CS276B

Text Information Retrieval, Mining, and Exploitation

Lecture 9
Text Classification IV
Feb 13, 2003

Today's Topics

- More algorithms
  - Vector space classification
  - Nearest neighbor classification
  - Support vector machines
- Hypertext classification

Vector Space Classification
K Nearest Neighbor Classification

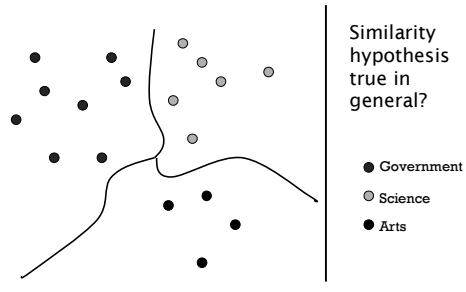
Recall Vector Space Representation

- Each document is a vector, one component for each term (= word).
- Normalize to unit length.
- Properties of vector space
  - terms are axes
  - *n* docs live in this space
  - even with stemming, may have 10,000+ dimensions, or even 1,000,000+

Classification Using Vector Spaces

- Each training doc a point (vector) labeled by its class
- Similarity hypothesis: docs of the same class form a contiguous region of space. Or: Similar documents are usually in the same class.
- Define surfaces to delineate classes in space

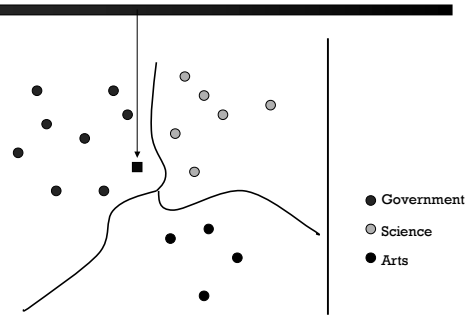
Classes in a Vector Space



Given a Test Document

- Figure out which region it lies in
- Assign corresponding class

Test Document = Government



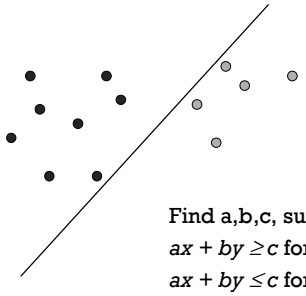
Binary Classification

- Consider 2 class problems
- How do we define (and find) the separating surface?
- How do we test which region a test doc is in?

Separation by Hyperplanes

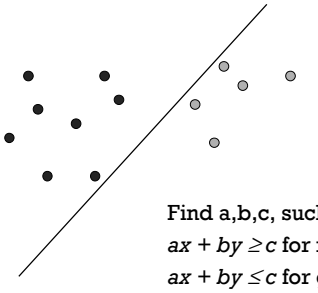
- Assume *linear separability* for now:
  - in 2 dimensions, can separate by a line
  - in higher dimensions, need hyperplanes
- Can find separating hyperplane by *linear programming* (e.g. perceptron):
  - separator can be expressed as $ax + by = c$

Linear Programming / Perceptron



Find a,b,c, such that
$ax + by \geq c$ for red points
$ax + by \leq c$ for green points.

Relationship to Naïve Bayes?

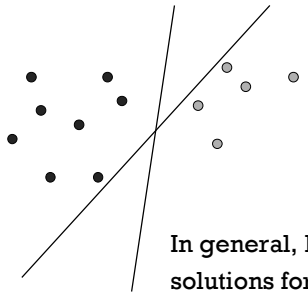


Find a,b,c, such that
$ax + by \geq c$ for red points
$ax + by \leq c$ for green points.

Linear Classifiers

- Many common text classifiers are linear classifiers
- Despite this similarity, large performance differences
  - For separable problems, there is an infinite number of separating hyperplanes. Which one do you choose?
  - What to do for non-separable problems?

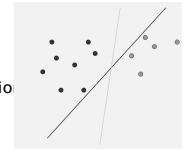
Which Hyperplane?



