Text Classification

Jason Rennie
jrennie@ai.mit.edu

Text Classification

- Assign text document a label based on content.
- Examples:
  - E-mail filtering
  - Knowedge-base creation
  - E-commerce
  - Question Answering
  - Information Extraction

Example: E-mail Classification

- Filter e-mail into folders set up by user.
- Aids searching for old e-mails
- Can be used to prioritize incoming e-mails
  - High priority to e-mails concerning your Ph.D. thesis
  - Low priority to "FREE Pre-Built Home Business"

Knowedge-Base Creation

- Company web sites provide large amounts of information about products, marketing contact persons, etc.
- Categorization can be used to find companies' web pages and organize them by industrial sector.
- This information can be sold to, e.g. person who wants to market "Flat Fixer" to tire company.

E-Commerce

- Users locate products in two basic ways: search and browsing.
- Browsing is best when user doesn't know exactly what he/she wants.
- Text classification can be used to organize products into a hierarchy according to description.
- EBay: Classification can be used to ensure that product fits category given by user.

Question Answering

- "When did George Washington die?"
- Search document database for short strings with answer.
- Rank candidates
- Many features (question type, proper nouns, noun overlap, verb overlap, etc)
- Problem: learn if string is the answer based on its feature values.

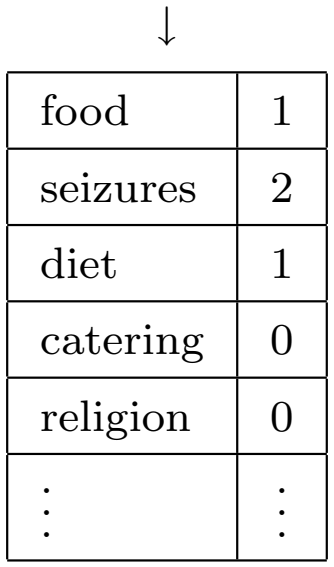
Information Extraction

- Want to extract information from talk announcements (room, time, date, title, speaker, etc)
- Many features may identify the information (keyword, punctuation, capitalization, numeric tokens, etc.)
- Problem: scan over text of document, filling buckets with desired information.
- Freitag (1998) showed that this approach could identify speaker (63%), location (76%), start time (99%) and end time (96%).

Representation

From: dyer@spdcc.com (Steve Dyer)
Subject: Re: food-related seizures?

My comments about the Feingold Diet have no relevance to your daughter's purported FrostedFlakes-related seizures. I can't imagine why you included it.



food	1
seizures	2
diet	1
catering	0
religion	0
⋮	⋮

Representation

- Punctuation is removed, case is ignored, words are separated into tokens. Known as "feature vector" or "bag-of-words" representation.
- Vector length is size of vocabulary. Common vocabulary size is 10,000-100,000. Classification problem is very high dimensional.
- Richer representations (word order, document structure) don't help classification.

Transformation

- Word vector has problems:
  - longer document $\Rightarrow$ larger vector
  - words tend to occur a little or a lot
  - rare words have same weight as common words
- SMART "ltc" transform:
  - $\text{new-tf}_i = \log(\text{tf}_i + 1.0)$
  - $\text{new-wt}_i = \text{new-tf}_i * \log \frac{\text{num-docs}}{\text{num-docs-with-term}}$
  - $\text{norm-wt}_i = \frac{\text{new-wt}_i}{\sqrt{\sum_i \text{new-wt}_i^2}}$

Example Word Vectors

- god 13 15 1 5 1 5 3 4 10 1 2 1 1 1 1 1 2 1 10 1 5 1 3 19 1 1 22 1 1 2 1 3 8 4 1 1 1 1 1 4 1 3 1 1 3 2 2 2 4 2 15 2 7 2 3 6 1 1 1 3 2 3 21 2 1 2 1 1 7
- planet 1 1 1 2 2 80 1 1 1 1 1 1 1
- context 1 5 1 1 3 1 3 3 1 10 1 1 2 1 1
- sleep 16 1 1
- jesus 1 8 2 1 1 1 6 2 2 1 4 4 9 1 9 1 2 20 2 1 5 1 1 19 3 1 1 1 4 8
- understand 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1

Types of Classification Problems

- Binary: label each new document as positive or negative.
  Is this a news article Tommy would want to read?
- Multiclass: give one of m labels to each new document.
  Which customer support group should respond to this e-mail?
- Multilabel: assign zero to m labels to each new document.
  Who are good candidates for reviewing this research paper?
- Ranking: rank categories by relevance.
  Help user annotate documents by suggesting good categories.

Ranking: E-mail categorization

- Problem: Any automated filter will occasionally put e-mail in wrong folder.
- Problem: It's cumbersome to move e-mail from Inbox to storage folders when you have lots of folders.
- Solution: Rank. Sort folders in order of relevance for this e-mail. Correct folder will usually be at top.
- Segal and Kephart (1999)

Why is text different?

- Near independence of features
- High dimensionality (often larger vocabulary than # of examples!)
- Importance of speed

Colocations

- Meaningful word-pairs, like "machine learning" should be useful in classification, shouldn't they?
- If you see "machine learning," is there any doubt that the document is about machine learning?
- But, how do you make that connection?

