CSCI 5417
Information Retrieval Systems
Jim Martin

Lecture 13
10/6/2011

Text classification

- First
  - Naïve Bayes
    - Simple, fast, low training and testing cost
- Then
  - K Nearest Neighbor classification
    - Simple, can easily leverage inverted index, high variance, non-linear
- Today
  - Linear classifiers
    - A very quick tour
  - SVMs
  - Some empirical evaluation and comparison
  - Text-specific issues in classification

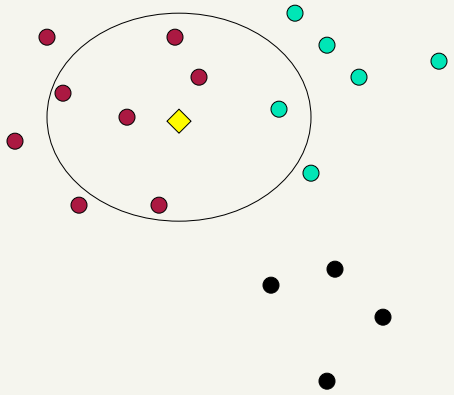
Where we are

- Classification and naïve Bayes
  - Chapter 13
- Vector space classification
  - Chapter 14
- Machine learning
  - Chapter 15

K Nearest Neighbors Classification

- To classify document *d* into class c
- Define *k*-neighborhood N as *k* nearest neighbors of *d*
- Count number of documents i in N that belong to c
- Estimate P(c|*d*) as i/k
- Choose as class argmax_c P(c|*d*)
  - I.e. majority class

Example: k=6 (6NN)



P(science|◇)?

Government

Science

Arts

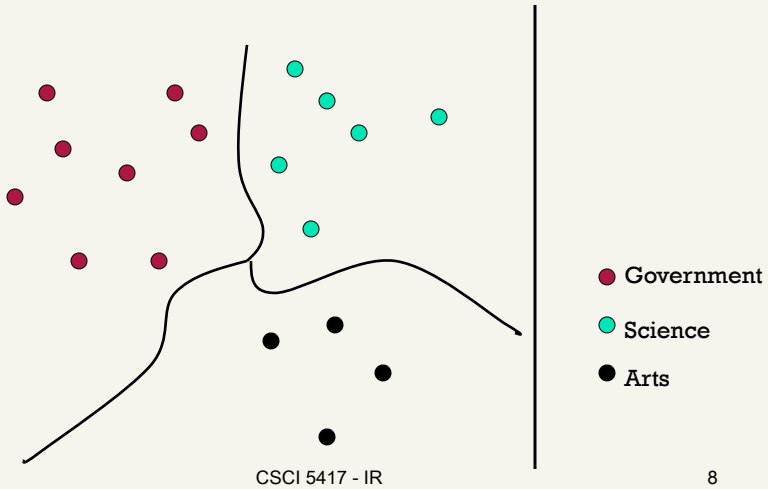
Nearest Neighbor with Inverted Index

- Naively finding nearest neighbors requires a linear search through |*D*| documents in collection
- But if cosine is the similarity metric then determining *k* nearest neighbors is the same as determining the *k* best retrievals using the test document as a query to a database of training documents.
- So just use standard vector space inverted index methods to find the *k* nearest neighbors.
- What are the caveats to this????

