Machine Learning Review

CptS 580 – Advanced Machine Learning
School of EECS
Washington State University

What is *Machine* Learning?

- Herbert Simon (1970)
  - Any process by which a system improves its performance.
- Tom Mitchell (1990)
  - A computer program that improves its performance at some task through experience.
- Ethem Alpaydin (2010)
  - Programming computers to optimize a performance criterion using example data or past experience.

Details, details

- How is knowledge represented?
- How is experience represented?
- What is the performance measure?
- Knowledge acquisition vs. skill acquisition

Why Do Machine Learning?

- Automated knowledge acquisition
- Discover new knowledge
- Understand human learning
- Systems need to adapt to unknown, dynamic environments

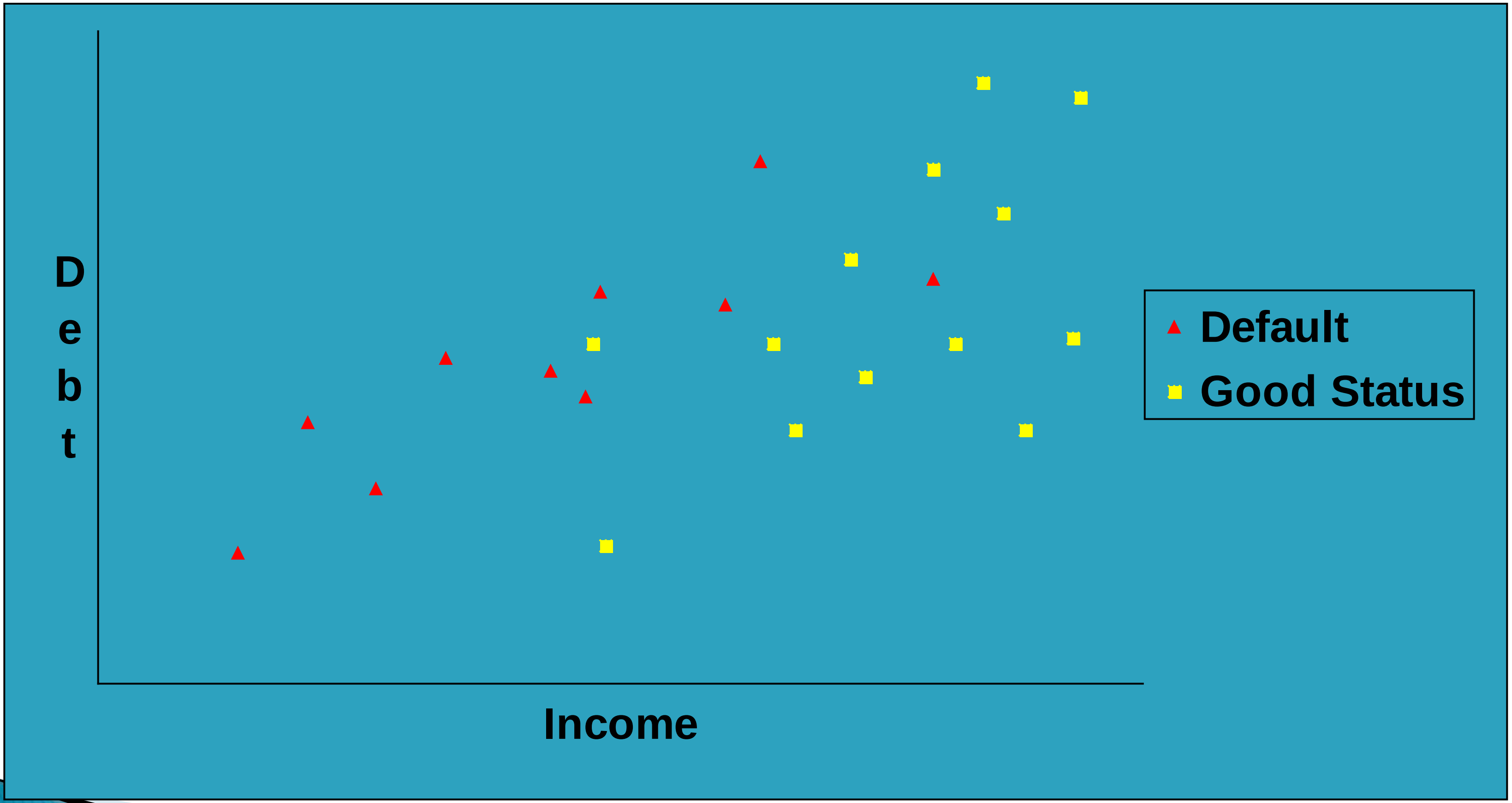
Applications

- Medical diagnosis
- Autonomous control (planes, trains, automobiles, robotics)
- Perception (speech, language, images, video)
- Recommendations (Amazon, Netflix)
- Prediction (business, financial, environment, health, energy, security, …)
- Fraud/intrusion detection
- …

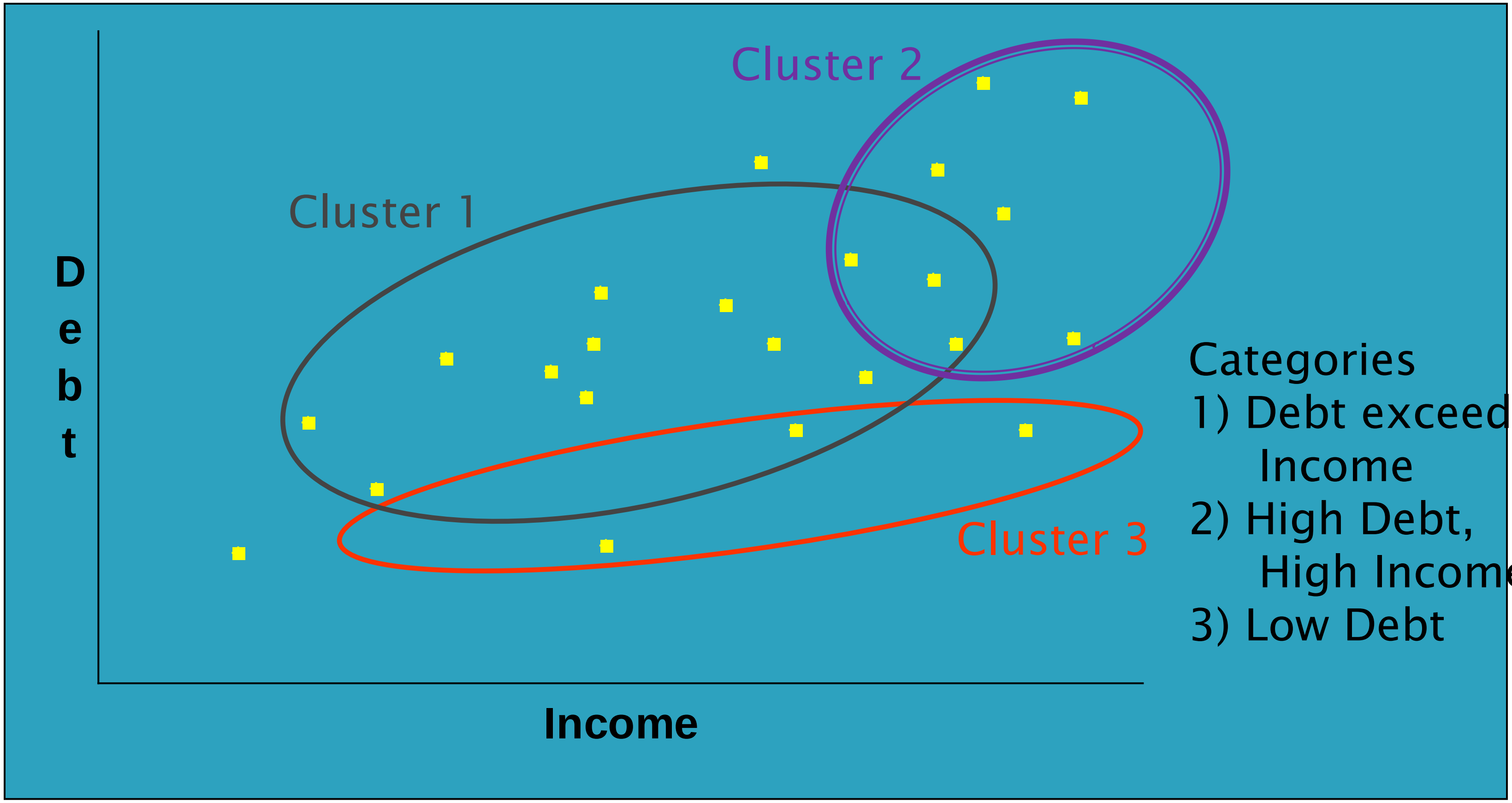
Approaches

- Unsupervised Learning
  - Clustering
- Supervised Learning
  - Classification
  - Regression
- Reinforcement Learning

