CIS 833 – Information Retrieval and Text Mining

Lecture 22

Text Classification

November 17, 2015

Credits for slides: Allan, Arms, Manning, Lund, Noble, Page.

Planning

- PageRank implementation: due Dec 1st
- Exam review: Dec 1st
- Final exam: Dec 3rd
- Project presentation: Dec. 17th (9:40 AM – 11:30 AM)
- Project report: Dec. 18th

Textbook Material

- Next – Text Classification
  - Chapter 13: Text Classification and Naïve Bayes
  - Chapter 14: Vector Space Classification
  - Chapter 15: Support Vector Machines

Learning Algorithms for Classification Tasks

- Relevance Feedback (Rocchio)
- k-Nearest Neighbors (simple, powerful)
- Naive Bayes (simple, common method)
- Support-vector machines (new, more powerful)
- … plus many other methods

Nearest-Neighbor Learning Algorithm

- Learning is just storing the representations of the training examples in *D*.
- Testing instance *x*:
  - Compute similarity between *x* and all examples in *D*.
  - Assign *x* the category of the most similar example in *D*.
- Does not explicitly compute a generalization or category prototypes.
- Also called:
  - Case-based
  - Memory-based
  - Lazy learning

K Nearest-Neighbor

- Using only the closest example to determine categorization is subject to errors due to:
  - A single atypical example.
  - Noise (i.e., error) in the category label of a single training example.
- More robust alternative is to find the *k* most-similar examples and return the majority category of these *k* examples.
- Value of *k* is typically odd to avoid ties, 3 and 5 are most common.

Similarity Metrics

- Nearest neighbor method depends on a similarity (or distance) metric.
- Simplest for continuous *m*-dimensional instance space is *Euclidian distance*.
- Simplest for *m*-dimensional binary instance space is *Hamming distance* (number of feature values that differ).
- For text, cosine similarity of TF-IDF weighted vectors is typically most effective.

3 Nearest Neighbor Illustration
(Euclidian Distance)

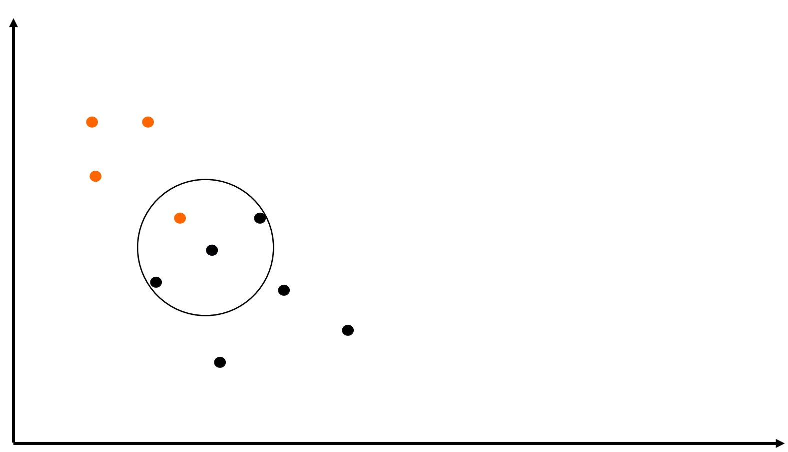
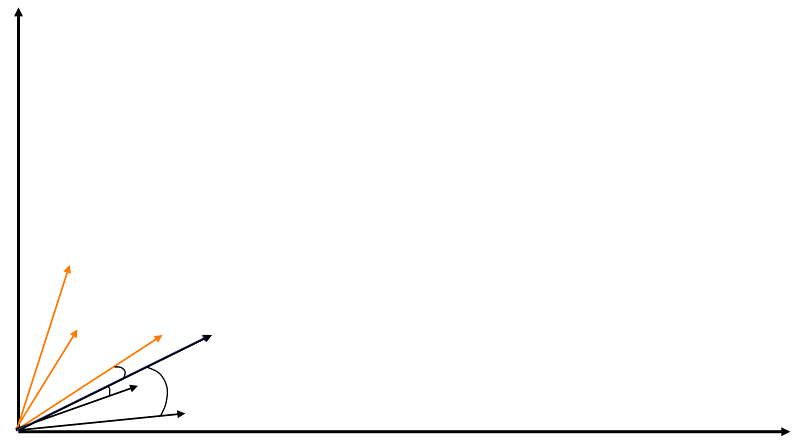
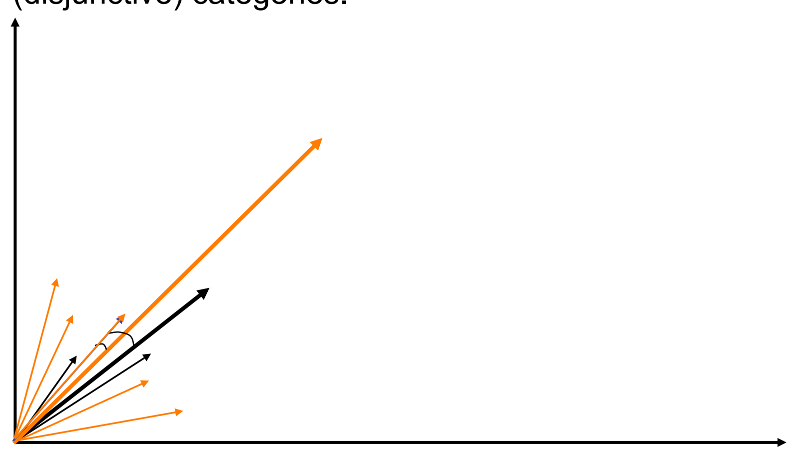


