Slides adapted from Prof Carpuat and Duraiswami

CMSC 422 Introduction to Machine Learning

Lecture 10 Practical Issues: Features, Evaluation, Debugging

Furong Huang / furongh@umd.edu

UNIVERSITY OF MARYLAND

Practical Issues

Classifier	Accuracy on test set
Team A	80.00
Team B	79.90
Team C	79.00
Team D	78.00

Which classifier is the best?

- This result table alone cannot give us the answer
- Solution: statistical hypothesis testing

Practical Issues

Classifier	Accuracy on test set
Team A	80.00
Team B	79.90
Team C	79.00
Team D	78.00

Is the difference in accuracy between A and B statistically significant?

What is the probability that the observed difference in performance was due to chance?

A confidence of 95%

- It means

"If I run this experiment 100 times, I expect A to perform better than B 95 times."

Practical Issues: Debugging!

- You've implemented a learning algorithm
- You try it on some train/dev/test data
- But it doesn't seem to learn

- What's going on?
  - Is the data too noisy?
  - Is the learning problem too hard?
  - Is the implementation of the learning algorithm buggy?

Strategies for Isolating Causes of Errors

- Is the problem with **generalization** to test data?
  - Can learner fit the training data?
  - Yes: problem is in generalization to test data
  - No: problem is in representation (need better features or better data)

- **Train/test mismatch**?
  - Try reselecting train/test by shuffling training data and test together

Strategies for Isolating Causes of Errors

- Is algorithm **implementation correct**?
  - Measure loss rather than accuracy
  - Hand-craft a toy dataset
- **Is representation adequate?**
  - Can you learn if you add a cheating feature that perfectly correlates with correct class?
- Do you have **enough data**?
  - Try training on 80% of the training set, how much does it hurt performance?

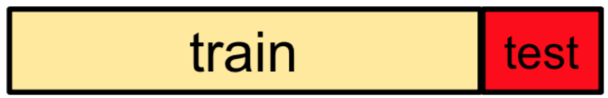
Practical Issues: hyperparameter tuning with dev set vs. cross-validation

Algorithm 8 CrossValidate(*LearningAlgorithm*, *Data*, K)

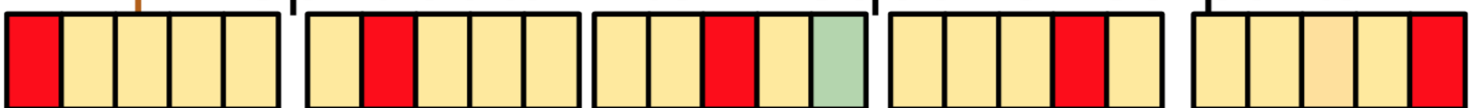
$\hat{\epsilon} \leftarrow \infty$ // store lowest error encountered so far
$\hat{\alpha} \leftarrow$ unknown // store the hyperparameter setting that yielded it
for all hyperparameter settings α **do**
    $err \leftarrow [\,]$ // keep track of the K-many error estimates
    for $k = 1$ **to** K **do**
        $train \leftarrow \{(\boldsymbol{x}_n, y_n) \in Data : n \mod K \neq k-1\}$
        $test \leftarrow \{(\boldsymbol{x}_n, y_n) \in Data : n \mod K = k-1\}$ // test every Kth example
        $model \leftarrow$ Run *LearningAlgorithm* on *train*
        $err \leftarrow err \oplus$ error of *model* on *test* // add current error to list of errors
    end for
    $avgErr \leftarrow$ mean of set *err*
    if $avgErr < \hat{\epsilon}$ **then**
        $\hat{\epsilon} \leftarrow avgErr$ // remember these settings
        $\hat{\alpha} \leftarrow \alpha$ // because they're the best so far
    end if
end for

N-fold cross validation

- Instead of a single test-training split:



- Split data into N equal-sized parts



- Train and test N different classifiers
- Report average accuracy and standard deviation of the accuracy

Practical Issues: Evaluation, beyond accuracy

- So far we've measured classification performance using **accuracy**
- But this is not a good metric when some errors matter mode than others
  - Given medical record, predict whether patient has cancer or not
  - Given a document collection and a query, find documents in collection that are relevant to query

