(Almost) All of Machine Learning

Rayid Ghani

THE UNIVERSITY OF CHICAGO

Slides liberally borrowed and customized from lots of excellent online sources

What we'll cover

- What is Machine Learning?
- Examples in the real world
- How to solve problems using ML?
- How to evaluate ML methods?
- Overview of ML Methods
- Doing ML with Python and sklearn

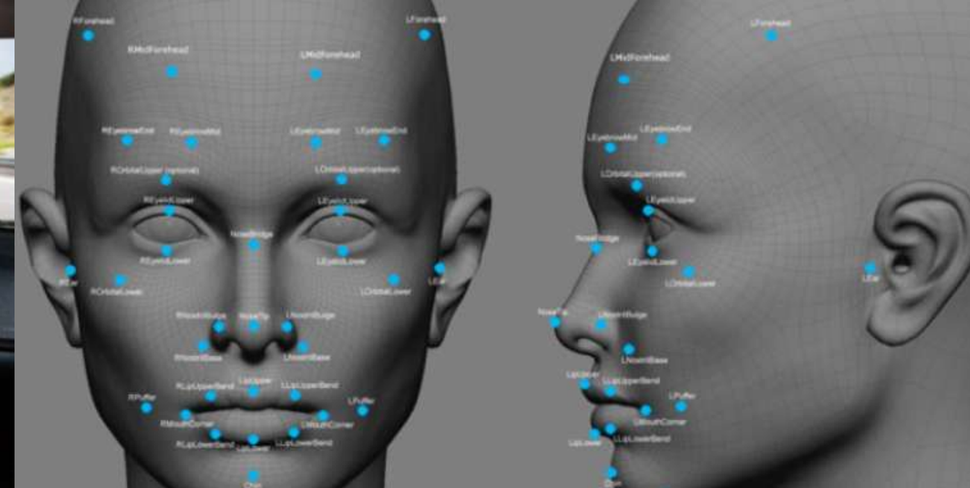
After these sessions, you should be able to

- Formulate a policy or social science problem as a machine learning problem

- Understand, Use, and Evaluate Machine Learning methods

Machine Learning

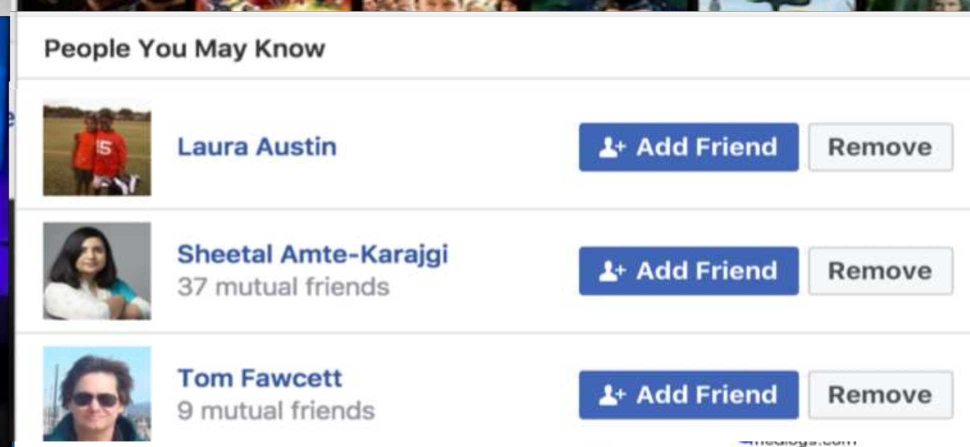
"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."




fraud


NETFLIX
Browse
Just for Kids
Continue Watching
Rango
Melancholia
Lilyhammer
Top 10 for you
Breaking Bad
DAMAGES
MAD MEN
ABDUCTION
Instant Queue
LILYHAMMER
CRAZY OUTSIDE


THINK
$4,800
$36,681
$10,400


People You May Know
Laura Austin
Add Friend
Remove
Sheetal Amte-Karajgi
37 mutual friends
Add Friend
Remove
Tom Fawcett
9 mutual friends
Add Friend
Remove

Why Machine Learning?

- Goal: Adaptive and scalable systems that are cost effective to build and maintain

- Existing Systems are often Rules-Based
  - Rigid
  - Difficult to create and maintain
- ML is more effective
  - Lots of data is available to “train” the system
  - Business experts are better at “training” the system compared to “building” the system

Machine Learning Meets Policy Problems

- Lots of examples at http://dssg.uchicago.edu/projects

Machine Learning Process

- Understand Business problem
- Map to Machine Learning problem
- Get (and integrate) the data
- Explore and Process the data
- Create "Features" (variables/covariates/predictors/independent variables)
- Select methods to try
- Evaluate methods
- Deploy, Maintain, Update

Types of ML tasks for Policy Problems

- Description (Understand the past)
- Detection (Anomalies, Events, Patterns)
- Prediction (Predict the Future)
- Behavior Change (Causal Inference)

Types of Learning

Unsupervised

Supervised

Clustering
PCA
Association Rules
…

Classification
Regression
…

Unsupervised Learning

- No outcome or dependent variable is present
- Goal is exploration, understanding historical data, finding patterns and/or groups in the data
- Examples
  - Clustering (cluster analysis)
  - Principal Components Analysis
  - Association Rules (beer and diapers)

- We’ll skip the details for now and come back to it

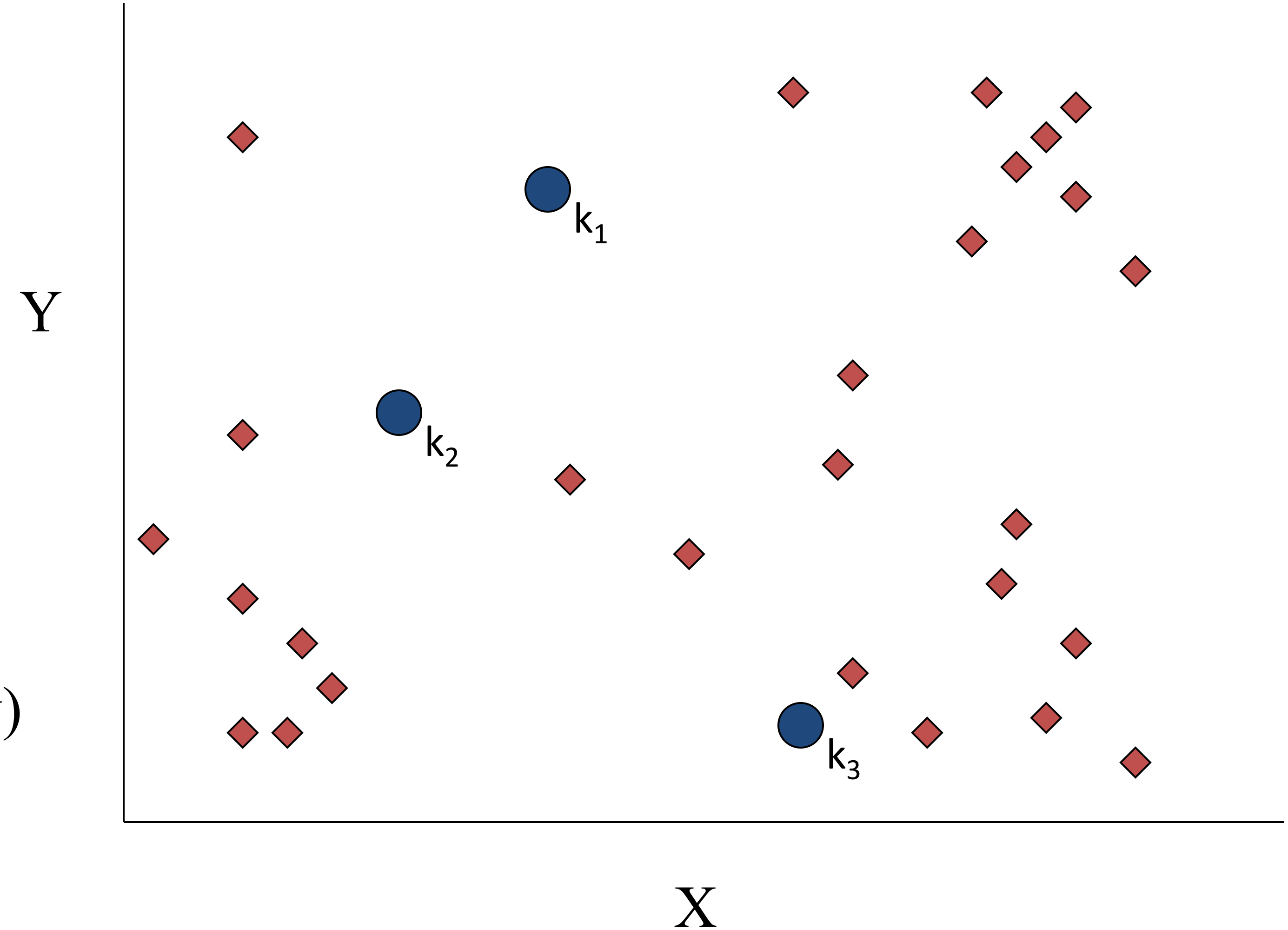
Clustering

- A good clustering method will produce clusters with
  - High intra-cluster similarity
  - Low inter-cluster similarity

- K-Means is the simplest and the most common algorithm

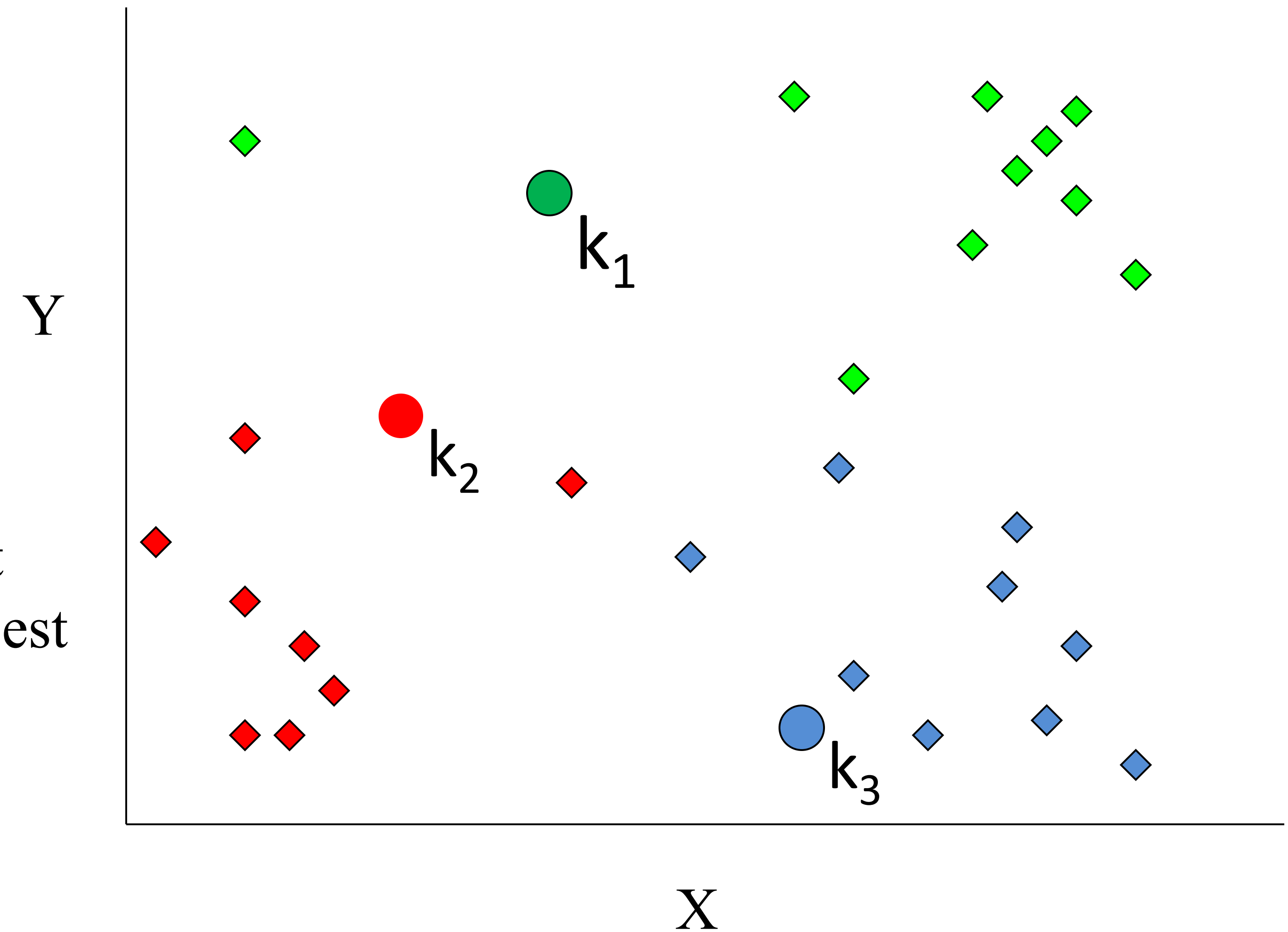
K-means example, step 1

Pick 3 initial cluster centers (randomly)