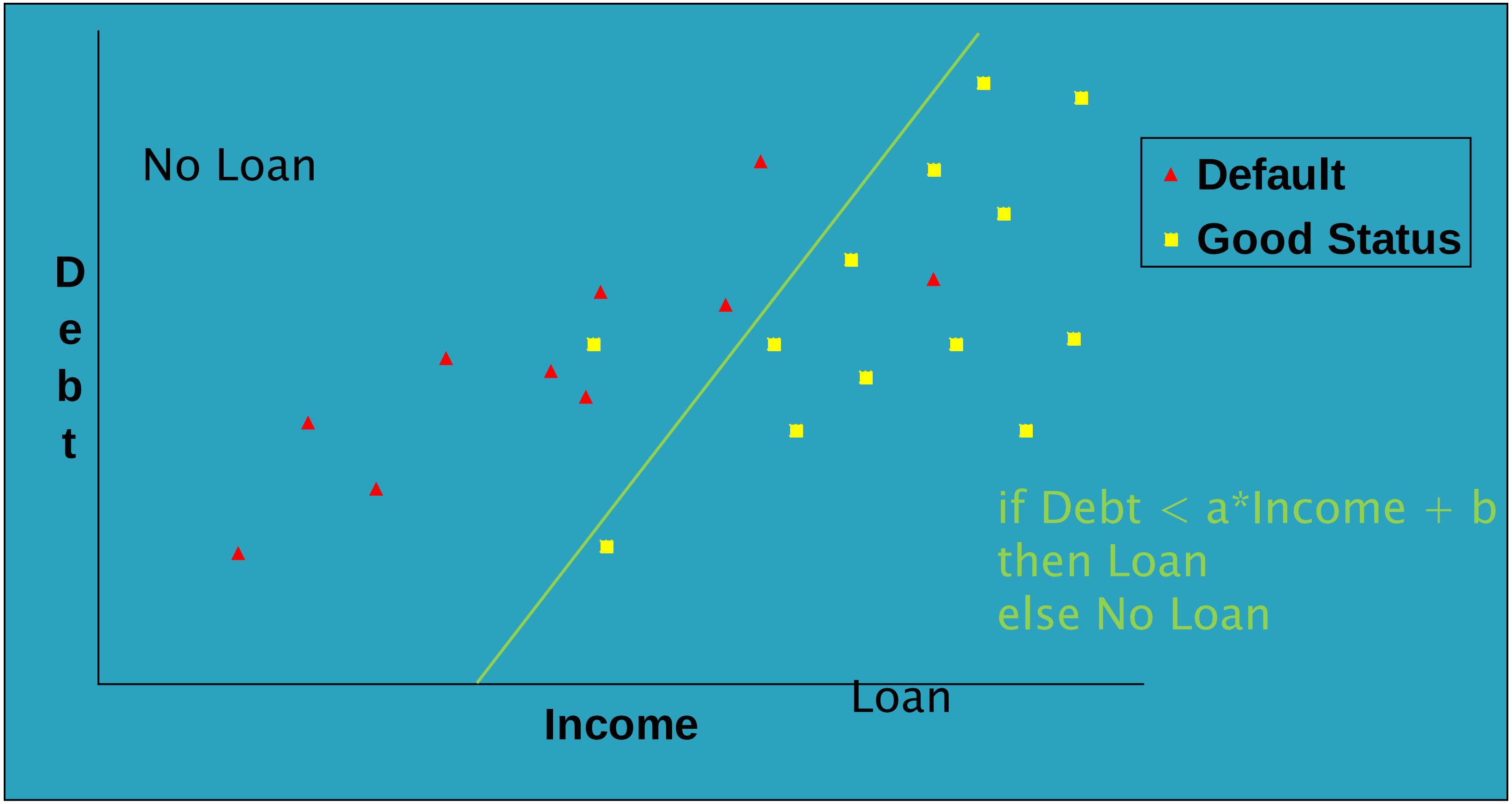
Bank Loan Example



Unsupervised Learning

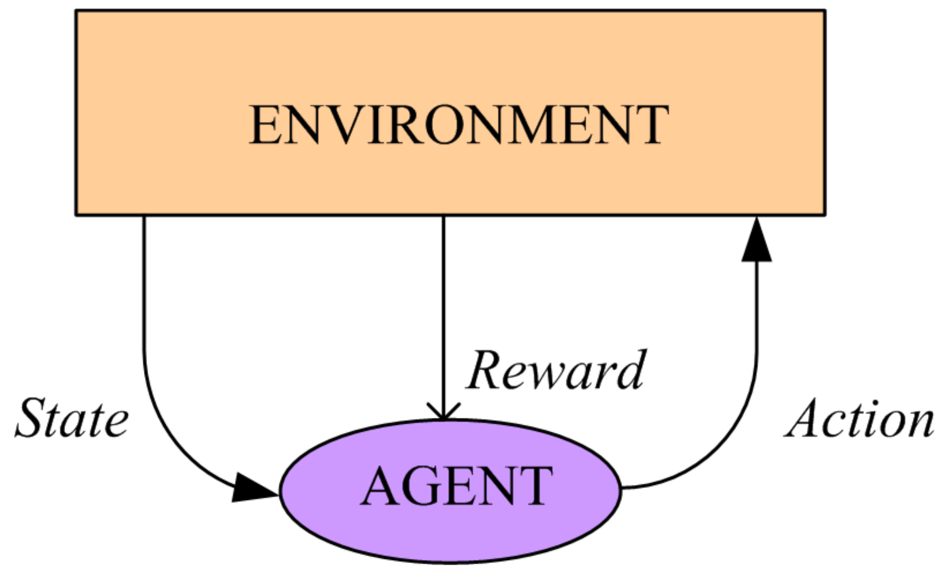


Supervised Learning



Reinforcement Learning

- Game-playing: Sequence of moves to win a game
- Robot in a maze: Sequence of actions to find a goal
- Agent has a state in an environment, takes an action and sometimes receives reward and the state changes
- Credit-assignment
- Learn a policy
  - π: State → Action



Other ML Issues

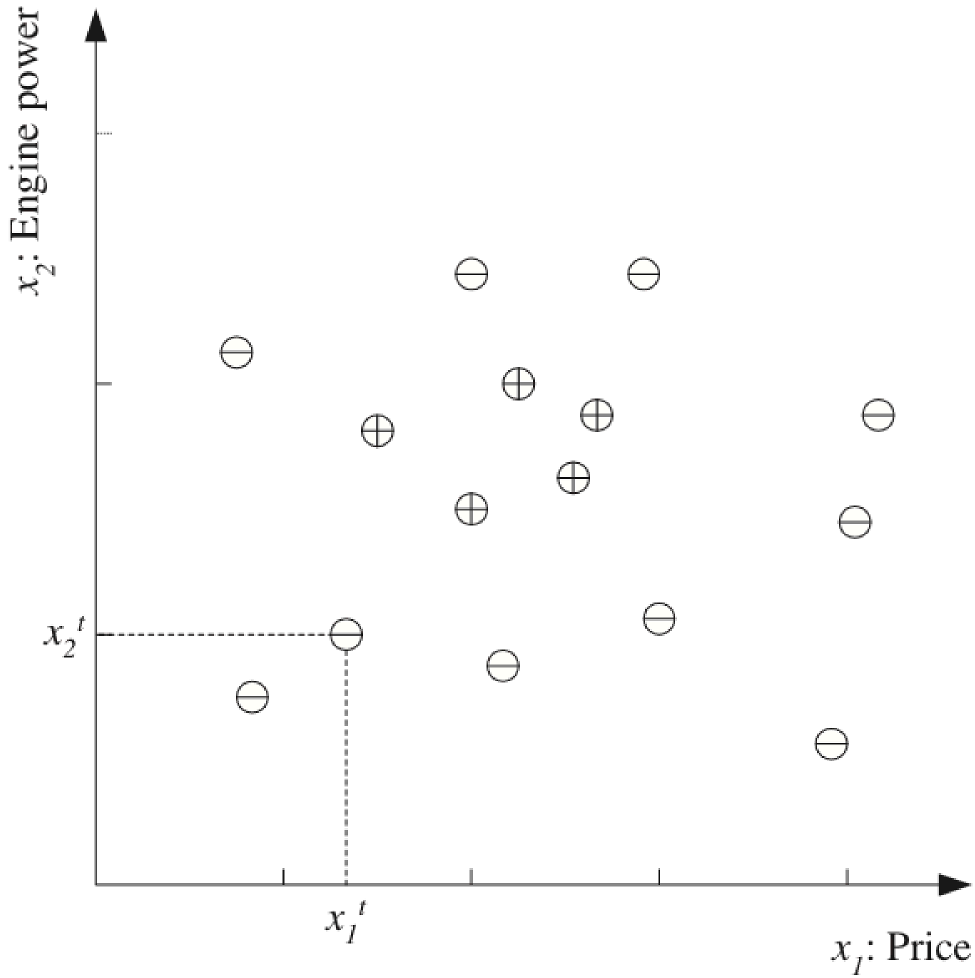
- Evaluation
  - Which learning approach is better
- Theoretical bounds
  - What is and is not learnable
- Scalability
  - Learning from massive, real-time datasets

Supervised Learning

Example: “Family Car”

- Learning task
  - Learn to classify cars into one of two classes: “family car” or “other”
  - Each car is represented by two features (attributes): “engine power” and “price”
  - Given several training examples of already-classified cars
  - Output classifier that accurately classifies cars

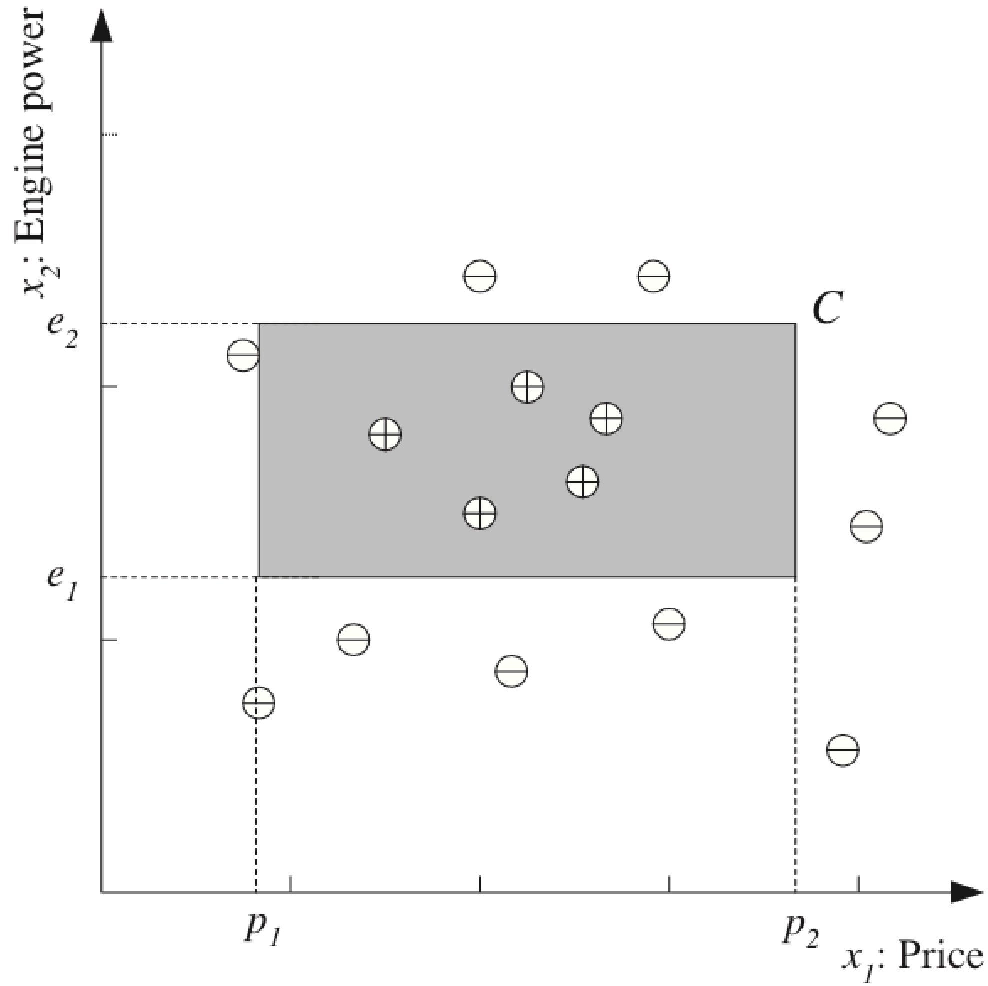
Example: "Family Car"



Definitions

- Feature (attribute): x_i
  - A property of the object to be classified
  - Discrete or continuous
  - E.g., “engine power”, “price”
- Instance: $\mathbf{x} = [x_1, x_2, \ldots, x_d]$
  - The feature values for a specific object
  - E.g., “engine power = 100”, “price = high”
- Instance space: $\mathcal{I}$
  - Space of all possible instances
- Class: C
  - Categorical feature of an object
  - Set of instances of objects in this category
  - E.g., “family car”