Colocations: Question

- Vocabulary sizes for reasonable-length documents (say 50+ words) are around 30k.
- This puts the number of possible colocations at 9×10^8.
- **With 10,000 training documents, how do you learn them?**

Colocations: Answer

- You can't![a]
- ...at least, not without some domain knowledge or a peculiar data set.

[a]Remember that we're using document vectors. We're really talking about finding "machine" and "learning" in the same document. But, to take advantage of "machine learning", you need a *much* richer feature set.

Colocations: Explanation

- Rifkin and I (2001) found no improvement in text classification on two data sets using non-linear kernels for the SVM.
- Yang and Liu (1999) made the same findings on Reuters-21578.
- Dumais (1998) found the linear SVM outperforming a Bayes net with pair-wise dependencies.
- But, can't linear classifiers learn some correlation information?

Colocations: Explanation

- Yes. They can. But, consider this experiment:
- Scramble word frequencies between documents (enforce the independent features assumption):

d_1		d_2	
dog	5	dog	2
cat	3	cat	7
head	1	head	4
joke	0	joke	2

$\rightarrow$

d'_1		d'_2	
dog	2	dog	5
cat	3	cat	7
head	4	head	1
joke	0	joke	2

Colocations: Explanation

- Both document sets yield the same accuracy for a linear SVM!
- 20 Newsgroups, 12,000 training documents, 10 random test/train splits:

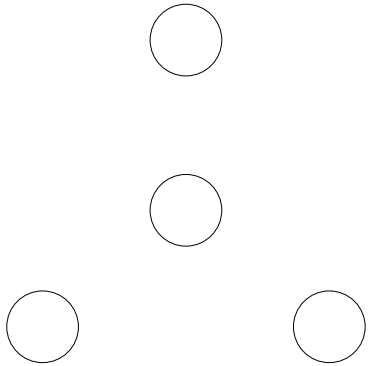
Regular	Scramble
0.1346	0.1346
0.1362	0.1362
0.1361	0.1361
0.1356	0.1356
⋮	⋮
0.1347	0.1347

- Identical!

One-vs-all

- How would you do multiclass classification with a binary classifier?
- Standard approach is one-vs-all.
- Construct 1 vs. $\{2, 3, \ldots, n\}$, 2 vs. $\{1, 3, \ldots, n\}$, ...
- Assign label of most confident classifier.
- Will this work if we're using a linear classifier?

One-vs-All



- Four points in two-dimensional space aren't necessarily separable with linear one-vs-all boundaries.
- In general, $n + 2$ points in n-dimensional space aren't separable with linear one-vs-all boundaries (recall that $n + 1$ points define a hyper-tetrahedron).

One-vs-All

- Rifkin and I (2001) ran text classification experiments and found that one-vs-all worked very well—just slightly worse than the best multiclass technique!
- Couldn't these decision boundaries be difficult to learn?

One-vs-All

20 Newsgroups		Industry Sector	
	Error		Error
OVA	0.131	OVA	0.072
BCH	0.125	BCH	0.067

- We'll talk about BCH later. For now, think of it as the multi-class technique to beat.

High Dimensionality: One-vs-All

- 20 Newsgroups & Industry Sector have vocabulary sizes of 62,061 and 55,197, respectively. # classes: 20 and 105. # training examples: 16,000 and 4900.
- KEY: Linear one-vs-all boundaries are easy to find in high-dimensional space.

More High Dimensionality

- Another interesting property of data living in high dimensional space is that data points are usually lineary independent.
- Take 20 Newsgroups: 20,000 news stories in 62,061 dimensional space.
- These documents span a 19,998-dimensional space (only two documents can be removed as linear constructs of the others!)
- So what?

The SVM

An SVM is trained via the following optimization problem:

$$\hat{w} = \arg\min_{w} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i, \tag{1}$$

with constraints

$$y_i(d_i \cdot w + b) \geq 1 - \xi_i \ \forall i, \tag{2}$$

$$\xi_i \geq 0 \tag{3}$$

where each d_i is a document vector, y_i is the label ($+1$ or -1) for d_i and $\hat{w}$ is the vector of weights that defines the optimal separating hyperplane. This form of the optimization is called the "primal." By incorporating the inequality constraints via Lagrange

multipliers, we arrive at the "dual" form of the problem,

$$\hat{w} = \arg\max_{w} \sum_{i} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (d_i \cdot d_j) \tag{4}$$

subject to

$$0 \leq \alpha_i \leq C \quad \forall i \tag{5}$$

$$\sum_{i} \alpha_i y_i = 0 \tag{6}$$

Given optimized values for the α_i, the optimal separating hyperplane is

$$\hat{w} = \sum_{i} \alpha_i y_i d_i. \tag{7}$$

High Dimensionality: Subsampling

- Question: What happens when you train a linear SVM on 15,000 20 Newsgroups documents?
- Answer: You get *lots* of support vectors—about 5,500.
- Implication: Subsampling is bad for text classification with the SVM.
- This is important because the fastest SVM implementations are (appx.) quadratic in the number of training examples (cn^2) (Joachims 1997)[a]; subsampling is a standard technique for speeding up super-linear ($> cn$) algorithms.

