Course Summary

- Introduction:
  - Basic problems and questions in machine learning.

- Linear Classifiers
  - Naïve Bayes
  - Logistic Regression
  - LMS

- Five Popular Algorithms
  - Decision trees (C4.5)
  - Neural networks (backpropagation)
  - Probabilistic networks (Naïve Bayes; Mixture models)
  - Support Vector Machines (SVMs)
  - Nearest Neighbor Method

- Theories of Learning:
  - PAC, Bayesian, Bias-Variance analysis

- Optimizing Test Set Performance:
  - Overfitting, Penalty methods, Holdout Methods, Ensembles

- Sequential Data
  - Hidden Markov models, Conditional Random Fields; Hidden Markov SVMs
Course Summary

- Goal of Learning
- Loss Functions
- Optimization Algorithms
- Learning Algorithms
- Learning Theory
- Overfitting and the Triple Tradeoff
- Controlling Overfitting
- Sequential Learning
- Statistical Evaluation
Goal of Learning

- Classifier: $\hat{y} = f(x)$ “Do the right thing!”
- Conditional probability estimator: $P(y|x)$
- Joint probability estimator: $P(x,y)$
  - compute conditional probability at classification time
Loss Functions

- Cost matrices and Bayesian decision theory
  - Minimize expected loss
  - Reject option

- Log Likelihood: \( \sum_k -I(y=k) \log P(y=k|x,h) \)

- 0/1 loss: need to approximate
  - squared error
  - mutual information
  - margin slack ("hinge loss")
Optimization Algorithms

- None: direct estimation of $\mu, \Sigma, P(y), P(x \mid y)$
- Gradient Descent: LMS, logistic regression, neural networks, CRFs
- Greedy Construction: Decision trees
- Boosting
- None: nearest neighbor
Learning Algorithms

- LMS
- Logistic Regression
- Multivariate Gaussian and LDA
- Naïve Bayes (gaussian, discrete, kernel density estimation)
- Decision Trees
- Neural Networks (squared error and softmax)
- k-nearest neighbors
- SVMs (dot product, gaussian, and polynomial kernels)
- HMMs/CRFs/averaged perceptron
The Statistical Problem: Overfitting

Goal: choose $h$ to optimize test set performance

Triple tradeoff: sample size, test set accuracy, complexity
- For fixed sample size, there is an accuracy/complexity tradeoff

Measures of complexity:
- $|H|$, VC dimension, log $P(h)$, $||w||$, number of nodes in tree

Bias/Variance analysis
- Bias: systematic error in $h$
- Variance: high disagreement between different $h$’s
- $\text{test error} = \text{Bias}^2 + \text{variance} + \text{noise}$ (square loss, log loss)
- $\text{test error} = \text{Bias} + \text{unbiased-variance} – \text{biased-variance} (0/1 \text{ loss})$

Most accurate hypothesis on training data is not usually most accurate on test data

Most accurate hypothesis on test data may be deliberately wrong (i.e., biased)
Controlling Overfitting

**Penalty Methods**
- Pessimistic pruning of decision trees
- Weight decay
- Weight elimination
- Maximum Margin

**Holdout Methods**
- Early stopping for neural networks
- Reduce-error pruning

**Combined Methods (use CV to set penalty level)**
- Cost-complexity pruning
- CV to choose pruning confidence, weight decay level, SVM parameters $C$ and $\sigma$

**Ensemble Methods**
- Bagging
- Boosting
## Off-The-Shelf Criteria

<table>
<thead>
<tr>
<th>Criterion</th>
<th>LMS</th>
<th>Logistic</th>
<th>LDA</th>
<th>Trees</th>
<th>Nets</th>
<th>NNbr</th>
<th>SVM</th>
<th>NB</th>
<th>Boosted Trees</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed data</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Missing values</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>some</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Outliers</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>disc</td>
<td>yes</td>
</tr>
<tr>
<td>Monotone transforms</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>some</td>
<td>no</td>
<td>no</td>
<td>disc</td>
<td>yes</td>
</tr>
<tr>
<td>Scalability</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Irrelevant inputs</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>some</td>
<td>no</td>
<td>no</td>
<td>some</td>
<td>some</td>
<td>yes</td>
</tr>
<tr>
<td>Linear combinations</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>some</td>
<td>yes</td>
<td>yes</td>
<td>some</td>
</tr>
<tr>
<td>Interpretable</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>some</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Accurate</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
What We’ve Skipped

- Unsupervised Learning
  - Given examples $X_i$
  - Find: $P(X)$
  - Clustering
  - Dimensionality Reduction
Reinforcement Learning: Agent interacting with an environment

- At each time step $t$
  - Agent perceives current state $s$ of environment
  - Agent choose action to perform according to a policy: $a = \pi(s)$
  - Action is executed, environment moves to new state $s'$ and returns reward $r$

- Goal: Find $\pi$ to maximizes long term sum of rewards
What We Skipped (3):
Semi-Supervised Learning

- Learning from a mixture of supervised and unsupervised data
- In many applications, unlabeled data is very cheap
  - BodyMedia
  - Task Tracer
  - Natural Language Processing
  - Computer Vision
- How can we use this?
Research Frontier

- More complex data objects
  - sequences, images, networks, relational databases
- More complex runtime tasks
  - planning, scheduling, diagnosis, configuration
- Learning in changing environments
- Learning online
- Combining supervised and unsupervised learning
- Multi-agent reinforcement learning
- Cost-sensitive learning; imbalanced classes
- Learning with prior knowledge