Illustration of 3 Nearest Neighbor for Text



Rocchio Anomaly

- Prototype models have problems with polymorphic (disjunctive) categories.

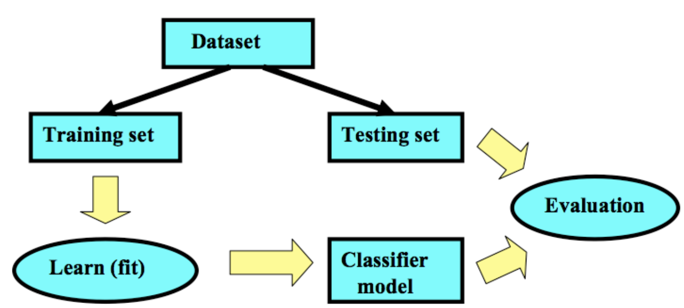


3 Nearest Neighbor Comparison

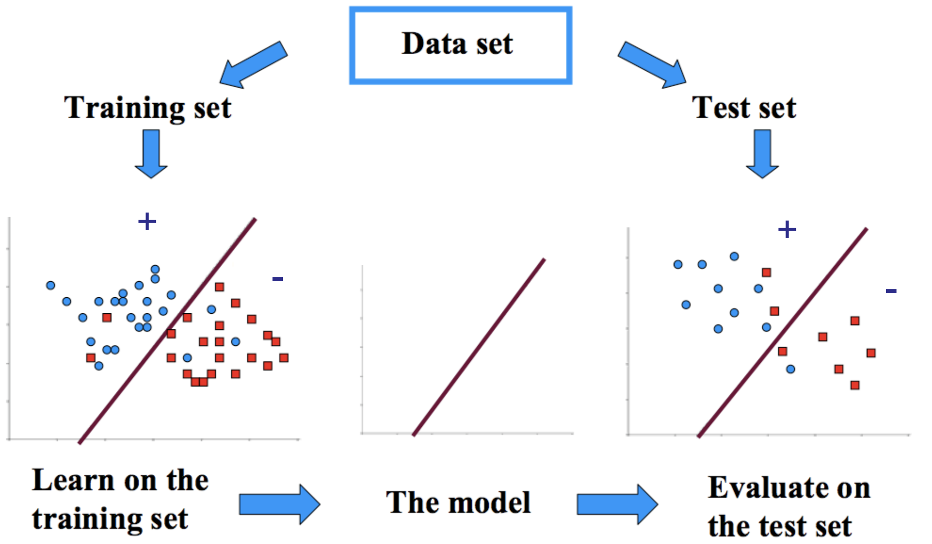
- Nearest Neighbor tends to handle polymorphic categories better.

Evaluation Framework

- We want our classifier to generalize well to future examples
- Problem: But we do not know future examples !!!
- Solution: evaluate the classifier on the test set that is withheld from the learning stage



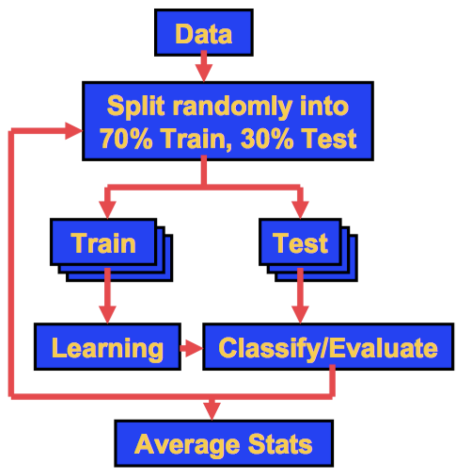
Evaluation Framework



Methods to Estimate Performance

- Holdout
  - Reserve ½ for training and ½ for testing
  - Reserve 2/3 for training and 1/3 for testing
- To limit the effect of one lucky or unlucky train/test split it is common to average through:
  - Random subsampling
    - Repeated holdout
  - Cross validation
    - Partition data into k disjoint subsets
    - k-fold: train on k-1 partitions, test on the remaining one
    - Leave-one-out: k=n
  - Stratified sampling
    - oversampling vs undersampling
  - Bootstrap sampling with replacement