[a]Rif has regularization network classifiers that work much faster and perform about as well, but they can still be slow compared to linear-time algorithms

High Dimensionality: α's

- Subsampling effectively constrains $\alpha_i = 0$ for the removed docs (let α^s be vector of α's in subsampled problem; think about mapping these back to the full set of documents)
- Concatenating documents imposes a different constraint: $\alpha_i = \alpha_j$.
- Concatenating also preserves mean statistics.
- Good, linear-time classifiers, like Naive Bayes and Rocchio are based on class means.

Concatenation

- Concatenate d_{n-1} and d_n to get $D = \{d_1^c, \ldots, d_{n-1}^c\}$.
- $d_i^c = d_i$ for $i \in \{1, \ldots, n-2\}$, $\quad d_{n-1}^c = d_{n-1} + d_n$.
- After training: $\alpha^c = \{\alpha_1^c, \ldots, \alpha_{n-1}^c\}$, $w^c = \sum_{i=1}^{n-1} \alpha_i^c y_i d_i^c$.
- Back to original documents...
  - Let $\alpha_i := \alpha_i^c$ for $i \in \{1, \ldots, n-2\}$ and $\alpha_{n-1} := \alpha_n := \alpha_{n-1}^c$.
  - Then, $w^c = \sum_{i=1}^{n} \alpha_i y_i d_i$.

Naive Bayes in One Slide

- Likelihood: $p(D^c|\theta) = \frac{N^c!}{\prod_k N_k^c!} \prod_k \theta_k^{N_k^c}$.
- Dirichlet prior: $p(\theta) = \mathrm{Dir}(\theta|\{\alpha_k\}) = \frac{\Gamma(\alpha)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k - 1}$.
- Let $\alpha_k = 2\ \forall k$. Estimated parameters are

$$\hat{\theta}_k^c = \arg\max_\theta p(D^c|\theta)p(\theta) = \frac{N_k^c + 1}{N^c + V}. \tag{8}$$

- Decision rule is

$$\hat{H}(d) = \arg\max_c \prod_k \left(\frac{N_k^c + 1}{N^c + V} \right)^{n_k} \tag{9}$$

- $\alpha_k = 2$ gives ficticious count of 1 for every estimate

Rocchio in One Slide

- First, apply a cool transform to the document vectors (like SMART ltc).
- Then, create a "prototype" vector for each class:

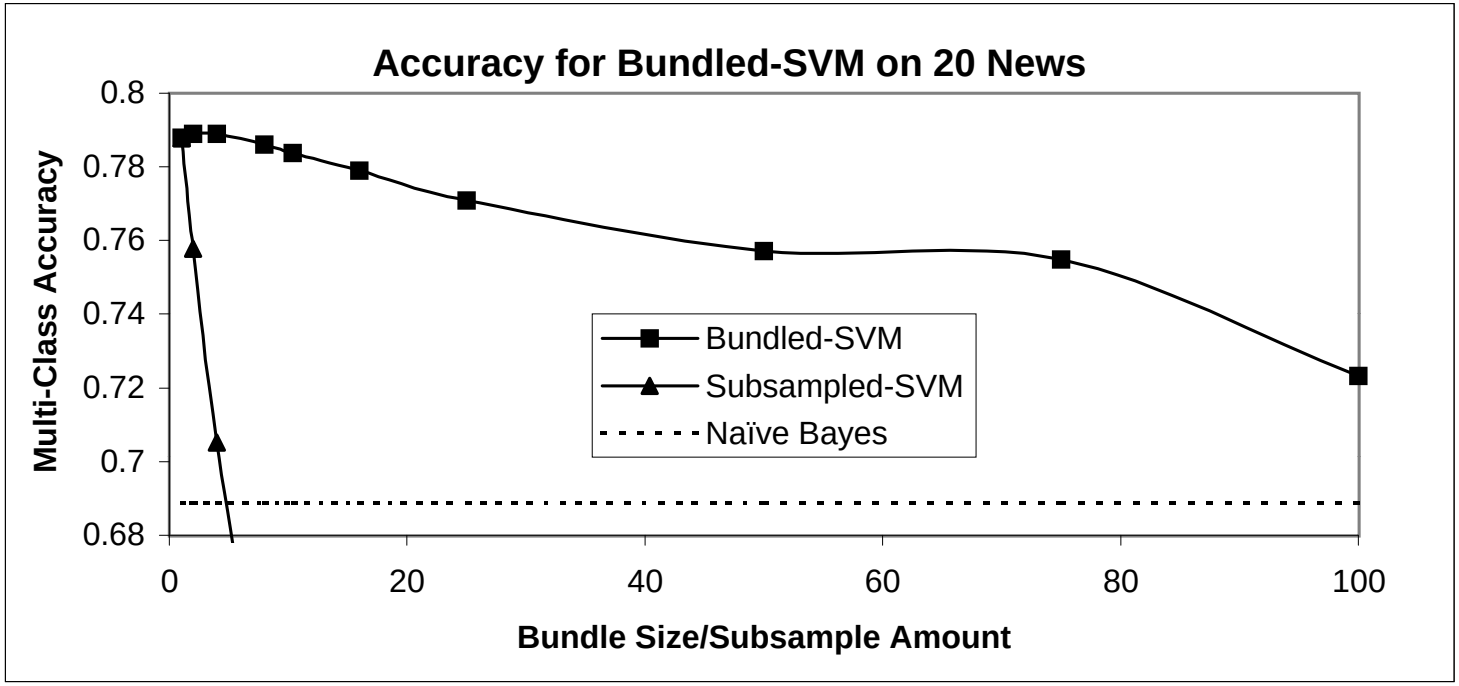
$$c_j = \alpha \frac{1}{|C_j|} \sum_{d \in C_j} \frac{d}{\|d\|} - \beta \frac{1}{|D \setminus C_j|} \sum_{d \in D \setminus C_j} \frac{d}{\|d\|} \qquad (10)$$

- α and β are mysterious parameters. The gods say they should be set to $\alpha = 16$ and $\beta = 4$ (or optimized via cross-validation).
- A new document is assigned the class with the smallest angle between that vector and the prototype vector.

