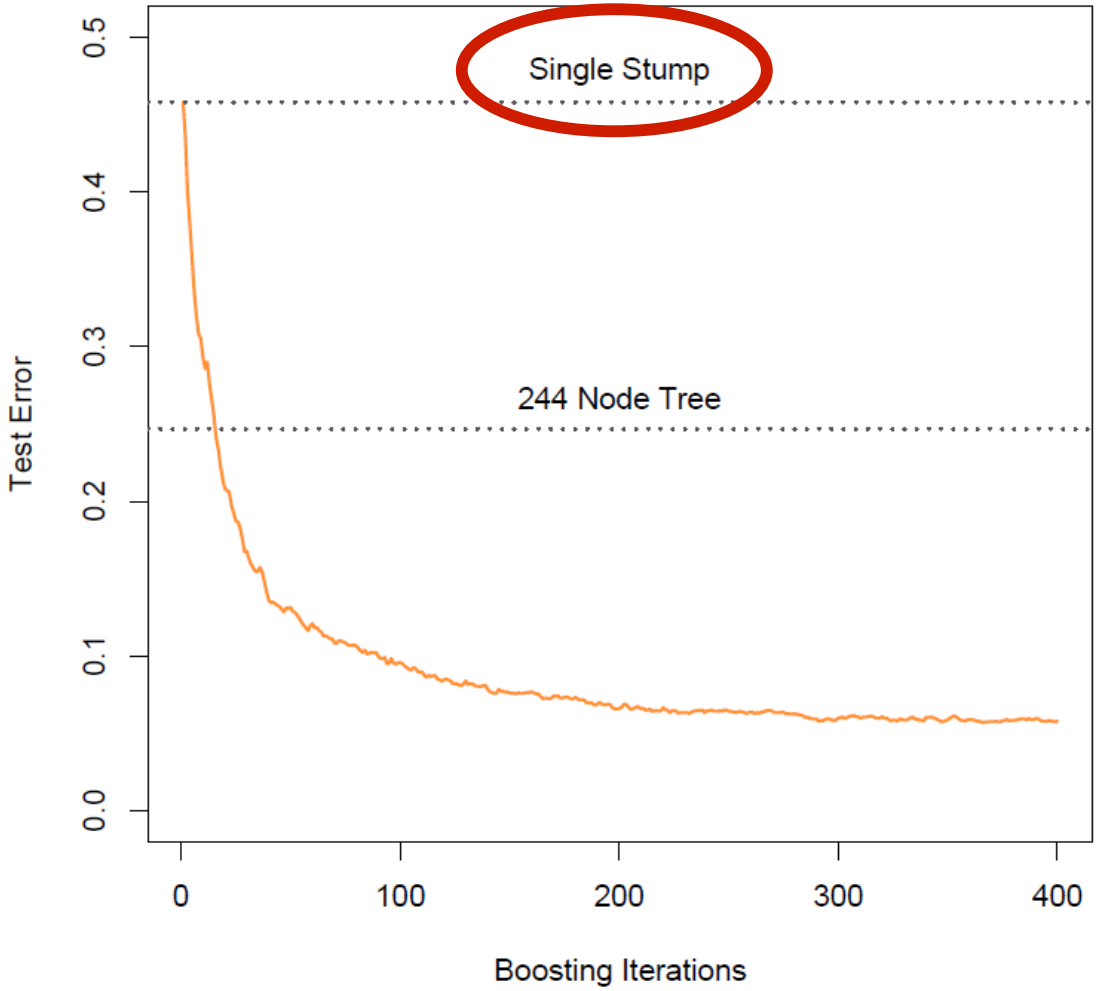


AdaBoost

- Initialize weights for data points
- For each iteration:
  - Fit classifier to training data
  - Compute weighted classification error
  - Compute weight for classifier from the error
  - Update weights for data points
- Final classifier is weighted sum of all single classifiers



AdaBoost



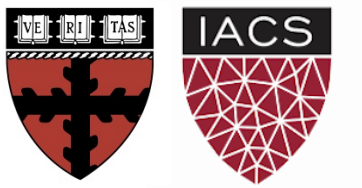
Hastie et al.,"The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer (2009)

AdaBoost

AdaBoost in Action

Kai O. Arras
Social Robotics Lab, University of Freiburg

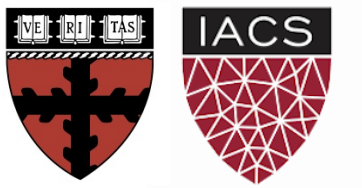
Nov 2009 Social Robotics Laboratory



AdaBoost

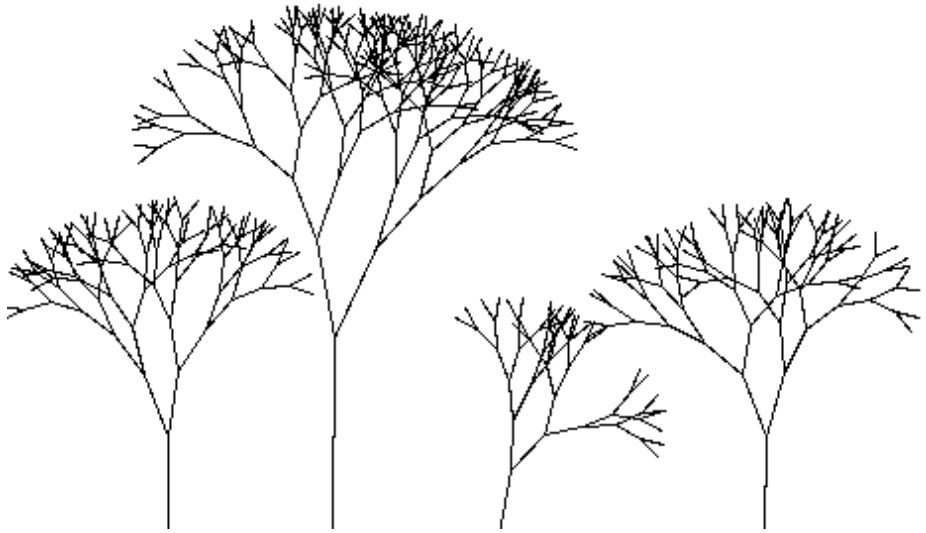
- Introduced by Freund and Schapire in 1995
- Worked great, nobody understood why!

- Then five years later (Friedman et al. 2000):
  - Adaboost minimizes exponential loss function.
- There still are open questions.