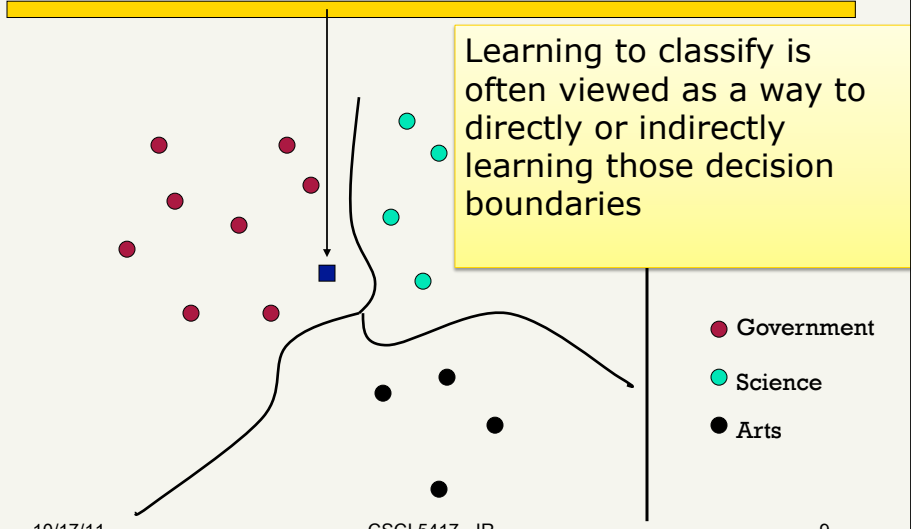
kNN: Discussion

- No feature selection necessary
- Scales well with large number of classes
  - Don't need to train *n* classifiers for *n* classes
- Scores can be hard to convert to probabilities
- No training necessary
  - Sort of… still need to figure out tf-idf, stemming, stop-lists, etc. All that requires tuning, which really is training.

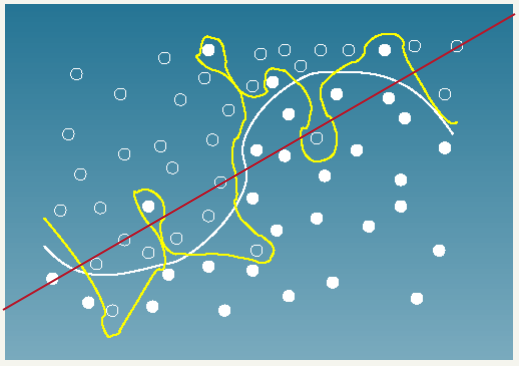
Classes in a Vector Space



Test Document = Government



Bias vs. Variance: Choosing the correct model capacity



kNN vs. Naive Bayes

- Bias/Variance tradeoff
  - Variance → Capacity
  - Bias → Generalization
- kNN has high variance and low bias.
  - Infinite memory
- NB has low variance and high bias.
- Consider: Is an object a tree?
  - Too much capacity/variance, low bias
    - Botanist who memorizes every tree
    - 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

Linear Classifiers

- Methods that attempt to separate data into classes by learning a linear separator in the space representing the objects.
- Unlike k-NN these methods explicitly seek a generalization (representation of a separator) in the space.
- Not a characterization of the classes though (ala naïve Bayes). These methods seek to characterize a way to separate the classes.

Example

Suppose you had collected data concerning the relationship between the use of vague adjectives in real estate ads and whether the house subsequently sold for more or less than the asking price (Levitt and Dubner, 2005) and by how much.

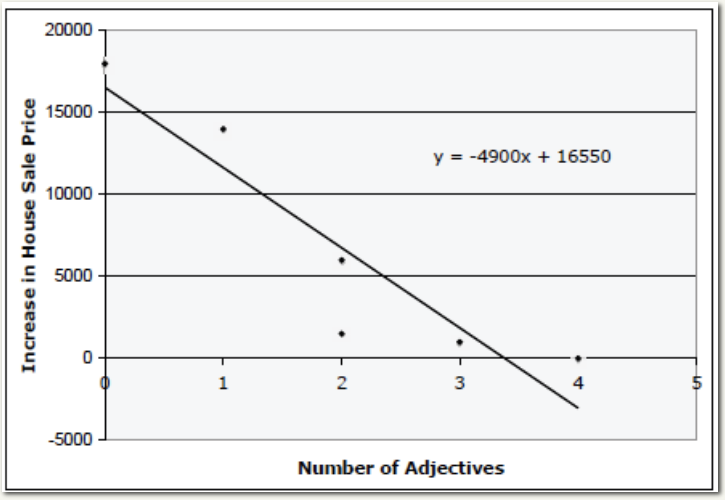
- Consider "cute" or "charming" vs. "stainless" or "granite".
- You might end up with a table like...

Classification Example

# of Vague Adjectives	Amount House Sold Over Asking Price
4	0
3	$1000
2	$1500
2	$6000
1	$14000
0	$18000

Clearly, hot properties are not associated with vague adjectives.

