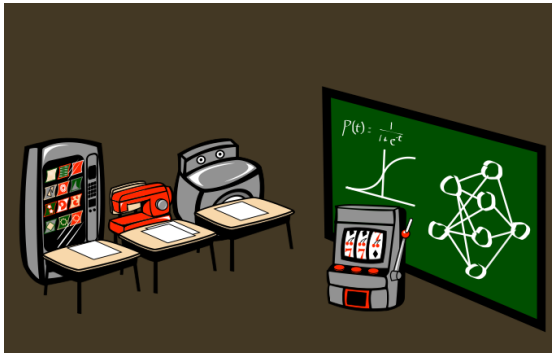
An Introduction to Machine Learning

LING-7800 Computational Lexical Semantics

Presented by Lee Becker and Shumin Wu

What is Machine Learning?



What is Machine Learning?

- AKA
  - Pattern Recognition
  - Data Mining

What is Machine Learning?

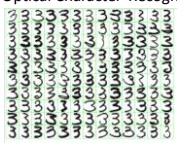
- Programming computers to do tasks that are (often) easy for humans to do, but hard to describe algorithmically.
- Learning from observation
- Creating models that can predict outcomes for unseen data
- Analyzing large amounts of data to discover new patterns

What is Machine Learning?

- Isn't this just statistics?
  - Cynic's response: Yes
  - CS response: Kind of
    - Unlike in statistics, machine learning is also concerned with the complexity, optimality, and tractability in learning a model
    - Statisticians are often dealing with much smaller amounts of data.

Problems / Application Areas

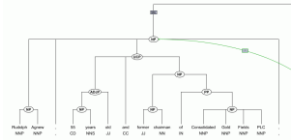
Optical Character Recognition



Face Recognition

Movie Recommendation

Speech and Natural Language Processing



Ok, so where do we start?

- Observations
  - Data! The more the merrier (usually)
- Representations
  - Often raw data is unusable, especially in natural language processing
  - Need a way to represent observations in terms of its properties (features)
- Feature Vector

Machine Learning Paradigms

- Supervised Learning
  - Deduce a function from labeled training data to minimize labeling error on future data
- Unsupervised Learning
  - Learning with unlabeled training data
- Semi-supervised Learning
  - Learning with (usually small amount of) labeled training data and (usually large amount of)
- Active Learning
  - Actively query for specific labeled training data
- Reinforcement Learning
  - Learn actions in environment to maximize (often long-term) reward

Supervised Learning

- Given a set of instances, each with a set of features, and their class labels, deduce a function that maps from feature values to labels:

Given:

$$\begin{pmatrix} x_{11}, x_{12}, x_{13} \dots x_{1m} \\ x_{21}, x_{22}, x_{23} \dots x_{2m} \\ \dots \\ x_{n1}, x_{n2}, x_{n3} \dots x_{nm} \end{pmatrix} \quad \begin{pmatrix} Y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix}$$

Find:

$f(\mathbf{x}) = \hat{y}$

$f(\mathbf{x})$ is called a classifier.
The way and/or parameters of $f(\mathbf{x})$ is chosen is called a classification model.

Supervised Learning

- Stages
  - Train model on data
  - Tune parameters of the model
  - Select best model
  - Evaluate

Evaluation

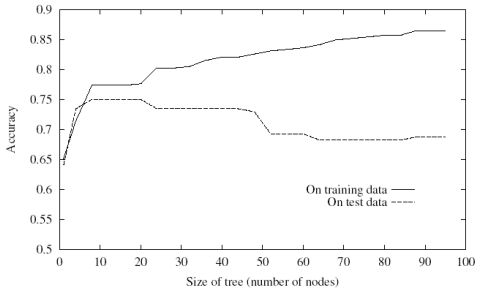
- How do we select the best model?
- How do we compare machine learning algorithms versus one another?
- In supervised learning / classification typically comparing model accuracy
  - Number of correctly labeled instances