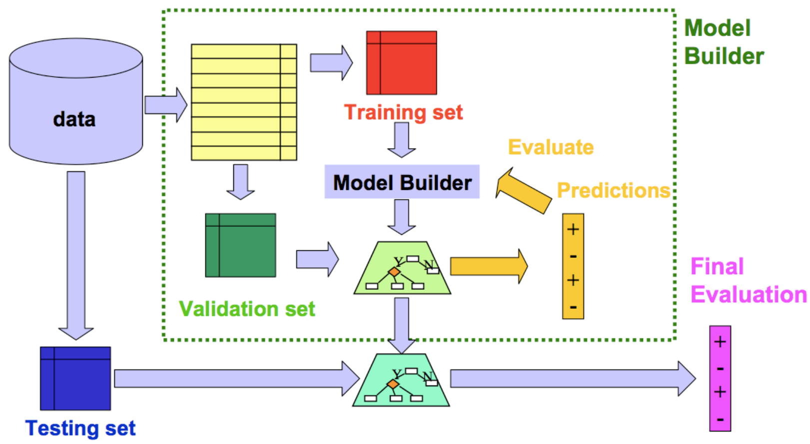
Random subsampling



A Note on Parameter Tuning

- It is important that the test data is not used in any way to create the classifier
- Some learning schemes operate in two stages:
  - Stage 1: builds the basic structure
  - Stage 2: optimizes parameter settings
- The test data can't be used for parameter tuning!
- Proper procedure uses three sets:
  - training data, validation data, and test data
- Validation data is used to optimize parameters

Train, Validation, and Test



Evaluation Measures

Start with a CONTINGENCY table

	actual +	actual -
predicted +	*TP*	*FP*
predicted -	*FN*	*TN*

Sensitivity/Recall

$$SN = \frac{TP}{TP+FN}$$

Specificity

$$SN = \frac{TN}{TN+FP}$$

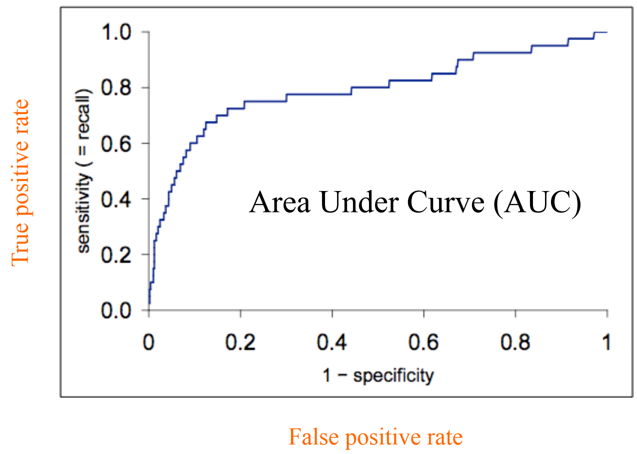
Precision

$$PR = \frac{TP}{TP+FP}$$

Accuracy $= \dfrac{TP+TN}{N}$ Error $= \dfrac{FP+FN}{N}$

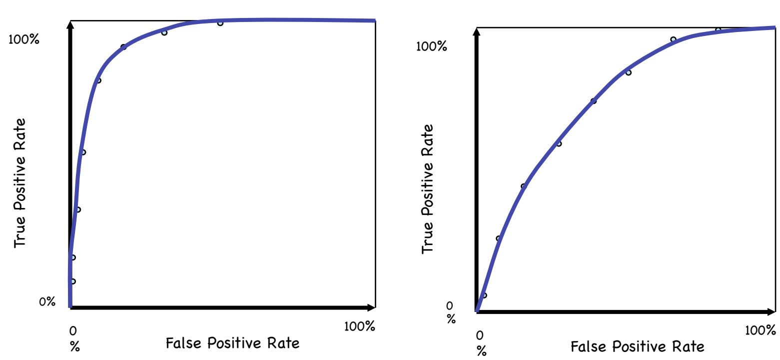
where N=TP+FP+FN+TN

ROC Curve



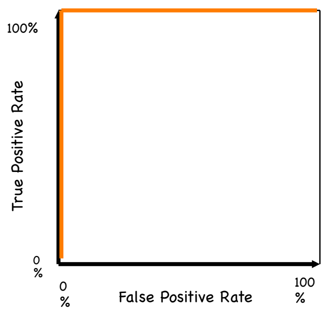
$$FPR = \frac{FP}{FP + TN}$$

ROC Curve Comparison



ROC Curve Extremes

Best classifier:



Worst classifier:

The distributions don't overlap at all

The distributions overlap completely