Bundled-SVM

- Concatenate together documents within class to create bundles.
- Apply SMART ltc transform, hand bundles to SVM.
- Size of bundles allows trade-off between speed and accuracy.
- Works much better than subsampling in tests (Shih, Chang, Rennie 2002).

Bundled-SVM


Accuracy for Bundled-SVM on 20 News
Multi-Class Accuracy
0.8
0.78
0.76
0.74
0.72
0.7
0.68
0
20
40
60
80
100
Bundle Size/Subsample Amount
Bundled-SVM
Subsampled-SVM
Naïve Bayes

Bundled-SVM

- Reuters: multi-label problem, 90 topics, # documents/topic range from 1,000 to 1 (many topics with few documents)
- On Reuters-21578, improves speed and accuracy! (micro: .857 $\rightarrow$ .884; macro: .631 $\rightarrow$ .681 for bundle-size=2)
  - Some Reuters documents are super-short.
  - In one-vs-all docs in "one" class often have nearly equal alphas (concatenation may be the right thing anyway).
  - We're still trying to explain this one...

Classifier Evaluation

- How do you evaluate a classifier?
- First, some terminology...

		Guess	
		+1	−1
True	+1	tp	fn
Label	−1	fp	tn

- recall = 1-miss = tp/(tp+fn)
- precision = tp/(tp+fp)
- false alarm = fp/(fp+tn)

Classifier Evaluation

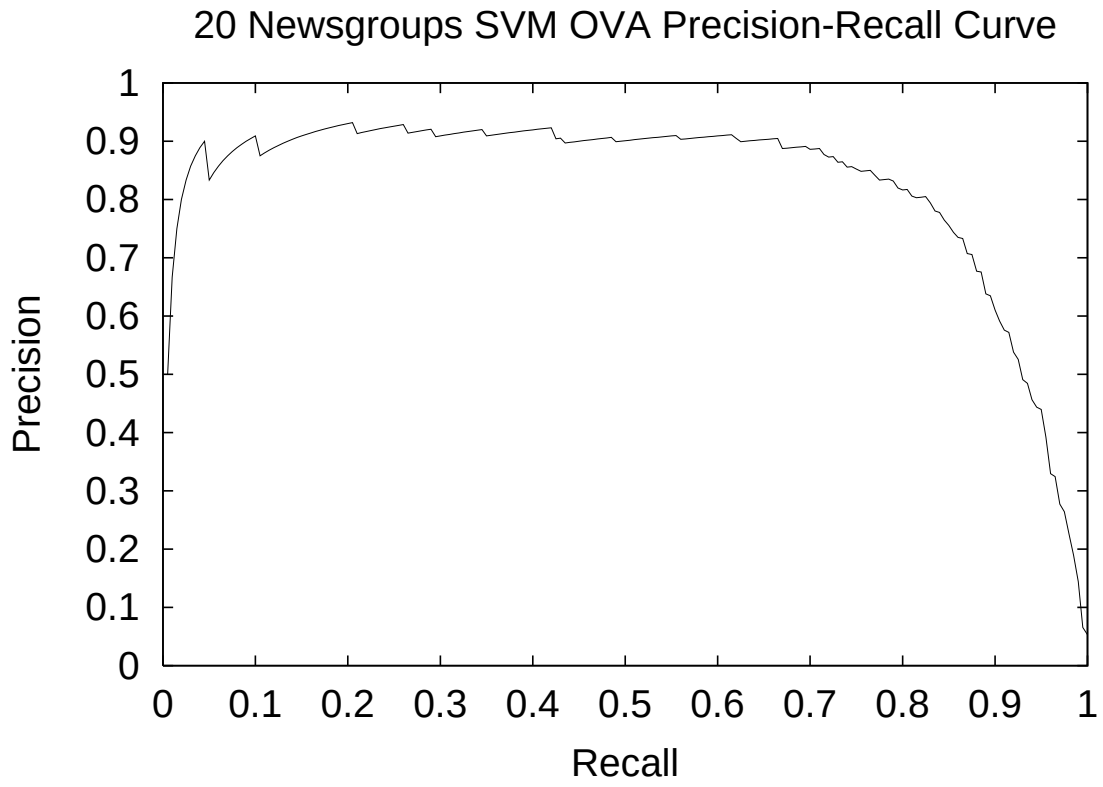
- Error is one way to evaluate, but what if positives are *really* important. Think information retrieval/search engines.
- Say you'd rather see some junk (low precision) than lose an e-mail from your advisor (low recall).
- Also, different classifiers have different objectives (minimize error, minimize hinge loss, maximize likelihood, etc.)
- How do you ensure that you're evaluating them on equal footing?
- Better classifier could have higher error!