Evaluation

- But what are we comparing against?
- Typically the data is divided into three parts
  - Training
  - Development
  - Test / Validation
- Typically accuracy on the validation set is reported
- Why all this extra effort?
  - The goal in machine learning is to select the model that does the best on unseen data
  - This divide is an attempt to keep our experiment honest
  - Avoids overfitting

Evaluation

- Overfitting



Types of Classification Models

- Generative Models
  - Model class-conditional pdfs and prior probabilities (Bayesian approach)
  - "Generative" since sampling can generate synthetic data points
  - Popular models:
    - naïve Bayes
    - Bayesian networks
    - Gaussian mixture model
- Discriminative Models
  - Directly estimate posterior probabilities
  - No attempt to model underlying probability distributions (frequentist approach)
  - Popular models:
    - linear discriminant analysis
    - support vector machine
    - decision tree
    - boosting
    - neural networks

heavily borrowed from Sargur N. Srihari

Naïve Bayes

- Assumes that when class label is known the features are independent:

$$f(\mathbf{x}) = \arg\max_{y} \; p(y)\prod_{i=1}^{m} p(x_i|y)$$

Naïve Bayes Dog vs Cat Classifier

- 2 features: weight & how frequent it chases mouse

mouse chase	weight	label
0.7	55	dog
0.05	15	dog
0.2	100	dog
0.25	42	dog
0.2	32	dog
0.6	25	cat
0.2	15	cat
0.55	8	cat
0.15	12	cat
0.4	15	cat

Given an animal that weighs no more than 20 lbs and chases mouse at least 21% of time, is it a cat or dog?

$f(dog, w \le 20, m \ge .21) =$
$p(dog)\,p(w \le 20 \mid dog)\,p(m \ge 0.21 \mid dog) =$
$0.5 \times 0.2 \times 0.4 = 0.04$

$f(cat, w \le 20, m \ge .21) =$
$p(cat)\,p(w \le 20 \mid cat)\,p(m \ge 0.21 \mid cat) =$
$0.5 \times 0.4 \times 0.6 = 0.12$

So, it's cat! In fact, naïve Bayes is 75% certain it's a cat over a dog.

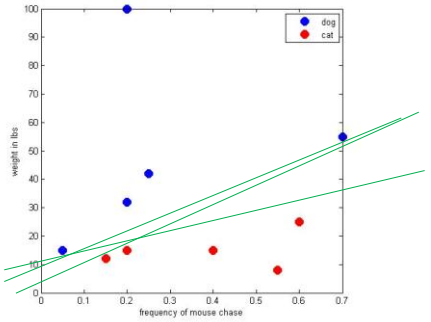
Linear Classifier

- Features have linear relationships with each other:

$$g(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \ldots + \beta_m x_m$$

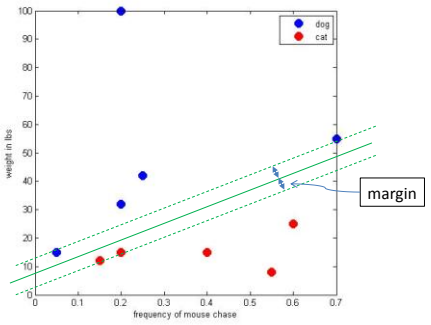
$$f(\mathbf{x}) = \begin{pmatrix} \text{class } 1 & \text{if } g(\mathbf{x}) \geq 0 \\ \text{class } 2 & \text{if } g(\mathbf{x}) < 0 \end{pmatrix}$$

Linear Classifier Example



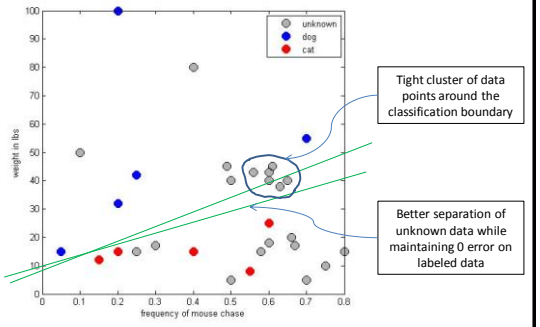
There are infinite number of answers… So, which one is the "best"???