The 2-by-2 contingency table

Imagine we are addressing a document retrieval task for a given query, where +1 means that the document is relevant -1 means that the document is not relevant

We can categorize predictions as:

- true/false positives
- true/false negatives

	Gold label = +1	Gold label = -1
Prediction = +1	tp	fp
Prediction = -1	fn	tn

Precision and recall

- **Precision**: % of positive predictions that are correct

$$Precision = \frac{tp}{tp + fp}$$

Denominator: # of positive predictions

- **Recall**: % of positive gold labels that are found

$$Recall = \frac{tp}{tp + fn}$$

Denominator: # of positive gold labels

	Gold label = +1	Gold label = -1
Prediction = +1	tp	fp
Prediction = -1	fn	tn

A combined measure: F

$$P = \frac{tp}{tp + fp}$$

$$R = \frac{tp}{tp + fn}$$

- A combined measure that assesses the P/R tradeoff is F measure

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

- People usually use balanced F-1 measure
  - i.e., with $\beta = 1$ (that is, $\alpha = \frac{1}{2}$)
  - Harmonic mean $F = \frac{2PR}{P+R}$

Formalizing Errors

The learned classifier

$\mathcal{F}$ set of all possible classifiers using a fixed representation

$$\text{error}(f) = \underbrace{\left[\text{error}(f) - \min_{f^* \in \mathcal{F}} \text{error}(f^*)\right]}_{\text{estimation error}} + \underbrace{\left[\min_{f^* \in \mathcal{F}} \text{error}(f^*)\right]}_{\text{approximation error}}$$

How far is the learned classifier f from the optimal classifier f*?

Quality of the model family
aka hypothesis class

The bias/variance trade-off

- Trade-off between
  - approximation error (bias)
  - estimation error (variance)

- Example:
  - Consider the always positive classifier
    - Low variance as a function of a random draw of the training set
    - Strongly biased toward predicting +1 no matter what the input

Recap: practical issues

- Learning algorithm is only one of many steps in designing a ML application
- Many things can go wrong, but there are practical strategies for
  - Improving inputs
  - Evaluating
  - Tuning
  - Debugging
- Fundamental ML concepts: estimation vs. approximation error

Imbalanced data distributions

- Sometimes training examples are drawn from an imbalanced distribution
- This results in an imbalanced training set
  - “needle in a haystack” problems
  - E.g., find fraudulent transactions in credit card histories
- Why is this a big problem for the ML algorithms we know?

Task: α-Weighted Binary Classification

Given:

1. An input space $\mathcal{X}$
2. An unknown distribution $\mathcal{D}$ over $\mathcal{X}\times\{-1,+1\}$

Compute: A function f minimizing: $\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\alpha^{y=1}\left[f(\boldsymbol{x}) \neq y\right]\right]$

Given a good algorithm for solving the binary classification problem, how can I solve the α-weighted binary classification problem?

We define cost of misprediction as:
α > 1 for y = +1
1 for y = -1

Solution: Train a binary classifier on an induced distribution

Algorithm 11 SubsampleMap($\mathcal{D}^{weighted}$, α)

```
while true do
    (x, y) ~ D^weighted                    // draw an example from the weighted distribution
    u ~ uniform random variable in [0, 1]
    if y = +1 or u < 1/α then
        return (x, y)
    end if
end while
```

Turn a weighted data distribution, into a (different) distribution over un-weighted binary examples

Subsampling optimality

Theorem: If the binary classifier achieves a binary error rate of ε, then the error rate of the α-weighted classifier is α ε

Proof omitted in class.

