Machine Learning

A Brief & Spotty Survey

L715/B659
(with many thanks to Sandra Kübler!)

Dept. of Linguistics, Indiana University

Fall 2016

Machine Learning for Author Profiling

Machine learning explores the study and construction of algorithms that can learn from and make predictions on data. (https:// en.wikipedia.org/ wiki/ Machine_learning, retrieved 8/22/16)

- Our focus will be on supervised learning, where the ML system learns from labeled examples
  - There will be some space for unsupervised learning, too

Motivation for ML in CL

Motivation for ML in computational linguistics:

- 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

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 & 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)

- Adapting to new domains is also a challenge

Machine Learning

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

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

- May be as simple as comparing classifier confidence for multiple guesses
- Makes the most sense for gradient properties

Example: Part-of-Speech Tagging

- Task: find the appropriate POS tag for a word in context
- e.g., `They` **`man`**$_{VB}$ `the boat.` vs. `They` **`man`**$_{NN}$ `in the boat.`
  - for English, accuracy > 96 %
  - for morphologically rich languages: many POS tags

Sample instance:

feature	word_{-2}	tag_{-2}	word_{-1}	tag_{-2}	word	POS tag
value	NULL	NULL	They	PRP	man	VB

Text Classification

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

- text classification $\subset$ classification $\subset$ 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:
  - http://en.wikipedia.org/wiki/Document_classification
  - http://nlp.stanford.edu/IR-book/html/htmledition/text-classification-and-naive-bayes-1.html

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:

Per	Num	Gen	Class
2	pl	neut	1

Real-valued features:

Per			Num		Gen			
1	2	3	Sg	Pl	M	F	N	Class
0	1	0	0	1	0	0	1	1

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 $h : X \to Y$ 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

- **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

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), ...

Evaluation

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
  - For some tasks (e.g., detecting deception), human performance may be below automatic systems

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
- Some tasks defined as classification may not naturally be classification

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.mlpack.org): “emphasis on scalability, speed, and ease-of-use”
- sofia-ml (https://code.google.com/p/sofia-ml/)
- Various R packages (https://www.quora.com/What-are-the-best-machine-learning-packages-in-R)

More packages

- Python Machine Learning kits:
  - Orange: http://orange.biolab.si/
  - mlpy: https://mlpy.fbk.eu/
  - PyML: http://pyml.sourceforge.net/
  - scikit-learn: http://scikit-learn.org/stable/
  - Anaconda Python has much pre-installed: https://store.continuum.io/cshop/anaconda/
- Various packages in R (http://cran.r-project.org/web/views/MachineLearning.html)
- Maximum Entropy-related software: http://homepages.inf.ed.ac.uk/s0450736/maxent.html
- ...

Next time, we'll work with the Python-based `scikit-learn`

Choosing a Classifier

Bias & Variance

- Bias: (non-)ability to approximate the data, degree to which model makes assumptions about data distribution
  - "High bias is related to under-fitting" (e.g., linear regression for a quadratic relationship)
- Variance: (non-)stability in the face of new training examples
  - "High variance is related to over-fitting" (e.g., k-NN changes a lot depending on training set)
- Regularization parameters help control bias-variance tradeoff
  - e.g., penalize complex models, set k higher

(http://followthedata.wordpress.com/2012/06/02/practical-advice-for-machine-learning-bias-variance/)

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://cs229.stanford.edu/materials/ML-advice.pdf

(http://followthedata.wordpress.com/2012/06/02/practical-advice-for-machine-learning-bias-variance/)

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

(http://nlp.stanford.edu/IR-book/html/htmledition/choosing-what-kind-of-classifier-to-use-1.html)

Other Ways to Boost Performance

- Combine mutliple 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)

(http://nlp.stanford.edu/IR-book/html/htmledition/improving-classifier-performance-1.html)

Next time

We'll spend some time getting into practical matters

- See your assignment for installations