Example: "Family Car" Class



Definitions

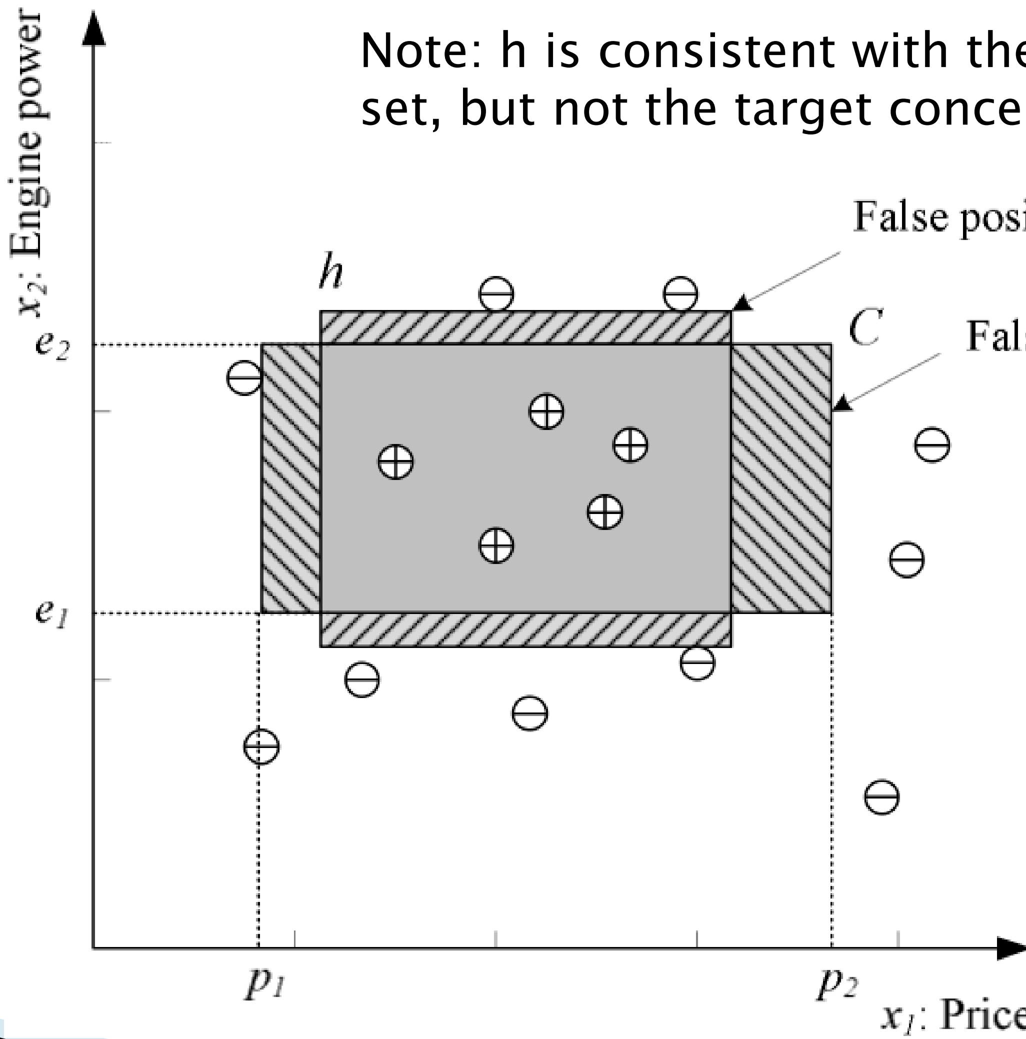
- Example: ($\mathbf{x}$,r)
  - Instance along with its class membership r
  - Positive example: member of class (r=1)
  - Negative example: not a member of class (r=0)
- Training set: $X = \{x^t, r^t\}$, $1 \leq t \leq N$
  - Set of N examples
- Target concept (C)
  - Correct expression of class
  - E.g., ($e_1 \leq$ engine power $\leq e_2$) and ($p_1 \leq$ price $\leq p_2$)

Definitions

- Hypothesis: $h(\mathbf{x}) \rightarrow \{0,1\}$
  - Approximation to target concept
- Hypothesis class: H
  - Space of all possible hypotheses
  - E.g., axis-aligned rectangles
  - E.g., axis-aligned ellipses
  - E.g., k-term-DNF
- Learning goal
  - Find hypothesis $h \in H$ that closely approximates target concept C
  - h is the output classifier
  - Target concept may not be in H

Example: Hypothesis Error

Note: h is consistent with the training set, but not the target concept C.



Definitions

- Empirical (sample) error
  - How well h classifies training set X

$$E(h \mid X) = \frac{1}{N}\sum_{t=1}^{N} 1(h(\mathbf{x}^t) \neq r^t)$$

1(expr) = 1 if expr is true, else 0

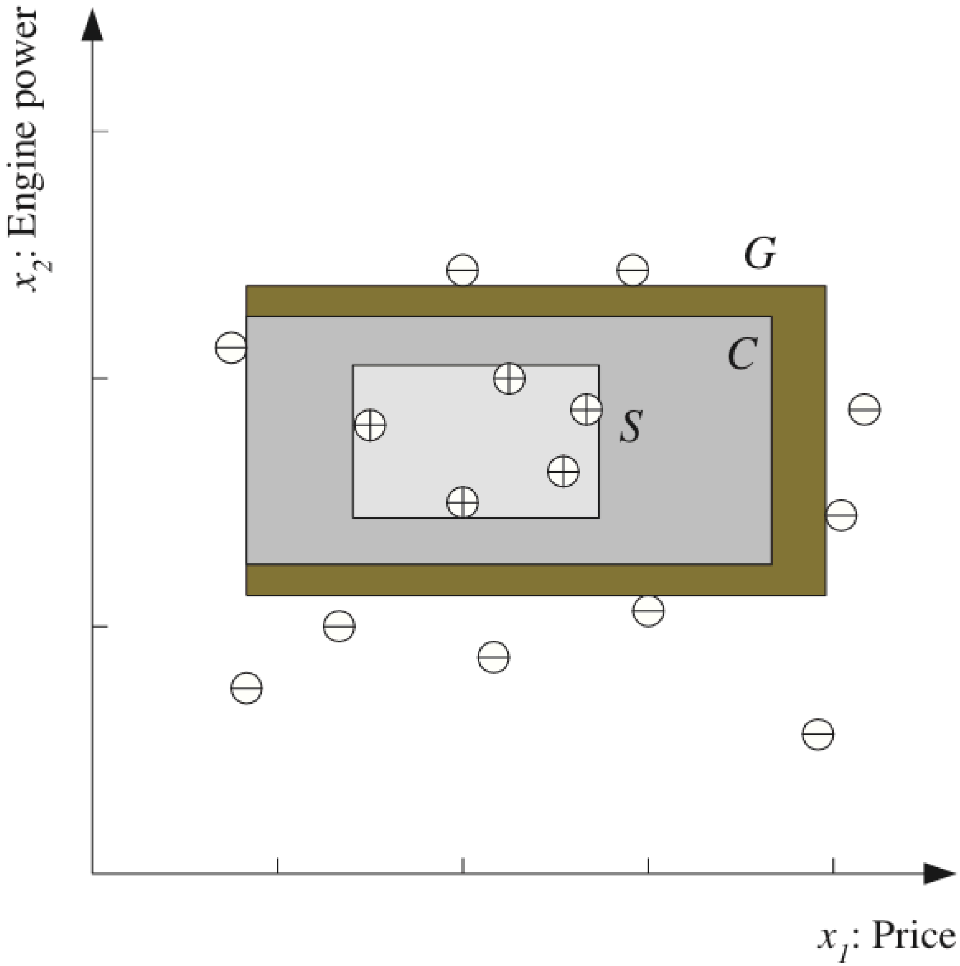
- Generalization error
  - How well h classifies instances not in X
- True error
  - How well h classifies entire instance space

$$E(h) = \frac{1}{|I|}\sum_{\mathbf{x}\in I} 1(h(\mathbf{x}) \neq C(\mathbf{x}))$$

Definitions

- Most specific hypothesis S
  - Consistent hypothesis covering fewest instances
- Most general hypothesis G
  - Consistent hypothesis covering most instances
- Version space
  - All hypotheses "between" S and G

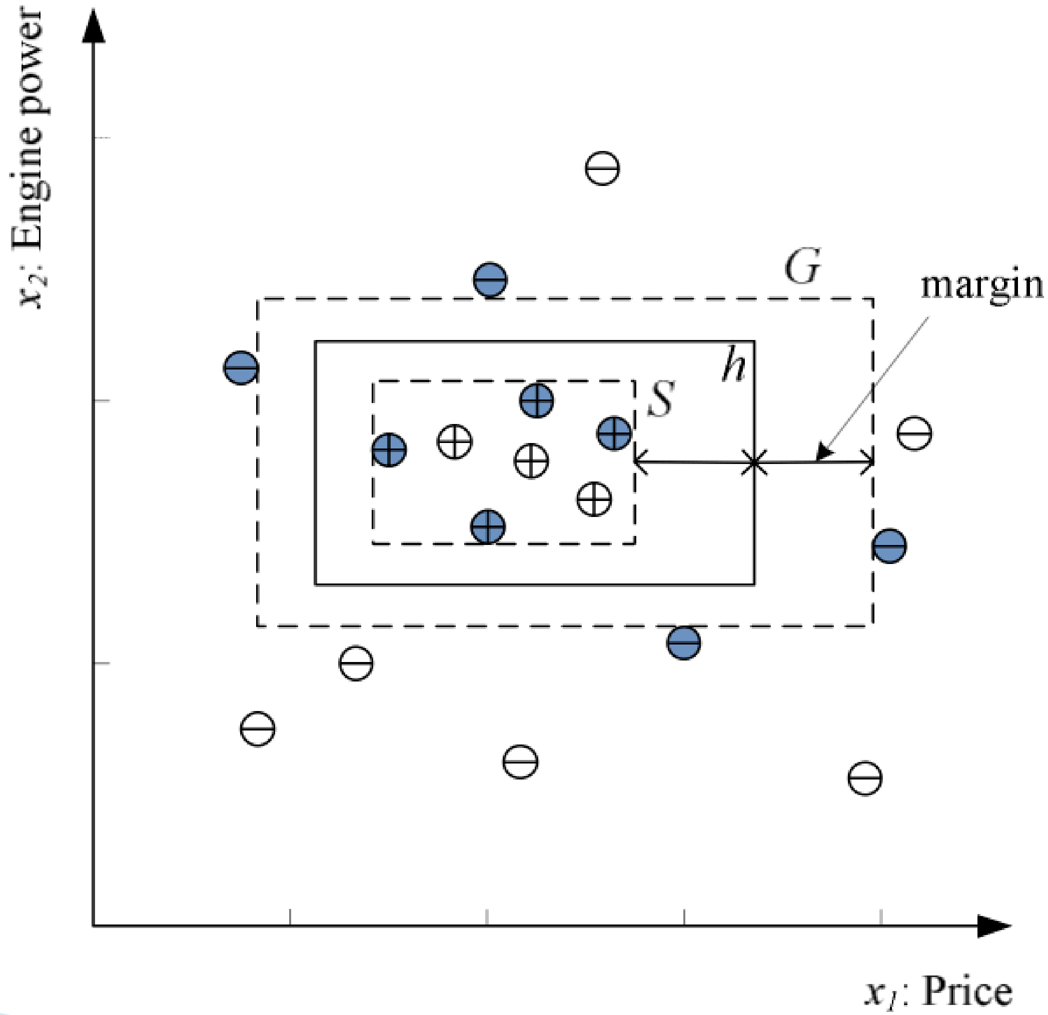
Example: Version Space



Version space: All rectangles within G and containing S.

Assuming we don't know C, which hypothesis in VS is the best?

Margin



<u>Margin</u>: The distance between h and the examples closest to h.