In general, lots of possibl solutions for *a,b,c.*

Which Hyperplane?

- Lots of possible solutions for *a,b,c.*
- Some methods find a separating hyperplane, but not the optimal one (e.g., perceptron)
- Most methods find an optimal separating hyperplane
- Which points should influence optimality?
  - All points
    - Linear regression
    - Naïve Bayes
  - Only "difficult points" close to decisio boundary
    - Support vector machines
    - Logistic regression (kind of)



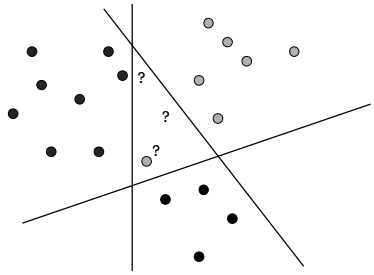
Hyperplane: Example

- Class: "interest" (as in interest rate)
- Example features of a linear classifier (SVM)
w_i	t_i	w_i	t_i
0.70	prime	-0.71	dlrs
0.67	rate	-0.35	world
0.63	interest	-0.33	sees
0.60	rates	-0.25	year
0.46	discount	-0.24	group
0.43	bundesbank	-0.24	dlr

More Than Two Classes

- One-of classification: each document belongs to exactly one class
  - How do we compose separating surfaces into regions?
- Any-of or multiclass classification
  - For n classes, decompose into n binary problems
- Vector space classifiers for one-of classification
  - Use a set of binary classifiers
  - Centroid classification
  - K nearest neighbor classification

Composing Surfaces: Issues

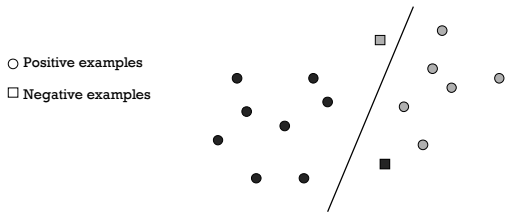


Set of Binary Classifiers

- Build a separator between each class and its complementary set (docs from all other classes).
- Given test doc, evaluate it for membership in each class.
- For one-of classification, declare membership in classes for class with:
  - maximum score
  - maximum confidence
  - maximum probability
- Why different from multiclass classification?

Negative Examples

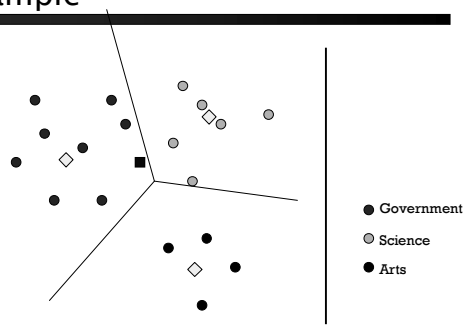
- Formulate as above, except negative examples for a class are added to its complementary set.



Centroid Classification

- Given training docs for a class, compute their centroid
- Now have a centroid for each class
- Given query doc, assign to class whose centroid is nearest.
- Compare to Rocchio

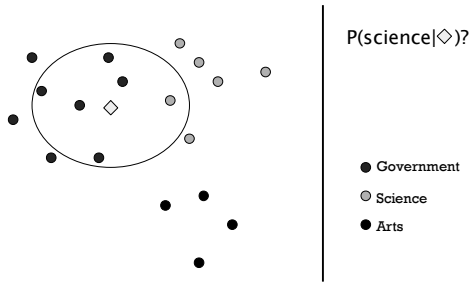
Example



k Nearest Neighbor Classification

- To classify document d into class c
- Define k-neighborhood N as k nearest neighbors of d
- Count number of documents l in N that belong to c
- Estimate P(c|d) as l/k

Example: k=6 (6NN)



Cover and Hart 1967

- Asymptotically, the error rate of 1-nearest-neighbor classification is less than twice the Bayes rate.
- Assume: query point coincides with a training point.
- Both query point and training point contribute error -> 2 times Bayes rate
- In particular, asymptotic error rate 0 if Bayes rate is 0.