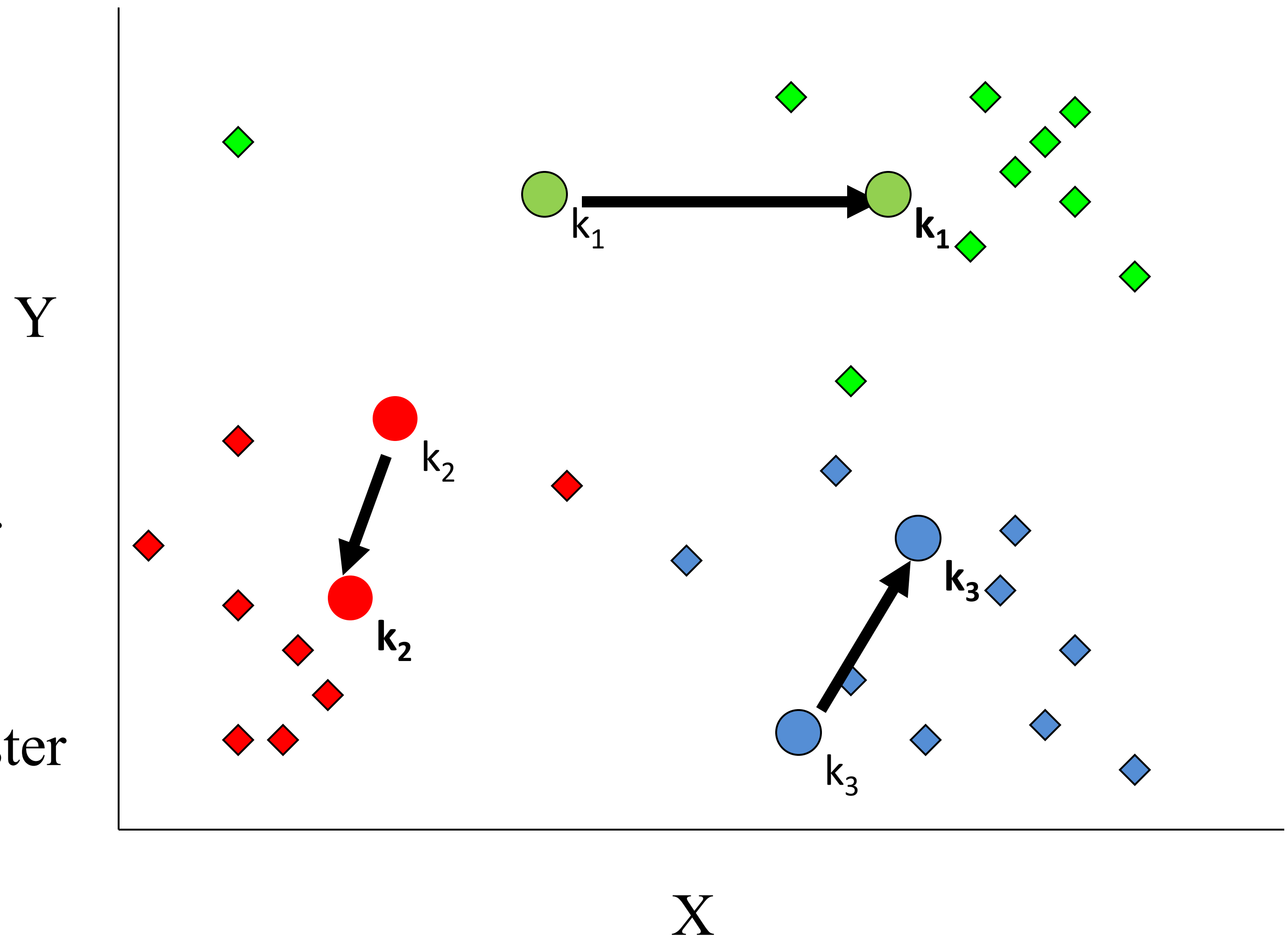


K-means example, step 2

Assign each point to the closest cluster center



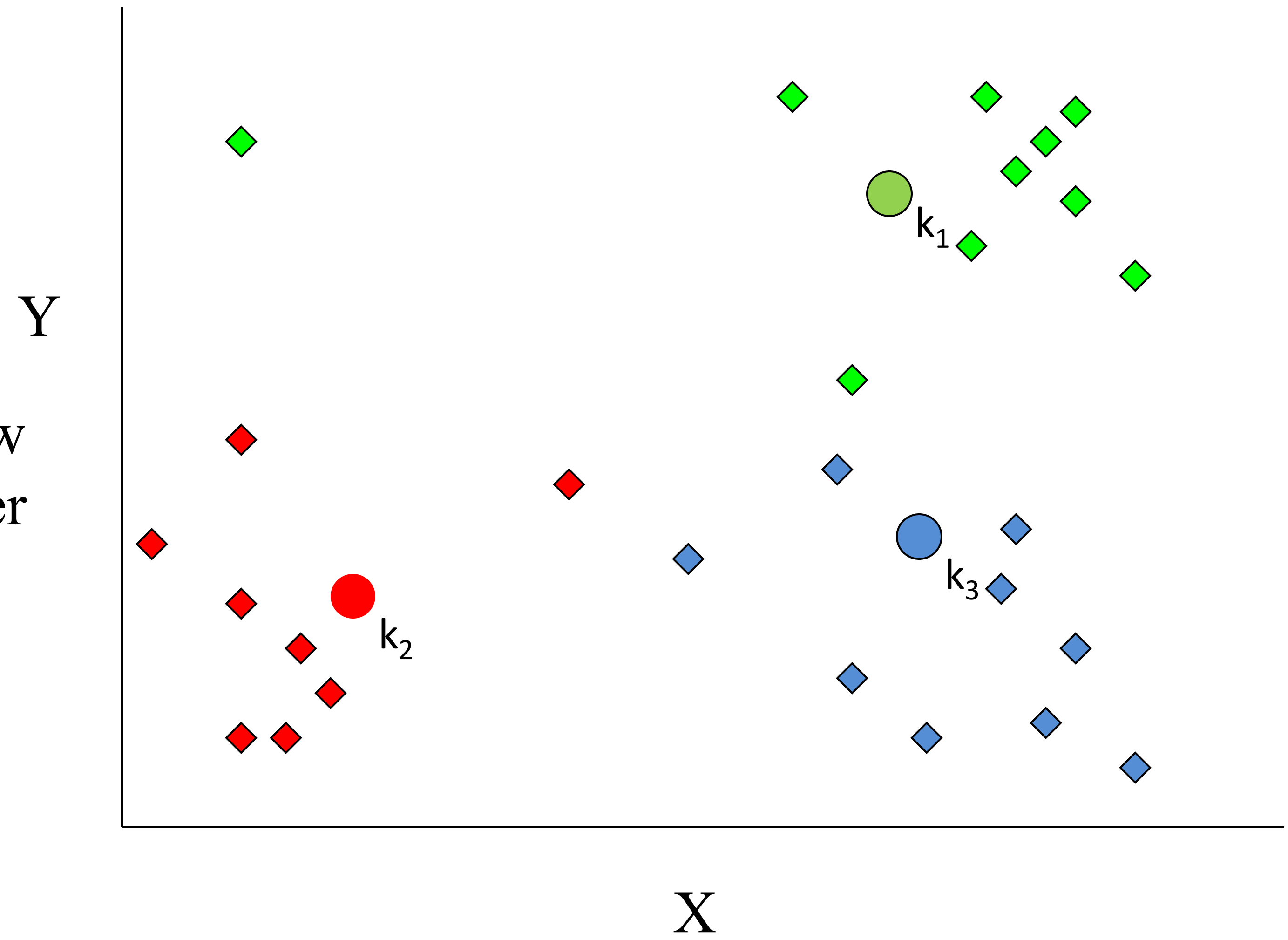
K-means example, step 3



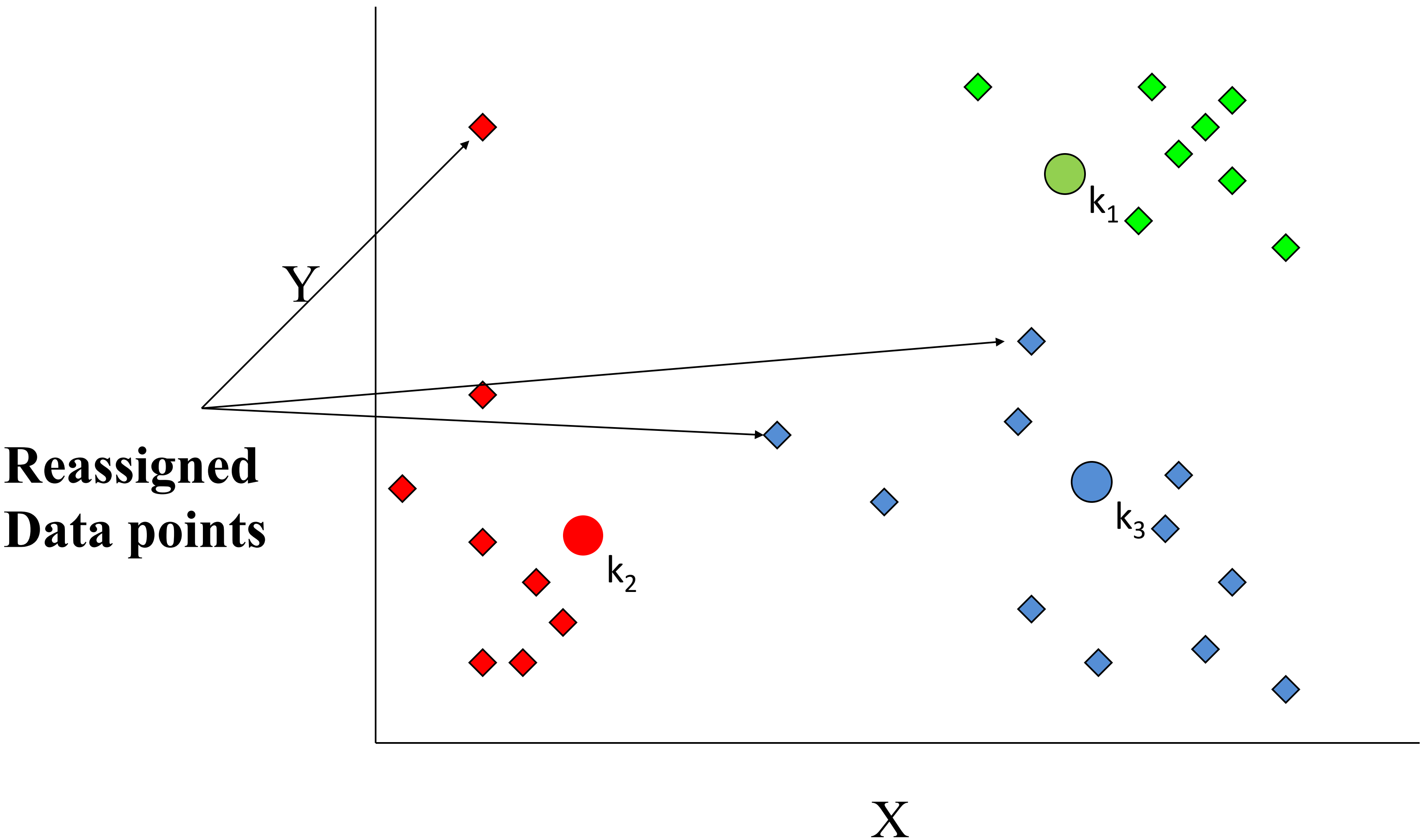
Move
each cluster
center
to the mean
of each cluster

