Machine learning is a subfield of computer science (CS) and artificial intelligence (AI) that deals with the construction and study of systems that can learn from data, rather than follow only explicitly programmed instructions. (http://en.wikipedia.org/wiki/Machine_learning, retrieved 8/11/14)

Our focus will be on supervised learning, where the ML system learns from labeled examples

Ideas incorporating unsupervised learning are of course welcome, too!

Motivation for ML in CL (2)

In contrast:

- NLP systems and language resources for NLP based on machine learning techniques
  - require less human effort
  - are data-driven and require large-scale data sources
  - achieve coverage directly proportional to the richness of the data source
  - are more adaptive to noisy data

One priority for us, then, is to obtain appropriate data (more on that in the coming weeks)

Motivation for ML in CL

- Manually developed NLP systems and language resources for NLP
  - require considerable human effort
  - are often based on limited inspection of the data with an emphasis on prototypical examples
    - (... though, precision can be quite high)
  - often fail to reach sufficient domain coverage
  - often lack sufficient robustness when input data is noisy

These issues are exacerbated with a task such as author profiling across different domains

Classification

- Classification:
  - Supervised learning = classification
  - Classification = assigning a label from a limited set of labels to an instance
    - Instance = pre-defined list of feature-value pairs
  - Unsupervised approaches: more general results (hierarchies, models)

We may also want to employ ranking amongst choices

- could be as simple as comparing classifier confidence for multiple guesses
- makes the most sense for gradient properties

Idea: computers are better at finding regularities than humans

- do not give the computer explicit rules
- let it extract knowledge from data

Assumptions of learning:

- from labeled data → supervised learning
- from unlabeled data → unsupervised learning

Ways to approach the problem:

- abstract over data → eager learning
- do not abstract over data → lazy learning
Example: Part-of-Speech Tagging

- Task: find the appropriate POS tag for a word in context
  - e.g., They man\text{VB} the boat. versus The man\text{NN} in the boat.
    - for English, accuracy > 96%
    - for morphologically rich languages: many POS tags

<table>
<thead>
<tr>
<th>feature</th>
<th>word\text{.2}</th>
<th>tag\text{.2}</th>
<th>word\text{.1}</th>
<th>tag\text{.1}</th>
<th>word\text{.0}</th>
<th>POS tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>NULL</td>
<td>NULL</td>
<td>They</td>
<td>PRP</td>
<td>man</td>
<td>VB</td>
</tr>
</tbody>
</table>

Text Classification

Our task this semester falls under the general field of document classification, or text classification

- text classification ⊆ classification ⊆ machine learning
- Classification might be in terms of topics/subjects, document type, etc.
  - Generally different from our task, documents may contain multiple topics (any-of problem)
  - Topics may also be hierarchical, e.g., poultry and coffee as subclasses of industries
- Some useful links:

The Learning Problem

- instance: a vector of feature values \( < f_1, f_2, \ldots, f_n > \)
  where the values are taken from the discrete or real-valued domain of the \( i \)th feature

- let \( X \) be the space of possible instances

- let \( Y \) be the set of classes

- the goal of the ML system is to learn a target function \( c : X \rightarrow Y \)

The Learning Problem (2)

Restrictions

Some learners require real-valued features, while some (e.g., timbl) allow for things like text features

Text features:

<table>
<thead>
<tr>
<th>Per</th>
<th>Num</th>
<th>Gen</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>pl</td>
<td>neut</td>
<td>1</td>
</tr>
</tbody>
</table>

Real-valued features:

<table>
<thead>
<tr>
<th>Per</th>
<th>Num</th>
<th>Gen</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>Sg</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>M</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>N</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Likewise, some learners require binary decisions

- though, they usually have techniques to convert \( n \)-ary decisions to binary internally

The Learning Problem (3)

- training example: instance \( x \in X \) labeled with the correct class \( c(x) \)
  - let \( D \) be the set of all training examples

- hypothesis space, \( H \): set of functions \( h : X \rightarrow Y \) of possible definitions
  - the goal is to find an \( h \in H \) such that for all \( x, c(x) \in D, h(x) = c(x) \)