Breakeven

- Tradition (for text) is to evaluation classifiers in terms of precision and recall.
- Breakeven is point where precision=recall.
- If one classifier is uniformly better (in p/r), it will have a higher breakeven.
- Macro-breakeven: average breakevens for all topics.
- Micro-breakeven: point where precision=recall with tp, tn, fp summed over all classifiers.

Problems with Breakeven

- P-R curve not convex.
- May need to interpolate to get precision=recall. Not necessarily achievable.



Problems with Breakeven

- P-R breakeven is also not particularly well defined!
- There can be multiple precision=recall points and there's no standard for handling them.
- Also, micro-breakeven is deceptively sensitive to minor changes in the data set...

Problems with Micro-Breakeven

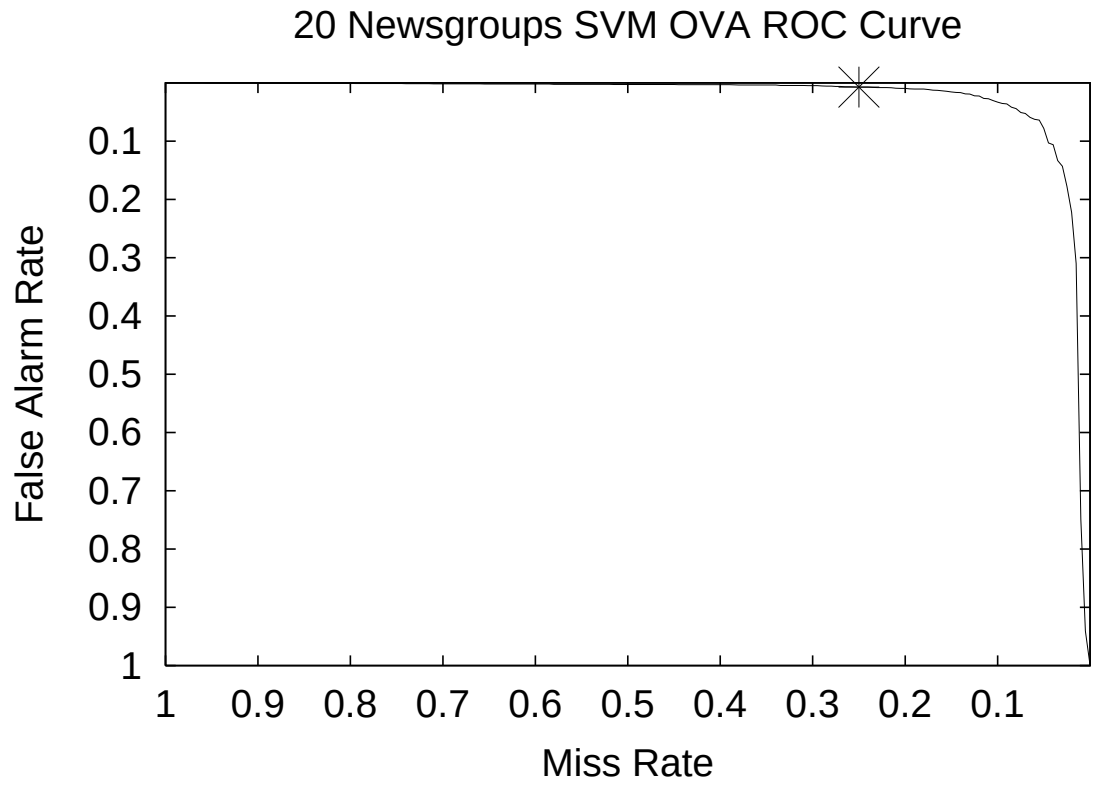
- Reuters-21578 has 90 topics with 1+ training docs, 1+ test docs.
- Reuters-21578 has 95 topics with 2+ training docs.
- Difference: 11 out of 3300 test documents (0.33%)
- One one-vs-all classifier per topic (multi-label problem).
- What will the difference in micro P-R breakeven be? (wait)

Problems with Micro-Breakeven

- 1.5% (86.3 vs. 87.8)
- Why so big a difference?
- Contribution from recall is small (limited by number of training documents.
- Precision can hurt. Lots of negative examples. Each has the possibility of being a false positive.

ROC Breakeven

- ROC Curve is the only true characterization of a classifier:



ROC Breakeven

- ROC Breakeven: point where curve intersections 45° line.
- Or, it's the point where miss=false alarm.
- Better than P-R Breakeven:
  - ROC Breakeven is always achievable (ROC curve is convex).
  - A $(0.8, 0.8)$ breakeven tells you that $(0.9, 0.4)$ and $(0.4, 0.9)$ points are achievable (convexity, $(0, 1)$ & $(1, 0)$).
  - Well-specified: ROC Breakeven is unique and always exists.

Summary

- Text is High Dimensional
- It has nearly independent features
- Speed is important
- Evaluating text classifiers (and classifiers in general) requires care.

Error-Correcting Output Coding

- How do you do multi-class classification with a binary learner?
- One-vs-all works fine, but are there other ways?
- What if we view a binary classifier as a noisy channel. Noise corresponds to classification errors.
- If errors are independent, we can indefinitely reduce errors.