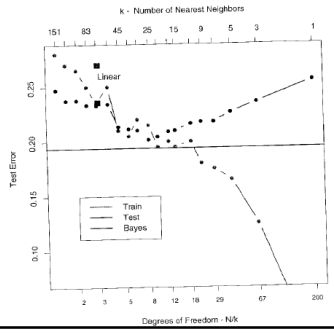
kNN vs. Regression

- Bias/Variance tradeoff
- Variance ≈ Capacity
- kNN has high variance and low bias.
- Regression has low variance and high bias.
- Consider: Is an object a tree? (Burges)
- Too much capacity/variance, low bias
  - Botanist who memorizes
  - Will always say "no" to new object (e.g., #leaves)
- Not enough capacity/variance, high bias
  - Lazy botanist
  - Says "yes" if the object is green

kNN: Discussion

- Classification time linear in training set
- No feature selection necessary
- Scales well with large number of classes
  - Don't need to train n classifiers for n classes
- Classes can influence each other
  - Small changes to one class can have ripple effect
- Scores can be hard to convert to probabilities
- No training necessary
  - Actually: not true. Why?

Number of Neighbors

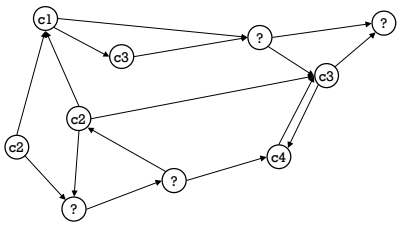


Hypertext Classification

Classifying Hypertext

- Given a set of hyperlinked docs
- Class labels for some docs available
- Figure out class labels for remaining docs

Example



Bayesian Hypertext Classification

- Besides the terms in a doc, derive cues from linked docs to assign a class to test doc.
- Cues could be any abstract features from doc and its neighbors.

Feature Representation

- Attempt 1:
  - use terms in doc + those in its neighbors.
- Generally does worse than terms in doc alone. Why?

Representation: Attempt 2

- Use terms in doc, plus tagged terms from neighbors.
- E.g.,
  - ***car*** denotes a term occurring in *d*.
  - ***car@I*** denotes a term occurring in a doc with a link into *d*.
  - ***car@O*** denotes a term occurring in a doc with a link from *d*.
- Generalizations possible: ***car@OIOI***

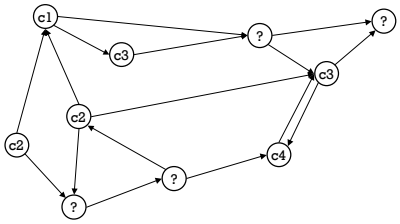
Attempt 2 Also Fails

- Key terms lose density
- e.g., ***car*** gets split into ***car, car@I, car@O***

Better Attempt

- Use class labels of (in- and out-) neighbors as features in classifying *d*.
  - e.g., docs about physics point to docs about physics.
- Setting: some neighbors have pre-assigned labels; need to figure out the rest.

Example



Content + Neighbors' Classes

- Naïve Bayes gives Pr[c_j|d] based on the words in d.
- Now consider Pr[c_j|N] where N is the set of labels of d's neighbors.
  (Can separate N into in- and out-neighbors.)
- Can combine conditional probs for c_j from text- and link-based evidence.

Training

- As before, use training data to compute Pr[N|c_j] etc.
- Assume labels of d's neighbors independent (as we did with word occurrences).
- (Also continue to assume word occurrences within d are independent.)

Classification

- Can invert probs using Bayes to derive Pr[c_j|N].
- Need to know class labels for all of d's neighbors.

Unknown Neighbor Labels

- What if all neighbors' class labels are not known?
- First, use word content alone to assign a tentative class label to each unlabelled doc.
- Next, iteratively recompute all tentative labels using word content as well as neighbors' classes (some tentative).