Maximum Margin Linear Classifier



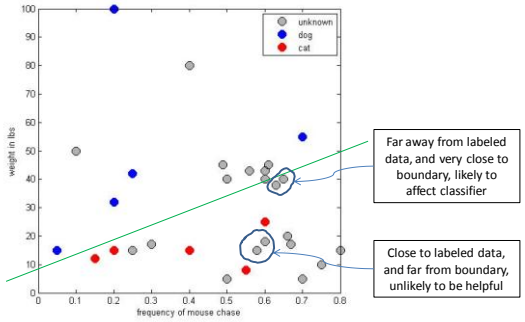
Choose the line that maximizes the margin. (What SVM does).

Semi-Supervised Learning



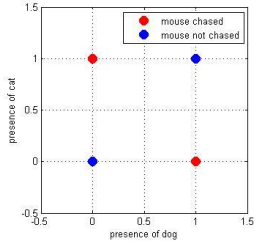
Active Learning

If we can choose to query the labels of a few unknown data points, which ones would be the most helpful?



Linear Classifier Limitation

Suppose we want to model whether the mouse will be chased in the presence of dog/cat. If either a dog or a cat is present, the mouse will be chased, but if both the dog and the cat is present, the dog will chase the cat and ignore the mouse.



Can we draw a straight line separating the 2 classes?

Decision Trees

- Reaches decision by performing a sequence of tests
  - Like a battery of *if… then* cases
  - Two Types of Nodes
    - Decision Nodes
    - Leaf Nodes
- Advantages
  - Output easily understood by humans
  - Able to learn complex rules that are impossible for a linear classifier to detect

Decision Trees

- Trivial (Wrong Approach)
  - Construct a decision tree that has one path to a leaf for each example
  - Enumerate rules for all attributes of all data points
  - Issues
    - Simply memorizes observations
    - Extracts no patterns
    - Unable to generalize

Decision Trees

- A better approach
  - Find the most important attribute first
  - Prune the tree based on these decision
  - Lather, Rinse, and Repeat as necessary

Decision Trees

- Choosing the best attribute
  - Measuring Information (Entropy):

$$I(P(v_1),\ldots,P(v_2)) = \sum_{i=1}^{n} -P(v_i)\log_2 P(v_i)$$

  - Examples:
    - Tossing a fair coin

$$I(P(heads),P(tails)) = I\left(\tfrac{1}{2},\tfrac{1}{2}\right) = -\tfrac{1}{2}\log_2\tfrac{1}{2} - \tfrac{1}{2}\log_2\tfrac{1}{2} = 1 \text{ bits}$$

    - Tossing a biased coin

$$I(P(heads),P(tails)) = I\left(\tfrac{1}{100},\tfrac{99}{100}\right) = -\tfrac{1}{100}\log_2\tfrac{1}{100} - \tfrac{99}{100}\log_2\tfrac{99}{100} = 0.08 \text{ bits}$$

    - Tossing a fair die

$$I(P(1),P(2),\ldots,P(6)) = I\left(\tfrac{1}{6},\tfrac{1}{6},\tfrac{1}{6},\tfrac{1}{6},\tfrac{1}{6},\tfrac{1}{6}\right) = 2.58 \text{ bits}$$

Decision Trees

- Choosing the best attribute cont'd
  - New information requirement due to an attribute

$$Remainder(A) = \sum_{i=1}^{v} I\left(\frac{p_i}{p_i+n_i},\frac{n_i}{p_i+n_i}\right)$$