Error-Correcting Output Coding

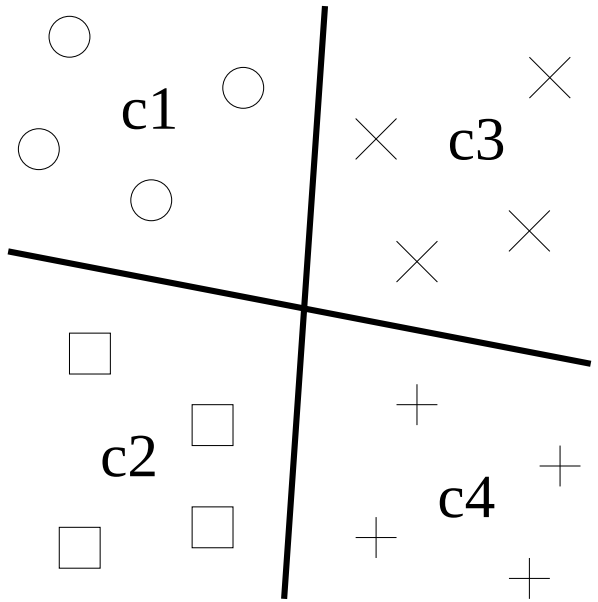
- Matrix specifies the code for each label (row of matrix)
- Each label is identified by unique bit-vector
- Motivation: errors can be corrected using more bits than are needed to partition labels.

$$R = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 \\ +1 & +1 & +1 & -1 & -1 \\ +1 & +1 & -1 & +1 & +1 \\ -1 & -1 & +1 & +1 & +1 \end{bmatrix} \tag{11}$$

Error-Correcting Output Coding

- Train one binary classifier per column of matrix.
- Code for example is output of binary classifiers.
- Classify by matching example with "closest" code (e.g. Hamming distance)

$$R = \begin{bmatrix} -1 & -1 \\ -1 & +1 \\ +1 & -1 \\ +1 & +1 \end{bmatrix}$$

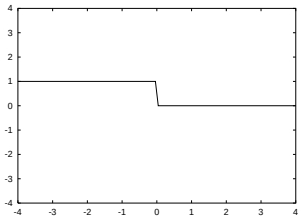


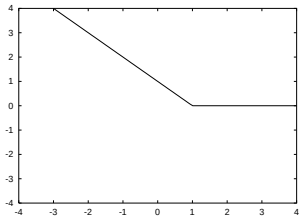
ECOC: The Loss Function

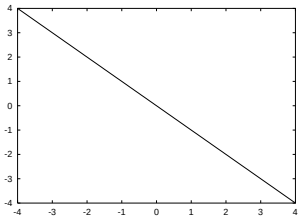
- General form of decision rule is

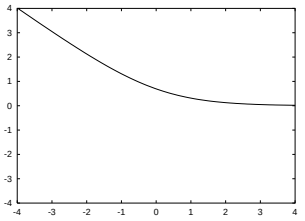
$$\hat{H}(x) = \arg \min_{c \in \{1,\ldots,m\}} \sum_{i=1}^{l} g(f_i(x) R_{ci}) \tag{12}$$

- $f_i(x)$ is the output of the binary classifier; $f_i(x) > 0$ is a $+1$ classification, $f_i(x) < 0$ indicates -1.
- The loss function allows a non-linear transform on the outputs:


Hamming


Hinge (SVM)


Linear


Logistic

ECOC: Example

- Let $R = \begin{bmatrix} -1 & -1 & +1 \\ -1 & +1 & -1 \\ +1 & -1 & -1 \\ +1 & +1 & +1 \end{bmatrix}$, $g(x) = (1-x)_+$ and $\begin{aligned} f_1(x) &= -2 \\ f_2(x) &= -1 \\ f_3(x) &= -5 \end{aligned}$.
-

	$c=1$	$c=2$	$c=3$	$c=4$
$\sum g(f_i(x)R_{ci})$	3	2	6	11

- ECOC selects class 2.

Previous Work: ECOC with Naive Bayes

- Berger used ECOC with Naive Bayes to do multiclass text classification [1999].
- He used random code matrices and a Hamming-like loss function. Minor improvements over Naive Bayes were found (10-20% less error).
- Ghani added BCH codes, used the Hamming loss and ran experiments on a data set with 105 classes and 27-102 documents/class [2000].
- Ghani found 66% less error than Naive Bayes.

Experiments: Questions to be answered

- Will the SVM dominance in binary problems convey to multiclass problems?
- What loss function works best with Naive Bayes and the SVM?
- What types of code matrices yield best performance?