K-means example, step 4

Reassign points closest to a different new cluster center

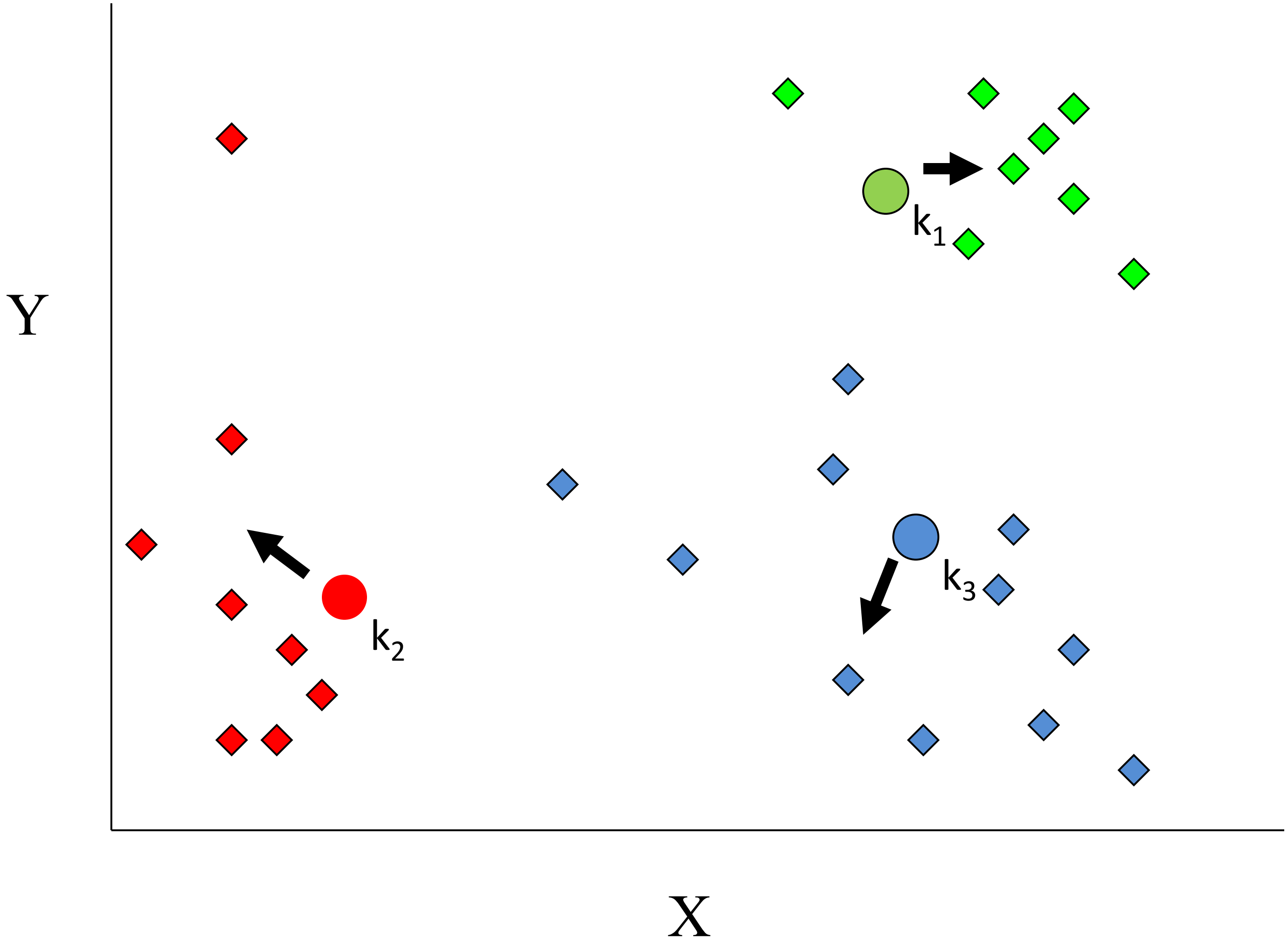


K-means example, step 4 …

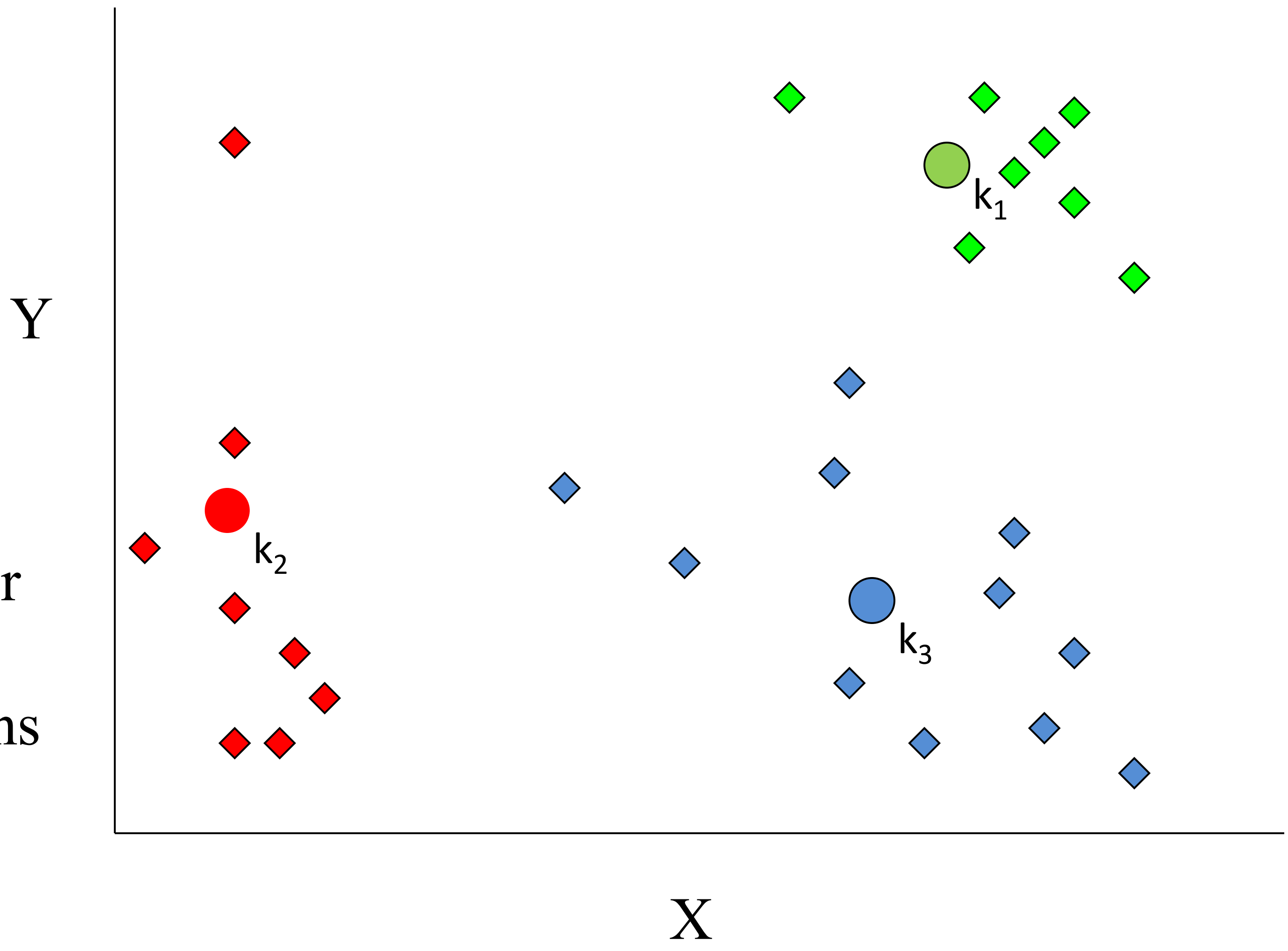


K-means example, step 4b

re-compute cluster means



K-means example, step 5



move cluster centers to cluster means

K-means algorithm

- Given *k*, the *k-means* algorithm works as follows:
  1. Randomly choose *k* data points (seeds) to be the initial centroids (cluster centers)
  2. Assign each data point to the closest centroid
  3. Re-compute the centroids using the current cluster memberships.
  4. If a convergence criterion is not met, go to 2).

More Clustering Methods

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

As we go down this chart, the clustering strategies have more tendency to transitively group points even if they are not nearby in feature space

Exploring and Understanding clusters

- Visualizing using PCA
- Doing descriptive stats for each cluster
- Use decision trees with cluster ids as labels to generate rules describing each cluster

Evaluating Clustering Methods

- Objective Evaluation
- Task-Specific Evaluation
- Same data can be clustered in different ways in different number of clusters

Evaluation

Chicago

Cleveland

New York

Minneapolis

Charlotte

London

Tokyo

Rio de Janeiro

Mexico City

Unsupervised Learning: Recap

- Used to understand patterns and groups in data

- Interactive (multivariate) data exploration

Supervised learning framework

$$y = f(x)$$

output

"Learned" function

Features/Predictors

future/generalization

Training or Learning: Find an f that minimizes error in recovering y

Slide credit: L. Lazebnik

Supervised learning framework

$$y = f(x)$$

output

"Learned" function

Features/Predictors

- **Training:** Given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function f that minimizes future generalization (out of sample) error
- **Scoring/Testing:** apply f to a new *example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

Slide credit: L. Lazebnik

Typical Problems

- Prediction (time dependent)
- Classification (not time dependent)

Steps

Data Preparation

↓

Construct Features

↓

Build Model

↓

Validate Model

↓

Deploy Model

How to solve a prediction problem

- Define and Create Label (outcome variable)
- Define and Create Features (predictors)
- Create Training and Validation Sets
- Train model(s) on Training Set
- Validate model(s) on Validation Set
- Select “best” model

Validation

- You’ve run a large number of different types of models

- You need to understand what types of models work when, **and**
- You need to decide which one(s) to use in the **future**

