Practical ML Advice

Based on slides from Jude Shavlik and Tom Dietterich
Proper Experimental Methodology Can Have a Huge Impact:

A 2002 paper in *Nature* (a major journal) needed to be corrected due to “training on the testing set”

Original report: 95% accuracy (5% error rate)

Corrected report (which still is buggy):

73% accuracy (27% error rate)

Error rate increased over 400%!!!
Some Typical ML Experiments

![Graph showing test set accuracy vs. number of training examples with confidence bars and learning curve]

Test set
Accuracy

# of Training Examples
(or ‘amount of noise’ or ‘amount of missing features’)

Confidence Bars (from multiple runs)

Algorithm 1

Algorithm 2

A ‘learning curve’
## Typical Experiments

<table>
<thead>
<tr>
<th></th>
<th>Test Set Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full System</td>
<td>80%</td>
</tr>
<tr>
<td>Without Module A</td>
<td>75%</td>
</tr>
<tr>
<td>Without Module B</td>
<td>62%</td>
</tr>
</tbody>
</table>
1) Start with a dataset of labeled examples
2) Randomly partition into $N$ groups
3a) $N$ times, combine $N-1$ groups into a train set
3b) Provide training set to learning system
3c) Measure accuracy on “left out” group (the test set)

Called $N$-fold cross validation
Validation Sets

- Often, an ML system has to choose when to stop learning, select among alternative answers, etc.

- One wants the model that produces the highest accuracy on future examples ("overfitting avoidance")

- It is a "cheat" to look at the test set while still learning

- Better method
  - Set aside part of the training set
  - Measure performance on this validation data to estimate future performance for a given set of hyperparameters
  - Use best hyperparameter settings, train with all training data (except test set) to estimate future performance on new examples
A typical Learning system

Statistical techniques such as 10-fold cross validation and t-tests are used to get meaningful results.

A collection of classified examples

- **Training examples**
  - Train' set
  - Generate solutions
  - Learner
  - Select best

- **Testing examples**
  - Tune set
  - Classifier

Expected accuracy on future examples
Multiple Tuning sets

• Using a **single** tuning set can be unreliable predictor, plus some data “wasted”

1) For each possible set of hyperparameters
   a) Divide training data into **train** and **valid.** sets, using **N-fold cross validation**
   b) Score this set of hyperparameter values: average **valid.** set accuracy over the \( N \) folds

2) Use **best** set of hyperparameter settings and **all** (train + valid.) examples

3) Apply resulting model to **test** set
EVALUATING ML MODELS
Contingency Tables

(special case of ‘confusion matrices’)

<table>
<thead>
<tr>
<th></th>
<th>True Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>n(1,1)</td>
</tr>
<tr>
<td></td>
<td>[true pos]</td>
</tr>
<tr>
<td>+</td>
<td>n(1,0)</td>
</tr>
<tr>
<td></td>
<td>[false pos]</td>
</tr>
<tr>
<td>-</td>
<td>n(0,1)</td>
</tr>
<tr>
<td></td>
<td>[false neg]</td>
</tr>
<tr>
<td>-</td>
<td>n(0,0)</td>
</tr>
<tr>
<td></td>
<td>[true neg]</td>
</tr>
</tbody>
</table>

Counts of occurrences
**TPR and FPR**

**True Positive Rate (TPR)**

\[ \text{TPR} = \frac{n(1,1)}{n(1,1) + n(0,1)} \]

- Correctly categorized +’s / total positives
- \( \sim \) \( P(\text{algo outputs } + | + \text{ is correct}) \)

**False Positive Rate (FPR)**

\[ \text{FPR} = \frac{n(1,0)}{n(1,0) + n(0,0)} \]

- Incorrectly categorized –’s / total neg’s
- \( \sim \) \( P(\text{algo outputs } + | - \text{ is correct}) \)

Can similarly define False Negative Rate and True Negative Rate
• **ROC: Receiver Operating Characteristics**