Random Forest

- Builds upon the idea of bagging
- Each tree build from bootstrap sample
- Node splits calculated from random feature subsets

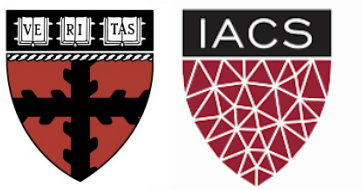


http://www.andrewbuntine.com/articles/about/fun

Random Forest

- All trees are fully grown
- No pruning

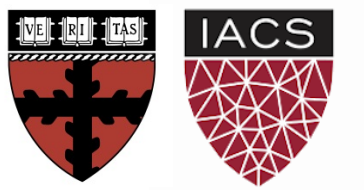
- Two parameters
  - Number of trees
  - Number of features



Random Forest Error Rate

- Error depends on:
  - Correlation between trees (higher is worse)
  - Strength of single trees (higher is better)

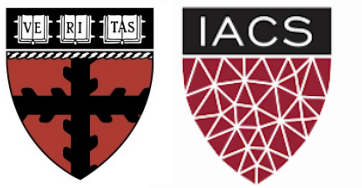
- Increasing number of features for each split:
  - Increases correlation
  - Increases strength of single trees



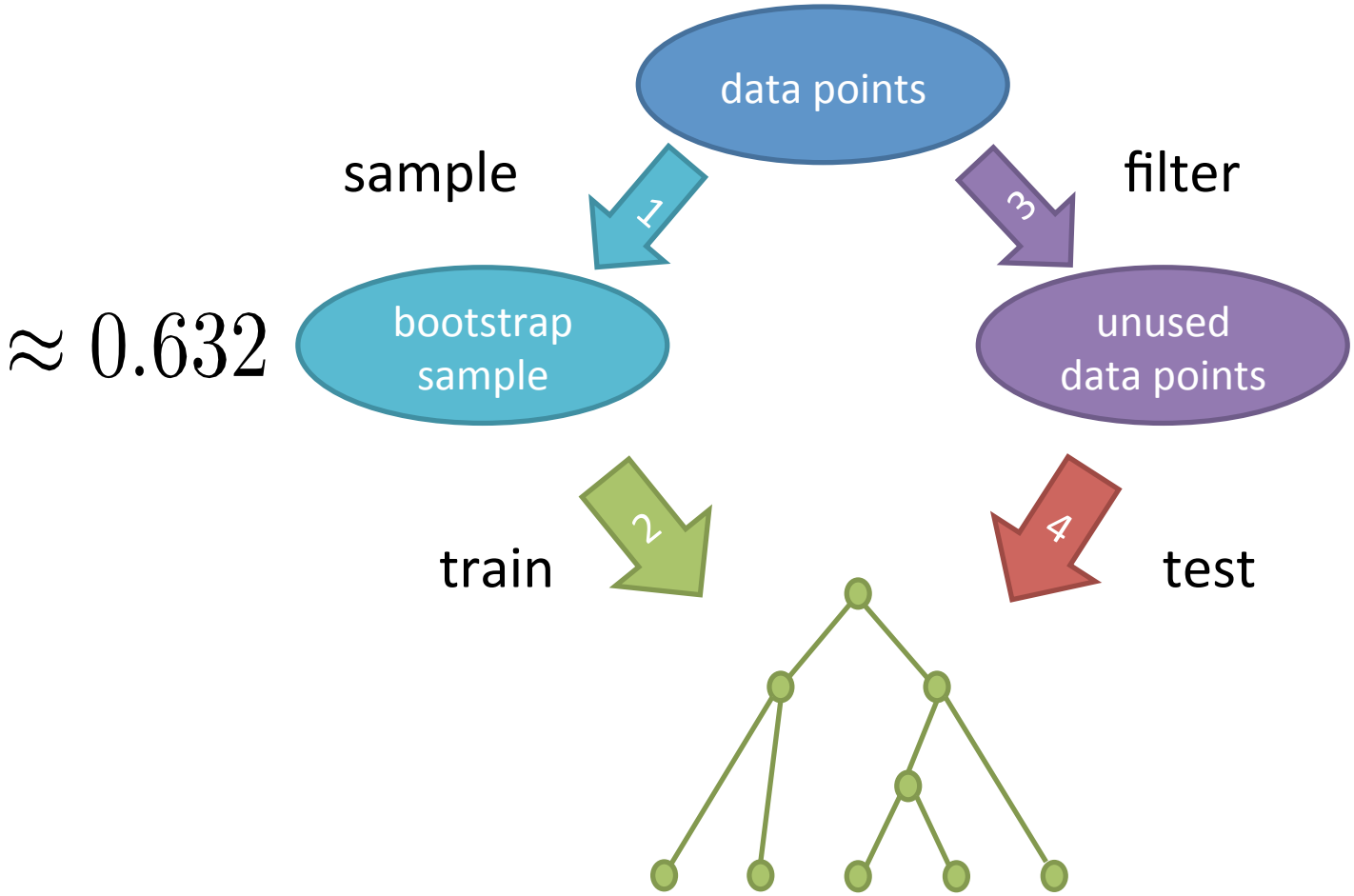
Out of Bag Error

- Each tree is trained on a bootstrapped sample
- About 1/3 of data points not used for training

- Predict unseen points with each tree
- Measure error



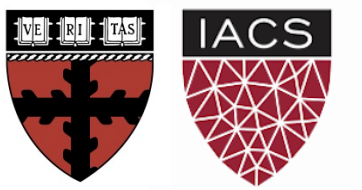
Out of Bag Error



≈ 0.632

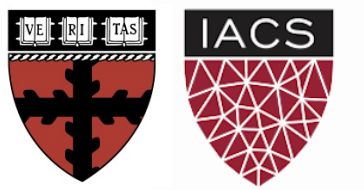
Out of Bag Error

- Very similar to cross-validation
- Measured during training
- Can be too optimistic



Variable Importance

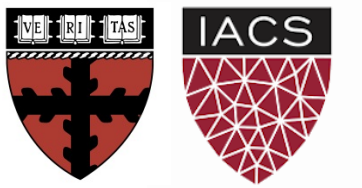
- Again use out of bag samples
- Predict class for these samples
- Randomly permute values of one feature
- Predict classes again
- Measure decrease in accuracy



Tempting Scenario

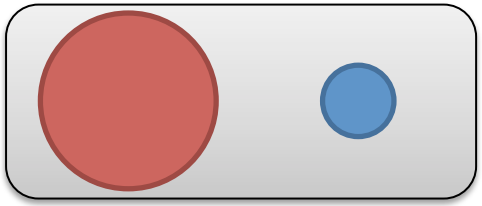
- Run random forest with all features
- Reduce number of features based on importance weights
- Run again with reduced feature set and report out of bag error

This does not measure test performance!