Convergence

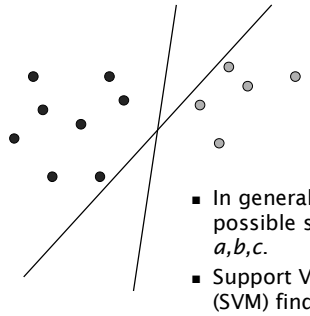
- This iterative relabeling will converge provided tentative labels "not too far off".
- Guarantee requires ideas from Markov random fields, used in computer vision.
- Error rates significantly below text-alone classification.

Typical Empirical Observations

- Training ~ 100's to 1000+ docs/class
- Accuracy
  ~ 90% in the very best circumstances
  below 50% in the worst

Support Vector Machines

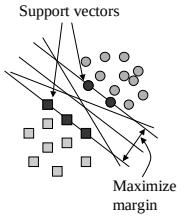
Recall: Which Hyperplane?



- In general, lots of possible solutions for *a,b,c*.
- Support Vector Machine (SVM) finds an optimal

Support Vector Machine (SVM)

- SVMs maximize the *margin* around the separating hyperplane.
- The decision function is fully specified by a subset of training samples, *the support vectors*.
- *Quadratic programming* problem
- Text classification method du jour



Maximum Margin: Formalization

- w: hyperplane normal
- x_i: data point i
- y_i: class of data point i (+1 or -1)
- Constraint optimization formalization:
- (1) $\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \quad \text{for } y_i = +1$
  $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \quad \text{for } y_i = -1$
- (2) maximize margin: 2/||w||

Quadratic Programming

- One can show that hyperplane w with maximum margin is:

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

- alpha_i: lagrange multipliers
- x_i: data point i
- y_i: class of data point i (+1 or -1)
- Where the a_i are the solution to maximizing:

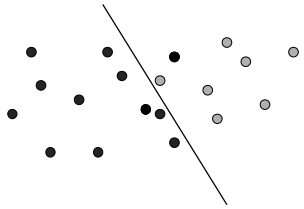
$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

Most alpha_i will be zer

Building an SVM Classifier

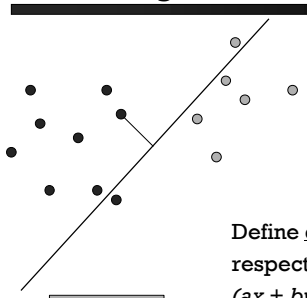
- Now we know how to build a separator for two linearly separable classes
- What about classes whose exemplary docs are not linearly separable?

Not Linearly Separable



Find a line that penalizes points on "the wrong side

Penalizing Bad Points

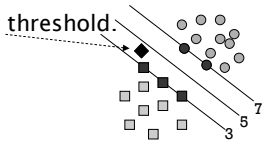


Define distance for each point with respect to separator $ax + by = c$: $(ax + by) - c$ for red points; $c - (ax + by)$ for green points.

Negative for bad points.

Solve Quadratic Program

- Solution gives "separator" between two classes: choice of a,b.
- Given a new point (x,y), can score its proximity to each class:
  - evaluate $ax+by$.
  - Set confidence threshold.



Predicting Generalization

- We want the classifier with the best generalization (best accuracy on new data).
- What are clues for good generalization?
  - Large training set
  - Low error on training set
  - Low capacity/variance (≈ model with few parameters)
- SVMs give you an explicit bound based on these.

Capacity/Variance: VC Dimension

- Theoretical risk boundary:

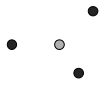
$$R(\alpha) \le R_{emp}(\alpha) + \sqrt{\left(\frac{h(\log(2l/h)+1) - \log(\eta/4)}{l}\right)}$$

- Remp - empirical risk, l - #observations, h – VC dimension, the above holds with prob. (1-η)
- VC dimension/Capacity: max number of points that can be shattered
- A set can be shattered if the classifier can learn every possible labeling.

