Machine Learning Coms-4771

Alina Beygelzimer

Tony Jebara, John Langford, Cynthia Rudin

February 3, 2008

(partially based on Yann LeCun's and Sam Roweis's slides; see links at the web page)

Logistics

- The course web page is
  `http://hunch.net/~coms-4771`
- If you have a question, email
  `beygel@us.ibm.com`
  or post it at
  `http://coms-4771.blogspot.com/`
- Do interrupt and ask questions during the class.
- The web page has notes on probability theory and statistics (if you need to refresh your memory)

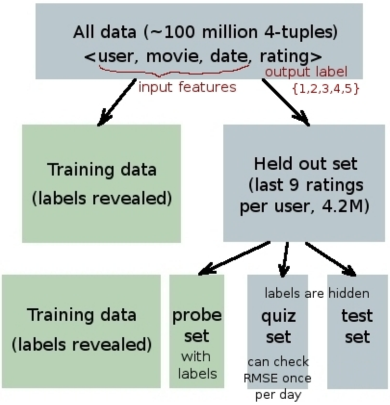
What is Machine Learning?

www.netflixprize.com

- In October 2006, Netflix announced a \$1M problem:

  Predict the rating a given user would assign to a given movie (based on 100 million past user-movie ratings).
- 10% improvement = \$1M
- 2500 teams; annual progress prize of \$50K went to KorBell from AT&T Labs (8.43%)

Netflix Problem Setup

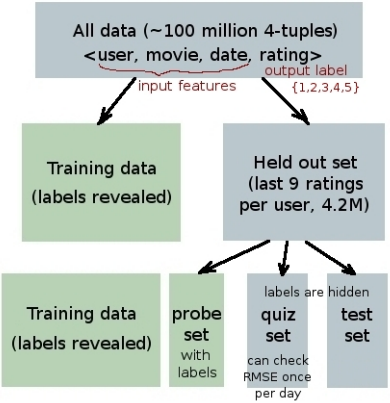


- Success is measured by the root mean squared error (RMSE) on the test set:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},$$

where y_i and $\hat{y}_i$ are the actual and predicted movie ratings.

Netflix Problem Setup

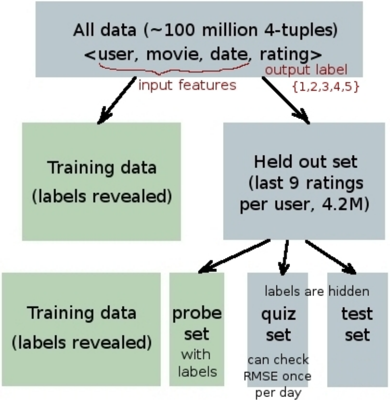


- Success is measured by the root mean squared error (RMSE) on the test set:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},$$

  where y_i and $\hat{y}_i$ are the actual and predicted movie ratings.
- Q: What's the role of the probe set?
- Q: Why are there both quiz and test sets?

Netflix Problem Setup



- Success is measured by the root mean squared error (RMSE) on the test set:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},$$

  where y_i and $\hat{y}_i$ are the actual and predicted movie ratings.
- Q: What's the role of the probe set?
- Q: Why are there both quiz and test sets?
- We have a well defined task. How would you go about solving it?

What is Machine Learning?

- We want robust, intelligent behavior. Hand-programming a solution directly is not going to work. The world is too complex.
- Learning approach = programming by example: Get the machine to program itself by showing it examples of the behavior we want. Learning is about improving performance through experience.
  - Learning is data driven. It can examine much larger amounts of data than you can.
  - Labeling examples is perhaps the easiest way to express knowledge.
  - Learning is general purpose—algorithm reuse!
- In reality, we specify a space of possible solutions, and let the machine find a good solution in this space.

Learning Problems, Structure of Learning Machines

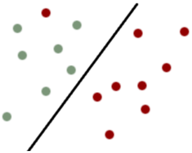
- Learning problem = (unknown) distribution of inputs and outputs D + (typically known) loss function L.
- Hypothesis space H = Space of functions mapping inputs to outputs (H is often indexed by a set of parameters the algorithm can tune to create different solutions)
- Learning algorithm searches (or prunes or tunes parameters in) H to find a hypothesis minimizing the expected L on D, based on a limited set of input-output examples.

The hardest part is deciding how to represent inputs/outputs and how to select appropriate L and H.

How do we incorporate prior information?

Supervised Learning

Given a set of labeled examples, predict outputs of future unlabeled examples.



Classification: Feature space X, discrete set of labels Y (categories). Find a decision boundary between the categories in Y.

Loss function: $\ell(y, y') = \mathbf{1}(y \neq y')$ (zero-one loss)

Distribution D over $X \times Y$. Find a classifier $h : X \to Y$ minimizing the expected loss on D given by $\mathbf{Pr}_{(x,y)\sim D}[h(x) \neq y] = \mathbf{E}_{(x,y)\sim D}\, \ell(y, h(x))$.