Experiments: Answers

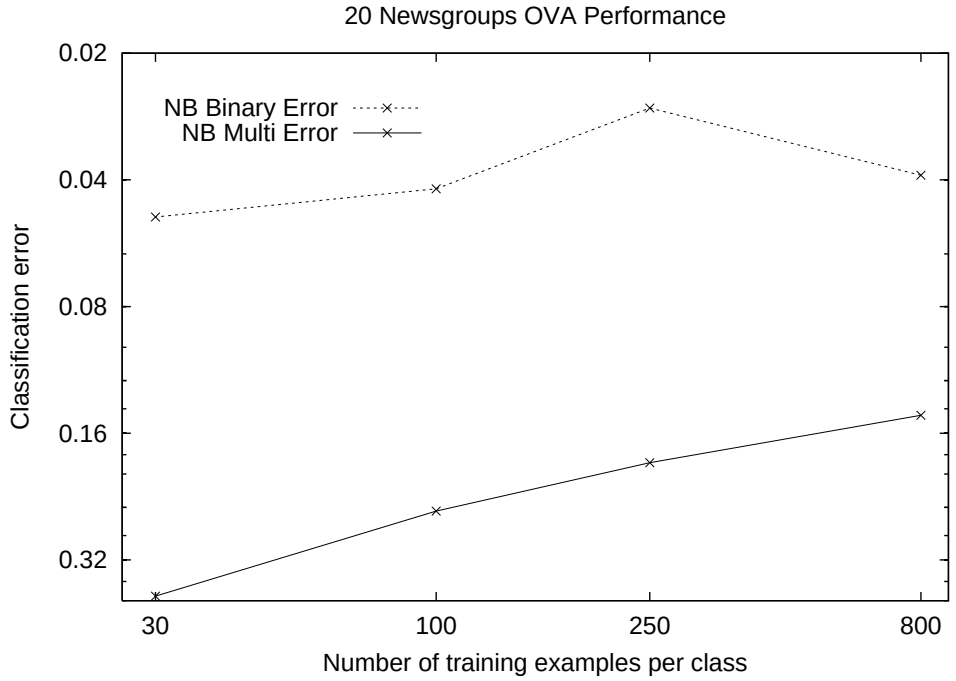
- Better SVM binary performance yields lower multiclass error.
- Hinge, $g(x) = (1 - x)_+$, and Linear, $g(x) = -x$, work best.
- BCH performs best, but OVA is nearly as good.

Multiclass Error

20 Newsgroups	800		250		100	
	SVM	NB	SVM	NB	SVM	NB
OVA	0.131	0.146	0.167	0.199	0.214	0.277
Random 63	0.129	0.154	0.171	0.198	0.222	0.256
BCH 63	0.125	0.145	0.164	0.188	0.213	0.245

Industry Sector	52		20		10	
	SVM	NB	SVM	NB	SVM	NB
OVA	0.072	0.357	0.176	0.568	0.341	0.725
Random 63	0.072	0.135	0.189	0.279	0.363	0.453
BCH 63	0.067	0.128	0.176	0.272	0.343	0.443

Binary Performance: What measure to use?



- Connection between binary error and multiclass classification can be very poor (guessing yields 5% error).

Binary Performance: What measure to use?

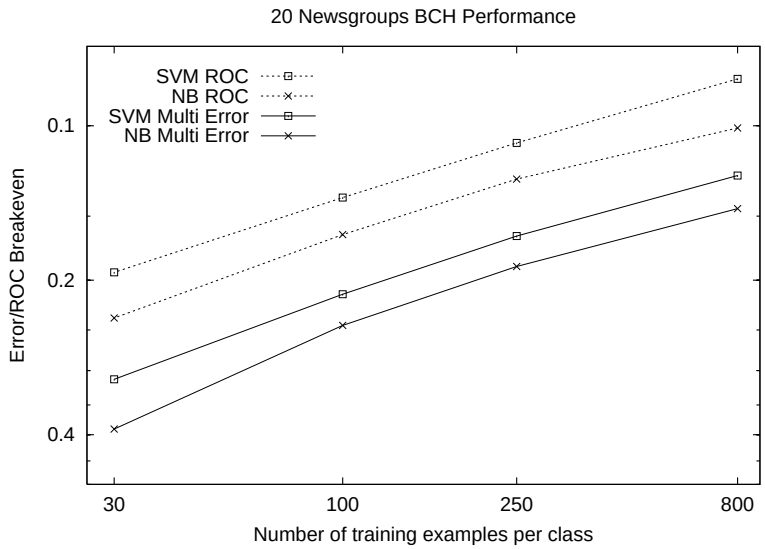
- When example distribution is skewed (as with OVA), binary error mainly judges performance on one class.
- If 95% of examples are class "+1", guessing achieves error of 5%.
- Classifier that guesses provides no information about the label of the example.
- Better characterization of performance evenly weights the classes.

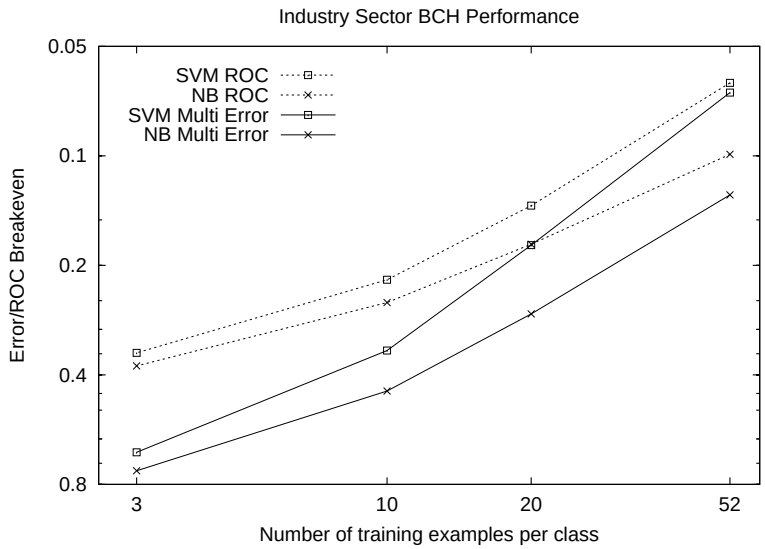
Binary Performance: ROC Breakeven

		Guess	
		+1	−1
True	+1	tp	fn
Label	−1	fp	tn

- New measure of multiclass performance: average of false alarm and miss rates when difference is minimized.
- Let FA $= fp/(tp + tn)$ and miss $= fn/(fn + tp)$.
- ROC breakeven is (FA + miss)/2 when |FA − miss| is minimized.
- Unlike precision-recall breakeven, ROC breakeven is convex combination of points on ROC curve & is always achievable.