Capacity of Hyperplanes?

Exercise

- Suppose you have *n* points in *d* dimensions, labeled red or green. How big need *n* be (as a function of *d*) in order to create an example with the red and green points not linearly separable?
- E.g., for *d=2, n ≥ 4*.



Capacity/Variance: VC Dimension

- Theoretical risk boundary:

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\left(\frac{h(\log(2l/h)+1) - \log(\eta/4)}{l}\right)}$$

- Remp - empirical risk, l - #observations, h – VC dimension, the above holds with prob. (1-η)
- VC dimension/Capacity: max number of points that can be shattered
- A set can be shattered if the classifier can learn every possible labeling.

Kernels

- Recall: We're maximizing:

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

- Observation: data only occur in dot products.
- We can map data into a very high dimensional space (even infinite!) as long as kernel computable.
- For mapping function Φ, compute kernel K(i,j) = Φ(xi) · Φ(xj)
- Example:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^2 \qquad \Phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ \sqrt{2}\, x_1 x_2 \\ x_2^2 \end{pmatrix}$$

Kernels

- Why use kernels?

Kernels

- Why use kernels?
  - Make non-separable problem separable.
  - Map data into better representational space
- Common kernels
  - Linear
  - Polynomial
  - Radial basis function

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2}$$

Performance of SVM

- SVM are seen as best-performing method by many.
- Statistical significance of most results not clear.
- There are many methods that perform about as well as SVM.
- Example: regularized regression (Zhang&Oles)
- Example of a comparison study: Yang&Liu

Yang&Liu: SVM vs Other Methods

Table 1: Performance summary of classifiers

method	miR	miP	miF1	maF1	error
SVM	.8120	.9137	.8599	.5251	.00365
KNN	.8339	.8807	.8567	.5242	.00385
LSF	.8507	.8489	.8498	.5008	.00414
NNet	.7842	.8785	.8287	.3765	.00447
NB	.7688	.8245	.7956	.3886	.00544

miR = micro-avg recall; miP = micro-avg prec.;
miF1 = micro-avg F1; maF1 = macro-avg F1.

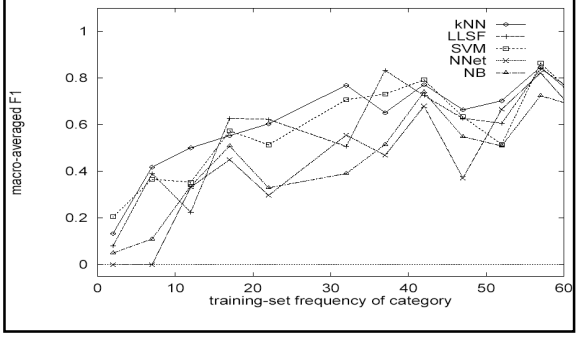
Yang&Liu: Statistical Significance

Table 2: Statistical significance test results

sysA	sysB	s-test	S-test	T-test	T'-test
SVM	kNN	$>$	$\sim$	$\sim$	$\sim$
SVM	LLSF	$\gg$	$\sim$	$\sim$	$\sim$
kNN	LLSF	$\gg$	$\sim$	$\sim$	$\sim$
SVM	NNet	$\gg$	$\gg$	$\gg$	$\gg$
kNN	NNet	$\gg$	$\gg$	$\gg$	$\gg$
LLSF	NNet	$\sim$	$\gg$	$\gg$	$\gg$
NB	kNN	$\ll$	$\ll$	$\ll$	$\ll$
NB	LLSF	$\ll$	$\ll$	$\ll$	$\ll$
NB	SVM	$\ll$	$\ll$	$\ll$	$\ll$
NB	NNet	$\ll$	$\sim$	$\sim$	$\sim$

"$\gg$" or "$\ll$" means P-value ≤ 0.01;
"$>$" or "$<$" means $0.01 <$ P-value ≤ 0.05;
"$\sim$" means P-value > 0.05.