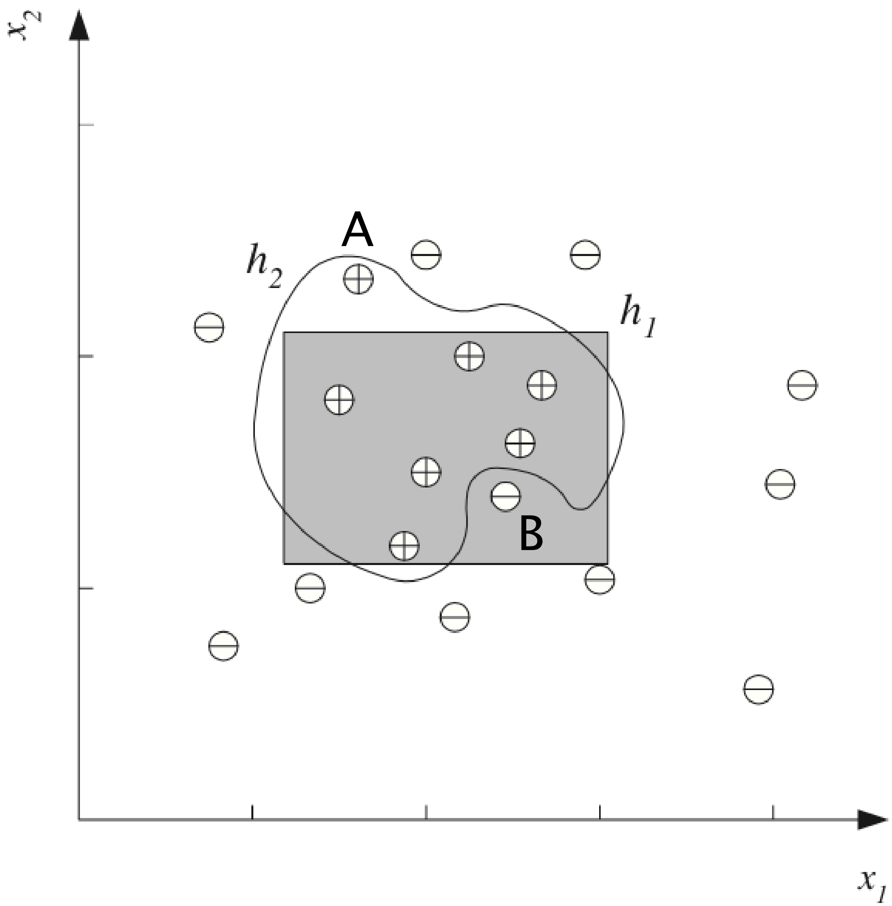
<u>Goal</u>: Find h maximizing margin.

<u>Note</u>: Only shaded examples needed to find h. These examples are the *support vector*.

Noise

- Sources
  - Incorrect feature values
  - Incorrect class labels
  - Hidden or latent features
- Impact
  - Overfitting: Trying too hard to fit h to the noise

Underfitting vs. Overfitting



If A and B are noise, then h_2 overfits.

If A and B are *not* noise, then h_1 underfits.

Bias vs. Variance

- Bias: Likelihood a learner will not change its hypothesis
- Variance: Ability of learner to change its hypothesis
- Simple models have high bias, low variance
- Complex models have low bias, high variance
- Want balanced tradeoff
- Depends on hypothesis class
  - Rectangles vs. arbitrary shape
- Occam's Razor: Prefer simpler models

Inductive Bias

- Given a training set X, there are many models that are consistent with X
- Preferring one of these models over another is an "inductive bias"
- For example
  - Preferring rectangles to arbitrary shapes
  - Preferring rectangle with largest margin
  - Preferring lower-degree polynomial
  - Preferring polynomial minimizing squared error
- How do we choose the right inductive bias?

Evaluation

- Empirical error too optimistic
- True error usually unobtainable
- General idea
  - Separate available examples into training set and test set
  - Learn hypothesis on training set
  - Evaluate hypothesis on test set
- Repeat above several times with different training/test sets and average results

Summary

- Supervised learning
  - Model: $g(\mathbf{x} \mid \theta)$
  - Loss function: $E(\theta \mid \mathcal{X}) = \sum_t L\left(r^t, g(\mathbf{x}^t \mid \theta)\right)$
  - Optimization procedure: $\theta^* = \arg\min_\theta E(\theta \mid \mathcal{X})$

- Choices for each constitute inductive bias

Bayesian Learning

The Gold Standard

Bayesian Learning

- Combines prior knowledge with evidence to make predictions
- Optimal (albeit impractical) classifier
- Naïve Bayes classifier (practical)
  - Assumes independence among features

Bayes Rule

$$P(C_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_i)P(C_i)}{p(\mathbf{x})}$$

Thomas Bayes

- C_i is the class, $1 \leq i \leq K$
- $\mathbf{x}$ is the feature vector of an instance
- $P(C_i|\mathbf{x})$ = probability that instance $\mathbf{x}$ belongs to class C_i (*posterior*)
- $p(\mathbf{x}| C_i)$ = probability that an instance drawn from class C_i would be $\mathbf{x}$ (*likelihood*)
- $P(C_i)$ = probability of class C_i (*prior*)
- $p(\mathbf{x})$ = probability of instance $\mathbf{x}$ (*evidence*)

Bayes Classifier

- Classify instance $\boldsymbol{x}$ as class C_i such that

$$i = \arg\max_{1 \le k \le K} P(C_k \mid \boldsymbol{x})$$

- Since only interested in maximum, can ignore denominator $p(\boldsymbol{x})$

$$i = \arg\max_{1 \le k \le K} p(\boldsymbol{x} \mid C_k) P(C_k)$$

- If prior probability distribution of classes is uniform, then can ignore $P(C_i)$

$$i = \arg\max_{1 \le k \le K} p(\boldsymbol{x} \mid C_k)$$

Bayes Classifier

- Practical issue
  - $p(\mathbf{x}|\ C_i)$ is a joint probability distribution
  - Need to know the probability of every possible instance given every possible class
  - Even for D boolean features and K classes, that's $K*2^D$ probabilities
- Solution
  - Assume features are independent of each other

$$p(x_1, x_2, \ldots, x_D \mid C_i) = \prod_{j=1}^{D} p(x_j \mid C_i)$$

Naïve Bayes Classifier

- Given training set X
- Estimate probabilities from X

$$P(C_i) = \frac{|\{(\mathbf{x}, r) \in X \mid r = C_i\}|}{|X|}$$

$$p(x_j = v \mid C_i) = \frac{|\{(\mathbf{x}, r) \in X \mid x_j = v \; and \; r = C_i\}|}{|\{(\mathbf{x}, r) \in X \mid r = C_i\}|}$$

- Classify new instance $\boldsymbol{x}$ as class C_i such that

$$i = \arg\max_{1 \le k \le K} P(C_k) * \prod_{j=1}^{D} p(x_j \mid C_k)$$

Naïve Bayes Classifier

- Independence assumption rarely true
  - E.g., is “price” independent from “engine power”?
- Naïve Bayes classifier still does surprisingly well
- Simple, effective baseline for other learners

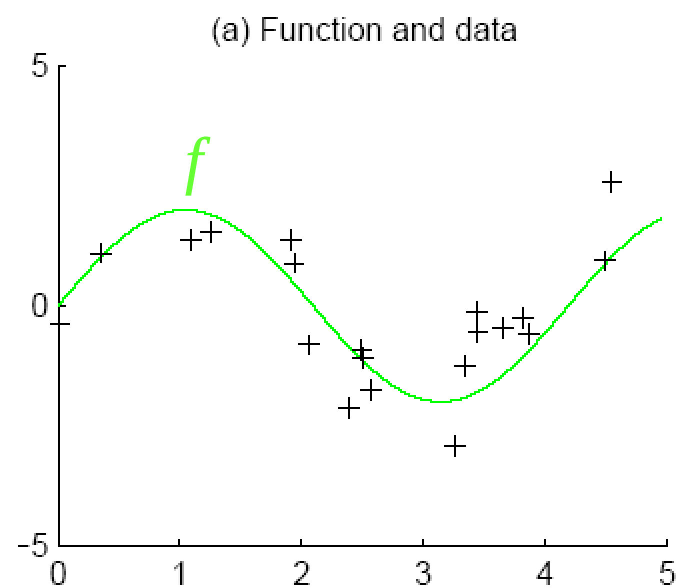
Parametric Methods

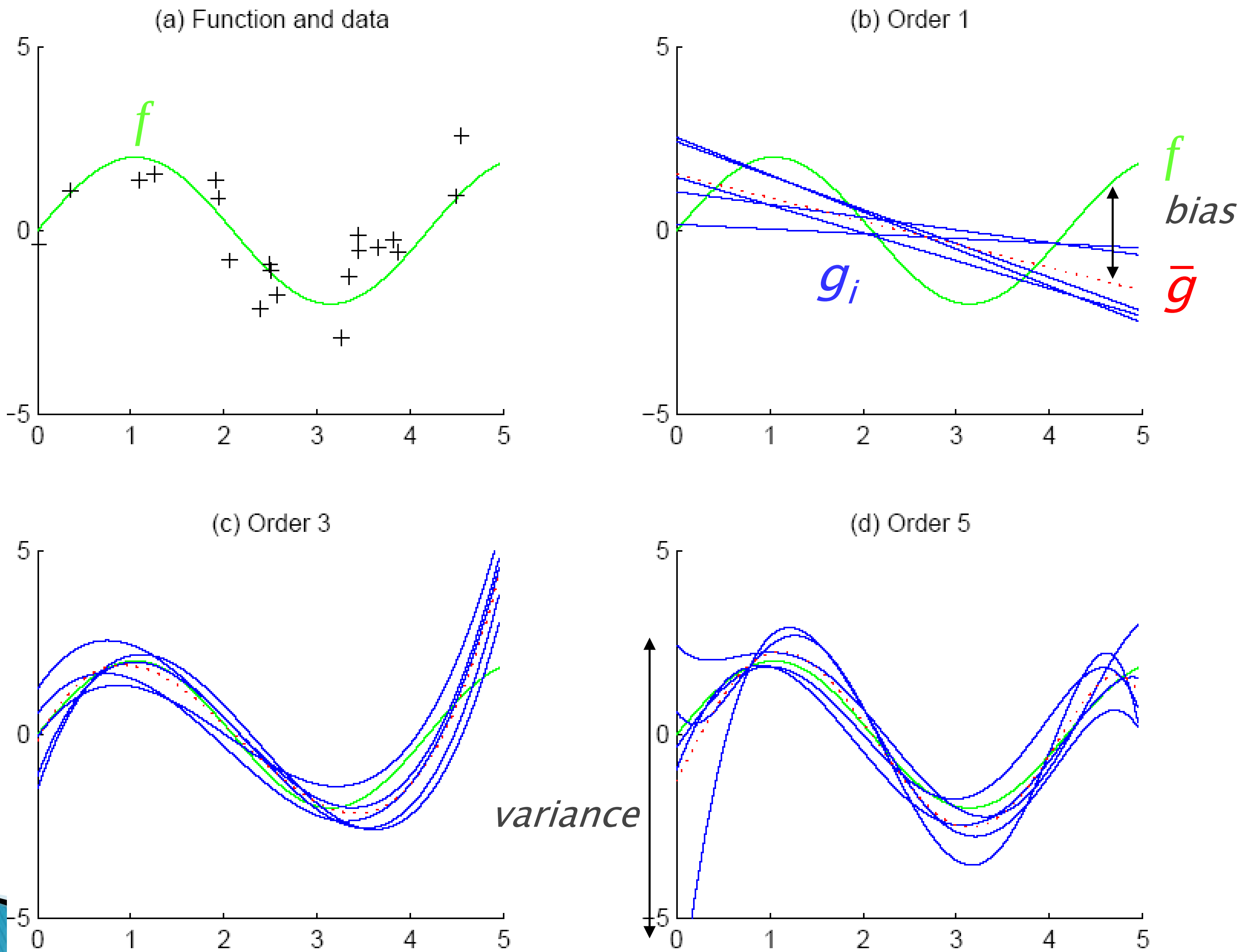
Parametric Methods

- Assume a model of the underlying distribution $p(x|\theta)$
- Estimate the parameters θ of the model based on the training set X
- Bias/variance dilemma
- Model selection