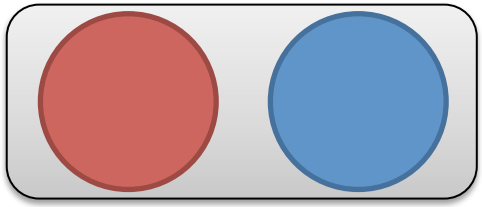


Unbalanced Classes

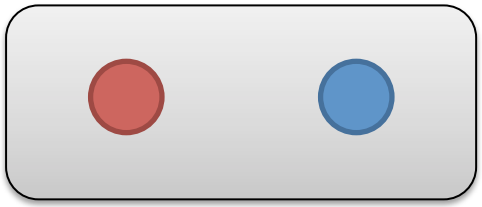
- The Problem:



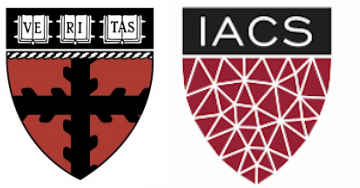
- Oversample:



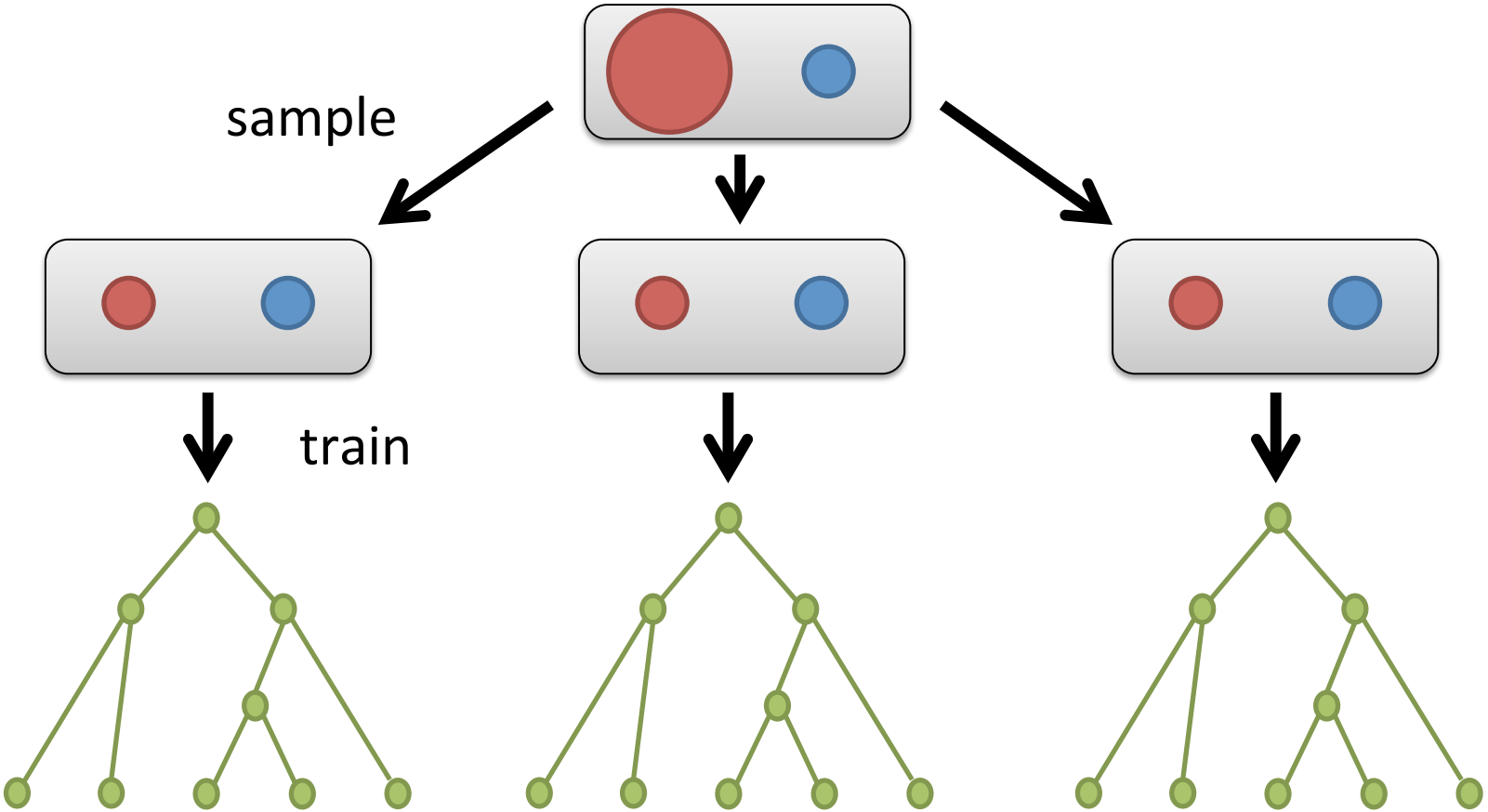
- Subsample:

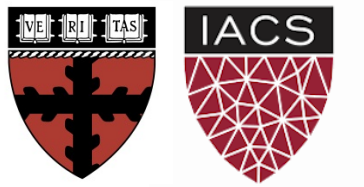


- Subsample for each tree!



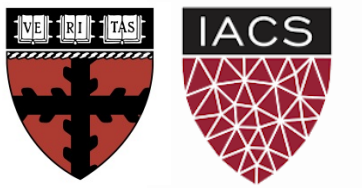
Random Forest Subsampling





Random Forest

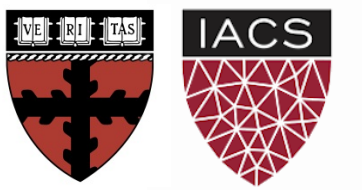
- Similar to Bagging
- Easy to parallelize
- Packaged with some neat functions:
  - Out of bag error
  - Feature importance measure
  - Proximity estimation