Yang&Liu: Small Classes



Results for Kernels (Joachims)

	Bayes	Rocchio	C4.5	k-NN	SVM (poly) degree $d=$ 1	2	3	4	5	SVM (rbf) width $\gamma=$ 0.6	0.8	1.0	1.2
earn	95.9	96.1	96.1	97.3	98.2	98.4	**98.5**	98.4	98.3	**98.5**	98.5	98.4	98.3
acq	91.5	92.1	85.3	92.0	92.6	94.6	**95.2**	95.2	95.3	95.0	95.3	95.3	**95.4**
money-fx	62.9	67.6	69.4	78.2	66.9	72.5	75.4	74.9	**76.2**	74.0	75.4	**76.3**	75.9
grain	72.5	79.5	89.1	82.2	91.3	93.1	**92.4**	91.3	89.9	**93.1**	91.9	91.9	90.6
crude	81.0	81.5	75.5	85.7	86.0	87.3	88.6	**88.9**	87.8	**88.9**	89.0	88.9	88.2
trade	50.0	77.4	59.2	77.4	69.2	75.5	76.6	77.3	**77.1**	76.9	78.0	**77.8**	76.8
interest	58.0	72.5	49.1	74.0	69.8	63.3	67.9	73.1	**76.2**	74.4	75.0	**76.2**	76.1
ship	78.7	83.1	80.9	79.2	82.0	85.4	86.0	**86.5**	86.0	**85.4**	86.5	87.6	87.1
wheat	60.6	79.4	85.5	76.6	83.1	84.5	85.2	**85.9**	83.8	**85.2**	85.9	85.9	85.9
corn	47.3	62.2	87.7	77.9	86.0	86.5	85.3	**85.7**	83.9	**85.1**	85.7	85.7	84.5
microavg.	**72.0**	**79.9**	**79.4**	**82.3**	84.2	85.1	85.9	86.2	85.9	86.4	86.5	86.3	86.2
					combined: **86.0**					combined: **86.4**			

SVM: Summary

- SVM have optimal or close to optimal performance.
- Kernels are an elegant and efficient way to map data into a better representation.
- SVM can be expensive to train (quadratic programming).
- If efficient training is important, and slightly suboptimal performance ok, don't use SVM?
- For text, linear kernel is common.
- So most SVMs are linear classifiers (like many others), but find a (close to) optimal separating hyperplane.

SVM: Summary (cont.)

- Model parameters based on small subset (SVs)
- Based on structural risk minimization

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\left(\frac{h(\log(2l/h)+1) - \log(\eta/4)}{l}\right)}$$

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

Resources

- Foundations of Statistical Natural Language Processing. Chapter 16. MIT Press. Manning and Schuetze.
- Trevor Hastie, Robert Tibshirani and Jerome Friedman, "Elements of Statistical Learning: Data Mining, Inference and Prediction" Springer-Verlag, New York.
- A Tutorial on Support Vector Machines for Pattern Recognition (1998) Christopher J. C. Burges
- Data Mining and Knowledge Discovery
- R.M. Tong, L.A. Appelbaum, V.N. Askman, J.F. Cunningham. Conceptual Information Retrieval using RUBRIC. Proc. ACM SIGIR 247-253, (1987).
- S. T. Dumais, Using SVMs for text categorization, IEEE Intelligent Systems, 13(4), Jul/Aug 1998
- Yiming Yang, S. Slattery and R. Ghani. A study of approaches to hypertext categorization Journal of Intelligent Information Systems, Volume 18, Number 2, March 2002.
- re-examination of text categorization methods (1999) Yiming Yang, Xin Liu 22nd Annual International SIGIR
- Tong Zhang, Frank J. Oles: Text Categorization Based on Regularized Linear Classification Methods. Information Retrieval 4(1): 5-31 (2001)