Regression Example

- f(x) = 2 sin (1.5x)
- Noise N(0,1)
- Five samples taken (one below)
- Fit order 1, 3 and 5 polynomials

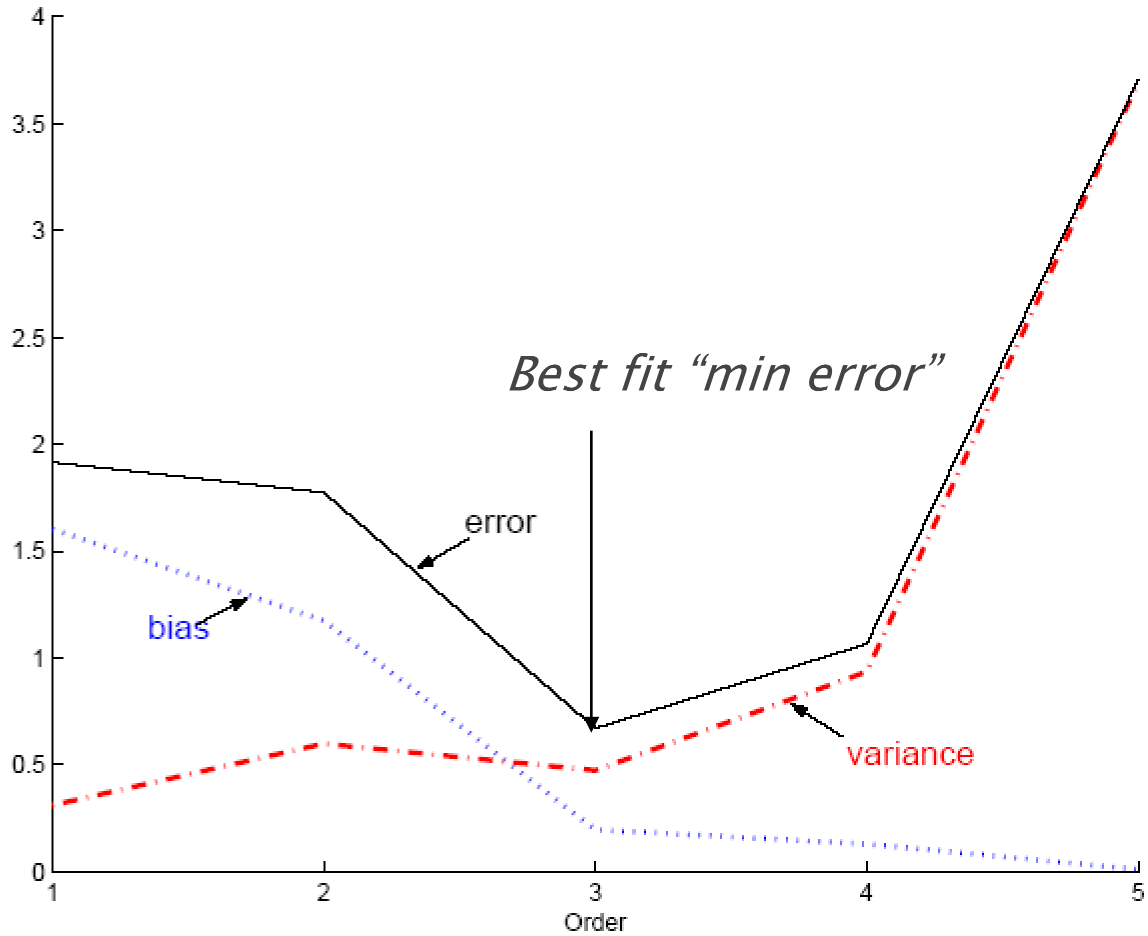



(a) Function and data
f
(b) Order 1
f
bias
g_i
$\bar{g}$
(c) Order 3
variance
(d) Order 5

Bias/Variance Dilemma

- Example
  - $g_i(x)=2$ has no variance and high bias
  - $g_i(x)= \sum_t r^t_i/N$ has lower bias with higher variance
- As we increase complexity,
  - Bias decreases (a better fit to data) and
  - Variance increases (fit varies more with data)

Polynomial Regression



Model Selection

- Cross-validation
  - Measure generalization accuracy by testing on data unused during training (validation set)
- Regularization
  - Penalize complex models
  - E'=error on data + λ * model complexity
- Minimum description length (MDL)
  - Best model minimizes description of model plus description of data given model

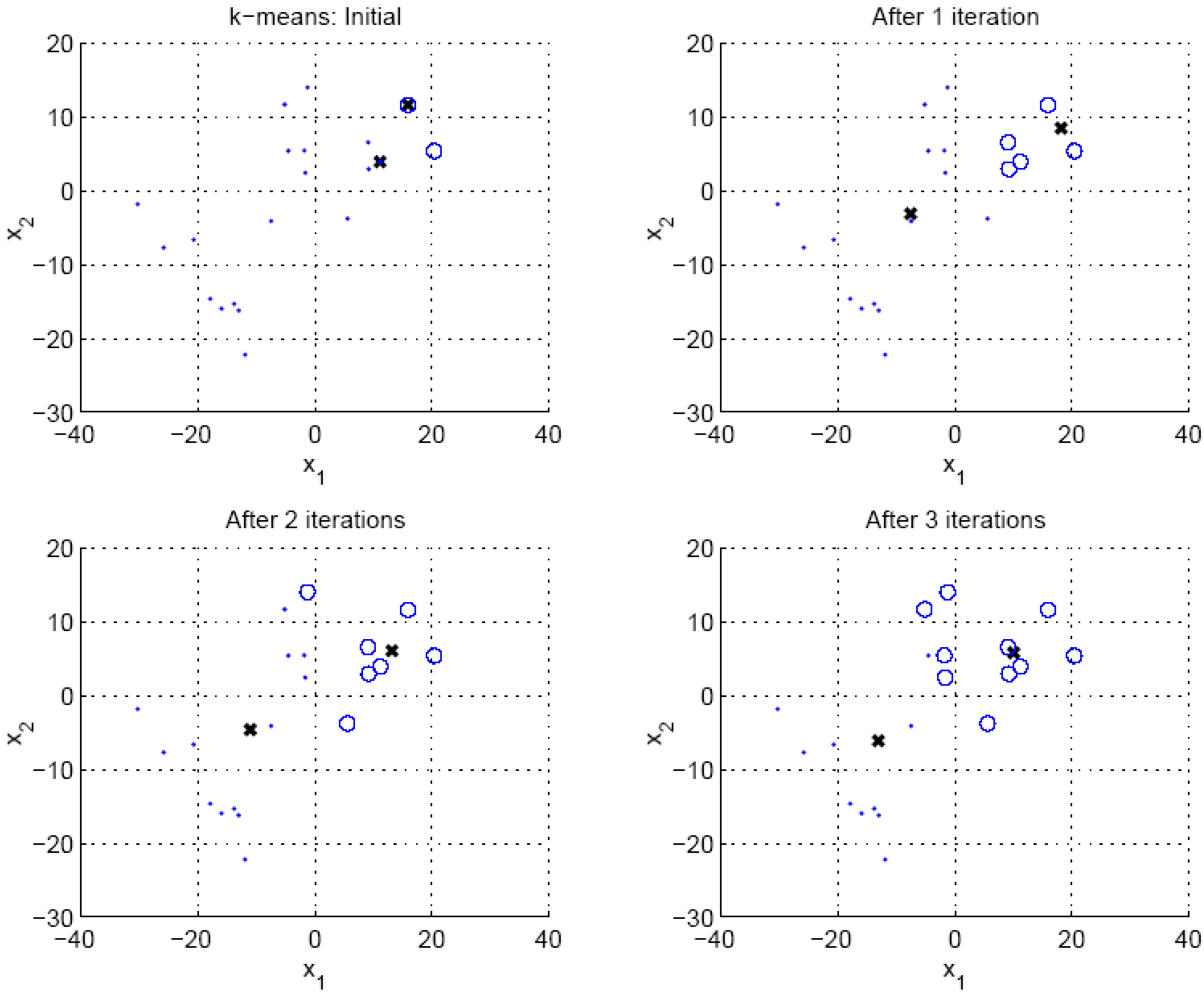
Nonparametric Methods

Nonparametric Methods

- Form of underlying distributions unknown
- But still want to perform classification and regression
- Clustering
  - k-means clustering
- Instance-based learning
  - k-nearest-neighbor classifier

k–means Clustering

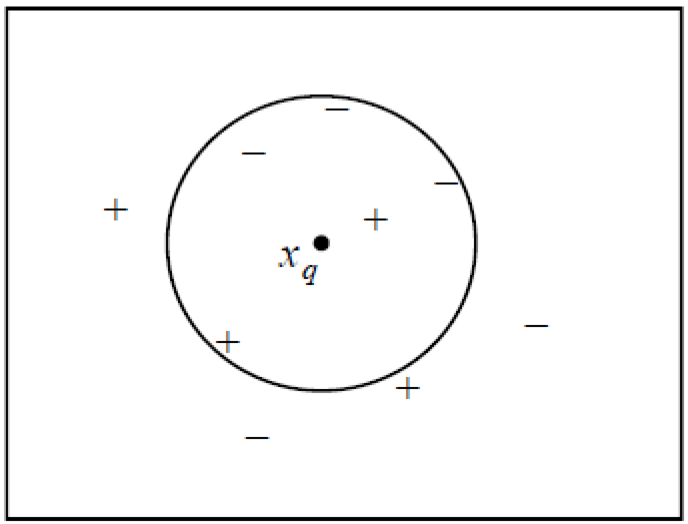
- Unsupervised learning
- Partition instances into k disjoint sets
- Each set has a representative instance m_i
- Place instance x into set i such that *distance*(x, m_i) is minimal
- Choose new central m_i for each set
- Repeat until m_i converge

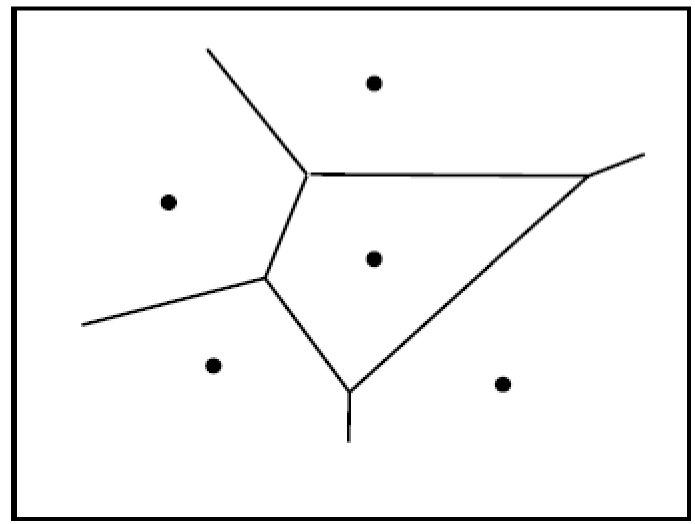

k−means: Initial
After 1 iteration
After 2 iterations
After 3 iterations
x_1
x_2

k-Nearest Neighbor Classifier

- Find k nearest neighbors to x
- Classification: g(x) = majority class among k neighbors
- Regression: g(x) = mean value of k neighbors

