Slides adapted from Prof Carpuat and Duraiswami

CMSC 422 Introduction to Machine Learning

Lecture 8 Practical Issues: Features, Evaluation, Debugging

Furong Huang / furongh@cs.umd.edu

UNIVERSITY OF MARYLAND

Theorem (Block & Novikoff, 1962)

If the training data $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$ is **linearly separable** with margin γ by a unit norm hyperplane w_* ($||w_*|| = 1$) with $b = 0$, then **perceptron training converges after $\frac{R^2}{\gamma^2}$ errors** during training (assuming ($||x|| < R$) for all x).

Proof:

- Margin of $\mathbf{w}_*$ on any *arbitrary example* $(\mathbf{x}_n, y_n)$: $\frac{y_n \mathbf{w}_*^T \mathbf{x}_n}{||\mathbf{w}_*||} = y_n \mathbf{w}_*^T \mathbf{x}_n \geq \gamma$
- Consider the $(k+1)^{th}$mistake: $y_n \mathbf{w}_k^T \mathbf{x}_n \leq 0$, and update $\mathbf{w}_{k+1} = \mathbf{w}_k + y_n \mathbf{x}_n$
- $\mathbf{w}_{k+1}^T \mathbf{w}_* = \mathbf{w}_k^T \mathbf{w}_* + y_n \mathbf{w}_*^T \mathbf{x}_n \geq \mathbf{w}_k^T \mathbf{w}_* + \gamma$ (why is this nice?)
- Repeating iteratively k times, we get $\mathbf{w}_{k+1}^T \mathbf{w}_* > k\gamma$ (1)
- $||\mathbf{w}_{k+1}||^2 = ||\mathbf{w}_k||^2 + 2y_n \mathbf{w}_k^T \mathbf{x}_n + ||\mathbf{x}||^2 \leq ||\mathbf{w}_k||^2 + R^2$ (since $y_n \mathbf{w}_k^T \mathbf{x}_n \leq 0$)
- Repeating iteratively k times, we get $||\mathbf{w}_{k+1}||^2 \leq kR^2$ (2)

Theorem (Block & Novikoff, 1962)

If the training data $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$ is **linearly separable** with margin γ by a unit norm hyperplane w_* ($||w_*||= 1$) with $b = 0$, then **perceptron training converges after $\frac{R^2}{\gamma^2}$ errors** during training (assuming ($||x||< R$) for all x).

What does this mean?

- Perceptron converges quickly when margin is large, slowly when it is small
- Bound does not depend on number of training examples N, nor on number of features d
- **Proof guarantees that perceptron converges, but not necessarily to the max margin separator**

What you should know

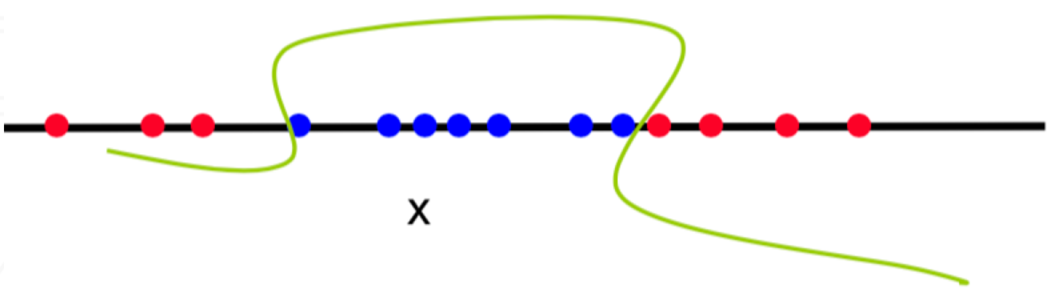
- Perceptron concepts
  - training/prediction algorithms (standard, voting, averaged)
  - convergence theorem and what practical guarantees it gives us
  - how to draw/describe the decision boundary of a perceptron classifier
- Fundamental ML concepts
  - Determine whether a data set is linearly separable and define its margin
  - Error driven algorithms, online vs. batch algorithms