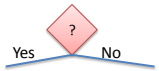
  - Gain = Original Information Requirement – New Information Requirement

$$Gain(A) = I\left(\frac{p}{p+n},\frac{n}{p+n}\right) - Remainder(A)$$

Decision Trees

Barks	Chase Mice (Freq)	Chase Ball (Freq)	Weight (Pounds)	Matching Eye Color	Category
TRUE	0.7	1	55	TRUE	Dog
TRUE	0.2	0.9	22	TRUE	Dog
TRUE	0.1	0.8	38	TRUE	Dog
TRUE	0.8	0.1	17	TRUE	Dog
TRUE	0.2	0	100	TRUE	Dog
FALSE	0.1	0.7	27	TRUE	Dog
FALSE	0.25	0.6	42	TRUE	Dog
FALSE	0.4	0.5	25	TRUE	Dog
FALSE	0.2	0.3	32	TRUE	Dog
FALSE	0.3	0.2	10	TRUE	Dog
FALSE	0.6	0.5	25	TRUE	Cat
FALSE	0.6	0.4	22	TRUE	Cat
FALSE	0.2	0.6	15	TRUE	Cat
FALSE	0.2	0.2	10	TRUE	Cat
FALSE	0.55	0.1	8	TRUE	Cat
FALSE	0.8	0	11	TRUE	Cat
FALSE	0.15	0.25	12	TRUE	Cat
FALSE	0.7	0.3	9	TRUE	Cat
FALSE	0.4	0	15	FALSE	Cat
FALSE	0.3	0	13	TRUE	Cat

Decision Trees



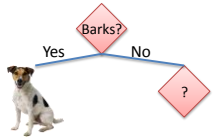
- Cats and Dogs
  - Step 1: Information Requirement

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = I\left(\frac{10}{20}, \frac{10}{20}\right) = 1 \text{ bits}$$

  - Information gain by attributes

Attribute	P(Dog|A)	P(Cat|A)	P(Dog|~A)	P(Cat|~A)	Remainder	Gain
Barks	1	0	.333	.667	.689	.311
Chases Mice	.286	.714	.615	.384	.927	.073
Chases Ball	.833	.167	.357	.642	.853	.147
Weight > 30	1	0	.333	.667	.689	.311
Eye Color Matches	.526	.473	0	1	.948	.052

Decision Trees



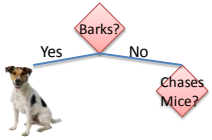
- Cats and Dogs
  - Step 2: Information Requirement

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = I\left(\frac{5}{15}, \frac{10}{15}\right) = .918 \text{ bits}$$

  - Information gain by attributes

Attribute	P(Dog|A)	P(Cat|A)	P(Dog|~A)	P(Cat|~A)	Remainder	Gain
Chases Mice	0	1	.5	.5	.667	.252
Chases Ball	.667	.333	.25	.75	.832	.086
Weight > 30	1	0	.231	.769	.675	.242
Eye Color Matches	.357	.642	.357	.643	.877	.041

Decision Trees



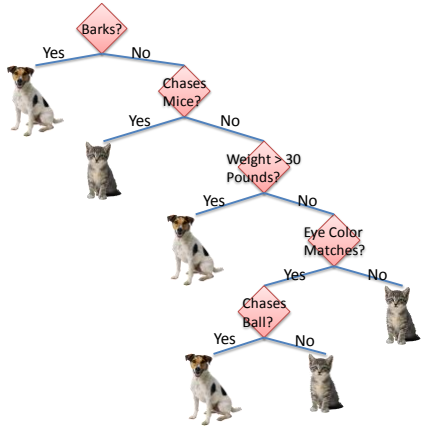
- Cats and Dogs
  - Step 3: Information Requirement

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = I\left(\frac{5}{10}, \frac{5}{10}\right) = 1 \text{ bit}$$

  - Information gain by attributes

Attribute	P(Dog|A)	P(Cat|A)	P(Dog|~A)	P(Cat|~A)	Remainder	Gain
Chases Ball	.667	.333	.429	.571	.965	.035
Weight > 30	1	0	.375	.625	.764	.236
Eye Color Matches	.556	.444	0	1	.892	.108