Regression ("curve fitting" or "function approximation"): $Y = \mathbb{R}$

Loss function: $\ell_{sq}(y, y') = (y - y')^2$ (squared loss)

Learn a continuous mapping $f : X \to \mathbb{R}$ minimizing $\mathbf{E}_{(x,y)\sim D}\, \ell_{sq}(y, f(x))$.

Training vs. Testing

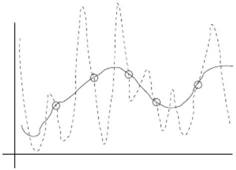
- Training (empirical) error: the average loss on the training data
- Test error: the average loss on the test data
- Ideally we want to minimize the test error, but we can't evaluate it! (Most of the time we don't even know future inputs.)
- Do we want to minimize the training error instead?

Training vs. Testing

- Training (empirical) error: the average loss on the training data
- Test error: the average loss on the test data
- Ideally we want to minimize the test error, but we can't evaluate it! (Most of the time we don't even know future inputs.)
- Do we want to minimize the training error instead?
- NO. Consider an algorithm that memorizes training examples. If a test example is in the training set, produce the memorized output. Otherwise, choose a random output.
- We are overfitting: Training error is 0. Test error is HUGE.
- Learning is not Memorization. We want to generalize from training examples to predict well on previously unseen examples.

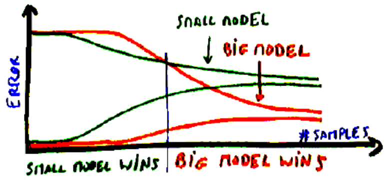
Inductive Bias

- There should be some hope that the data is predictive.
- No Free Lunch Theorems: an unbiased learner can never generalize. Inductive bias = set of assumptions that favor some predictors over others.



- Ways of incorporating bias: assumptions on the data distribution (test data is drawn from the same distribution as the training data, examples are independent), choice of the hypothesis class
- Examples: Occam's Razor (choose the simplest consistent hypothesis), maximum margin (attempt to maximize the width of the boundary), nearest neighbors (guess the label based on closest training examples).

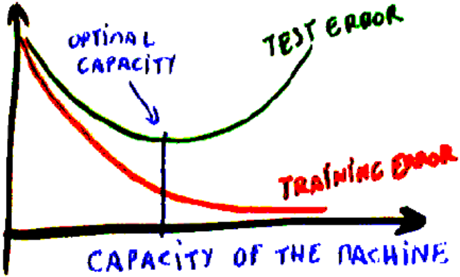
Choosing Hypothesis Space



- The number of training examples for which the training error and test error start converging = "capacity" of the learning machine.
- Can we bound the expected loss as a function of the empirical loss, the capacity of the family of functions, and the size of the training set? (Yes, sometimes.)
- Problem: if the class is too rich, there is a risk of overfitting the data. If the class if too simple, there is a risk of not being able to approximate the data well.

Q: How to choose the hypothesis space so that it is large enough to contain a solution to your problem, yet small enough to ensure generalization from the training sets you have?

Choosing Hypothesis Space



For each training set size, there is an optimal capacity for the machine.

Choosing the Loss Function

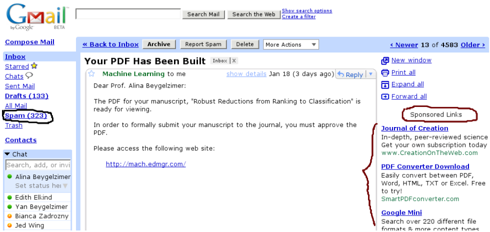
- L quantifies what it means to do well/poorly on the task
- Tradeoff: L captures what we actually want to minimize vs. L is computationally easy to optimize.
- Loss function semantics:
  - Optimizing squared loss means predicting the conditional mean $\mathbf{E}_{(x,y)\sim D}(y \mid x)$.
  - Optimizing the absolute loss $|y - y'|$ means predicting the conditional median.
- Good rule: start with what you want and try to derive a loss.

Other Types of Learning

- Unsupervised learning: given only inputs, automatically discover structure (clustering, outlier detection, embedding in low-dimensional manifold, compression). Not so well defined.
- Semi-supervised learning: labels are expensive, unlabeled data is cheap. Use the unlabeled data to help you learn.
- Active learning: ask for labels on unlabeled examples of your choice, direct the learning process.
- Reinforcement learning: learn how to act in a way that maximizes your expected reward; your actions change the distribution of future inputs.

Some Applications of Machine Learning

- Handwritten character recongition, speech recognition, speaker verification, object detection, tracking objects in videos
- Search, targeted ads, recommendation systems, spam filtering, auctions



- Credit card fraud, insurance premium prediction, product pricing, stock market analysis (Wall Street uses a lot of machine learning)
- Medical diagnosis and prognosis, fMRI analysis
- Game playing (adaptive opponents)
- Robotics, adaptive decision making under uncertainty