Expressivity

- Many functions are linear
  - Conjunctions:
    - $y = x_1 \cap x_3 \cap x_5$
    - $y = sign(1 \times x_1 + 1 \times x_3 + 1 \times x_5 - 3)$, $\mathrm{w} = [1{,}0{,}1{,}0{,}1]$
  - At least m of n:
    - $y = at\ least\ 2\ of\ (x_1, x_3, x_5)$
- Many functions are not
  - Xor: $y = x_1 \cap x_2 \cup \neg x_1 \cap \neg x_2$
  - Non trival DNF: $y = x_1 \cap x_2 \cup x_3 \cap x_4$
- But can be made linear

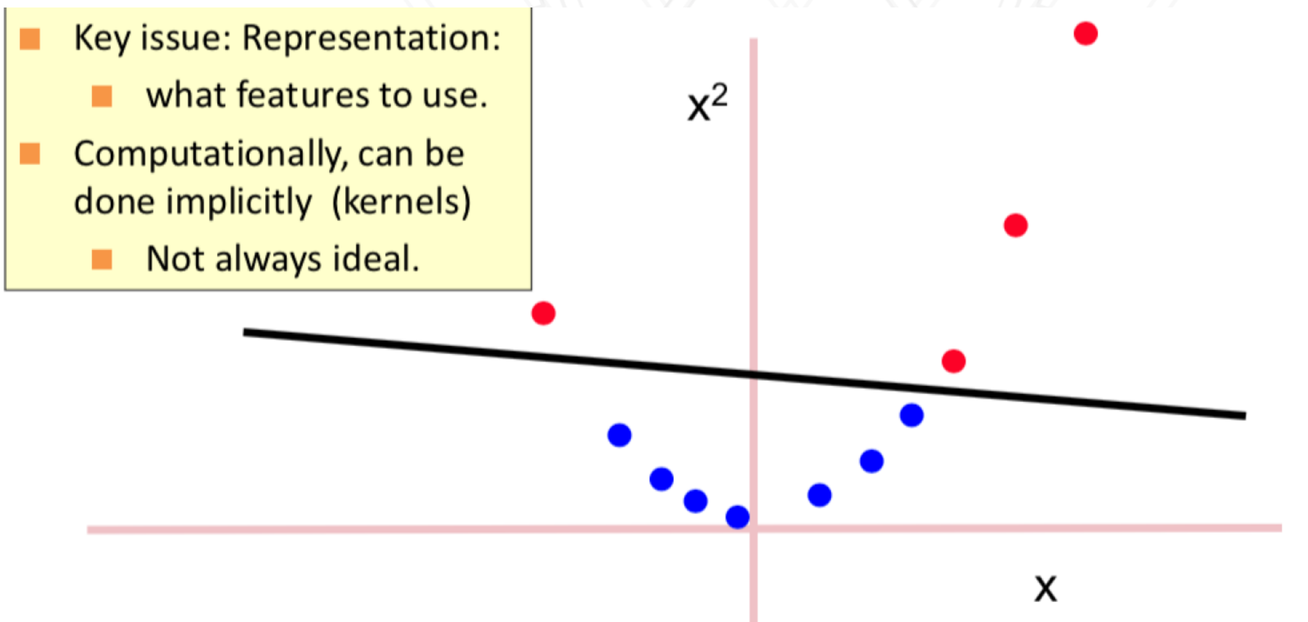
Functions Can be Made Linear

- Data are not linearly separable in one dimension
- Not separable if you insist on using a specific class of functions