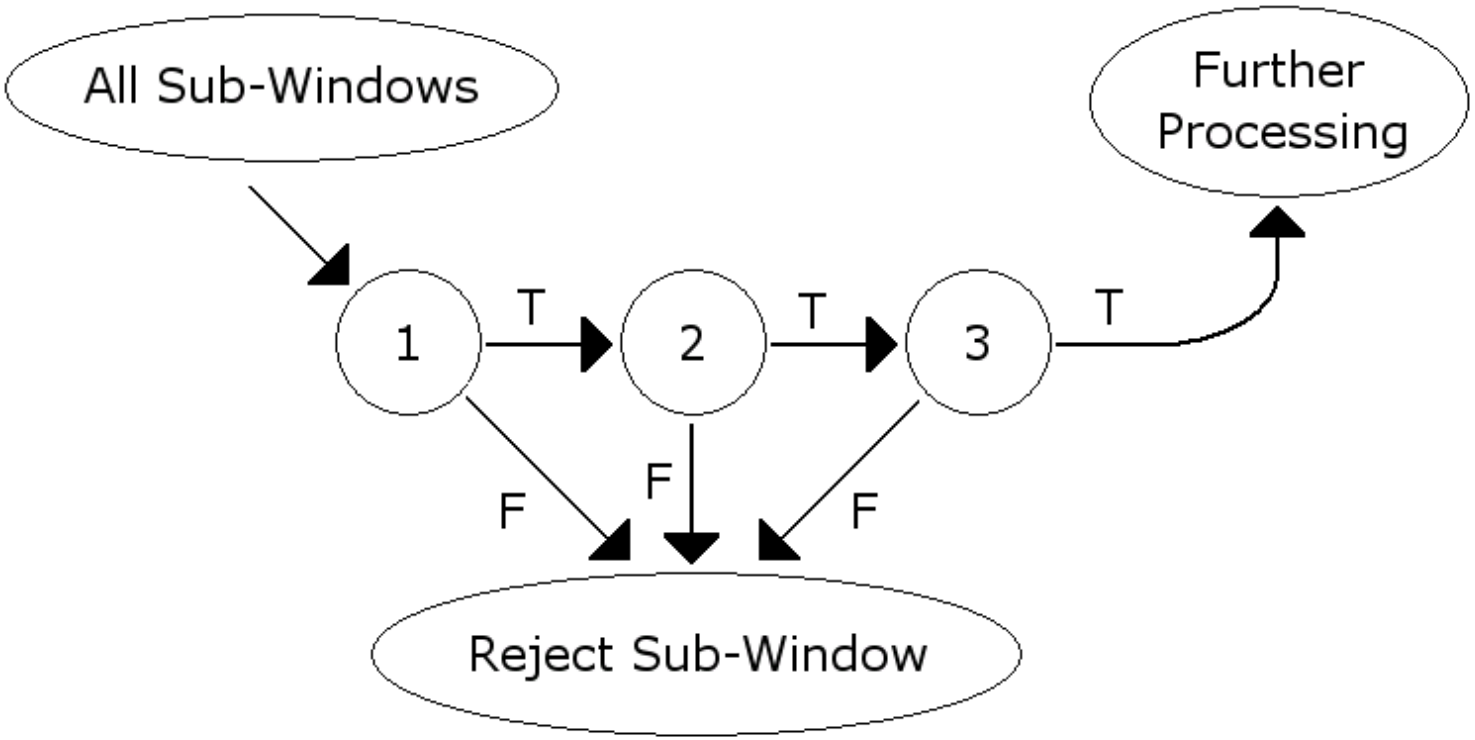
Cascade Classifier

- Ensemble methods are strong
- But prediction is slow
- Solution: Make prediction faster

Idea: Build a cascade



Cascade Classifier



http://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

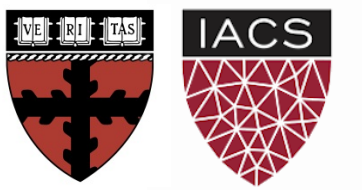
Viola Jones Face Detection

http://cvdazzle.com/

Viola Jones Face Detection

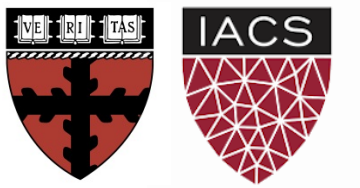
- Takes long to train
- Prediction in real time!

- Widely used today

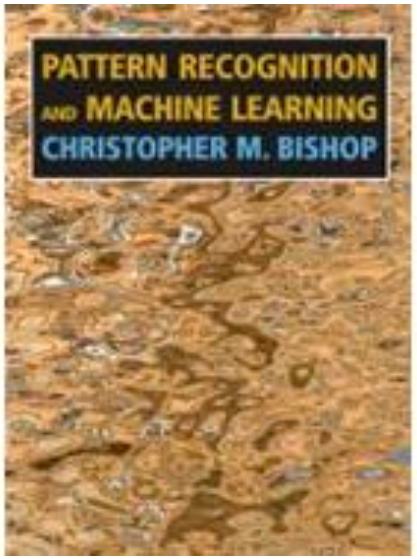


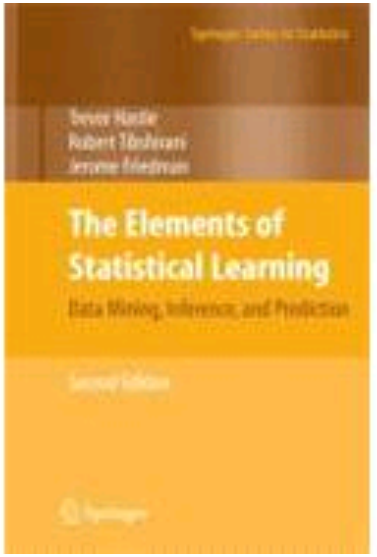
Summary

- SVMs
- Decision Trees
- Bootstrap, Bagging, Boosting
- Random Forest
- Cascade Classifier



Further Reading

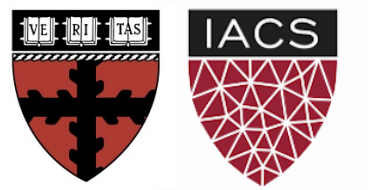




Error Measures

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

		predicted 1	predicted -1
true 1		tp	fn
true -1		fp	tn



TPR and FPR

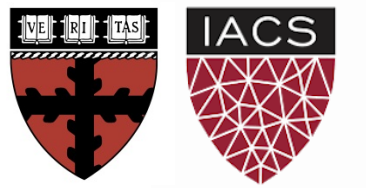
- True Positive Rate:

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

- False Positive Rate:

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

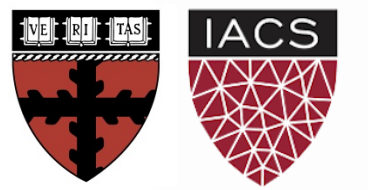
		predicted 1	predicted -1
true	1	tp	fn
	-1	fp	tn



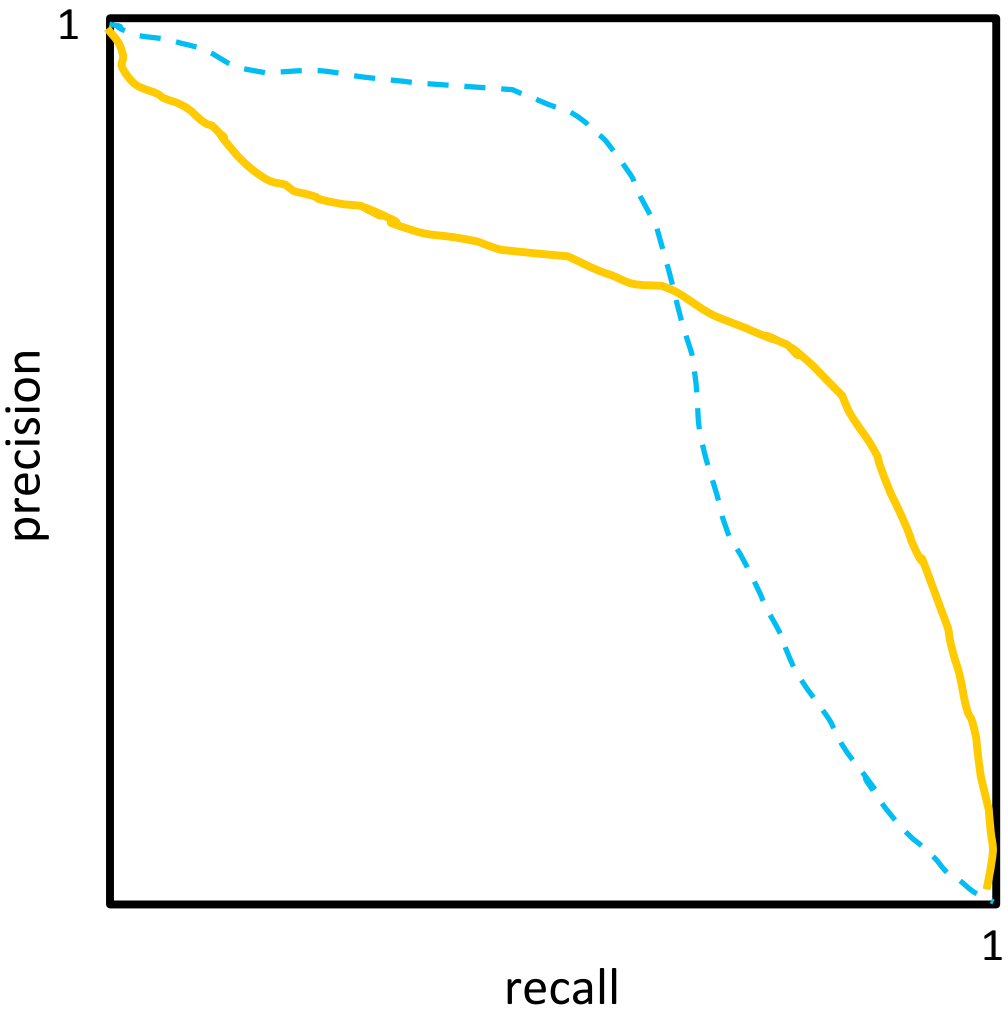
Precision Recall

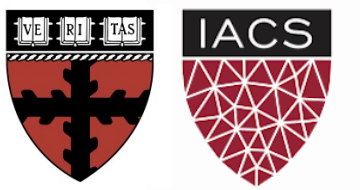
- Recall: $\frac{tp}{tp+fn}$
- Precision: $\frac{tp}{tp+fp}$