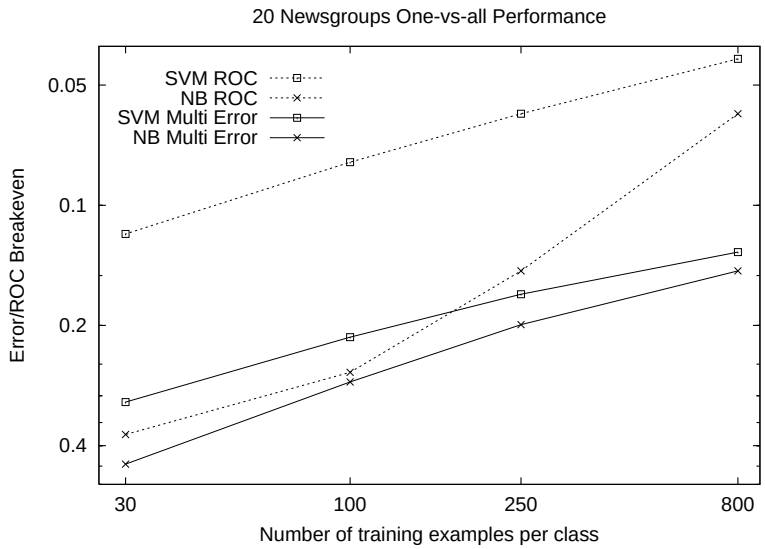
ROC Breakeven follows BCH performance

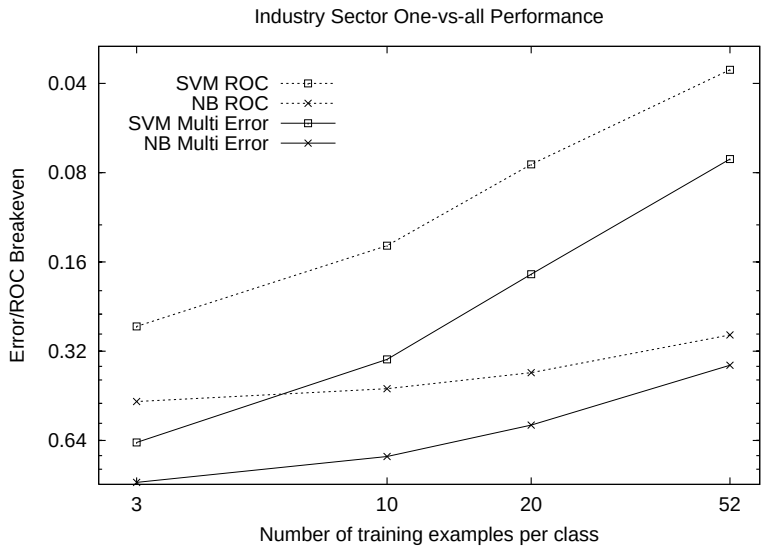




- 20 Newsgroups: consistent gap between ROC breakeven scores leads to similar multiclass error gap.
- Industry Sector: divergent ROC breakeven scores gives large difference in multiclass error.

ROC Breakeven follows OVA performance





- 20 Newsgroups: large ROC breakeven changes result in small multiclass error changes.
- Industry Sector: improved SVM ROC breakeven scores causes much lower multiclass error.

The Loss Function

20 Newsgroups	Hinge	Linear
OVA/SVM	0.131	0.131
OVA/NB	0.146	0.146
BCH 63/SVM	0.125	0.126
BCH 63/NB	0.145	0.144

Industry Sector	Hinge	Linear
OVA/SVM	0.072	0.072
OVA/NB	0.357	0.357
BCH 63/SVM	0.067	0.067
BCH 63/NB	0.128	0.127

- Hinge and Linear loss functions show nearly identical multiclass error.
- Hamming yields worst errors: contributes no confidence info.
- Linear and shifted version of Hinge, $g(x) = (-x)_+$, are equivalent for ECOC.
- Binary loss fn. is not necessarily "correct" ECOC loss fn.

Conclusions

- When used with ECOC, SVM is better multiclass text classifier than NB.
- Improvements over NB are because of better binary performance.
- OVA performs well with confidence information and good binary learner.
- Important aspect of loss function is transmission of confidence information.
- Text classification (bag-of-words) may be linear problem. Linear SVM/Linear loss fn.

Future Work

- Matrix design: Different data sets have different natural partitionings. Can we learn the best matrix for a particular data set? [Crammer and Singer, 2001]
- Boosting: ECOC partitions according to class boundaries. Can we gain additional benefit from weighting examples within classes? [Guruswami and Sahal, 1999] [Schapire, 1997]