Voronoi Diagram of k-NN





k-NN Distance Metrics

- Euclidean distance for numeric features
  - Normalize feature values
- Hamming distance for discrete features
  - Distance = 1 if feature values differ, else 0

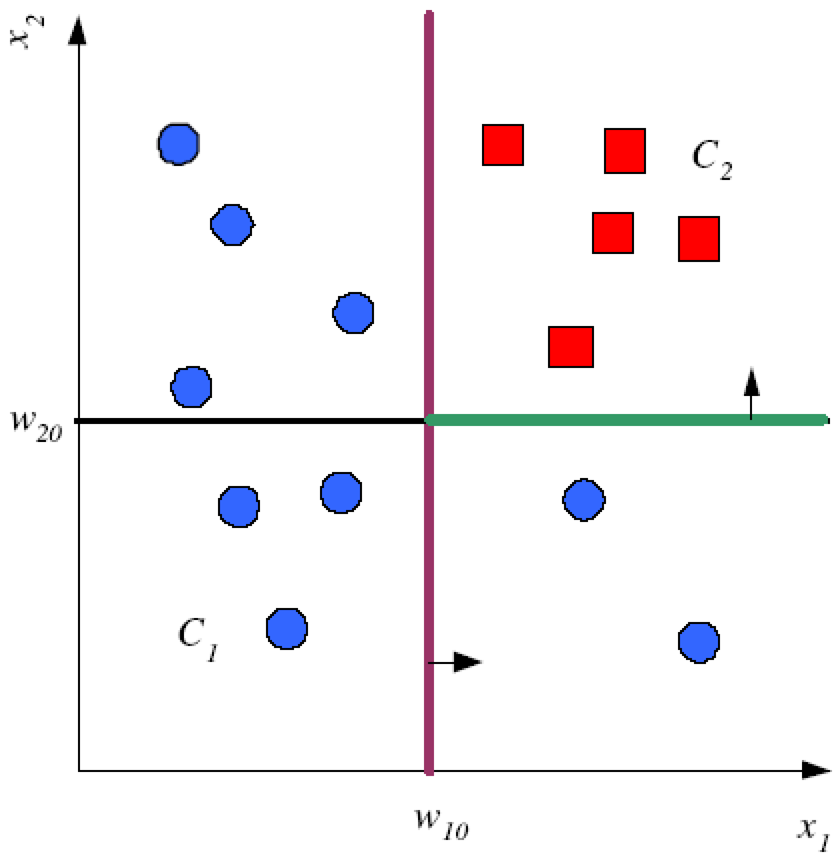
How to choose k?

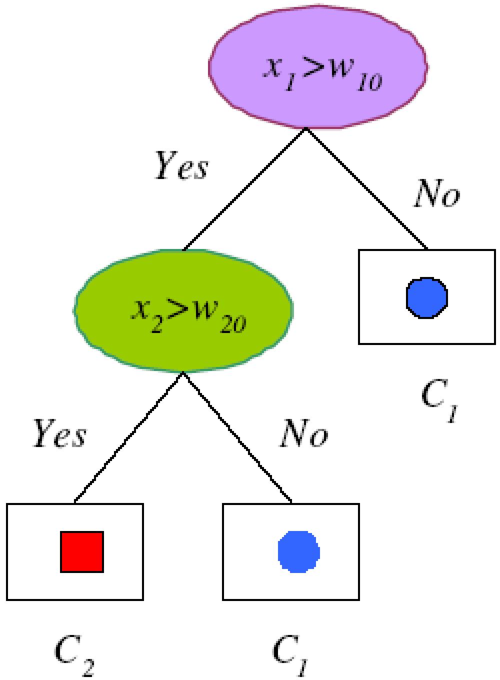
- When *k* is small, single instances matter
  - Bias is small, variance is large (undersmoothing)
  - High complexity
- As *k* increases, we average over more instances
  - Variance decreases but bias increases (oversmoothing)
  - Low complexity
- Cross-validation is used to tune *k*

Decision Trees

Popular Non-Parametric Method

Decision Tree Example





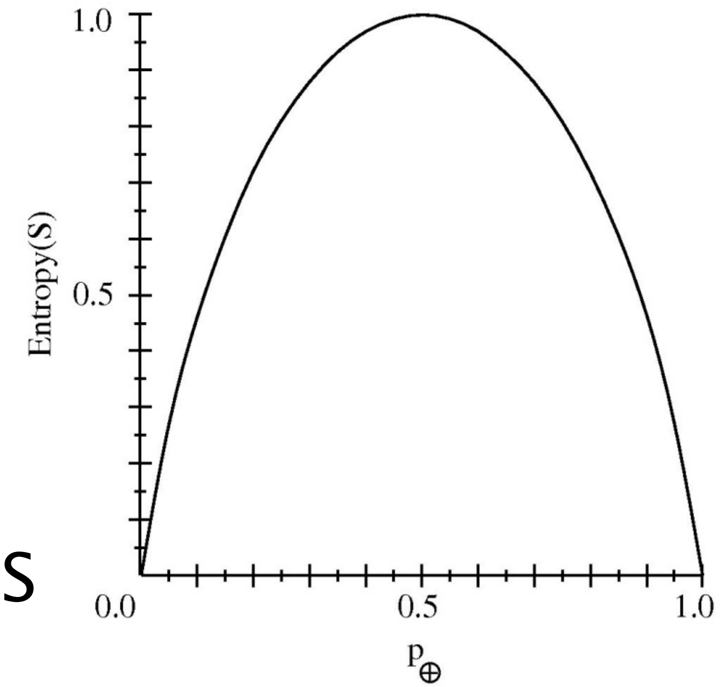
Learning Algorithm

- Create root node with all examples X
- Call GenerateTree (root, X)

```
GenerateTree (node, X)
   If node is "pure" enough
   Then assign class to node and return
   Else Choose "best" split feature F
          Foreach value v of F
              X' = examples in X where F = v
              Create childNode with examples X'
              GenerateTree (childNode, X')
```

Pure/Best based on Entropy

- S = set of training examples
- $p_{\oplus}$ = proportion of positive examples in S
- $p_{\ominus}$ = proportion of negative examples in S
- $Entropy(S) \equiv - p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$
- Entropy measures impurity of S
- "Pure" means low entropy
- "Best" means maximally reduces entropy

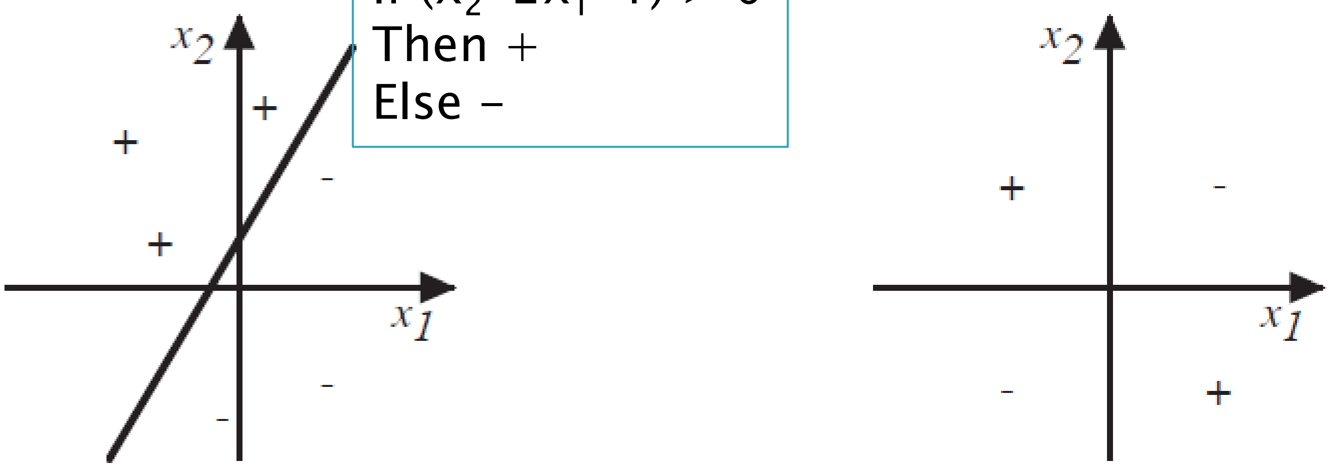


Linear Discrimination

Linear Discrimination

- Assume instances of classes are linearly separable
- Estimate parameters of linear discriminant

If $(x_2 - 2x_1 - 1) > 0$
Then +
Else –



(a) (b)

Discriminant-based vs. Likelihood-based Classification

- Classification (K classes)

$$choose\ C_i\ if\ \ g_i(\boldsymbol{x}) = \max_{j=1}^{K} g_j(\boldsymbol{x})$$

- Likelihood-based classification
  - Estimate priors $P(C_i)$ and likelihoods $p(\boldsymbol{x}|C_i)$
  - Define $g_i(\boldsymbol{x})$ in terms of the posteriors

$$g_i(\boldsymbol{x}) = \log P(C_i \mid \boldsymbol{x})$$

  - Requires knowledge of types of densities

Discriminant-based vs. Likelihood-based Classification

- Discriminant-based classification
  - Learn model of boundaries between classes, instead of densities of bounded regions

$$g_i(\boldsymbol{x} \mid \Phi_i)$$

  - Where Φ_i are model parameters of the boundary

Linear Discriminant

- Simple
- Requires only O(d) space to store and O(d) time for classification
- Weight w_i indicates importance of feature x_i
- Try linear model before trying more complicated model

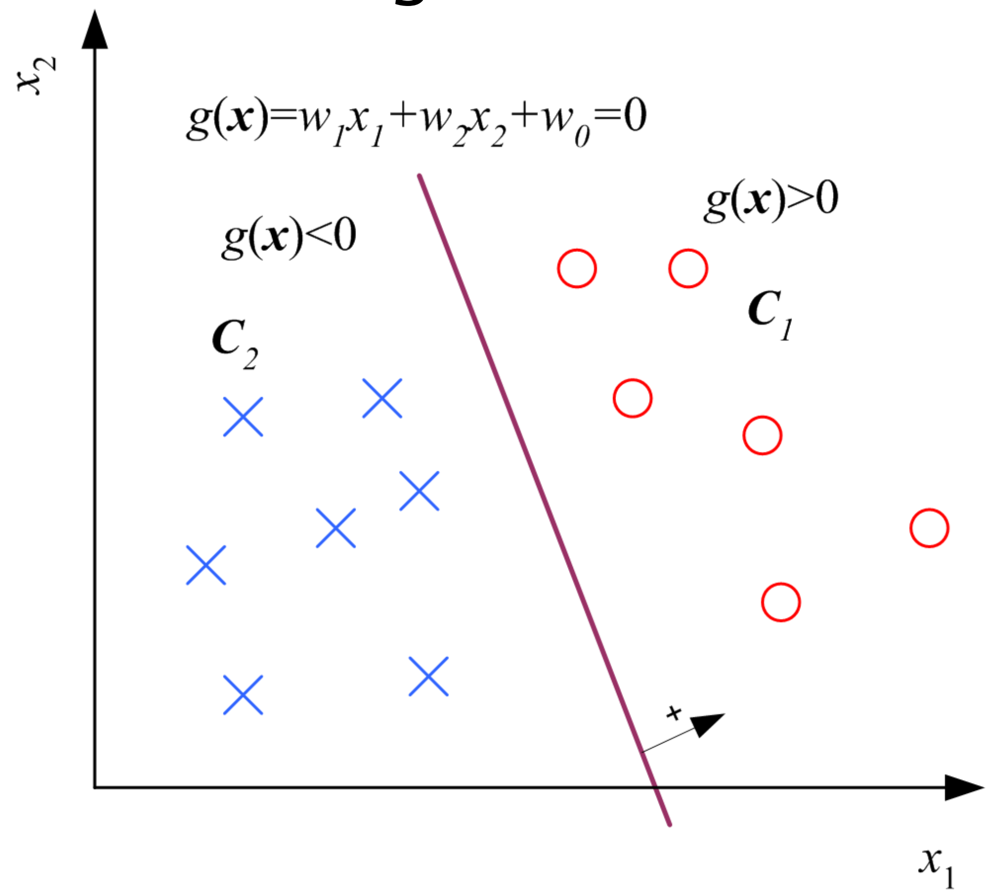
$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T \mathbf{x} + w_{i0} = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$$