What’s required to validate

- Methodology
- Metric(s)
- Comparing to baselines

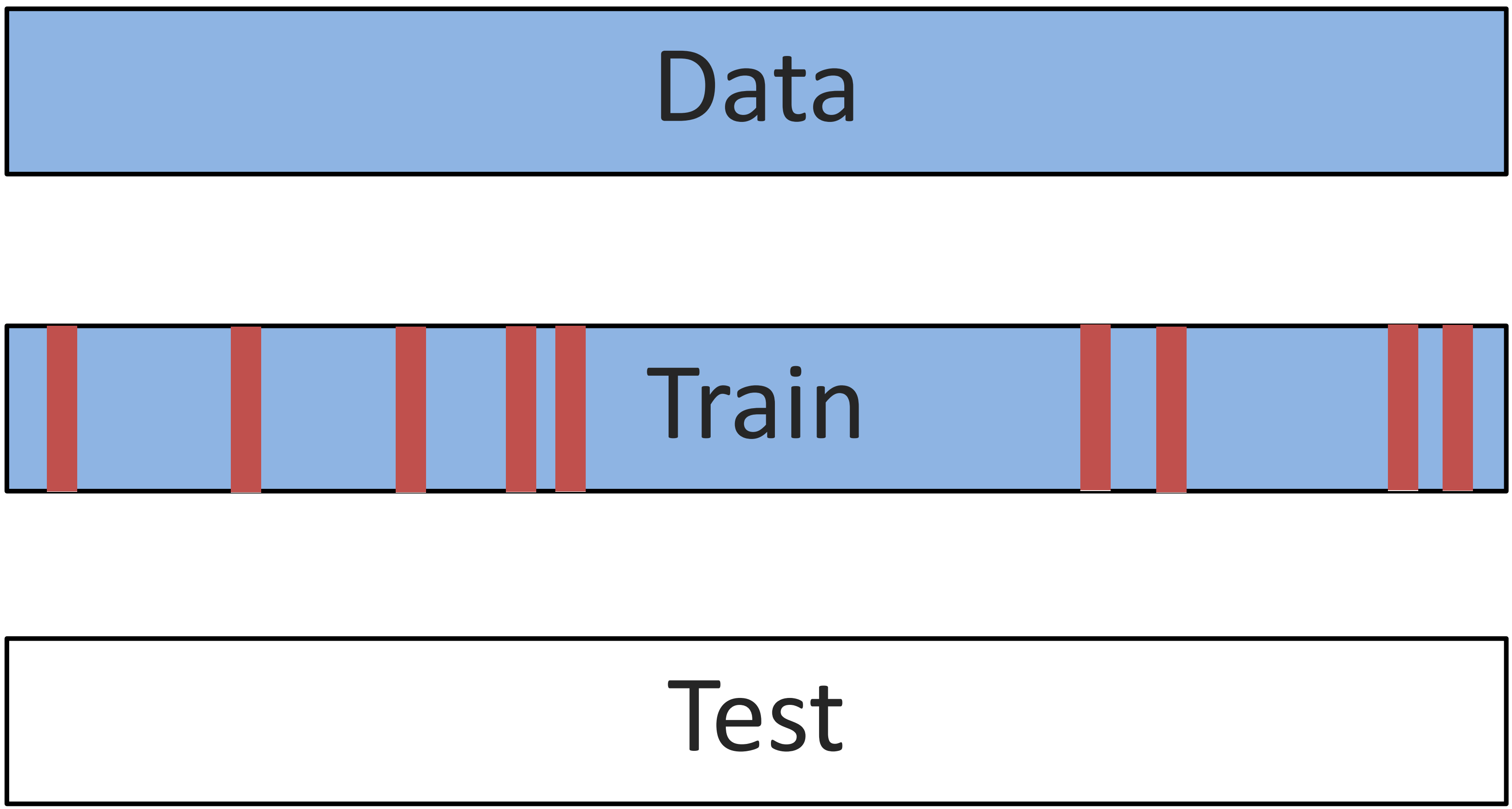
In-sample

Data

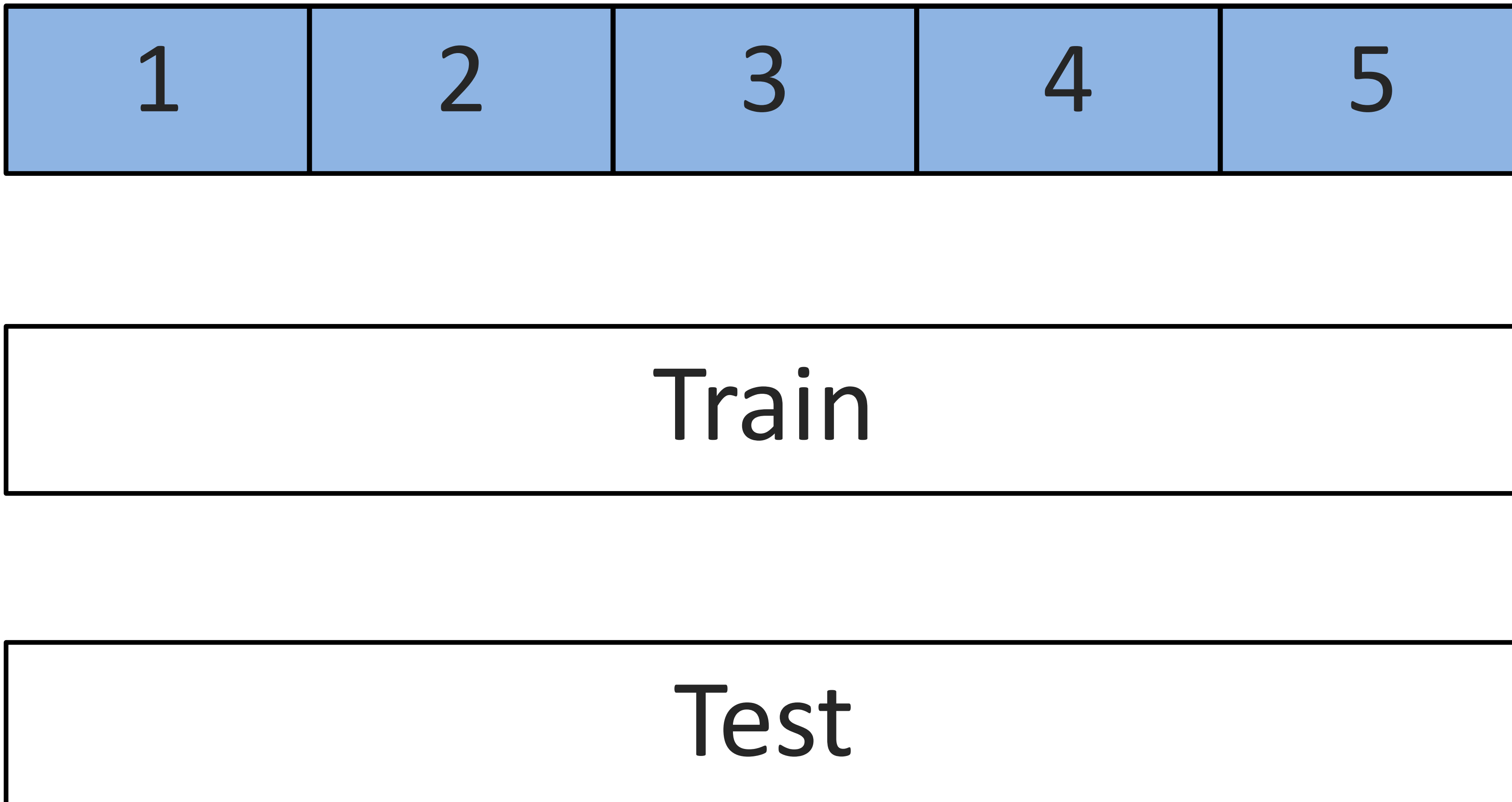
Train

Test

Out-of-Sample



k-fold Cross-Validation



k-fold Cross-Validation

k-fold Cross-Validation

k-fold Cross-Validation

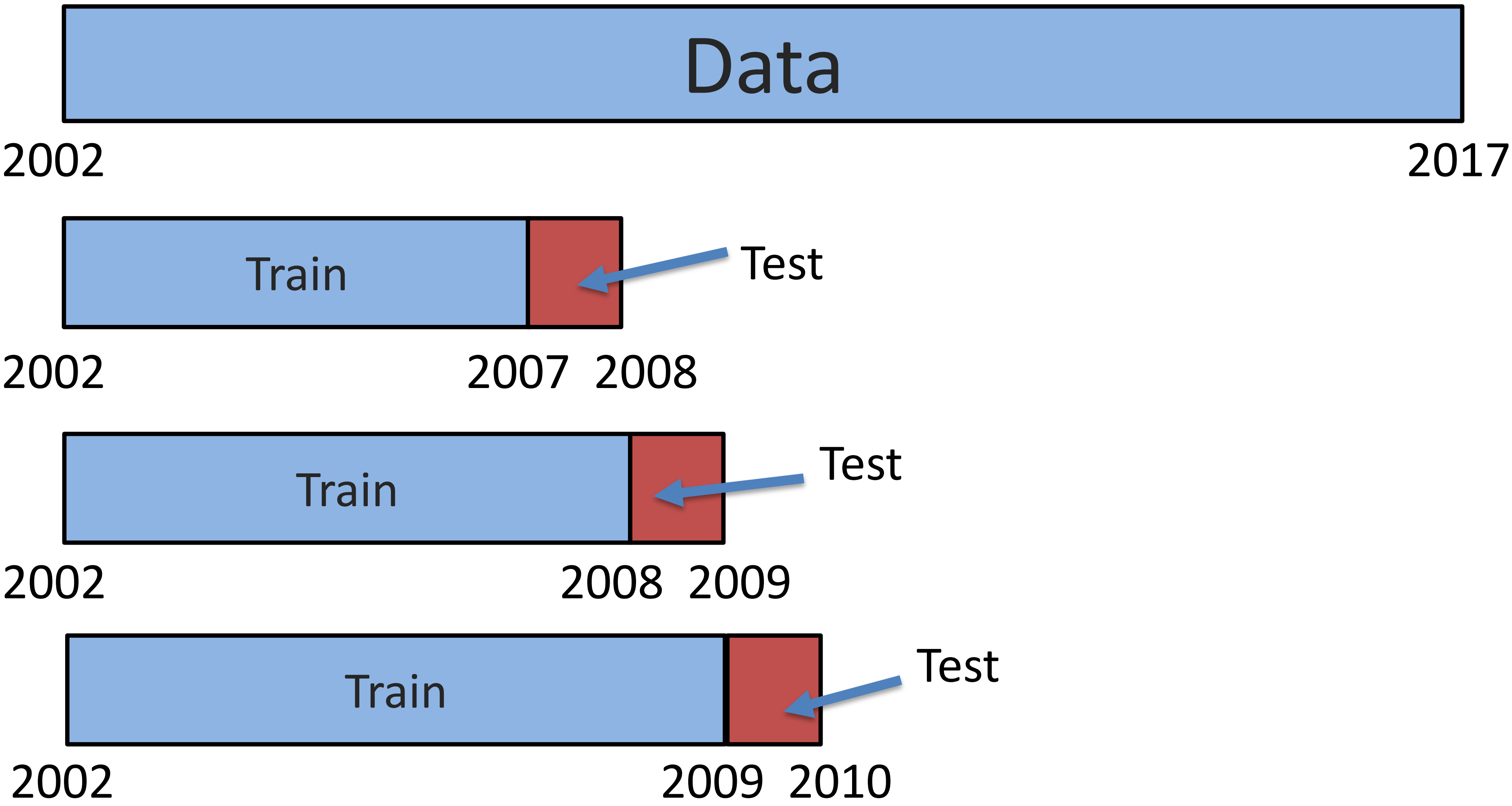
1	2	3	4	5

Train

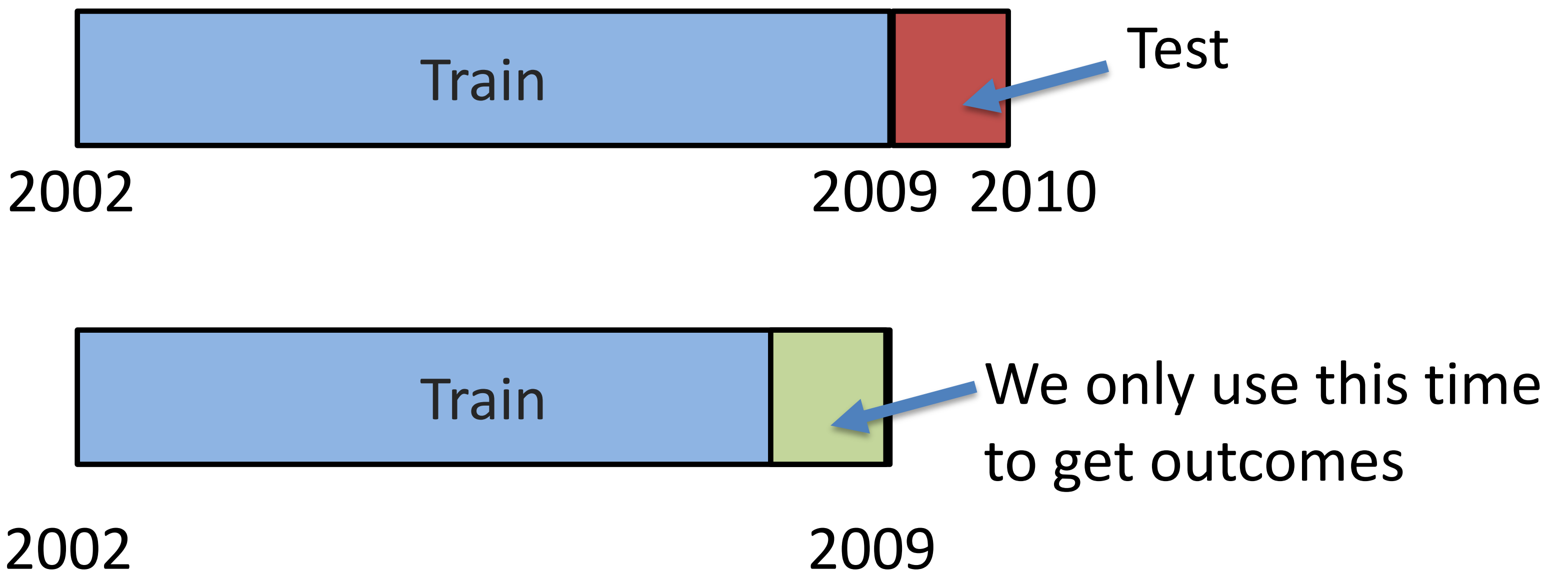
Test

k-fold Cross-Validation

Temporal Holdouts



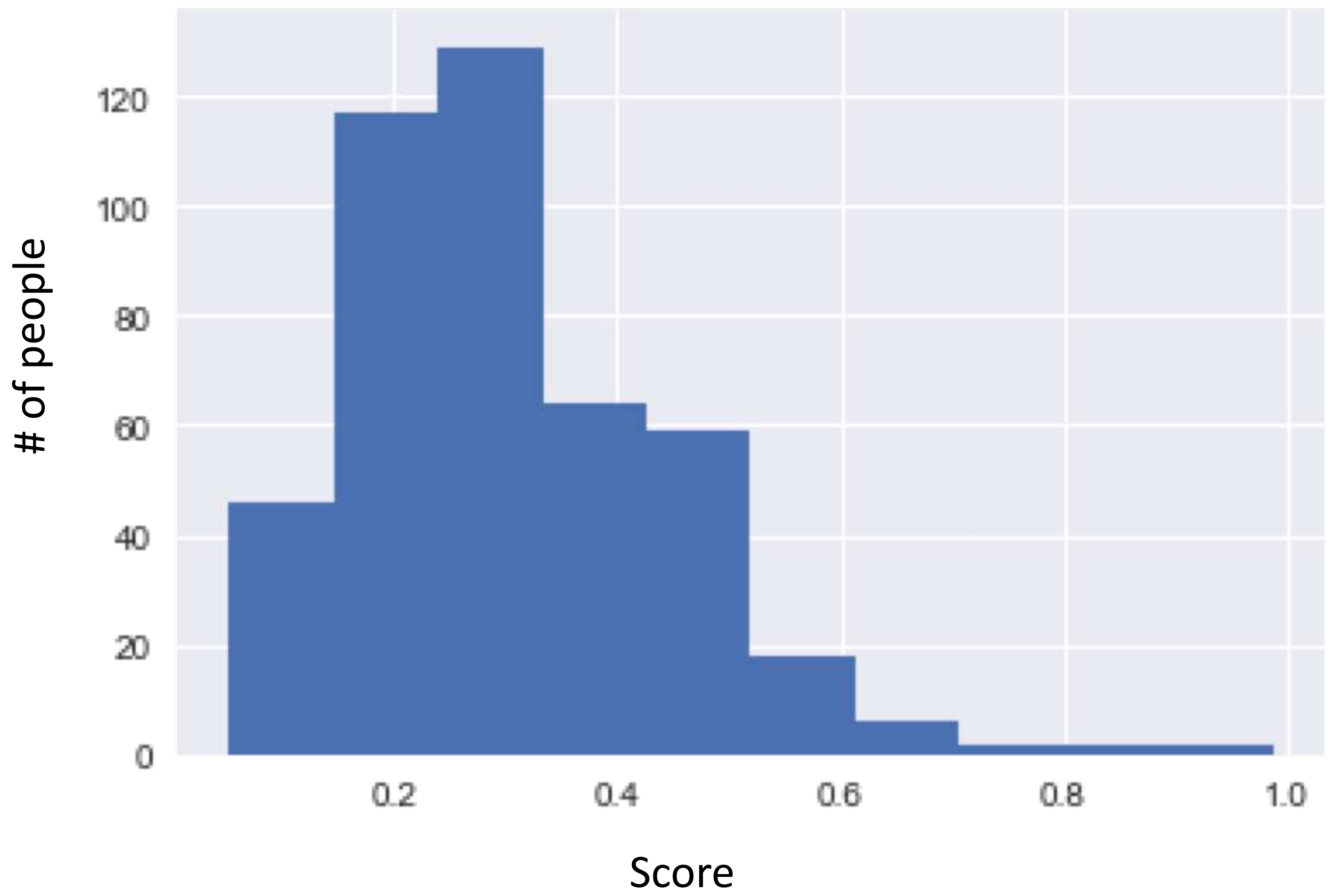
Training - Time splits



Evaluation - Methodology

- In-sample
- Out of sample
- Multiple Out-of-sample (Hold-out) Splits
- Cross Validation
  - Leave one out (LOO)
  - k fold
- Temporal Holdouts

Score Distribution on the Test Set



Evaluation - Metrics

- Predictions are often scores between 0 and 1
- We need to first turn them into 0 or 1 by selecting a threshold

Predicted Class

Actual Class

	Yes	No
Yes	**True Positives (TP)**	**False Negatives (FN)**
No	**False Positives (FP)**	**True Negatives (TN)**

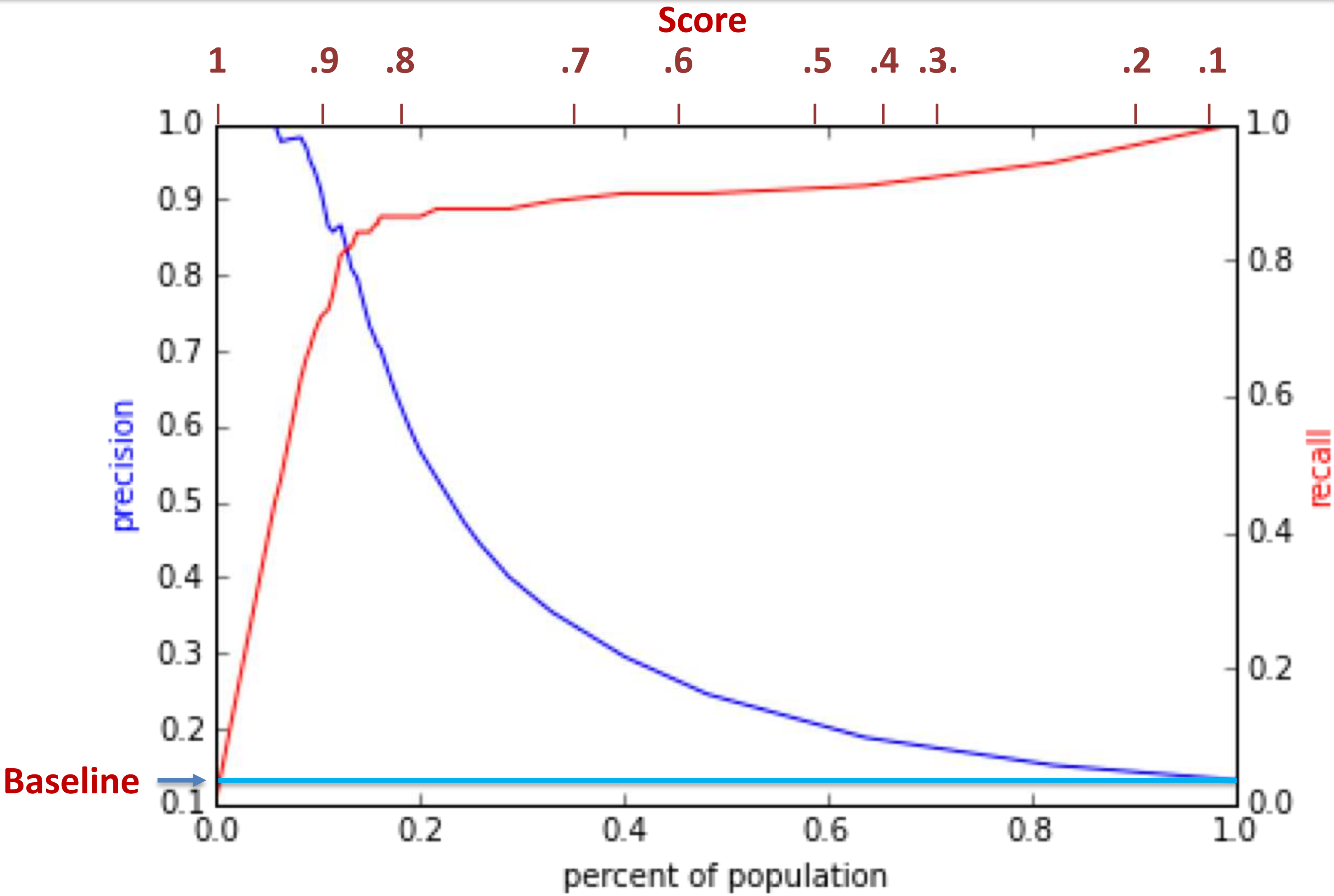
Evaluation – Metrics (at a threshold k)

- Accuracy = **(TP + TN) / (TP + TN + FP + FN)**
- Precision (or PPV) = **TP / (TP + FP)**
- Recall (or Sensitivity) = **TP / (TP + FN)**
- Specificity = **TN / (TN + FN)**