Final Decision Tree



Other Popular Classifiers

- Support Vector Machines (SVM)
- Maximum Entropy
- Neural Networks
- Perceptron

Machine Learning for NLP (courtesy of Michael Collins)

- The General Approach:
  - Annotate examples of the mapping you're interested in
  - Apply some machinery to learn (and generalize) from these examples
- The difference from classification
  - Need to induce a mapping from one complex set to another (e.g. strings to trees in parsing, strings in machine translation, strings to database entries in information extraction)
- Motivation for learning approaches (as opposed to "hand-built" systems
  - Often, a very large number of rules is required.
  - Rules interact in complex and subtle ways.
  - Constraints are often not "categorical", but instead are "soft" or violable.
  - A classic example: Speech Recognition

Unsupervised Learning

- Given a set of instances, each with a set of features, but WITHOUT any labels, find how the data are organized:

Given:

$$\begin{pmatrix} x_{11}, x_{12}, x_{13} \dots x_{1m} \\ x_{21}, x_{22}, x_{23} \dots x_{2m} \\ \dots \\ x_{n1}, x_{n2}, x_{n3} \dots x_{nm} \end{pmatrix} \quad \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix}$$

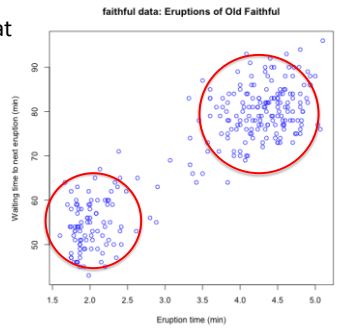
Find:

$f(\mathbf{x}) = \hat{y}$

$f(\mathbf{x})$ is called a classifier.
The way and/or parameters of $f(\mathbf{x})$ is chosen is called a classification model.

Clustering

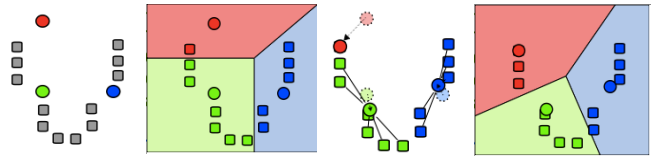
- Splitting a set of observations into a subsets (clusters), so that observations are grouped together in similar sets
- Related to problem of density estimation
- Example: Old Faithful Dataset
  - 272 Observations
  - Two Features
    - Eruption Time
    - Time to Next Eruption



K-Mean Clustering

- Aims to partition *n* observations into *k* clusters. Wherein each observation is in the cluster with the nearest mean.
- Iterative 2-stage process
  - Assignment Step
  - Update Step

K-Mean Clustering*



1) k initial "means" (in this case k=3) are randomly selected from the data set (shown in color).

2) k clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

3) The centroid of each of the k clusters becomes the new means.

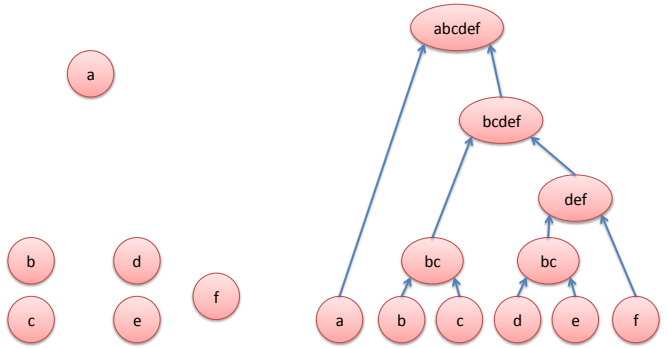
4) Steps 2 and 3 are repeated until convergence has been reached.