Linear Regression Example



Regression Example

- Definition of a line y= mx + b
  - Slope (m) and intercept (b)
  - $$$ = w_0 + w_1*Num_Adjectives
    - 16550 + -4900*Num_Adjectives
- What if you had more features?

$$price = w_0 + w_1 * \text{Num\_Adjectives} + w_2 * \text{Mortgage\_Rate} + w_3 * \text{Num\_Unsold\_Houses}$$

- In general

linear regression: $$y = \sum_{i=0}^{N} w_i \times f_i$$

Learning

- How to learn the weights?
  - The slope and intercept in our case?
- Search through the space of weights for the values that optimize some goodness metric
  - In this case, sum of the squared differences between the training examples and the predicted values.

Regression to Classification

- Regression maps numbers (features) to numbers and we're interested in mapping features to discrete categories...
- Let's think first about the binary case

Regression to Classification

- For the regression case, the line we learned is used to compute a value.
- But, given a set of +/- values we could just have easily search for a line that best separates the space into two regions (above and below) the line
  - Points above are + and the values below are -.
  - If we move beyond 2 dimensions (features) than we have a hyperplane instead of a line.

Regression to Classification

- Training in this case is a little different. We're not learning to produce a number, we're trying to best separate points.
  - That is, the y values are 0/1 (one for each class) the features are weighted.
  - Find the set of weights that best separates the training examples
  - The simplest answer is to find a hyperplane that minimizes the number of misclassifications
    - In the best case, places one set of points on one side

Break

- Quiz average was 34.

ML Course at Stanford

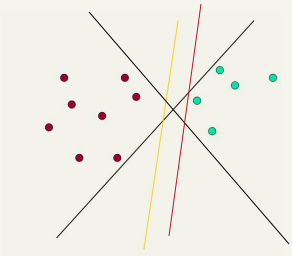
Problems

- There may be an infinite number of such separators. Which one should we choose?
- There may be no separators that can perfectly distinguish the 2 classes. What then?
- What do you do if you have more than 2 classes?

Problem 1: Which Hyperplane?

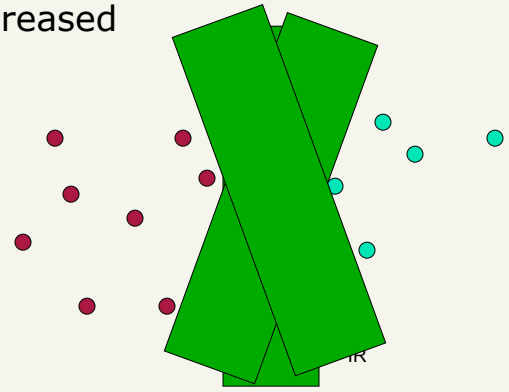
Most methods find a separating hyperplane, but not necessarily an optimal one
- E.g., perceptrons, linear regression

- Support Vector Machines (SVM) find optimal solutions
  - Maximize the distance between the hyperplane and the "difficult points" close to decision boundary
  - One intuition: if there are no points near the decision surface, then there are no very uncertain classification decisions



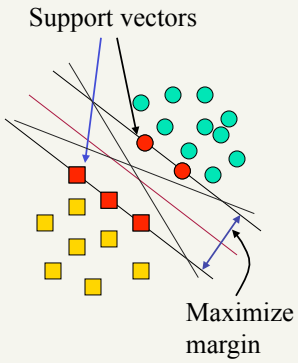
Intuition 1

- If you have to place a fat separator between classes, you have fewer choices, and so the capacity of the model has been decreased



Support Vector Machine (SVM)

- SVMs maximize the *margin* around the separating hyperplane.
  - A.k.a. large margin classifiers
- The decision function is fully specified by a subset of training samples, *the support vectors*.
- *Quadratic programming* problem
- Probably the most effective current text classification method



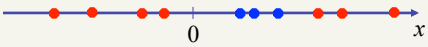
Problem 2: No Clean Separation

- In the case of no clean separation, you could just choose the linear separator with the best margin that minimizes the number of mistakes.
- Or you could find a way to warp the space so that you can find a linear separator in that space

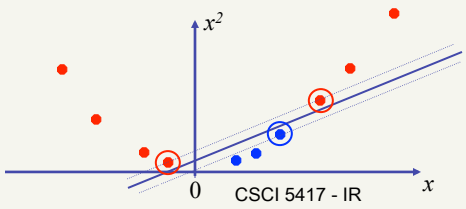
Non-linear SVMs

- Datasets that are linearly separable work out great:

- But what are we going to do if the dataset is just too hard?