Blown Up Feature Space

Data are separable in $<x, x^2>$ space

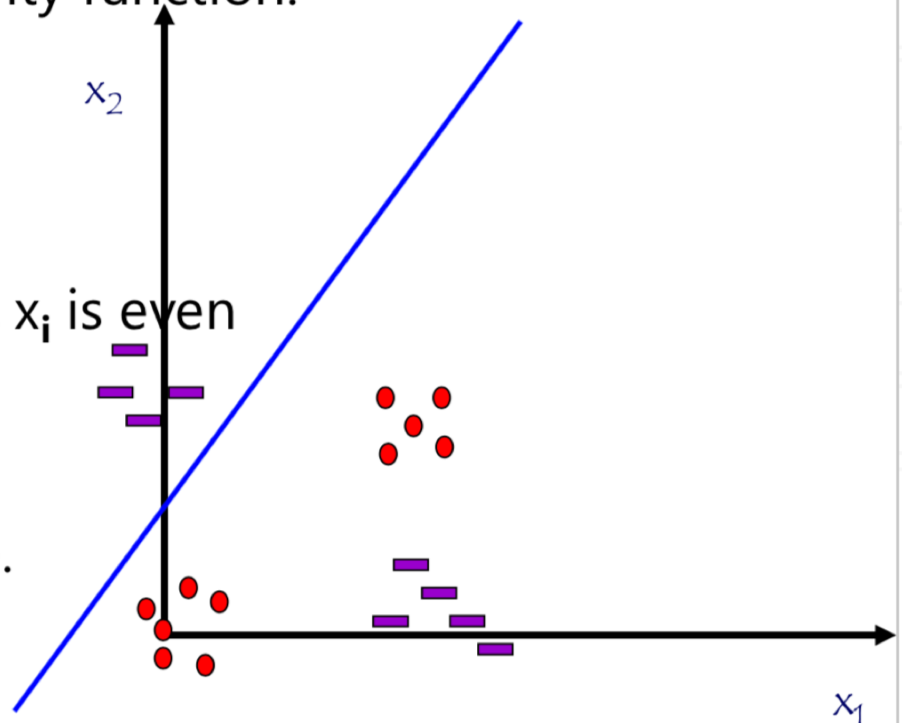


Exclusive-OR (XOR)

- $y = x_1 \wedge x_2 \vee \neg x_1 \wedge \neg x_2$

- In general: a parity function.
- $x_i \in \{0,1\}$
- $f(x_1, x_2, \ldots, x_n) = 1$ iff $\sum x_i$ is even

This function is not linearly separable.



Practical Issues

- "garbage in, garbage out"
  - Learning algorithms can't compensate for useless training examples
    - E.g., if all features are irrelevant
  - Feature design can have bigger impact on performance than tweaking the learning algorithm


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

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%

- 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 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

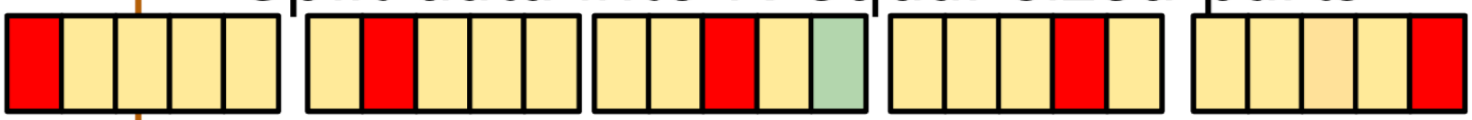
Algorithm 8 CrossValidate(*LearningAlgorithm*, *Data*, K)

$\hat{\epsilon} \leftarrow \infty$ // 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**
    $train \leftarrow \{(x_n, y_n) \in Data : n \mod K \neq k - 1\}$
    $test \leftarrow \{(x_n, y_n) \in Data : n \mod K = k - 1\}$ // 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

Improving Input Representations

- Feature pruning
- Feature normalization

$$
\begin{aligned}
\text{Centering:} \quad & x_{n,d} \leftarrow x_{n,d} - \mu_d & (5.1)\\
\text{Variance Scaling:} \quad & x_{n,d} \leftarrow x_{n,d} / \sigma_d & (5.2)\\
\text{Absolute Scaling:} \quad & x_{n,d} \leftarrow x_{n,d} / r_d & (5.3)\\
\textit{where:} \quad & \mu_d = \frac{1}{N} \sum_n x_{n,d} & (5.4)\\
& \sigma_d = \sqrt{\frac{1}{N-1} \sum_n (x_{n,d} - \mu_d)^2} & (5.5)\\
& r_d = \max_n \left| x_{n,d} \right| & (5.6)
\end{aligned}
$$

- Example normalization

$$\boldsymbol{x}_n \leftarrow \boldsymbol{x}_n / \, ||\boldsymbol{x}_n||$$

See CIML 5.3

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

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

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

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

A combined measure: F

- 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

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

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

UNIVERSITY OF MARYLAND

Furong Huang

3251 A.V. Williams, College Park, MD 20740

301.405.8010 / furongh@cs.umd.edu