*Example taken from http://en.wikipedia.org/wiki/K-means_clustering

Hierarchical Clustering

- Build a hierarchy of clusters
- Find successive clusters using previously established clusters
- Paradigms
  - Agglomerative: Bottom-up
  - Divisive: Top-down

Agglomerative Hierarchical Clustering*

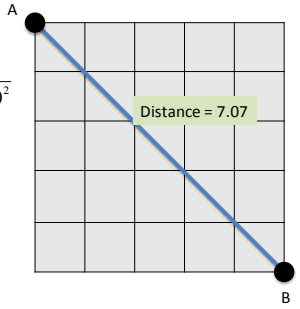


*Example courtesy of http://en.wikipedia.org/wiki/Data_clustering#Hierarchical_clustering

Distance Measures

- Euclidean Distance

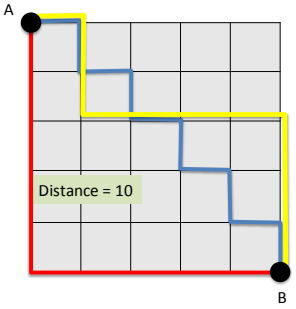
$$d(A,B) = \sqrt{(A_1 - B_1)^2 (A_2 - B_2)^2 + \ldots (A_n - B_n)^2}$$



Distance Measures

- Manhattan (aka Taxicab) distance

$$d(A,B) = \sum_{i=1}^{n} |A_i - B_i|$$



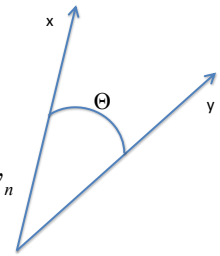
Distance Measures

- Cosine Distance

$$\theta = \arccos\left(\frac{x \cdot y}{|x||y|}\right)$$

where $x \cdot y = x_1 y_1 + x_2 y_2 + \ldots + x_n y_n$

and $|x| = \sqrt{x \cdot x}$



Cluster Evaluation

- Purity
  - Percentage of cluster members that are in the cluster's majority class

$$purity(\Omega, C) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|$$

where the set of clusters $\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$
and the set of classes $C = \{c_1, c_2, \ldots, c_j\}$

  - Drawbacks
    - Requires members to have labels
    - Easy to get perfect purity with lots of clusters

0.80 0.50 0.67 Avg = 0.66

Cluster Evaluation

- Normalized Mutual Information

$$NMI(\Omega, C) = \frac{I(\Omega, C)}{[H(\Omega) + H(C)]/2}$$

where the set of clusters $\Omega = \{\omega_1, \omega_2, ..., \omega_k\}$

and the set of classes $C = \{c_1, c_2, ..., c_j\}$

  - Drawbacks
    - Requires members to have labels

Application: Automatic Verb Class Identification

- Goal: Given significant amounts of sentences, discover verb classes
  - Example:
    - Steal-10.5: Abduct, Annex, Capture, Confiscate, …
    - Butter-9.9: Asphalt, Butter, Brick, Paper, …
    - Roll-51.3.1: Bounce, Coil, Drift, Drop….
- Approach:
  - Determine meaningful feature representation for each verb
  - Extract set of observations from a corpus
  - Apply clustering algorithm
  - Evaluate

Application: Automatic Verb Class Identification

- Feature Representation:
  - Want features that provide clues to the sense used
    - Word co-occurrence
      - *The ball rolled down the hill.*
      - *The wheel rolled away.*
      - *The ball bounced.*
    - Selectional Preferences
      - Part of Speech
      - Semantic Roles
    - Construction
      - Passive
      - Active
    - Other
      - Is the verb also a noun?

Application: Automatic Verb Class Identification

		P(*ball*|verb)	P(*hill*|verb)	P(*bread*|verb)	P(subj=agent)	P(subj=theme)	P(POS w-1=Adj)	P(POS w+1=Adj)	Is verb also a Noun?	
Roll	…	.050	.040	.005	.18	.27	.003	.000	1	…
Bounce	…	.06	.038	.002	.15	.22	.004	.000	1	…
Butter	…	.001	.000	.200	.09	.23	.000	.001	1	…
Disturb	…	.004	.003	.000	.33	.21	.000	.005	0	…

Application: Automatic Verb Class Identification