Predicted

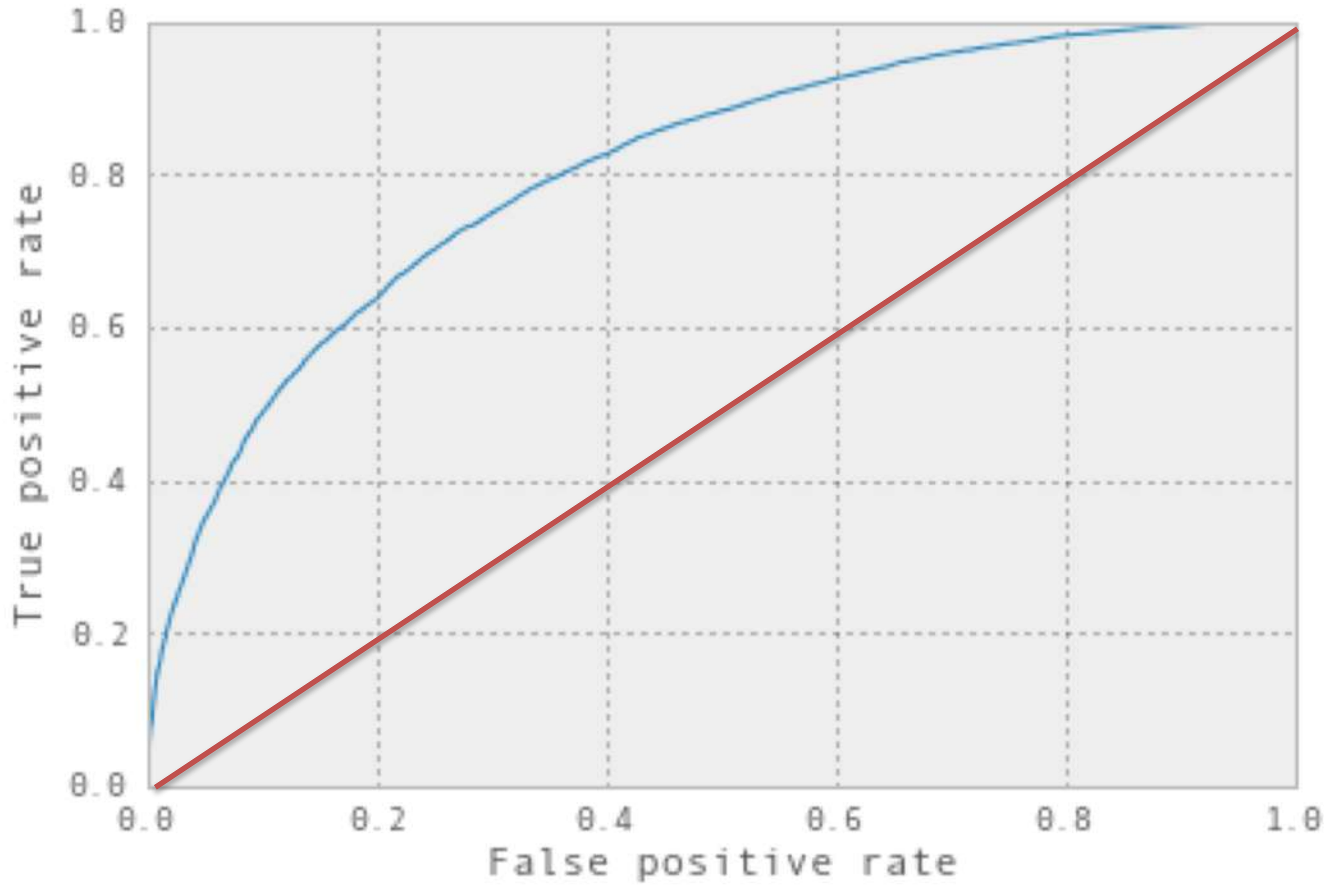
Actual		Yes	No
	Yes	**True Positives (TP)**	**False Negatives (FN)**
	No	**False Positives (FP)**	**True Negatives (TN)**

Varying the Threshold



ROC Curve

Receiver Operator Characteristic Curve



AUC (Area Under Curve)

- Overall measure of performance
  - 1 if all 1s are ranked above all 0s
  - 0 if all 0s are above all 1s

Evaluation - Baselines

- Random (predict most frequent class)
- Simple heuristics
- Expert heuristics (what may be in use today)

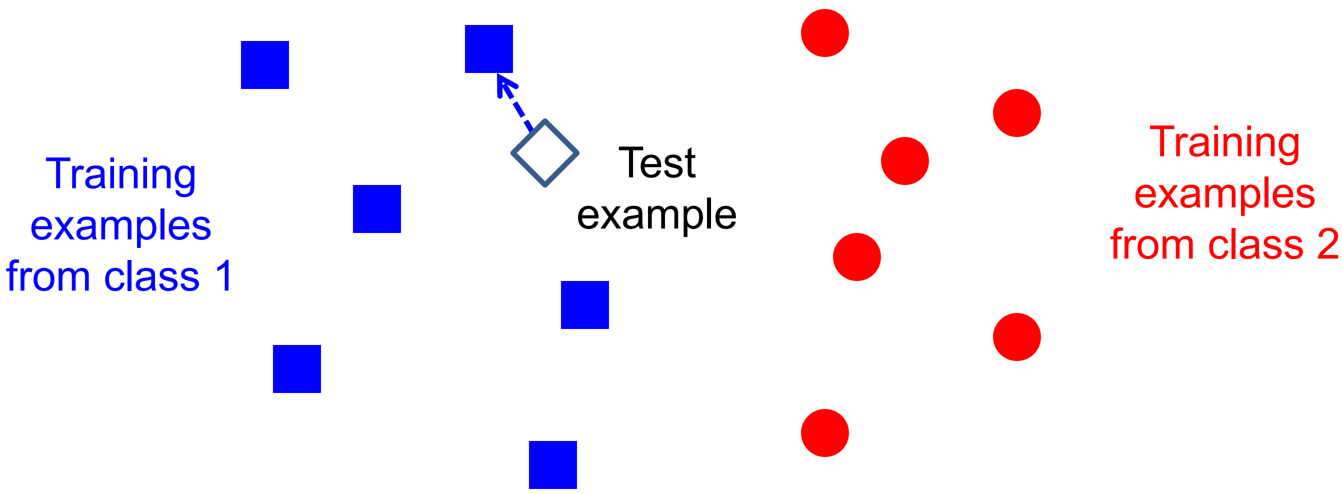
Supervised Learning - Methods

- Overfitting vs Underfitting

(Some) Popular/Common Methods

- Nearest neighbor
- Decision Trees
- Regression
- Support Vector Machines
- Bayes Classifier (not going to cover)
- Neural Networks (not going to cover)
- Ensembles
  - Bagging
  - Boosting
  - Random Forests

Classifiers: Nearest neighbor

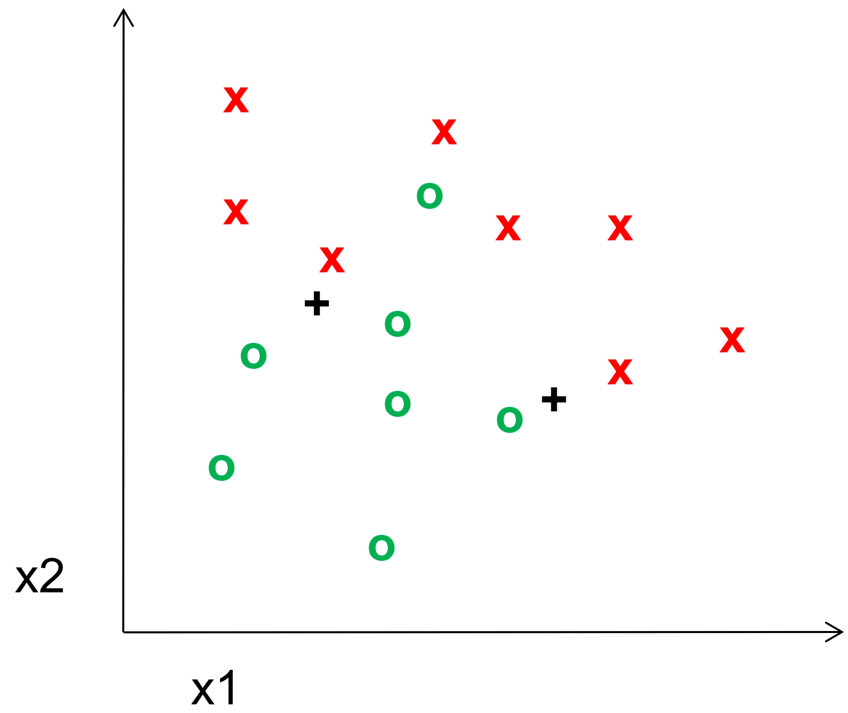


f($\mathbf{x}$) = label of the training example nearest to $\mathbf{x}$

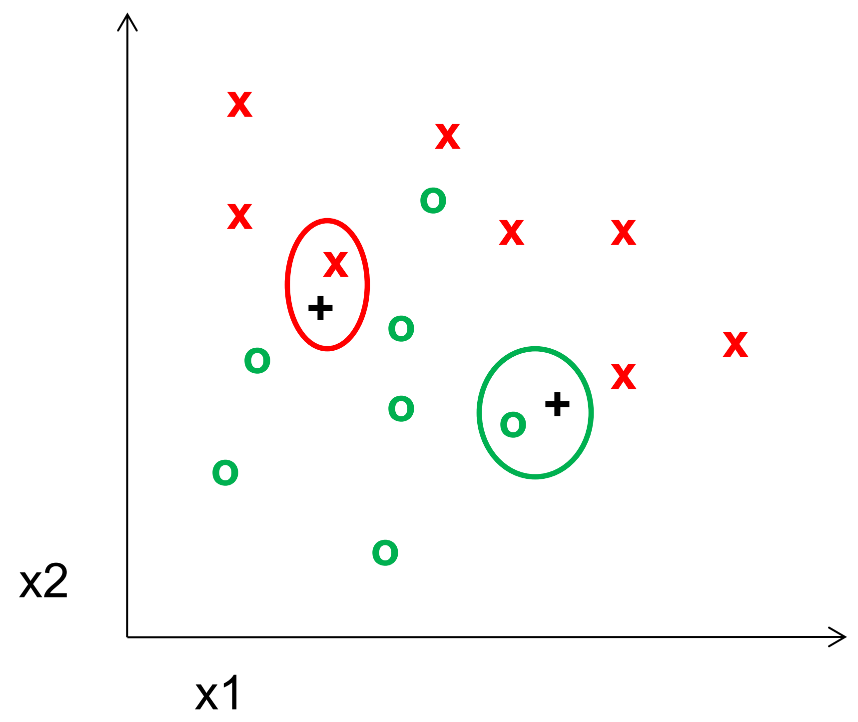
- **All we need is a distance function for our inputs**
- No training required!