• Started for radar research during WWII

• Judging algorithms on accuracy alone may not be good enough when *getting a positive wrong* costs more than *getting a negative wrong* (or vice versa)
  • e.g., medical tests for serious diseases
  • e.g., a movie-recommender system
Different algorithms can work better in different parts of ROC space. This depends on cost of false + vs false -
Creating an ROC Curve

The Standard Approach:

- You need an ML algorithm that outputs **NUMERIC** results such as `prob(example is +)`

- You can use ensemble methods to get this from a model that only provides Boolean outputs
  - e.g., have 100 models vote & count votes
Alg. for Creating ROC Curves

**Step 1**: Sort predictions on test set

**Step 2**: Locate a *threshold* between examples with opposite categories

**Step 3**: Compute TPR & FPR for each threshold of Step 2

**Step 4**: Connect the dots
### Plotting ROC Curves - Example

**ML Algo Output (Sorted)** | **Correct Category**
--- | ---
Ex 9 | .99 | +
Ex 7 | .98 | TPR=(2/5), FPR=(0/5) +
Ex 1 | .72 | TPR=(2/5), FPR=(1/5) -
Ex 2 | .70 | +
Ex 6 | .65 | TPR=(4/5), FPR=(1/5) +
Ex 10 | .51 | -
Ex 3 | .39 | TPR=(4/5), FPR=(3/5) -
Ex 5 | .24 | TPR=(5/5), FPR=(3/5) +
Ex 4 | .11 | -
Ex 8 | .01 | TPR=(5/5), FPR=(5/5) -

Algorithm predicts + if its output is $\geq 0$.
Area Under ROC Curve

• A common metric for experiments is to numerically integrate the ROC Curve
  • Usually called AUC
  • Probability that ML alg. will “rank” a randomly chosen positive instance higher than a randomly chosen negative one
    • Given a randomly selected positive example and a randomly selected negative example, AUC is the probability that the classifier will be able to distinguish them
  • Can summarize the curve too much in practice
ROC’s & Skewed Data

• One strength of ROC curves is that they are a good way to deal with skewed data ($|+| >> |-|$) since the axes are fractions (rates) independent of the # of examples.

• You must be careful though!
  • Low FPR * (many negative ex) = sizable number of FP
  • Possibly more than # of TP
Precision vs. Recall

• Think about search engines...

• **Precision** = (# of relevant items retrieved) / (total # of items retrieved)
  = \( \frac{n(1,1)}{n(1,1) + n(1,0)} \)

• **Recall** = (# of relevant items retrieved) / (# of relevant items that exist)
  = \( \frac{n(1,1)}{n(1,1) + n(0,1)} \)
  = TPR

• Notice that \( n(0,0) \) is not used in either formula
  Therefore you get no credit for filtering out irrelevant items
ROC vs. Precision-Recall

You can get very different visual results on the same data!

Produced by varying threshold for positive identification, e.g., say 1 if $p(1|x) > .5$ in logistic regression.
You can get very different visual results on the same data!

The reason for this is that there may be lots of – ex’s (e.g., might need to include 100 neg’s to get 1 more pos)
Rejection Curves

• In most learning algorithms, we can specify a threshold for making a rejection decision

  • Probabilistic classifiers: adjust cost of rejecting versus cost of FP and FN

  • Decision-boundary method: if a test point $\mathbf{x}$ is within $\theta$ of the decision boundary, then reject

    • Equivalent to requiring that the “activation” of the best class is larger than the second-best class by at least $\theta$
Rejection Curves

- Vary $\theta$ and plot fraction correct versus fraction rejected
The F1 Measure

• Figure of merit that combines precision and recall

\[ F_1 = 2 \cdot \frac{P \cdot R}{P + R} \]

where \( P = \) precision; \( R = \) recall. This is twice the harmonic mean of \( P \) and \( R \).

• We can plot \( F1 \) as a function of the classification threshold \( \theta \)