true \ predicted	1	-1
1	tp	fn
-1	fp	tn

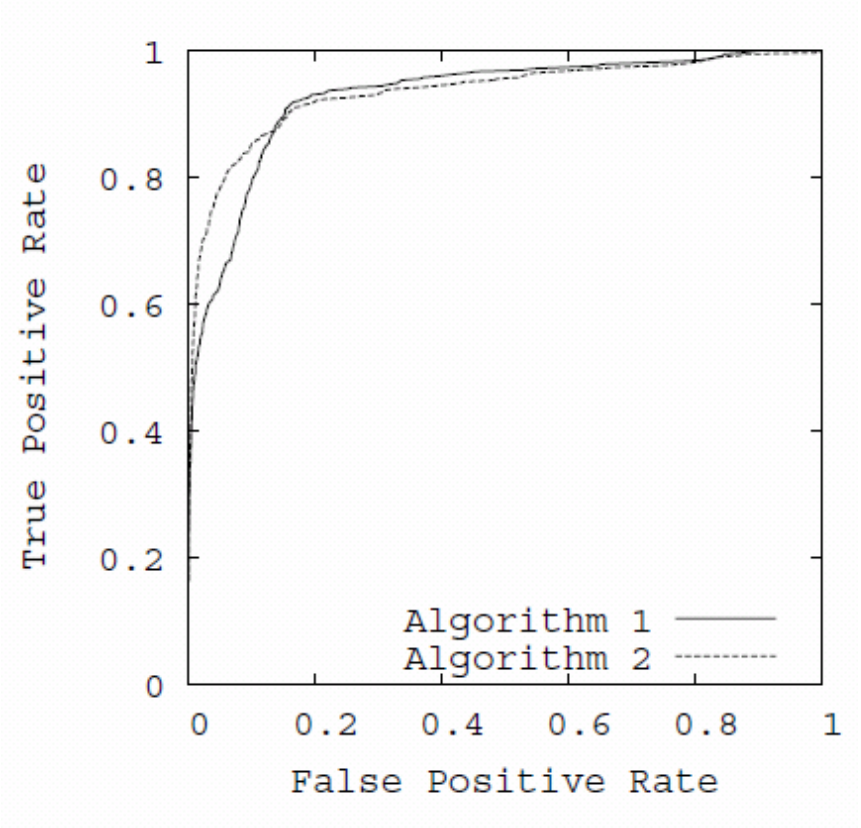


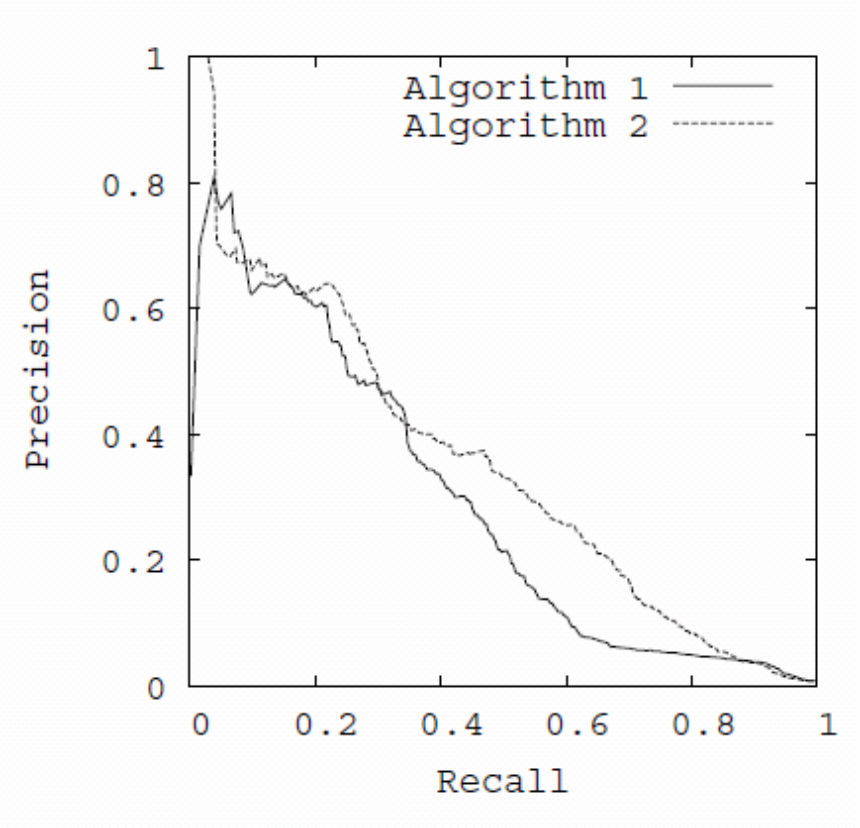
Precision Recall Curve





Comparison





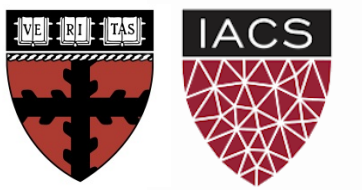
J. Davis & M. Goadrich,
"The Relationship Between Precision-Recall and ROC Curves.",
ICML (2006)

F-measure

- Weighted average of precision and recall

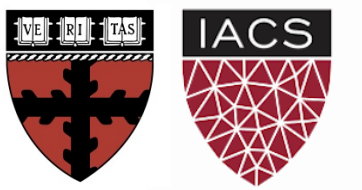
$$F_\beta = \frac{(\beta^2+1)\cdot P\cdot R}{\beta^2\cdot P+R}$$

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



Clustering Evaluation Criteria

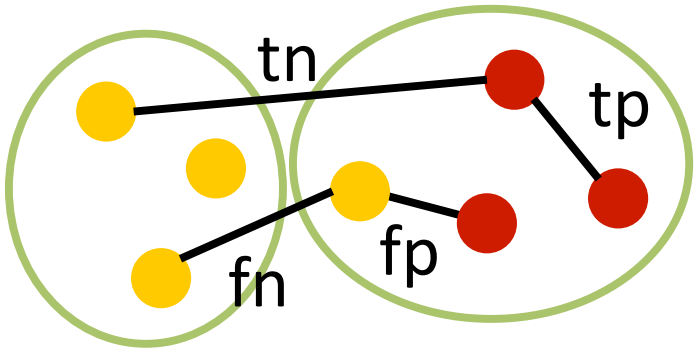
- Based on expert knowledge
- Debatable for real data
- Hidden Unknown structures could be present
- Do we even want to just reproduce known structure?



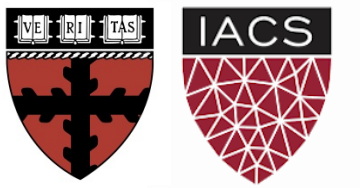
Rand Index

- Percentage of correct classifications
- Compare pairs of elements:

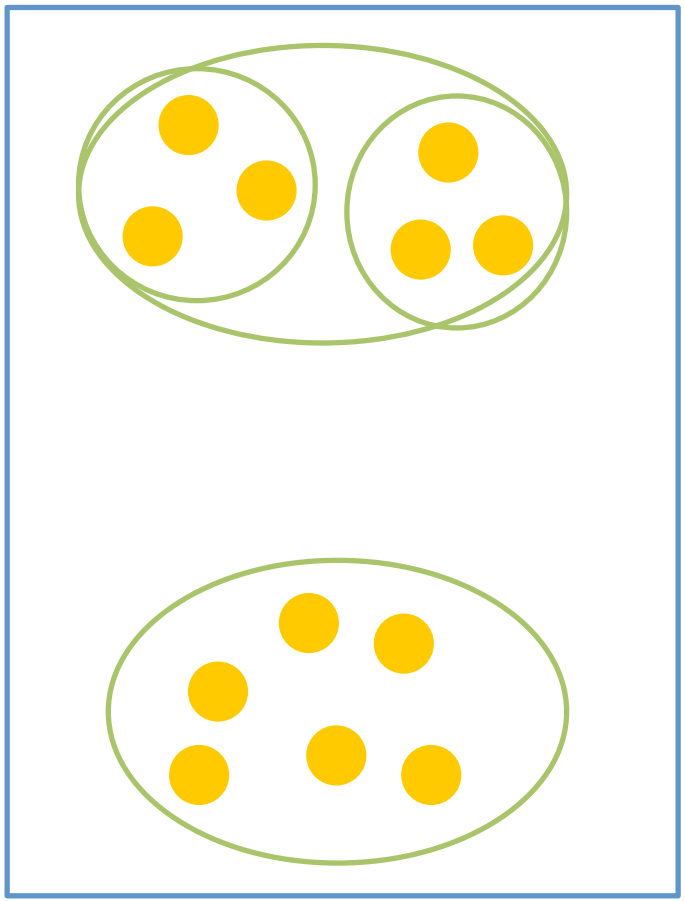
$$R = \frac{tp+tn}{tp+tn+fp+fn}$$

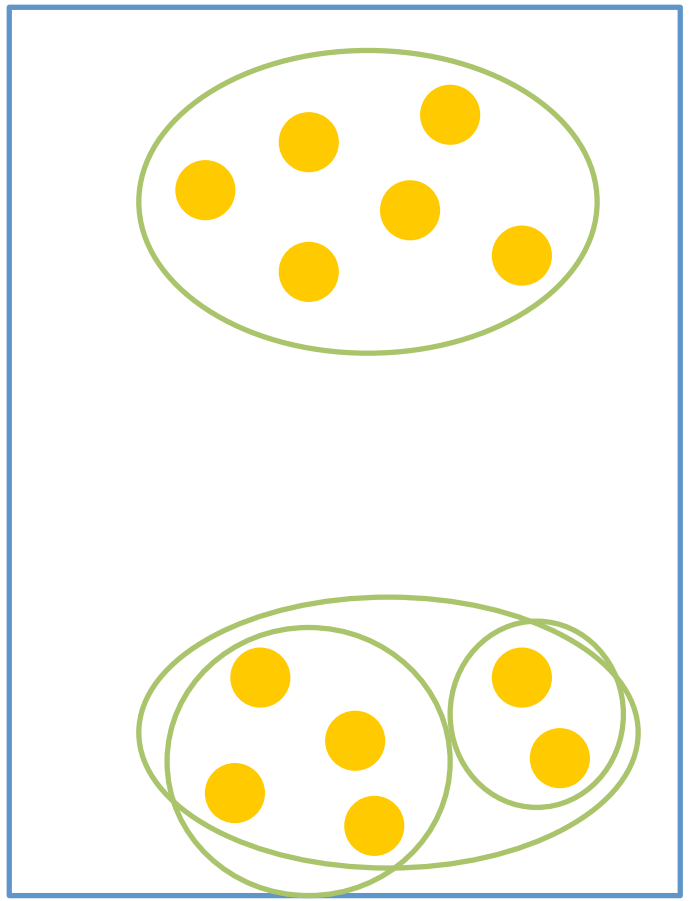


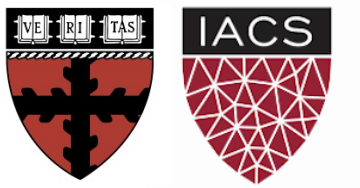
- Fp and fn are equally weighted



Stability



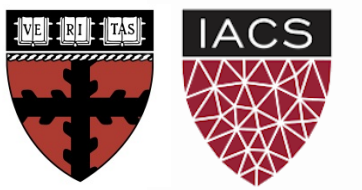




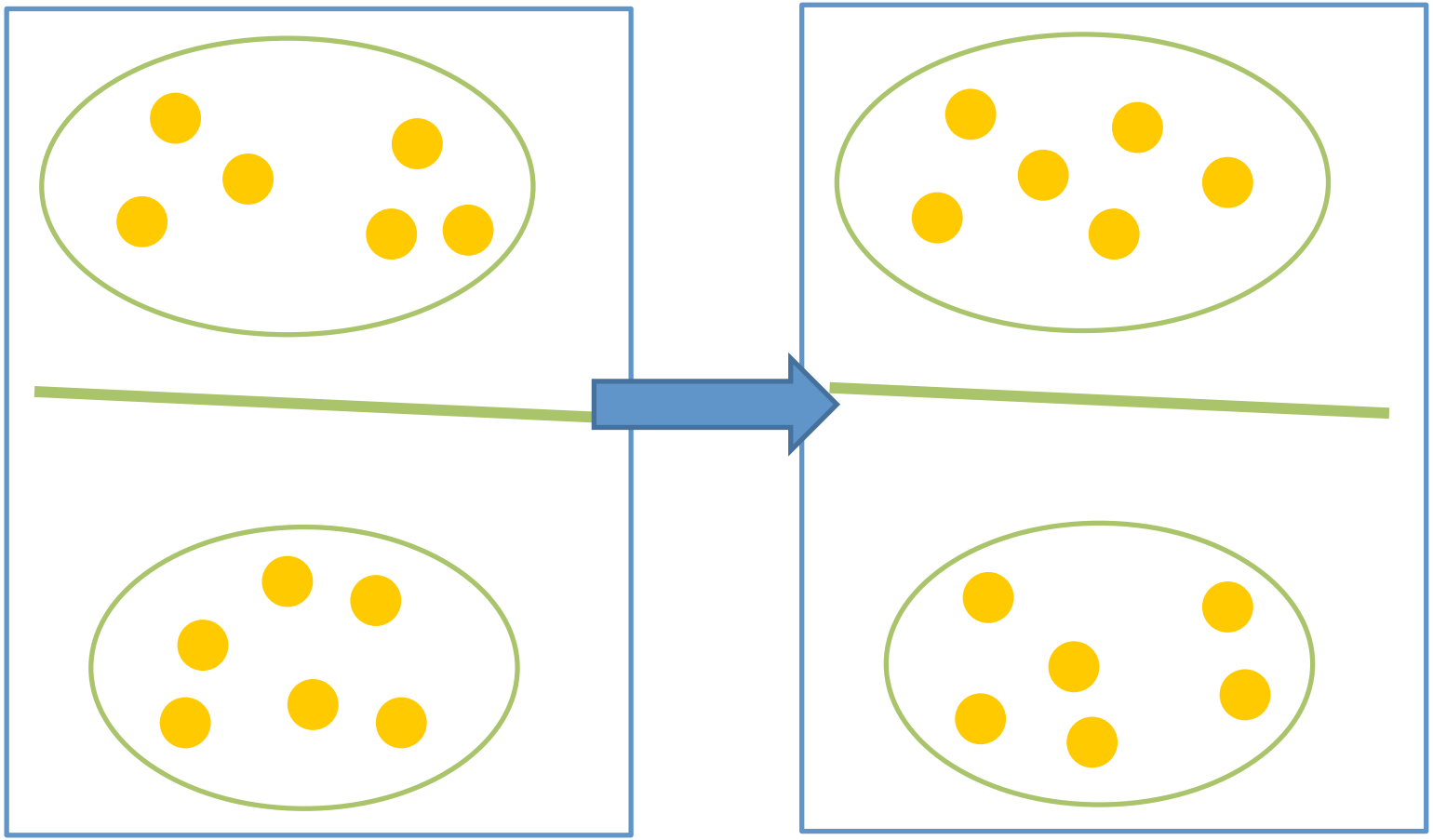
Stability

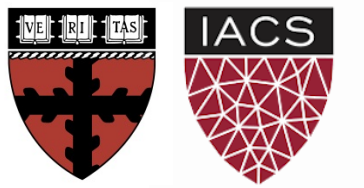
- What is the right number of clusters?
- What makes a good clustering solution?