Slide credit: L. Lazebnik

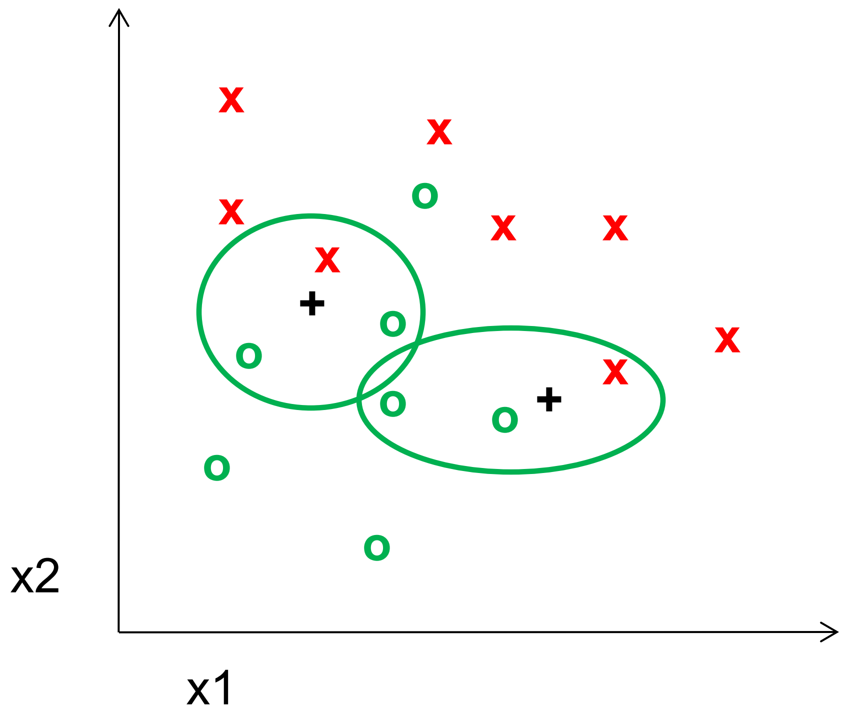
K-nearest neighbor



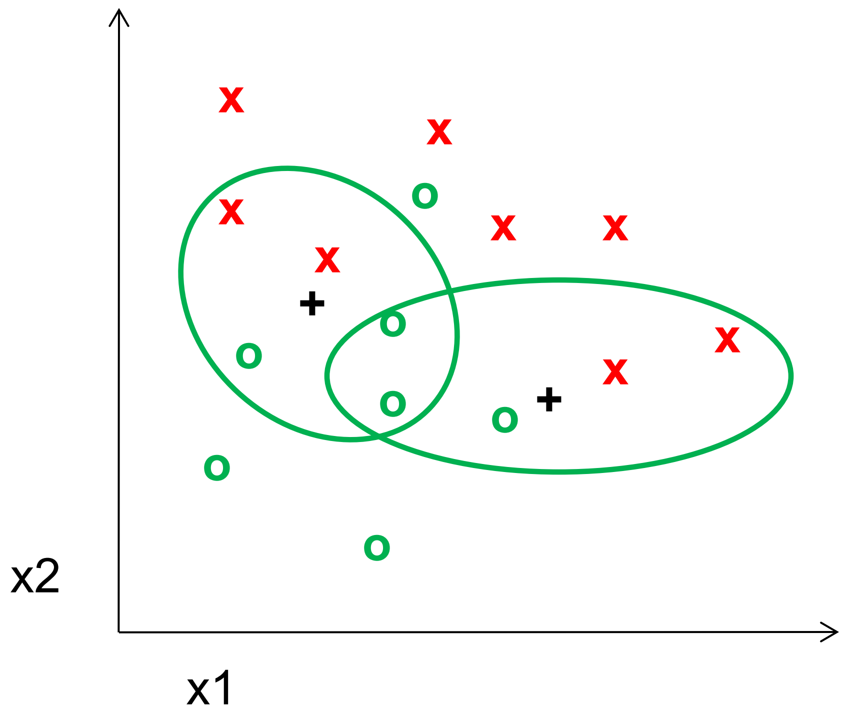
1-nearest neighbor



3-nearest neighbor



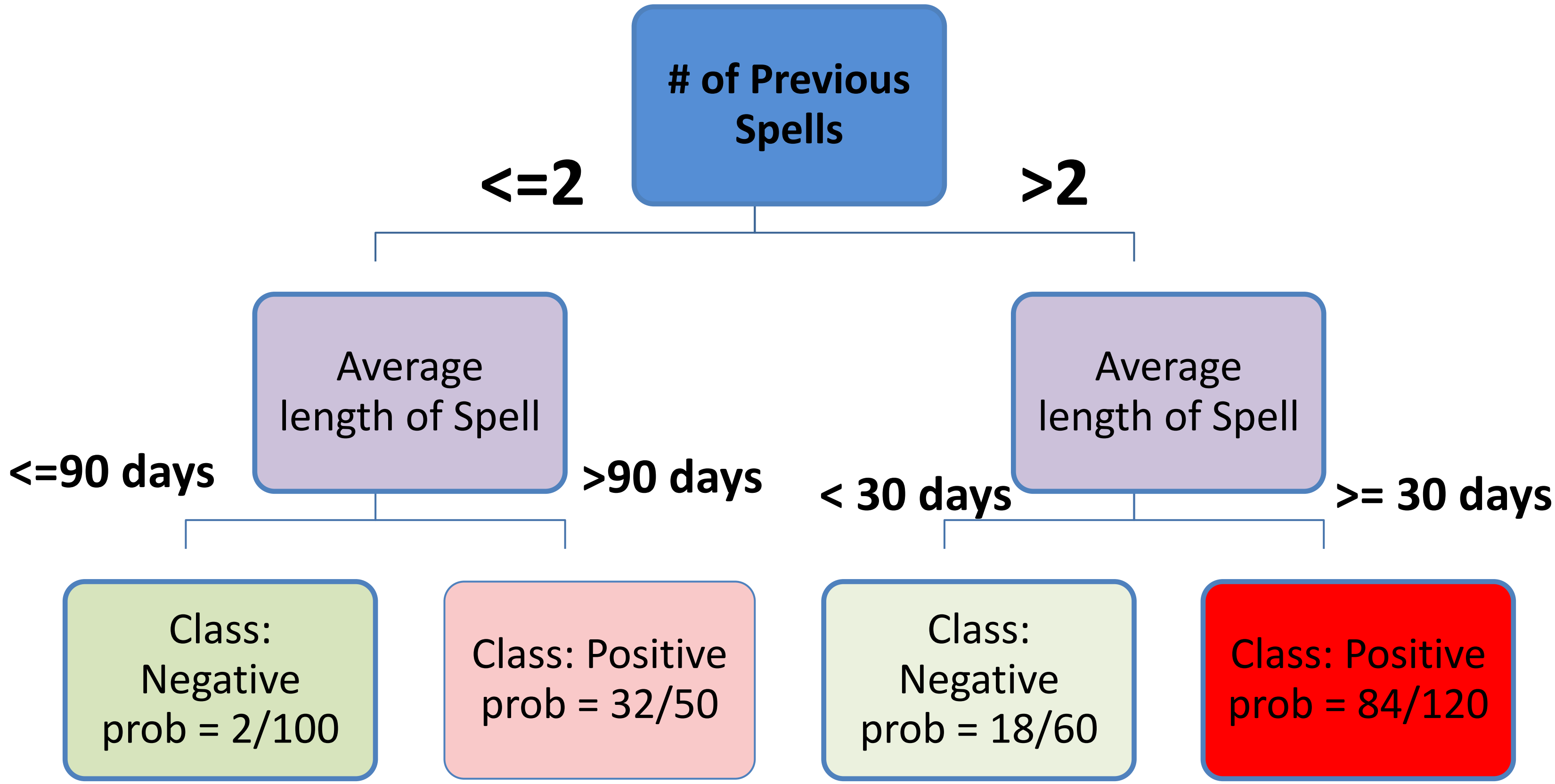
5-nearest neighbor



Using K-NN

- Simple
- “Cheap” to train – just involves storing data
- Scoring new data can be slow (needs to compute distances)

Classifiers: Decision Trees



What do we need to build a tree?

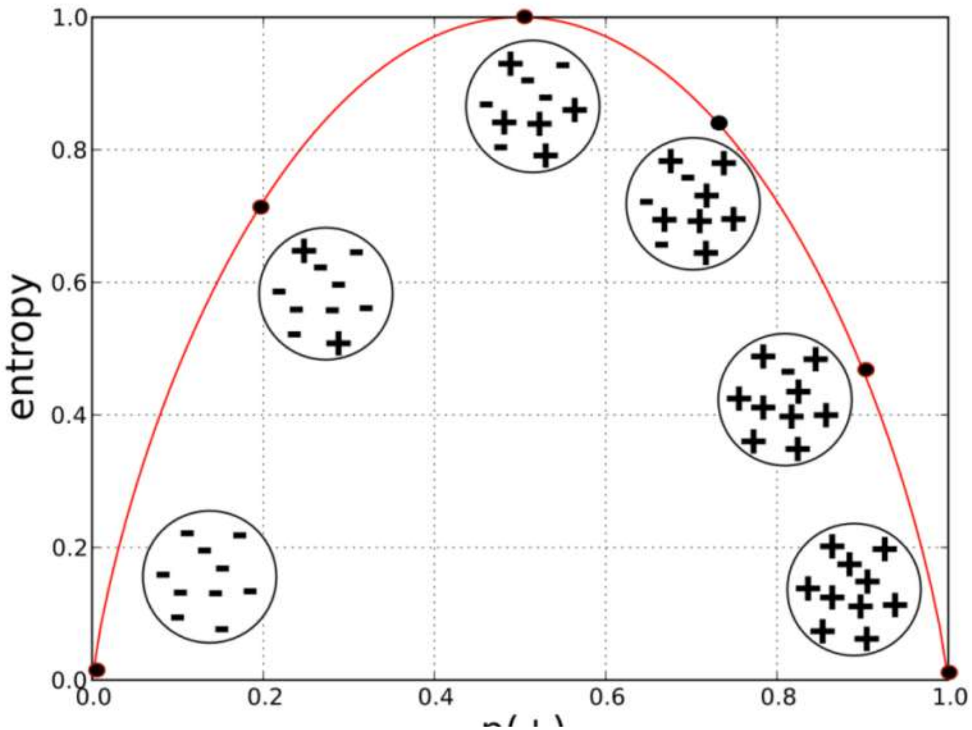
- How to create a split?
- When to stop?

How to create a split?

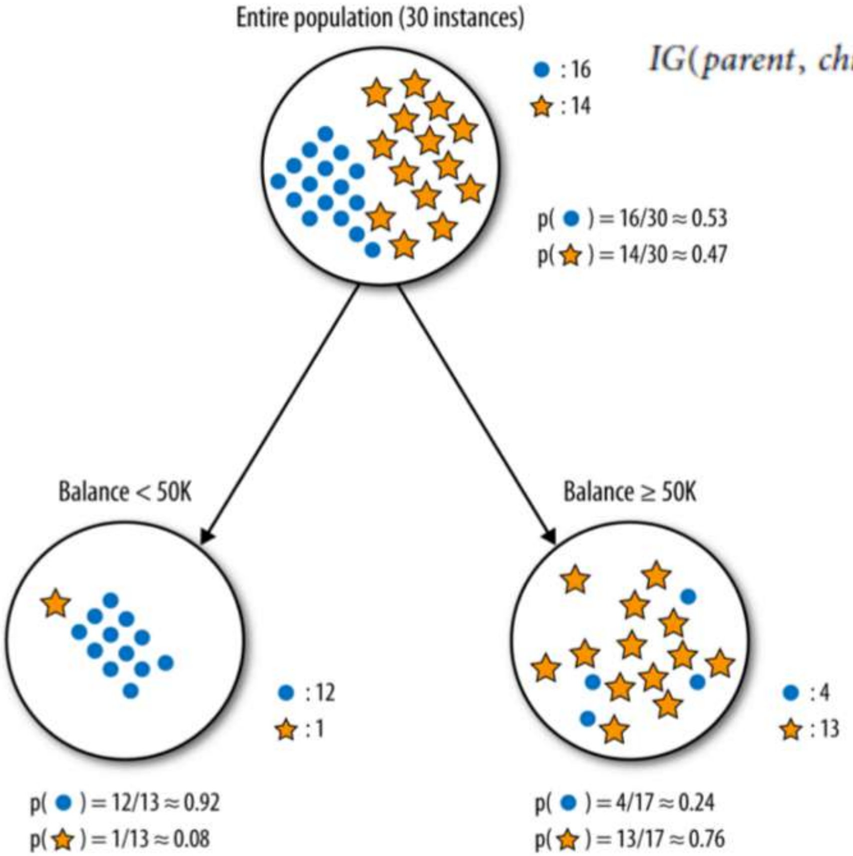
- Which of the many features/predictors/variables do we choose to split?
- What do we want the split to result in?
  - Purity of the leaf node
- How do we measure purity?

Information Gain

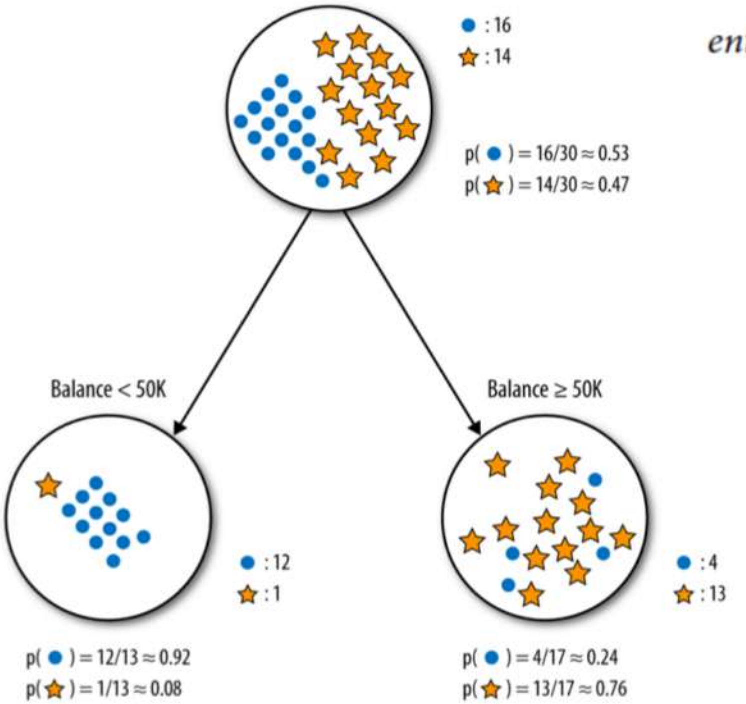
- The most common splitting criterion is called **information gain** (IG)
  - It is based on a **purity measure** called **entropy**
    - $entropy = -p_1 \log_2(p_1) - p_2 \log_2(p_2) - ..$
    - Measures the general disorder of a set



- Information gain measures the *change* in entropy due to any amount of new information being added



$$IG(parent, children) = entropy(parent) - [p(c_1) \times entropy(c_1) + p(c_2) \times entropy(c_2) + \cdots]$$



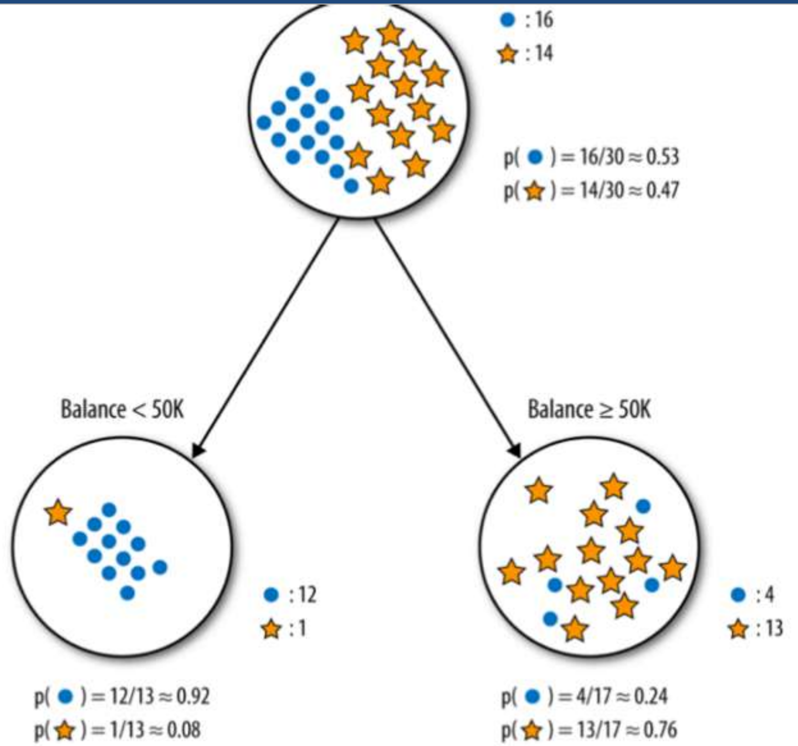
$$
\begin{aligned}
entropy(parent) &= -[p(\bullet) \times \log_2 p(\bullet) + p(\star) \times \log_2 p(\star)] \\
&\approx -[0.53 \times -0.9 + 0.47 \times -1.1] \\
&\approx 0.99 \quad \text{(very impure)}
\end{aligned}
$$

The entropy of the *left* child is:

$$
\begin{aligned}
entropy(Balance < 50K) &= -[p(\bullet) \times \log_2 p(\bullet) + p(\star) \times \log_2 p(\star)] \\
&\approx -[0.92 \times (-0.12) + 0.08 \times (-3.7)] \\
&\approx 0.39
\end{aligned}
$$