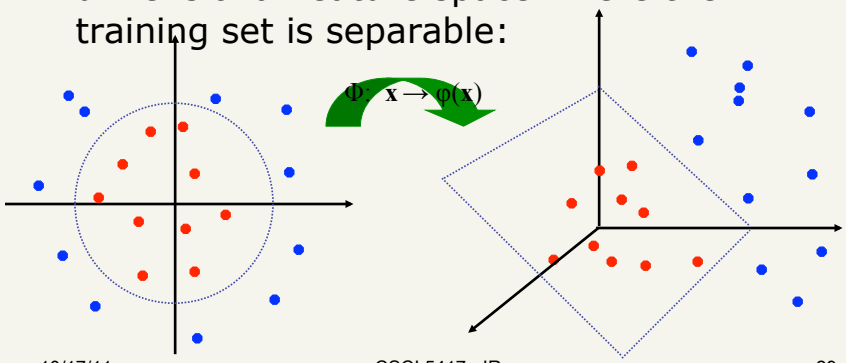


- How about … mapping data to a higher-dimensional space:



Non-linear SVMs: Feature spaces

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



SVMs: Practical Considerations

- Choice of Kernel
- Feature encoding
- Multiclass labeling

SVM: Kernels

- Start simple and move up the chain
  - Linear
  - Polynomial
  - RBF…

SVM: Kernels...

- From the text

Extending SVM algorithms to nonlinear SVMs… standard increases training complexity by a factor of |D| making them impractical… In practice, it can often be cheaper to materialize the higher-order features and train a linear SVM.

In English

- How to deal with phrases like "ethnic cleansing", where the meaning of the phrase is only vaguely a function of the words within it.
  - Use a quadratic kernel
    - Polynomial order 2
  - Or use a linear kernel with bigrams as your features

SVM: Feature Encoding

- Simplest method…
  - Length-normalized TF-IDF vectors.
    - Features are from the vocab
    - Values are real valued
    - Vectors are very sparse

Problem 3: Multiway Classification

- One vs. All
  - For M classes, train M classifiers. Each trained with the positive class against all others.
  - For classification, pass each instance to each classifier. Record the positive responses
    - And…
- All vs All
- $\binom{M}{2}$ Train each class against each other class giving classifiers.
  - For classification, aggregate the responses across the classifiers…
    - And argmax

Evaluation: Classic Reuters Data Set

- Most (over)used data set
- 21578 documents
- 9603 training, 3299 test articles (ModApte split)
- 118 categories
  - An article can be in more than one category
  - Learn 118 binary category distinctions
- Average document: about 90 types, 200 tokens
- Average number of classes assigned
  - 1.24 for docs with at least one category
- Only about 10 out of 118 categories are large

Common categories (#train, #test)

- Earn (2877, 1087)
- Acquisitions (1650, 179)
- Money-fx (538, 179)
- Grain (433, 149)
- Crude (389, 189)
- Trade (369, 119)
- Interest (347, 131)
- Ship (197, 89)
- Wheat (212, 71)
- Corn (182, 56)

Reuters Text Categorization data set (**Reuters-21578)** document

<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="12981" NEWID="798">

<DATE> 2-MAR-1987 16:51:43.42</DATE>

<TOPICS><D>livestock</D><D>hog</D></TOPICS>

<TITLE>AMERICAN PORK CONGRESS KICKS OFF TOMORROW</TITLE>

<DATELINE> CHICAGO, March 2 - </DATELINE><BODY>The American Pork Congress kicks off tomorrow, March 3, in Indianapolis with 160 of the nations pork producers from 44 member states determining industry positions on a number of issues, according to the National Pork Producers Council, NPPC.