(see also CIML 6.1)

Proof of Theorem 3. Let $\mathcal{D}^w$ be the original distribution and let $\mathcal{D}^b$ be the induced distribution. Let f be the binary classifier trained on data from $\mathcal{D}^b$ that achieves a binary error rate of ϵ^b on that distribution. We will compute the expected error ϵ^w of f on the weighted problem:

$$
\begin{aligned}
\epsilon^w &= \mathbb{E}_{(x,y)\sim\mathcal{D}^w}\left[\alpha^{y=1}[f(x) \neq y]\right] && (6.1)\\
&= \sum_{x\in\mathcal{X}}\sum_{y\in\pm1} \mathcal{D}^w(x,y)\alpha^{y=1}[f(x)\neq y] && (6.2)\\
&= \alpha \sum_{x\in\mathcal{X}} \left(\mathcal{D}^w(x,+1)[f(x)\neq +1] + \mathcal{D}^w(x,-1)\frac{1}{\alpha}[f(x)\neq -1]\right) && (6.3)\\
&= \alpha \sum_{x\in\mathcal{X}} \left(\mathcal{D}^b(x,+1)[f(x)\neq +1] + \mathcal{D}^b(x,-1)[f(x)\neq -1]\right) && (6.4)\\
&= \alpha\mathbb{E}_{(x,y)\sim\mathcal{D}^b}[f(x)\neq y] && (6.5)\\
&= \alpha\epsilon^b && (6.6)
\end{aligned}
$$

And we're done! (We implicitly assumed $\mathcal{X}$ is discrete. In the case of continuous data, you need to replace all the sums over x with integrals over x, but the result still holds.) □

Strategies for inducing a new binary distribution

- Undersample the negative class (dominant class)
- Oversample the positive class (subordinate class)

Strategies for inducing a new binary distribution

- Undersample the negative class
  - More computationally efficient
- Oversample the positive class
  - Base binary classifier might do better with more training examples
  - Efficient implementations incorporate weight in algorithm, instead of explicitly duplicating data!

Reductions

Idea is to re-use simple and efficient algorithms for binary classification to perform more complex tasks

Learning with Imbalanced Data is an Example of Reduction

TASK: α-WEIGHTED BINARY CLASSIFICATION

Given:

1. An input space $\mathcal{X}$
2. An unknown distribution $\mathcal{D}$ over $\mathcal{X}\times\{-1,+1\}$

Compute: A function f minimizing: $\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\alpha^{y=1}[f(\boldsymbol{x}) \neq y]\right]$

Subsampling Optimality Theorem:
If the binary classifier achieves a binary error rate of ε, then the error rate of the α-weighted classifier is α ε

Multiclass classification

- Real world problems often have multiple classes (text, speech, image, biological sequences…)
- How can we perform multiclass classification?
  - Straightforward with decision trees or KNN
  - Can we use the perceptron algorithm?

Reductions for Multiclass Classification

TASK: MULTICLASS CLASSIFICATION

Given:

1. An input space $\mathcal{X}$ and number of classes K
2. An unknown distribution $\mathcal{D}$ over $\mathcal{X} \times [K]$

Compute: A function f minimizing: $\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[f(\boldsymbol{x}) \neq y\right]$

TASK: BINARY CLASSIFICATION

Given:

1. An input space $\mathcal{X}$
2. An unknown distribution $\mathcal{D}$ over $\mathcal{X} \times \{-1, +1\}$

Compute: A function f minimizing: $\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[f(\boldsymbol{x}) \neq y\right]$

How many classes can we handle in practice?

- In most tasks, number of classes K < 100
- For much larger K
  - we need to frame the problem differently
  - e.g, machine translation or automatic speech recognition

What you should know

- How can we take the standard binary classifier and adapt it to handle problems with
  - Imbalanced data distributions
  - Multiclass classification problems
- Algorithms & guarantees on error rate
- Fundamental ML concept: reduction

Reduction 1: OVA

- “One versus all” (aka “one versus rest”)
  - Train K-many binary classifiers
  - classifier k predicts whether an example belong to class k or not

  - At test time,
    - If only one classifier predicts positive, predict that class
    - Break ties randomly

Algorithm 12 $\textsc{OneVersusAllTrain}(\mathbf{D}^{multiclass}, \textsc{BinaryTrain})$

for $i = 1$ **to** K **do**
    $\mathbf{D}^{bin} \leftarrow$ relabel $\mathbf{D}^{multiclass}$ so class i is positive and $\neg i$ is negative
    $f_i \leftarrow \textsc{BinaryTrain}(\mathbf{D}^{bin})$
end for
return $f_1, \ldots, f_K$

Algorithm 13 $\textsc{OneVersusAllTest}(f_1, \ldots, f_K, \hat{\boldsymbol{x}})$

$score \leftarrow \langle 0, 0, \ldots, 0 \rangle$ // initialize K-many scores to zero
for $i = 1$ **to** K **do**
    $y \leftarrow f_i(\hat{\boldsymbol{x}})$
    $score_i \leftarrow score_i + y$
end for
return $\operatorname{argmax}_k score_k$

Time complexity

- Suppose you have N training examples, in K classes. How long does it take to train an OVA classifier
  - if the base binary classifier takes O(N) time to learn?
  - if the base binary classifier takes O(N^2) time to learn?