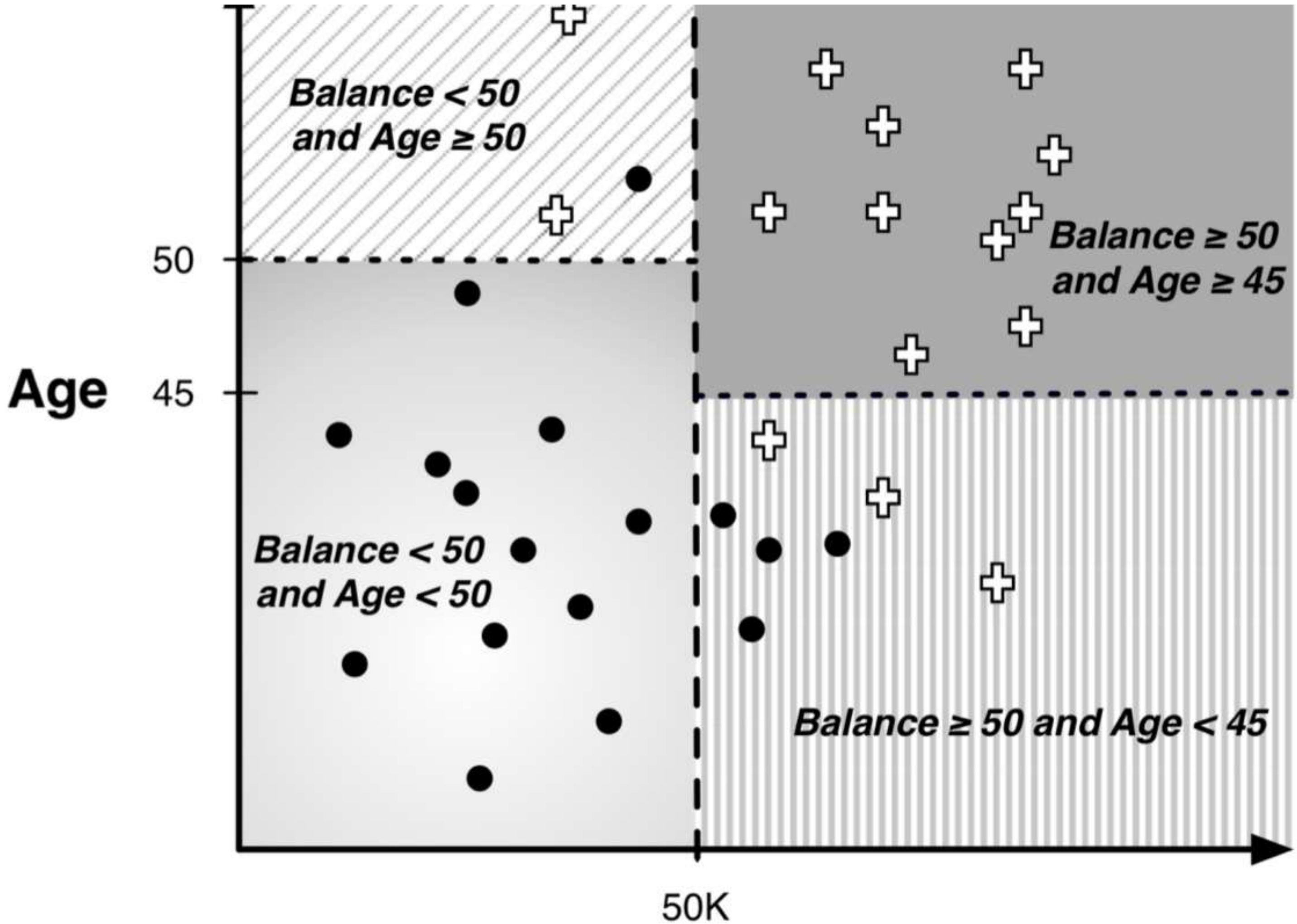
The entropy of the *right* child is:

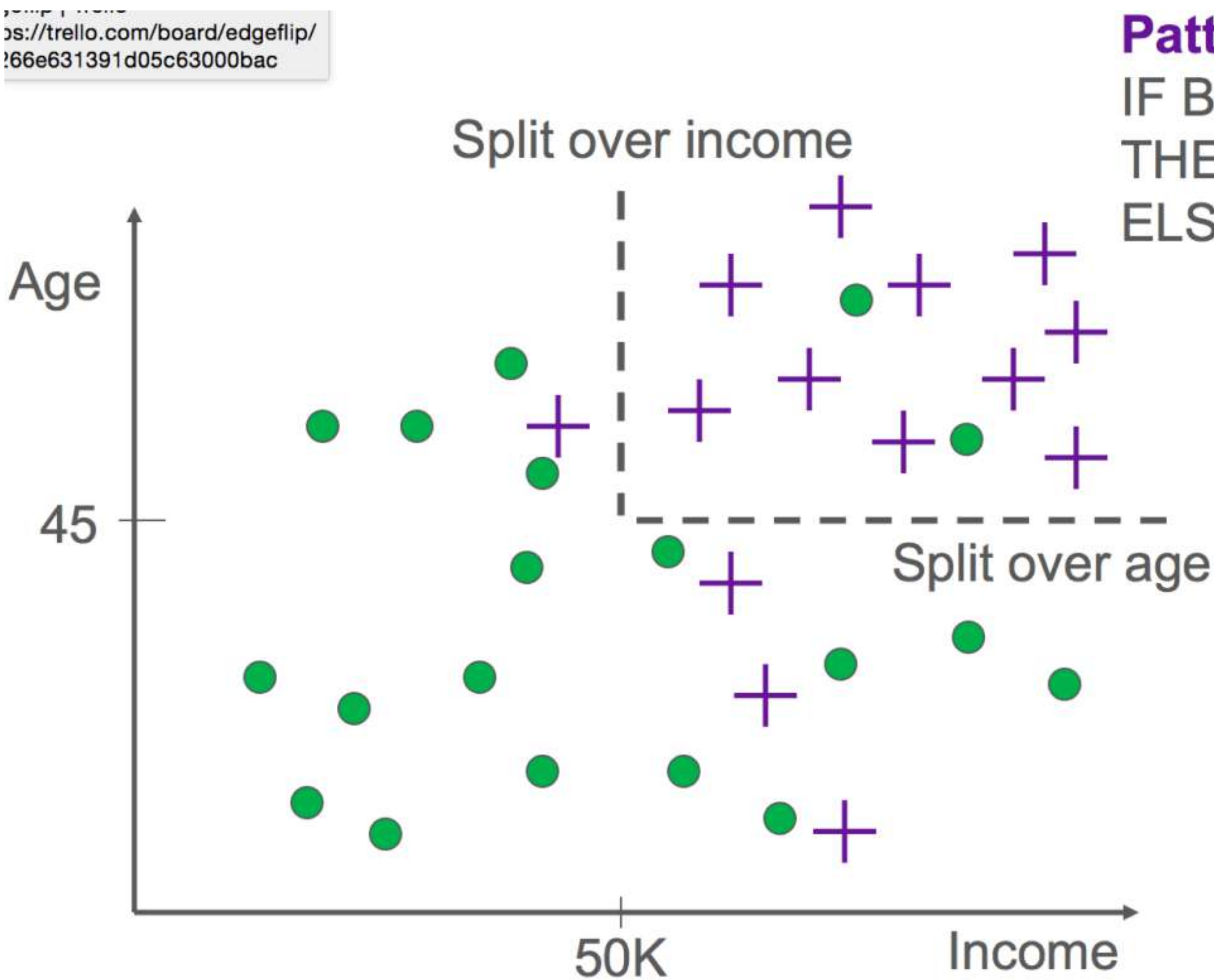
$$
\begin{aligned}
entropy(Balance \geq 50K) &= -[p(\bullet) \times \log_2 p(\bullet) + p(\star) \times \log_2 p(\star)] \\
&\approx -[0.24 \times (-2.1) + 0.76 \times (-0.39)]
\end{aligned}
$$



$$\begin{aligned} IG &= \mathit{entropy}(\mathit{parent}) - [p(\text{Balance} < 50\text{K}) \times \mathit{entropy}(\text{Balance} < 50\text{K}) \\ &\quad + p(\text{Balance} \geq 50\text{K}) \times \mathit{entropy}(\text{Balance} \geq 50\text{K})] \\ &\approx 0.99 - [0.43 \times 0.39 + 0.57 \times 0.79] \end{aligned}$$

Another way of looking at Decision Trees





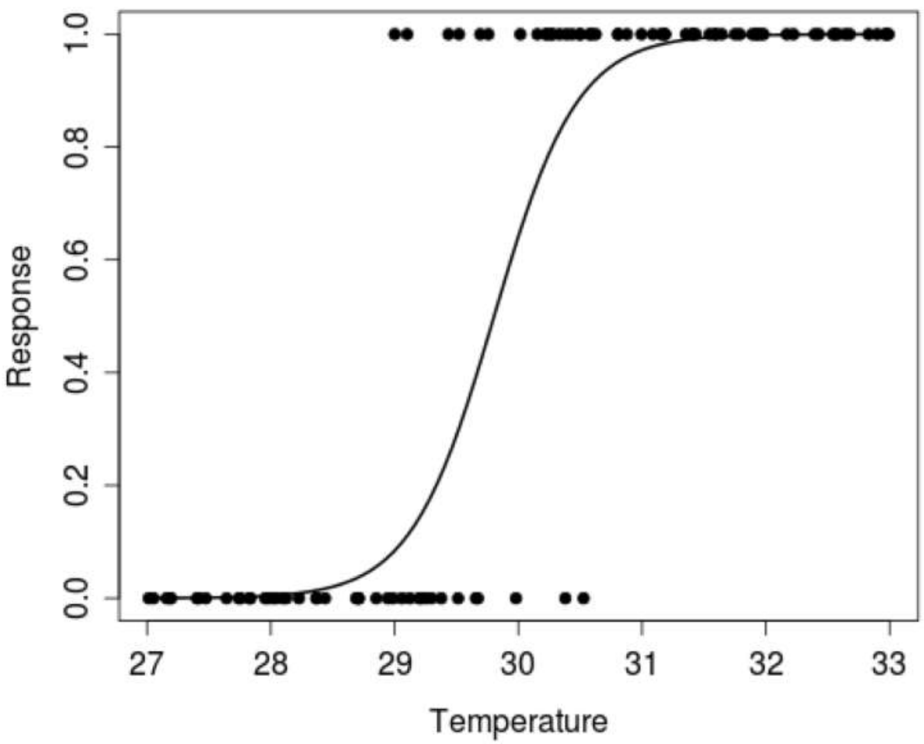
Pattern:
IF Balance >= 50K & Age > 45
THEN Default = 'no'
ELSE Default = 'yes'

Decision Trees: Summary

- Easy to build
- Performance is ok
- Sometimes easy to understand
- Computationally Fast/Cheap

Classifiers: Logistic Regression

Maximize likelihood of label/outcome given data



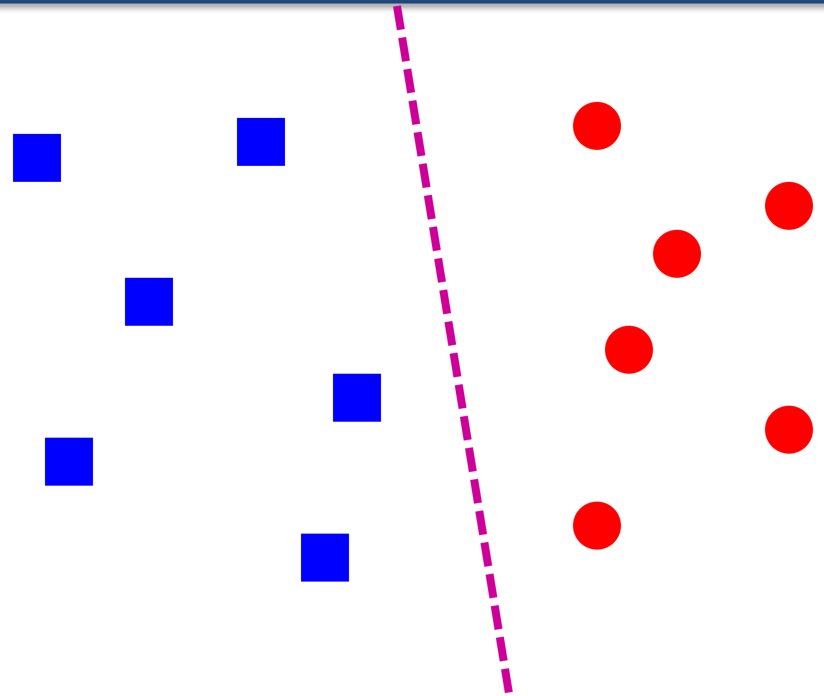
$$\log \frac{P(x_1, x_2 \mid y=1)}{P(x_1, x_2 \mid y=-1)} = \mathbf{w}^T \mathbf{x}$$

$$P(y=1 \mid x_1, x_2) = 1/\left(1+\exp\left(-\mathbf{w}^T \mathbf{x}\right)\right)$$

Using Logistic Regression

- Quick, simple classifier (try it first)
- Outputs a "probability"
- Use L2 or L1 regularization
  – L1 does feature selection and is robust to irrelevant features but slower to train