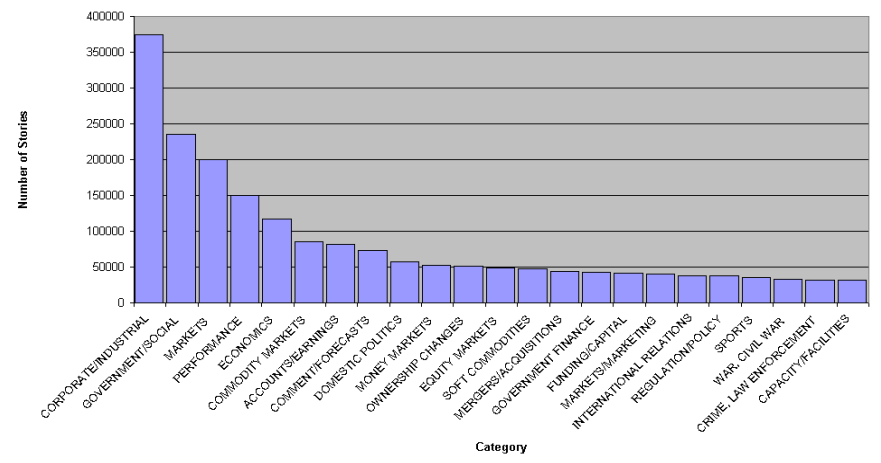
Delegates to the three day Congress will be considering 26 resolutions concerning various issues, including the future direction of farm policy and the tax law as it applies to the agriculture sector. The delegates will also debate whether to endorse concepts of a national PRV (pseudorabies virus) control and eradication program, the NPPC said.

A large trade show, in conjunction with the congress, will feature the latest in technology in all areas of the industry, the NPPC added. Reuter

&#3;</BODY></TEXT></REUTERS>

Newer Reuters: RCV1: 810,000 docs

- Top topics in Reuters RCV1



Per class evaluation measures

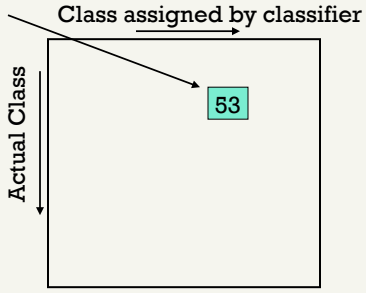
- Recall: Fraction of docs in class *i* classified correctly.
- Precision: Fraction of docs assigned class *i* that are actually about class *i*.
- Accuracy (1- error rate) Fraction of docs classified correctly
  - Useless usually

Dumais et al. 1998: Reuters - Accuracy

	Rocchio	NBayes	Trees	LinearSVM	
earn	92.9%	95.9%	97.8%	98.2%	
acq	64.7%	87.8%	89.7%	92.8%	
money-fx	46.7%	56.6%	66.2%	74.0%	
grain	67.5%	78.8%	85.0%	92.4%	
crude	70.1%	79.5%	85.0%	88.3%	
trade	65.1%	63.9%	72.5%	73.5%	
interest	63.4%	64.9%	67.1%	76.3%	
ship	49.2%	85.4%	74.2%	78.0%	
wheat	68.9%	69.7%	92.5%	89.7%	
corn	48.2%	65.3%	91.8%	91.1%	
Avg Top 10	64.6%	81.5%	88.4%	91.4%	
Avg All Cat	61.7%	75.2%	na	86.4%	

Good practice: Confusion matrix

This (i, j) entry means 53 of the docs actually in class i were put in class j by the classifier.



- In a perfect classification, only the diagonal has non-zero entries

The Real World

P. Jackson and I. Moulinier: *Natural Language Processing for Online Applications*

- "There is no question concerning the commercial value of being able to classify documents automatically by content. There are myriad potential applications of such a capability for corporate Intranets, government departments, and Internet publishers"
- "Understanding the data is one of the keys to successful categorization, yet this is an area in which most categorization tool vendors are extremely weak. Many of the 'one size fits all' tools on the market have not been tested on a wide range of content types."

The Real World

- Gee, I'm building a text classifier for real, now!
- What should I do?

- How much training data do you have?
  - None
  - Very little
  - Quite a lot
  - A huge amount and its growing

Manually written rules

- No training data, adequate editorial staff?
- Never forget the hand-written rules solution!
  - If (wheat or grain) and not (whole or bread) then
    - Categorize as grain
- In practice, rules get a lot bigger than this
  - Can also be phrased using tf or tf.idf weights
- With careful crafting (human tuning on development data) performance is high:
  - *Construe*: 94% recall, 84% precision over 675 categories (Hayes and Weinstein 1990)
- Amount of work required is huge
  - Estimate 2 days per class … plus maintenance

Very little data?