Some Important Concepts

- generalization: generalize from experience
- abstraction
  - vs. lazy learning: generalize when needed
  - non-abstraction can be useful when there are many exceptions & sub-regularities
- online learning: learn one instance at a time & thus continually refine model
- offline learning: learn as a batch
Some Machine Learning Algorithms

- **supervised:**
  - decision tree learning
  - memory-based learning (k-nearest neighbors)
  - support vector machines (SVM)
  - maximum entropy learning
  - neural networks
  - genetic programming
  - naive Bayes

- **unsupervised:**
  - clustering
  - minimum description length

Evaluation Metrics

Common metrics include:

- **accuracy:** percentage of correctly classified instances from test set
- **recall:** percentage of the items in the gold standard that were found by the ML program
- **precision:** percentage of the items selected by the ML program that are correct

Sometimes also: sensitivity, specificity, area under the ROC curve (http://gim.unmc.edu/dxtests/roc3.htm), ...

Problems with ML

- Difficult to distinguish between noise and subregularities, irregularities
- Feature selection mainly by intuition
  - though, see, e.g., deep learning (http://nlp.stanford.edu/courses/NAACL2013/)
  - irrelevant information can deteriorate performance
- Skewed class distribution deteriorates performance
- Defining tasks as classification which may not naturally be classification

Evaluation

- **training set:** data on which the ML program is trained
- **test set:** data on which the performance of the ML program is measured
  - **gold standard:** data against which the ML program is evaluated
  - **tenfold cross validation:** split data into 10 parts:
    - 10 rounds: use 1 part as test set and remaining parts as training set

Useful comparisons by which to gauge results:

- **baseline:** simple method, often heuristic; gives the lower estimate of the difficulty of the problem
  - Note in the PAN data sets how baseline systems are included
- **upper bound:** what can be reached in the optimal case, often human performance

Do's and Don't's in ML

**Don't:**
- report without evaluation
- test on (any part of) your training set

**Do:**
- make your features independent (required for most algorithms)
- optimize your parameters
- optimize features & parameters at the same time
  - comparisons of learning algorithms are only meaningful if both are optimized
- beware of overfitting
- do what you can to get more data
Some Available Packages

Packages of multiple algorithms:
- weka (http://www.cs.waikato.ac.nz/ml/weka/)
- mallet (http://mallet.cs.umass.edu): MAchine Learning for Language Toolkit
- MLlib (https://spark.apache.org/mllib/): good for very large-scale learning
- MLPack (http://www.mlfpack.org): “emphasis on scalability, speed, and ease-of-use”
- sofia-ml (https://code.google.com/p/sofia-ml/)

Some specific algorithms
- timbl (http://ilk.uvt.nl/timbl): memory-based learning
- svmlight (http://svmlight.joachims.org)

Choosing a Classifier

Accounting for bias & variance

What to do? (originally from Andrew Ng)
- High variance?
  - get more training examples
  - try smaller sets of features
  - High bias?
  - try new (more/different) features
To find whether it’s high variance or bias, compare training & testing (i.e., development) learning curves:
- cf. slides 7 & 8 of:
  (http://followthedata.wordpress.com/2012/06/02/
  practical-advice-for-machine-learning-bias-variance/)

Other Ways to Boost Performance

- Combine multiple classifiers
- Engage in feature engineering
  - Group together similar features (i.e., with similar votes)
  - Create new features for potentially informative concepts (e.g., subwords)
  - Account for document structure (e.g., upweight items in a title, use separate feature spaces for different document zones)

Next time (Sep. 3)

We’ll spend some time playing with off-the-shelf ML systems
- Meet in Memorial Hall (MM) 401

Choosing a classifier

Some advice from Manning, Raghavan, & Schütze (2008):
- Can use rules, if the task is relatively simple
  - Rules can also work well for post-processing ML output
- If you have little data, use a classifier with high bias (e.g., Naive Bayes)
  - Also can try semi-supervised training methods (e.g., bootstrapping, EM)
- If you have a huge amount of data, the classifier may have less of an impact on accuracy
  - Speed & ease of use become bigger questions

choosing-what-kind-of-classifier-to-use-1.html)