- Clustering should generalize!



Stability





Gini Impurity

Example:

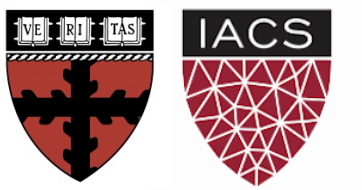
4 **red**, 3 **green**, 3 **blue** data points

- random sample:
  - red: 4/10 green: 3/10 blue: 3/10

- misclassification:
  - red: 4/10 * (3/10 + 3/10)

true class

wrong prediction



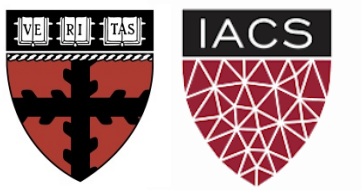
Gini Impurity

- Number of classes: C
- Number of data points: N
- Number of data points of class i: N_i

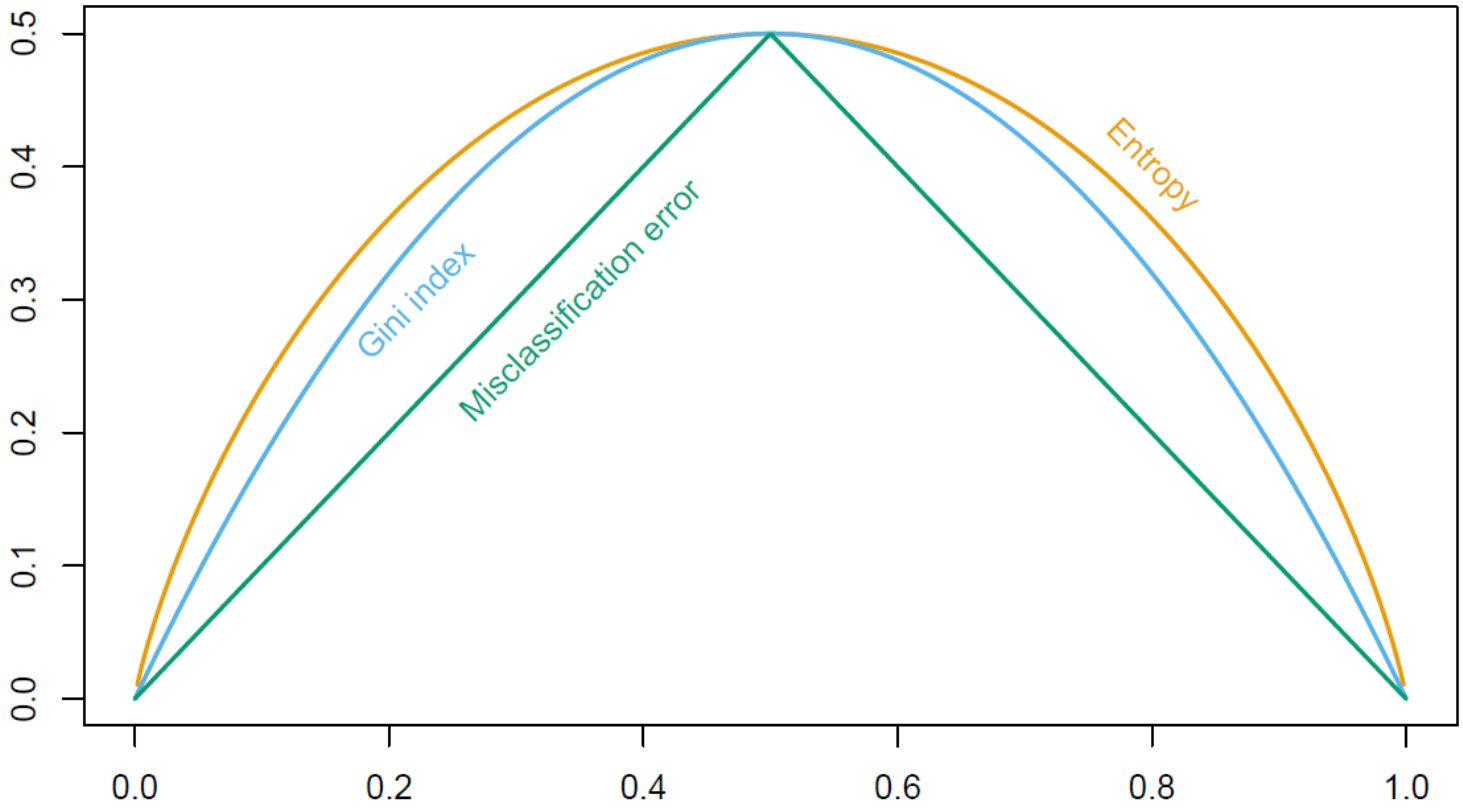
$$I_G = \sum_{i=1}^{C} \frac{N_i}{N}\left(1 - \frac{N_i}{N}\right)$$

true class

wrong prediction



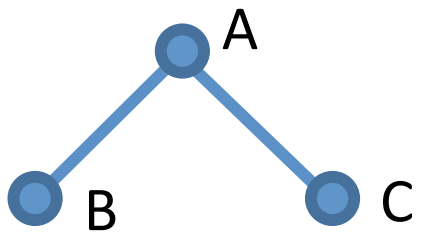
Gini Impurity



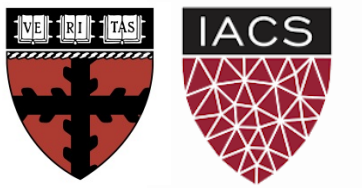
Hastie et al.,"The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer (2009)

Node Purity Gain

- Compare:
  - Gini impurity of parent node
  - Gini impurity of child nodes

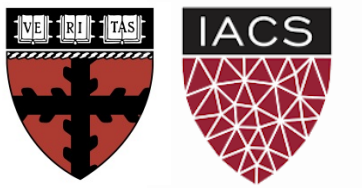


$$\Delta I_G = I_G(A) - \frac{N(B)}{N(A)} I_G(B) + \frac{N(C)}{N(A)} I_G(C)$$



Pseudocode

- Check for base cases
- For each attribute *a*
  - Calculate the gain from splitting on *a*
- Let *a_best* be the attribute with highest gain
- Create a decision *node* that splits on *a_best*
- Repeat on the sub-nodes



http://en.wikipedia.org/wiki/C4.5_algorithm