ML in Practice: Dealing with imbalanced data

CMSC 422
MARINE CARPUAT
marine@cs.umd.edu
Topics

• A few practical issues
  – CIML Chapter 4

• Dealing with imbalanced data distributions
  – Evaluation metrics (CIML 4.5)
  – Learning with imbalanced data (CIML 5.1)
Practical Issues

• “garbage in, garbage out”
  – Learning algorithms can’t compensate for useless training examples
    • e.g., if we only have irrelevant features
  – Feature design often has a bigger impact on performance than tweaking the learning algorithm
Practical Issues

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy on test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team A</td>
<td>80.00</td>
</tr>
<tr>
<td>Team B</td>
<td>79.90</td>
</tr>
<tr>
<td>Team C</td>
<td>79.00</td>
</tr>
<tr>
<td>Team D</td>
<td>78.00</td>
</tr>
</tbody>
</table>

Which classifier is the best?
– this result table alone cannot give us the answer
– solution: statistical hypothesis testing
## Practical Issues

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy on test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team A</td>
<td>80.00</td>
</tr>
<tr>
<td>Team B</td>
<td>79.90</td>
</tr>
<tr>
<td>Team C</td>
<td>79.00</td>
</tr>
<tr>
<td>Team D</td>
<td>78.00</td>
</tr>
</tbody>
</table>

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%

• does NOT mean

“There is a 95% chance than classifier A is better than classifier B”

• 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 your implementation buggy?
Practical Issues: Debugging

• You probably have a bug
  – if the learning algorithm cannot overfit the training data
  – if the predictions are incorrect on a toy 2D dataset hand-crafted to be learnable
Topics

• A few practical issues
  – CIML Chapter 4

• Dealing with imbalanced learning problems
  – Evaluation metrics (CIML 4.5)
  – Learning with imbalanced data (CIML 5.1)
Evaluation metrics: beyond accuracy/error

• Example 1
  – Given medical record,
  – Predict whether a patient has cancer or not

• Example 2
  – Given a document collection and a query
  – Find documents in collection that are relevant to query

• Accuracy is not a good metric when some errors matter more than others!
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

<table>
<thead>
<tr>
<th></th>
<th>Gold label = +1</th>
<th>Gold label = -1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prediction = +1</td>
<td>tp</td>
<td>fp</td>
</tr>
<tr>
<td>Prediction = -1</td>
<td>fn</td>
<td>tn</td>
</tr>
</tbody>
</table>
**Precision and recall**

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

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

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Gold label = +1</th>
<th>Gold label = -1</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>tp</td>
<td>fp</td>
</tr>
<tr>
<td>-1</td>
<td>fn</td>
<td>tn</td>
</tr>
</tbody>
</table>
A combined measure: $F$

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

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

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

• A few practical issues
  – CIML Chapter 4

• Dealing with imbalanced learning problems
  – Evaluation metrics (CIML 4.5)
  – Learning with imbalanced data (CIML 5.1)
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?
Learning with imbalanced data

• We need to let the learning algorithm know that we care about some examples more than others!

• 2 heuristics to balance the training data
  – Subsampling
  – Weighting
Recall: Machine Learning as Function Approximation

Problem setting
• Set of possible instances $X$
• Unknown target function $f : X \to Y$
• Set of function hypotheses $H = \{ h | h : X \to Y \}$

Input
• Training examples $\{(x^{(1)}, y^{(1)}), \ldots (x^{(N)}, y^{(N)})\}$ of unknown target function $f$

Output
• Hypothesis $h \in H$ that best approximates target function $f$
Recall: Loss Function

\[ l(y, f(x)) \] where \( y \) is the truth and \( f(x) \) is the system’s prediction

e.g. \( l(y, f(x)) = \begin{cases} 0 & \text{if } y = f(x) \\ 1 & \text{otherwise} \end{cases} \)

Captures our notion of what is important to learn
Recall: Expected loss

- $f$ should make good predictions
  - as measured by loss $l$
  - on future examples that are also drawn from $D$

Formally

- $\epsilon$, the expected loss of $f$ over $D$ with respect to $l$ should be small

$$\epsilon \triangleq \mathbb{E}_{(x,y) \sim D}\{l(y, f(x))\} = \sum_{(x,y)} D(x, y)l(y, f(x))$$
**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: $\mathbb{E}_{(x,y) \sim \mathcal{D}} [f(x) \neq y]$
We define cost of misprediction as:

\[ \alpha > 1 \text{ for } y = +1 \]

\[ 1 \text{ for } y = -1 \]

Given a good algorithm for solving the binary classification problem, how can I solve the \( \alpha \)-weighted binary classification problem?
Solution: Train a binary classifier on an induced distribution

**Algorithm 11** \texttt{SubsampleMap}(\texttt{D}^{\text{weighted}}, \alpha)

1: \textbf{while} true \textbf{do}
2: \quad \textbf{if} \ y = +1 \textbf{ or } u < \frac{1}{\alpha} \textbf{ then}
3: \quad \quad \textbf{return} \ (x, y)
4: \quad \textbf{end if}
5: \textbf{end while}

\[
\begin{aligned}
\text{(}x, y\text{)} &\sim \texttt{D}^{\text{weighted}} \quad \text{// draw an example from the weighted distribution} \\
u &\sim \text{uniform random variable in } [0, 1] \\
\end{aligned}
\]
Subsampling optimality

• **Theorem:** If the binary classifier achieves a binary error rate of $\varepsilon$, then the error rate of the $\alpha$-weighted classifier is $\alpha \varepsilon$

• Proof (CIML 5.1)
Strategies for inducing a new binary distribution

• Undersample the negative class

• Oversample the positive 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!
Algorithm 1 DecisionTreeTrain(data, remaining features)

1: guess ← most frequent answer in data  // default answer for this data
2: if the labels in data are unambiguous then
3:    return Leaf(guess)  // base case: no need to split further
4: else if remaining features is empty then
5:    return Leaf(guess)  // base case: cannot split further
6: else
7:    for all $f \in$ remaining features do
8:        NO ← the subset of data on which $f=\text{no}$
9:        YES ← the subset of data on which $f=\text{yes}$
10:       score[f] ← # of majority vote answers in NO
11:          + # of majority vote answers in YES
12:          // the accuracy we would get if we only queried on $f$
13:    end for
14:    $f \leftarrow$ the feature with maximal score(f)
15:    NO ← the subset of data on which $f=\text{no}$
16:    YES ← the subset of data on which $f=\text{yes}$
17:    left ← DecisionTreeTrain(NO, remaining features \ \{f\})
18:    right ← DecisionTreeTrain(YES, remaining features \ \{f\})
19:    return Node(f, left, right)
20: end if
What you should know

• Be aware of practical issues when applying ML techniques to new problems

• How to select an appropriate evaluation metric for imbalanced learning problems

• How to learn from imbalanced data using $\alpha$-weighted binary classification, and what the error guarantees are