Two-Class Case

- Weight vector $\mathbf{w}$ defines a hyperplane dividing the instance space into two regions

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

$$\text{Choose} \begin{cases} C_1 & \text{if } g(\mathbf{x}) > 0 \\ C_2 & \text{otherwise} \end{cases}$$



Gradient Descent

- Start with random **w**
- Update **w** in the opposite direction of the gradient vector

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \forall i$$

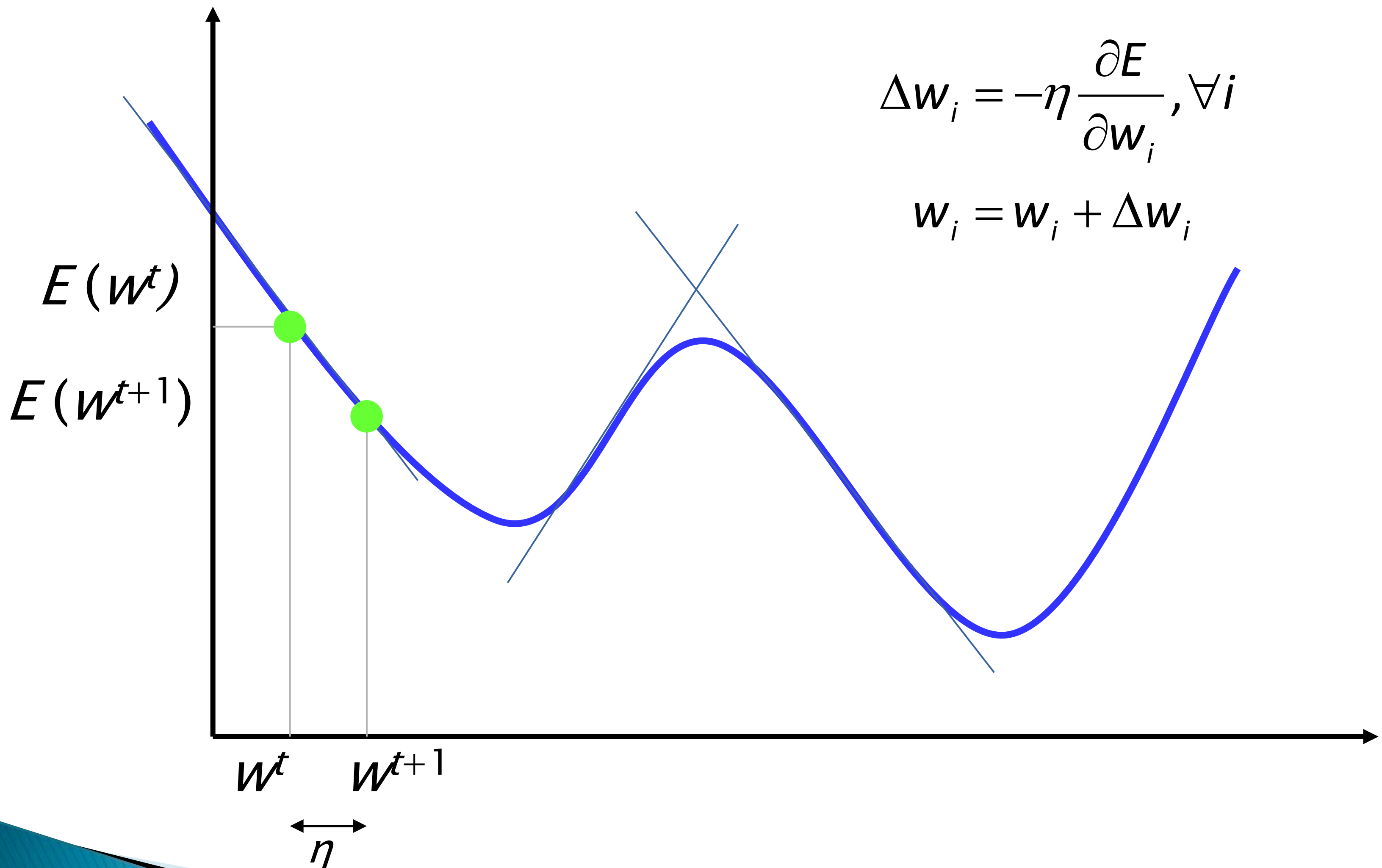
$$w_i = w_i + \Delta w_i$$

- Distance of update determined by step size (or learning factor) η
- Continue update until gradient is zero
  - May be a local minimum

Gradient Descent

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \forall i$$

$$w_i = w_i + \Delta w_i$$



Evaluation

(Ab)using Statistics

M.I.T. unveils its 24-hour All-Sports Network

Error

- Given sample S from all possible examples D
- Learner L learns hypothesis h based on S
- Sample error: $error_S(h)$
- True error: $error_D(h)$

- Example
  - Hypothesis h misclassifies 12 of 40 examples in S
  - $error_S(h) = 0.3$
  - What is $error_D(h)$?

Error

- Learner A learns hypothesis h_A on sample S
- Learner B learns hypothesis h_B on sample S
- Observe: $error_S(h_A) < error_S(h_B)$
- Is $error_D(h_A) < error_D(h_B)$?
- Is learner A better than learner B?

Evaluation

- How can we estimate the true error of a classifier?
- How can we determine if one learner is better than another?
- Using sample error is too optimistic
- Using error on a separate test set is better, but might still be misleading
- Repeating above for multiple iterations, each with different training/testing sets, yields better estimate of true error

Evaluation Issues

- Be careful not to give learner any information about data used to test performance
- Most common violation: Tweaking parameters after seeing test performance
  - Red flag: “We chose parameter x based on experience.”

Train/Test Split

- Given dataset X
- For each of K trials
  - Randomly divide X into training set (2/3) and testing set (1/3)
  - Learn classifier on training set
  - Test classifier on testing set (compute error)
- Compute average error over K trials
- Problem
  - Training and testing sets overlap between trials
  - Biases the results

K-fold Cross Validation

- Given dataset X
- Partition X into K disjoint sets $X_1, \ldots, X_K$
- For i = 1 to K
  - Learn classifier on training set $X - X_i$
  - Test classifier on testing set X_i (compute error)
- Compute average error over K trials
- Testing sets no longer overlap
- Training sets still overlap

Measuring Classifier Performance

- Confusion matrix

	Predicted class		
True class	Positive	Negative	Total
Positive	tp: true positive	fn: false negative	p
Negative	fp: false positive	tn: true negative	n
Total	p’	n’	N

Performance Measures (2-class)

Name	Formula
error accuracy	(fp + fn)/N (tp + tn)/N
tp-rate fp-rate	tp/p fp/n
precision recall	tp/p' tp/p = tp_rate
sensitivity specificity	tp/p = tp_rate tn/n = 1 – fp_rate

F-measure: $$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Example: OneR on Labor

OneR learns classifier based on one attribute (high bias).

```
=== Run information ===

Scheme:       weka.classifiers.rules.OneR -B 6
Relation:     labor-neg-data
Instances:    57
Attributes:   17
              duration
              wage-increase-first-year
              wage-increase-second-year
              wage-increase-third-year
              cost-of-living-adjustment
              working-hours
              pension
              standby-pay
              shift-differential
              education-allowance
              statutory-holidays
              vacation
              longterm-disability-assistance
              contribution-to-dental-plan
              bereavement-assistance
              contribution-to-health-plan
              class
…
```

Example: OneR on Labor (cont.)

```
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

wage-increase-first-year:
        < 2.9  -> bad
        >= 2.9 -> good
        ?      -> good
(48/57 instances correct)


Time taken to build model: 0 seconds

…
```

Example: OneR on Labor (cont.)

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          43       75.4386 %
Incorrectly Classified Instances        14       24.5614 %
Kappa statistic                          0.4063
Mean absolute error                      0.2456
Root mean squared error                  0.4956
Relative absolute error                 53.6925 %
Root relative squared error            103.7961 %
Coverage of cases (0.95 level)          75.4386 %
Mean rel. region size (0.95 level)      50      %
Total Number of Instances               57

…
```

Example: OneR on Labor (cont.)

```
=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.45      0.081      0.75      0.45      0.563      0.684    bad
                 0.919     0.55       0.756     0.919     0.829      0.684    good
Weighted Avg.    0.754     0.385      0.754     0.754     0.736      0.684