- If you're just doing supervised classification, you should stick to something with high bias
  - There are theoretical results that naïve Bayes should do well in such circumstances (Ng and Jordan 2002 NIPS)
- An interesting theoretical question is to explore semi-supervised training methods:
  - Bootstrapping, EM over unlabeled documents, …
- The practical answer is to get more labeled data as soon as you can
  - How can you insert yourself into a process where humans will be willing to label data for you??

A reasonable amount of data?

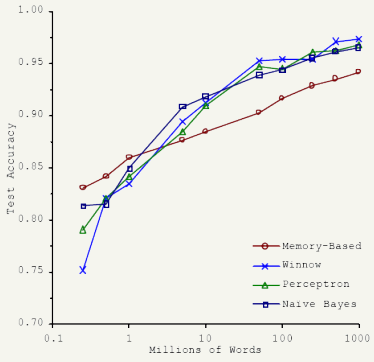
- Perfect, use an SVM
- But if you are using a supervised ML approach, you should probably be prepared with the "hybrid" solution
  - Users like to hack, and management likes to be able to implement quick fixes immediately
  - Hackers like perl

A huge amount of data?

- This is great in theory for doing accurate classification…
- But it could easily mean that expensive methods like SVMs (training time) or kNN (testing time) are quite impractical
- Naïve Bayes can come back into its own again!
  - Or other methods with linear training/test complexity like regularized logistic regression

A huge amount of data?

- With enough data the choice of classifier may not matter much, and the best choice may be unclear
  - Data: Brill and Banko on context-sensitive spelling correction
- But the fact that you have to keep doubling your data to improve performance is a little unpleasant



How many categories?

- A few (well separated ones)?
  - Easy!
- A zillion closely related ones?
  - Think: Yahoo! Directory, Library of Congress classification, legal applications
  - Quickly gets difficult!
    - Classifier combination is always a useful technique
      - Voting, bagging, or boosting multiple classifiers
    - Much literature on hierarchical classification
      - Mileage fairly unclear
    - May need a hybrid automatic/manual solution

How can one tweak performance?

- Aim to exploit any domain-specific useful features that give special meanings or that zone the data
  - E.g., an author byline or mail headers
- Aim to collapse things that would be treated as different but shouldn't be.
  - E.g., part numbers, chemical formulas

Do "hacks" help?

- You bet!
- You can get a lot of value by differentially weighting contributions from different document zones:
  - Upweighting title words helps (Cohen & Singer 1996)
    - Doubling the weighting on the title words is a good rule of thumb
  - Upweighting the first sentence of each paragraph helps (Murata, 1999)
  - Upweighting sentences that contain title words helps (Ko *et al,* 2002)

Measuring Classification Figures of Merit

- Not just accuracy; in the real world, there are economic measures:
  - Your choices are:
    - Do no classification
      - That has a cost (hard to compute)
    - Do it all manually
      - Has an easy to compute cost if doing it like that now
    - Do it all with an automatic classifier
      - Mistakes have a cost
    - Do it with a combination of automatic classification and manual review of uncertain/difficult/"new" cases
  - Commonly the last method is most cost efficient and is adopted

A common problem: Concept Drift

- Categories change over time
- Example: "president of the united states"
  - 1999: clinton is great feature
  - 2002: clinton is bad feature
- One measure of a text classification system is how well it protects against concept drift.
  - Can favor simpler models like Naïve Bayes
- Feature selection: can be bad in protecting against concept drift

Summary

- Support vector machines (SVM)
  - Choose hyperplane based on support vectors
    - Support vector = "critical" point close to decision boundary
  - (Degree-1) SVMs are linear classifiers.
  - Perhaps best performing text classifier
    - But there are other methods that perform about as well as SVM, such as regularized logistic regression (Zhang & Oles 2001)
  - Partly popular due to availability of SVMlight
    - SVMlight is accurate and fast – and free (for research)
    - Also libSVM, tinySVM, Weka…
- Comparative evaluation of methods
- Real world: exploit domain specific structure!