Error bound

- **Theorem:** Suppose that the average error of the K binary classifiers is ε, then the error rate of the OVA multiclass classifier is at most (K-1) ε

- To prove this: how do different errors affect **the maximum ratio of the probability of a multiclass error to the number of binary errors** (“**efficiency**”)?

Error bound proof

1. If we have a **false negative** on one of the binary classifiers (assuming all other classifiers correctly output negative)

What is the probability that we will make an incorrect multiclass prediction?

$$(K - 1) / K$$

Efficiency: $[(K - 1) / K] / 1 = (K - 1) / K$

Error bound proof

2. If we have m **false positives** with the binary classifiers

What is the probability that we will make an incorrect multiclass prediction?

- If there is also a false negative: 1
  - Efficiency =1 / (m + 1)
- Otherwise m / (m + 1)
  - Efficiency = [m / (m + 1)] / m = 1 / (m + 1)

Error bound proof

3. What is the worst case scenario?

False negative case: efficiency is (K-1)/K

Larger than false positive efficiencies

There are K-many opportunities to get false negative, **overall error bound is (K-1) ε**

Reduction 2: AVA

All versus all (aka all pairs)

How many binary classifiers does this require?

Algorithm 14 $\textsc{AllVersusAllTrain}(\mathbf{D}^{multiclass}, \textsc{BinaryTrain})$

$f_{ij} \leftarrow \varnothing, \forall 1 \leq i < j \leq K$
for $i = 1$ **to** K-1 **do**
  $\mathbf{D}^{pos} \leftarrow$ all $\boldsymbol{x} \in \mathbf{D}^{multiclass}$ labeled i
  for $j = i$+1 **to** K **do**
    $\mathbf{D}^{neg} \leftarrow$ all $\boldsymbol{x} \in \mathbf{D}^{multiclass}$ labeled j
    $\mathbf{D}^{bin} \leftarrow \{(\boldsymbol{x}, +1) : \boldsymbol{x} \in \mathbf{D}^{\text{pos}}\} \cup \{(\boldsymbol{x}, -1) : \boldsymbol{x} \in \mathbf{D}^{\text{neg}}\}$
    $f_{ij} \leftarrow \textsc{BinaryTrain}(\mathbf{D}^{bin})$
  end for
end for
return all f_{ij}s

Algorithm 15 $\textsc{AllVersusAllTest}(\text{all } f_{ij}, \hat{\boldsymbol{x}})$

$score \leftarrow \langle 0, 0, \ldots, 0 \rangle$ // initialize K-many scores to zero
for $i = 1$ **to** K-1 **do**
  for $j = i$+1 **to** K **do**
    $y \leftarrow f_{ij}(\hat{\boldsymbol{x}})$
    $score_i \leftarrow score_i + y$
    $score_j \leftarrow score_j - y$
  end for
end for
return $\text{argmax}_k \; score_k$

Time complexity

- Suppose you have N training examples, in K classes. How long does it take to train an AVA classifier
  - if the base binary classifier takes O(N) time to learn?
  - if the base binary classifier takes O(N^2) time to learn?

Error bound

Theorem: Suppose that the average error of the K binary classifiers is ε, then the error rate of the AVA multiclass classifier is at most 2(K-1) ε

Question: Does this mean that AVA is always worse than OVA?

Extensions

Divide and conquer

- Organize classes into binary tree structures

Use confidence to weight predictions of binary classifiers

- Instead of using majority vote

Topics

Given an arbitrary method for binary classification, how can we learn to make multiclass predictions?

OVA, AVA

Fundamental ML concept: reductions

Furong Huang

4124 IRB, College Park, MD 20740

301.405.8010 / furongh@umd.edu