=== Confusion Matrix ===

  a  b   <-- classified as
  9 11 |  a = bad
  3 34 |  b = good
```

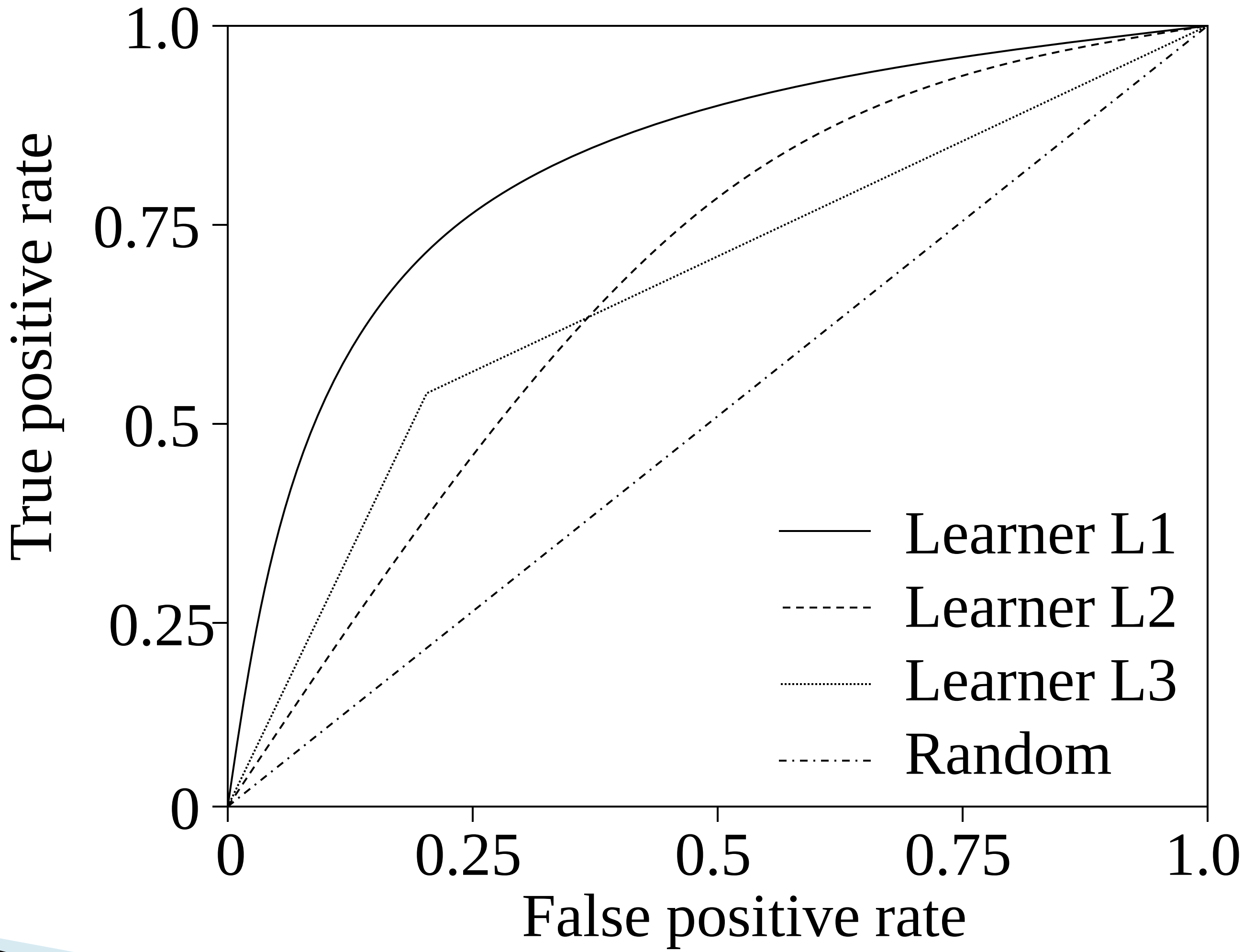
ROC Curve

- Most comparisons of machine learning algorithms use classification error
- Problems with this approach
  - May be different costs associated with false positive and false negative errors
  - Training data may not reflect true class distribution

ROC Curve

- Receiver Operating Characteristic (ROC)
  - Originated from signal detection theory
  - Common in medical diagnosis
  - Becoming common in ML evaluations
- ROC curves assess predictive behavior independent of error costs or class distributions
- Area Under ROC Curve (AUC)
  - Single measure of learning algorithm performance independent of error costs and class distributions

ROC Curve



1.0
0.75
0.5
0.25
0
True positive rate
0
0.25
0.5
0.75
1.0
False positive rate
Learner L1
Learner L2
Learner L3
Random

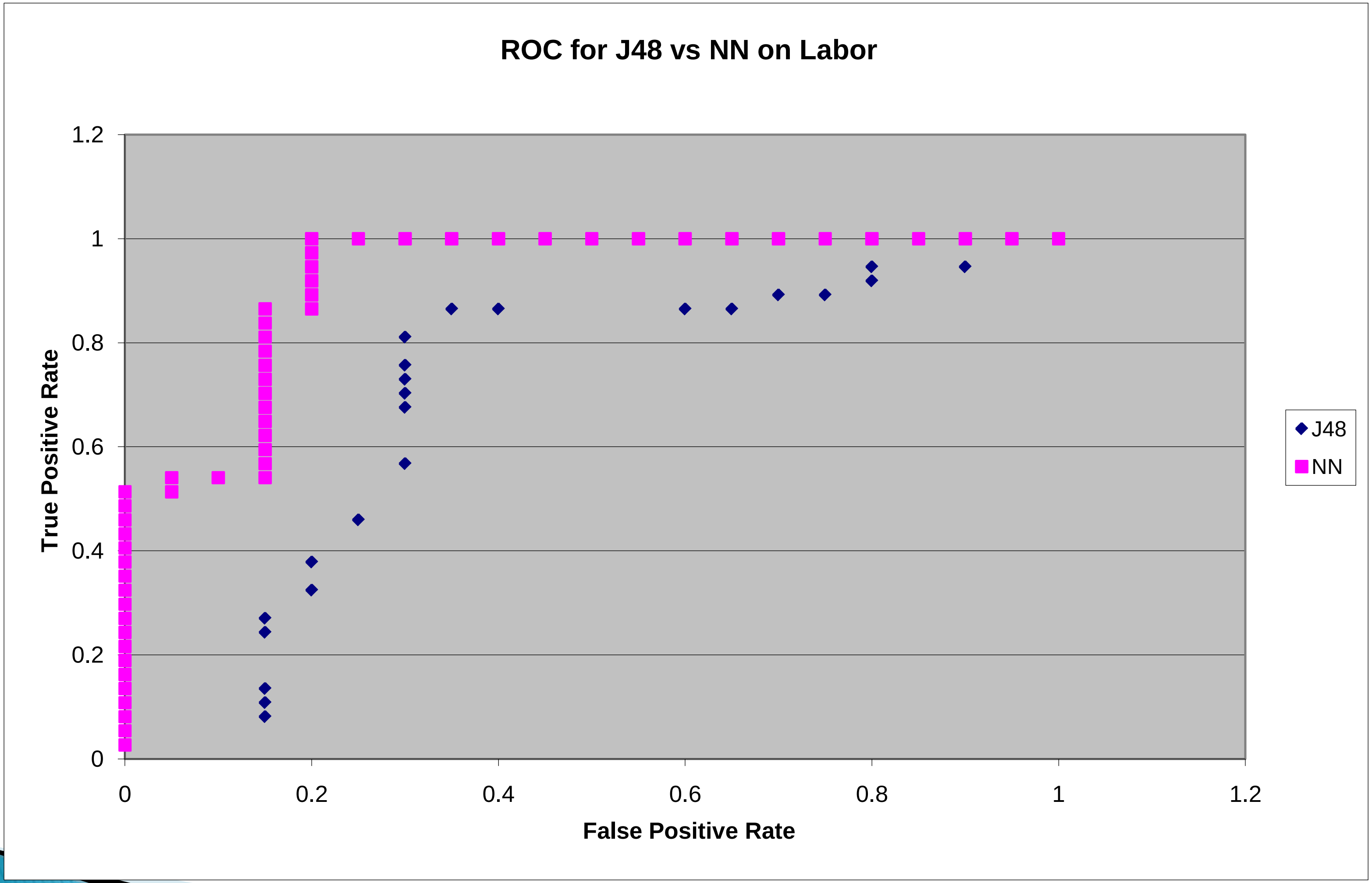
Domination in ROC Space

- Learner L1 dominates L2 if L1's ROC curve is always above L2's curve
- If L1 dominates L2, then L1 better than L2 for all possible error costs and class distributions
- If neither dominates (L2 and L3), then different classifiers are better under different conditions

Generating ROC Curve

- Assume classifier outputs $P(C|x)$ instead of just C (the predicted class for instance x)
- Let θ be a threshold such that if $P(C|x) > \theta$, then x is classified as C, else not C
- Compute fp-rate and tp-rate for different values of θ from 0 to 1
- Plot each (fp-rate, tp-rate) and interpolate (or convex hull)
- If multiple points with same fp-rate, then average tp-rates

Decision Tree vs. Neural Network



Hypothesis Testing

- Want to claim a hypothesis H_1
  - E.g., H_1 : $error_D(h) < 0.10$
- Define the opposite of H_1 to be the null hypothesis H_0
  - E.g., H_0 : $error_D(h) \geq 0.10$
- Perform experiment collecting data about $error_D(h)$
- With what probability can we reject H_0?

Comparing Two Learners

- K-fold cross-validated paired t test
  - Paired test: Both learners get same train/test sets
  - Use K-fold CV to get K training/testing folds
  - p_i^1, p_i^2 : Errors of learners 1 and 2 on fold i
  - $p_i = p_i^1 - p_i^2$: Paired difference on fold i
  - Null hypothesis is whether p_i has mean 0

$$H_0 : \mu = 0 \text{ vs. } H_1 : \mu \neq 0$$

$$m = \frac{\sum_{i=1}^{K} p_i}{K} \qquad s^2 = \frac{\sum_{i=1}^{K} (p_i - m)^2}{K-1}$$

$$\frac{\sqrt{K}(m-0)}{s} = \frac{\sqrt{K} \cdot m}{s} \sim t_{K-1} \text{ Accept if in } \left(-t_{\alpha/2,K-1}, t_{\alpha/2,K-1}\right)$$

Comparing Two Learners

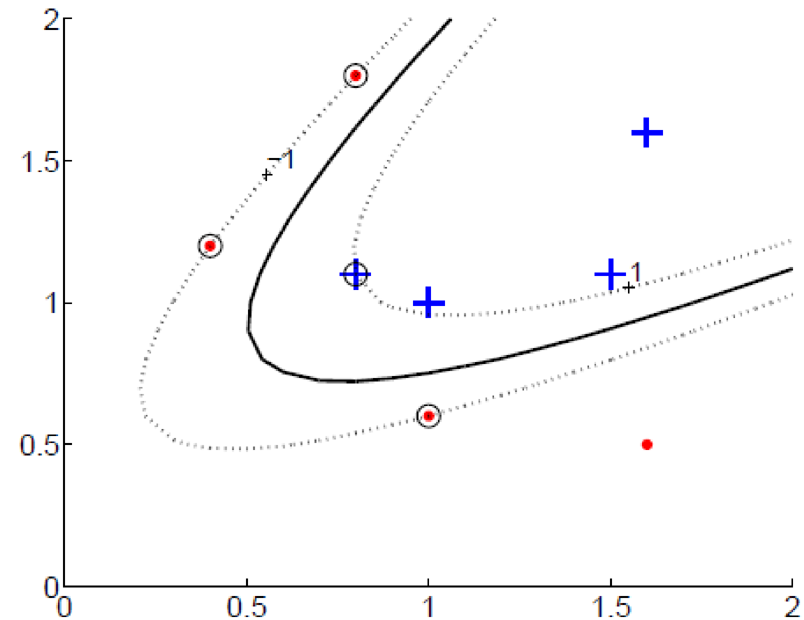
```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   8
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       10/6/10 12:00 AM


Dataset                     (1) rules.On | (2) bayes
--------------------------------------------------
loan                      (100)   39.50 |   84.50 v
contact-lenses            (100)   72.17 |   76.17
iris                      (100)   93.53 |   95.53
labor-neg-data            (100)   72.77 |   93.57 v
segment                   (100)   63.33 |   81.12 v
soybean                   (100)   39.75 |   92.94 v
weather                   (100)   36.00 |   67.50
weather.symbolic          (100)   38.00 |   57.50
--------------------------------------------------
                               (v/ /*) |   (4/4/0)
```

Conclusions

Machine Learning

- Computational process that improves performance based on experience
- Important concepts
  - Bias vs. variance
  - Model selection
  - Discriminant-based vs. likelihood-based classification



Methods

- Bayesian learning
- Parametric methods (regression)
- Nonparametric methods (nearest neighbor)
- Decision trees
- Linear discrimination
- Neural networks
- Kernel machines
- Ensembles
- Relational learning

Evaluation

- Estimate true error based on sample error
- Measure performance of learning algorithm
- Compare performance of learning algorithms
- Hypothesis testing and statistical significance
- Cross-validation
- ROC curve

Other Topics

- Dimensionality reduction (feature selection)
- Semi-supervised learning
- Hidden Markov models
- Belief networks
- Scalability
- Learning theory
- Privacy and ethics

Summary

- Machine learning seeks to give computers the ability to improve their performance based on experience
- Many mature methods and theoretical results
- Basis of multi-billion dollar industry
- Much research left to be done