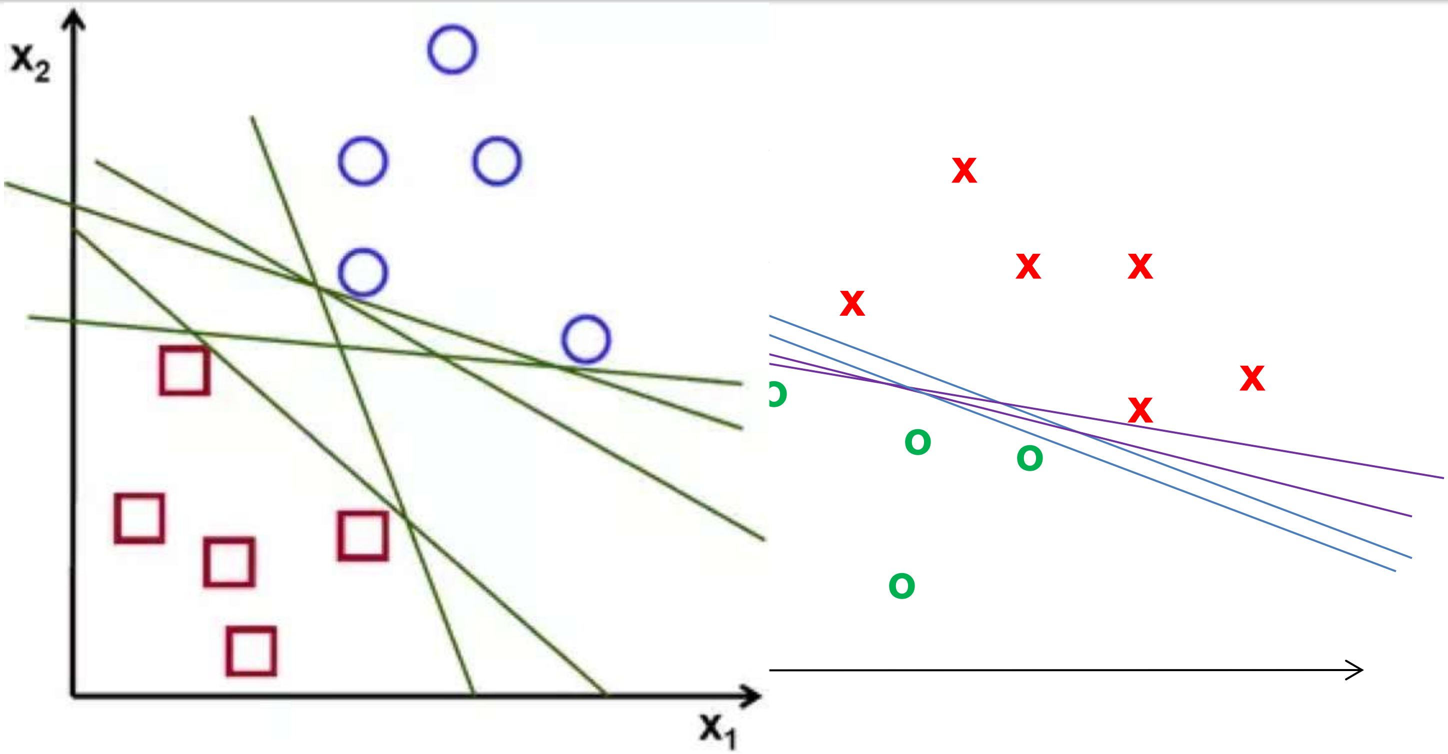
Classifiers: Linear



- Find a *linear function* to separate the two classes

Slide credit: L. Lazebnik

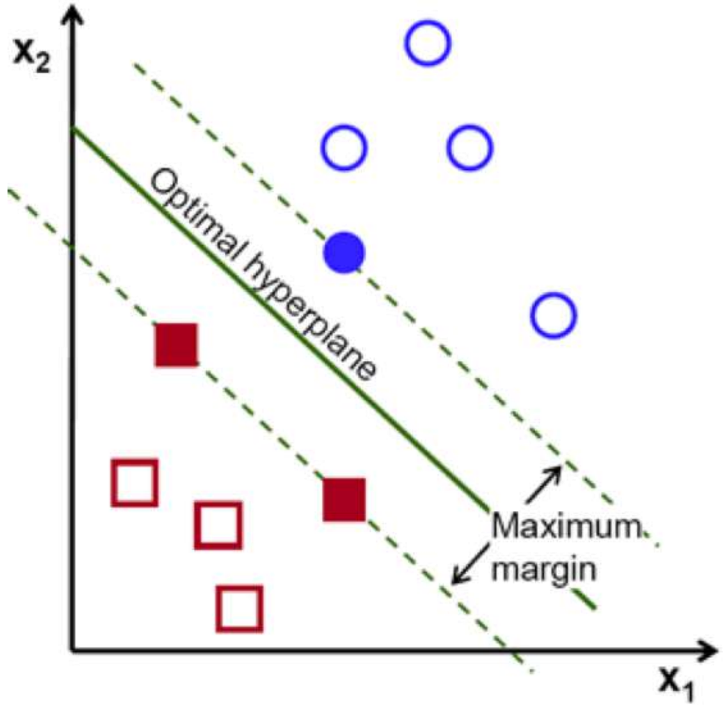
Classifiers: Support Vector Machines



- Find a *linear function* to separate the classes:

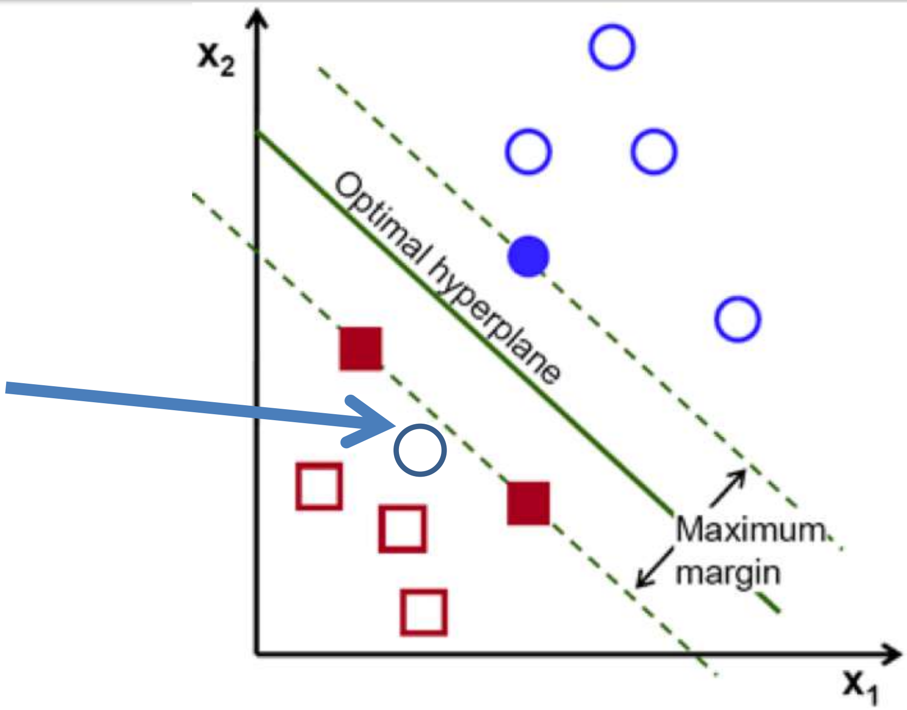
$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

Classifiers: Linear SVM



- Too many possible boundaries
- SVMs attempt to maximize the “margin”
- Optimization problem

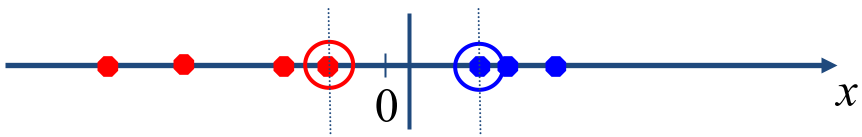
Classifiers: Linear SVM



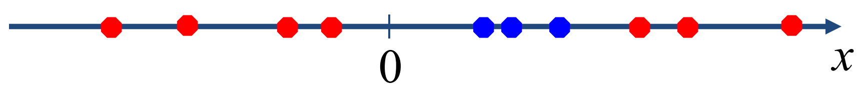
- Allow errors to keep models simple

Nonlinear SVMs

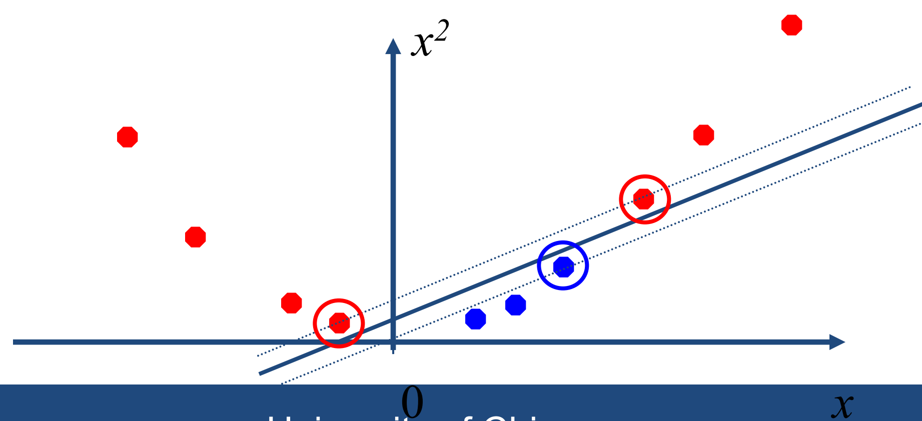
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?



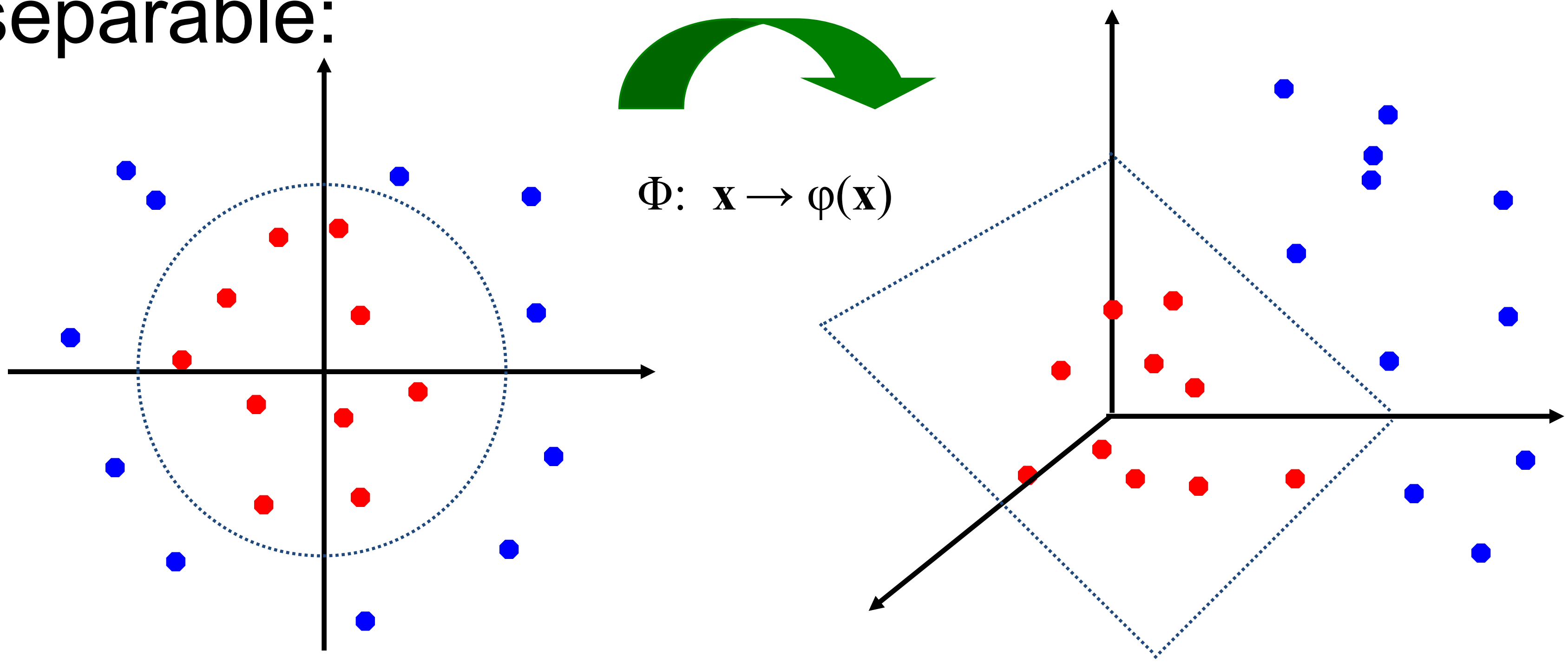
- We can map it to a higher-dimensional space:



Slide credit: Andrew Moore

Nonlinear SVMs

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation $\varphi(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$$

- (to be valid, the kernel function must satisfy *Mercer's condition*)
- This gives a nonlinear decision boundary in the original feature space:

$$\sum_i \alpha_i y_i \varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}) + b = \sum_i \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b$$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Ensembles – Intuition

- Generate models that are diverse but decent
- Combine diverse, uncorrelated models to get more robust predictions
- Question:
  - How to generate diverse, somewhat uncorrelated models?

Ensembles

- Different methods
- Algorithms with different choice for parameters
- Data set with different features (e.g. random subspace)
- Data set = different subsets (e.g. bagging, boosting)

Ensemble Methods

- Bagging (Bootstrap Aggregation)
- Boosting
- Random Forests
- Stacking

Ensemble Methods: Bagging

- Create ensembles by repeatedly randomly resampling the training data (Brieman, 1996).
- Given a training set of size *n*, create *m* samples of size *n* by drawing *n* examples from the original data, ***with replacement***.
  - Each ***bootstrap sample*** will on average contain 63.2% of the unique training examples, the rest are replicates.
- Combine the *m* resulting models using simple majority vote.

Ensemble Methods: Bagging

- For i = 1 .. M
  - Draw samples with replacement
  - Learn classifier C_i
- Final classifier is a vote of C_1 .. C_M
- Increases classifier stability/reduces variance

figure from Friedman et al. 2000

Boosting

- Examples are given weights.
- At each iteration, a new hypothesis is learned and the examples are reweighted to focus the system on examples that the most recently learned classifier got wrong.

Boosting

- General Loop:

  Set all examples to have equal uniform weights.
  For *t* from 1 to *T* do:
  Learn a classifier, C_t, from the weighted examples
  Increase the weights of examples C_t classifies incorrectly

- Base (weak) learner must focus on correctly classifying the most highly weighted examples while strongly avoiding over-fitting.

- During testing, each of the *T* hypotheses get a weighted vote proportional to their accuracy on the training data.

Ensemble Methods: Boosting

- Improves classification accuracy

- Can be used with many different types of classifiers

figure from Friedman et al. 2000

Random Forests

- Motivation: reduce error correlation between classifiers
- Main idea: build a larger number of un-pruned decision trees
- Key: using a random selection of features to split on at each node

How Random Forests Work

- Each tree is grown on a **bootstrap** sample of the training set of **N** cases.
- A number **m** is specified much smaller than the total number of variables **M** (e.g. m = sqrt(M)).
- At each node, m variables are selected at random out of the M.
- The split used is the best split on these **m** variables.
- Final classification is done by averaging or majority vote across trees.

Advantages of random forest

- Works well
- More robust with respect to noise.
- More efficient on large data - parallelizable
- Provides an estimation of the importance of features in determining classification
- More info at: http://stat-www.berkeley.edu/users/breiman/RandomForests/cc_home.htm

Factors to consider

- Complexity
- Overfitting
- Robustness/Stability over time
- Interpretability
- Training Time
- Test Time

What to remember about classifiers

- Better to have smart features and simple classifiers than simple features and smart classifiers

- Need more training data with increasingly powerful/complex classifiers

Slide credit: D. Hoiem

Other tools

- Languages
  - Python, R, Matlab (not open source)
- Software Packages
  - Knime, Rapidminer, Weka (mostly research use)
- Cloud
  - Amazon, Google, Microsoft
- Commercial
  - SAS enterprise miner,Ibm/spss , Skytree, GraphLab, H2O

Things to be careful about

- Bias
- Explanations
- Predictions may not be causal
- Scores may not be probabilities

Reference Books & Articles

- Data Science for Business by Provost and Fawcett
  - Good introduction to the ML process
- Machine Learning by Peter Flach
  - Good introduction to algorithms/models/methods
- Big Data: New Tricks for Econometrics by Hal Varian

Online Resources

- Introduction to Statistical Learning http://www-bcf.usc.edu/~gareth/ISL/
- Videos: https://www.youtube.com/user/dataschool/playlists?shelf_id=4&sort=dd&view=50
- Mining Massive Datasets http://www.mmds.org/