Text Classification and Naïve Bayes

Is this spam?

Subject: **Important notice!**
From: Stanford University <newsforum@stanford.edu>
Date: October 28, 2011 12:34:16 PM PDT
To: undisclosed-recipients:;

Greats News!

You can now access the latest news by using the link below to login to Stanford University News Forum.

http://www.123contactform.com/contact-form-StanfordNew1-236335.html

Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information
about the new services.

Positive or negative movie review?

- unbelievably disappointing
- Full of zany characters and richly applied satire, and some great plot twists
- this is the greatest screwball comedy ever filmed
- It was pathetic. The worst part about it was the boxing scenes.

What is the subject of this article?

MeSH Subject Category Hierarchy

MEDLINE Article

Syntactic frame and verb bias in aphasia: Plausibility judgments of undergoer-subject sentences

?

- Antogonists and Inhibitors
- Blood Supply
- Chemistry
- Drug Therapy
- Embryology
- Epidemiology
- …

Text Classification

- Assigning subject categories, topics, or genres
- Spam detection
- Authorship identification
- Age/gender identification
- Language Identification
- Sentiment analysis
- …

Text Classification: definition

- *Input*:
  - a document d
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_k\}$

- *Output*: a predicted class $c \in C$

Classification Methods: Hand-coded rules

- Rules based on combinations of words or other features
  - spam: black-list-address OR (“dollars” AND“ have been selected”)
- Accuracy can be high
  - If rules carefully refined by expert
- But building and maintaining these rules is expensive

Classification Methods: Supervised Machine Learning

- *Input:*
  - a document *d*
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set of *m* hand-labeled documents $(d_1,c_1),\ldots,(d_m,c_m)$
- *Output:*
  - a learned classifier $\gamma:d \rightarrow c$

Generative vs Discriminative classifier

- A probabilistic classifier additionally will tell us the probability of an observation being in the class C.
- Two ways of doing classification.
  - Generative classifiers (Ex. Naive Bayes)
    - build a model of each class.
    - given an observation, they return the class most likely to have generated the observation.
  - Discriminative classifiers (EX. logistic regression)
    - learn what features from the input are most useful to discriminate between the different possible classes.

Classification Methods: Supervised Machine Learning

- Any kind of classifier
  - Naïve Bayes
  - Logistic regression
  - Support-vector machines
  - k-Nearest Neighbors
  - …

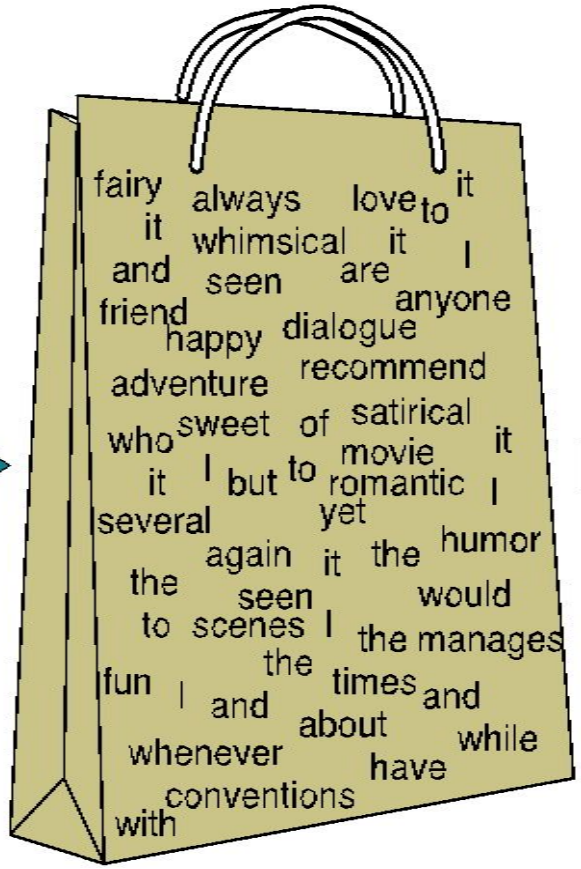
Text Classification and Naïve Bayes

Naïve Bayes Intuition

- Simple ("naïve") classification method based on Bayes rule
- Called multinomial naive Bayes classifier
  - because it is a Bayesian classifier that makes a simplifying (naive) assumption about how the features interact
- Relies on very simple representation of document
  - Bag of words: an unordered set of words with their position ignored, keeping only their frequency in the document

The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

Bayes' Rule Applied to Documents and Classes

- Naive Bayes is a probabilistic classifier, meaning that for a document d, out of all classes c $\in$ C the classifier returns the class c_{MAP} which has the maximum posterior probability for the document.

- For a document *d* and a class *c*

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}$$

Naïve Bayes Classifier (I)

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c)$$

Dropping the denominator

P(d) doesn't change for each class; we are always asking about the most likely class for the same document d, which must have the same probability P(d); hence P(d) is dropped.

Naïve Bayes Classifier (II)

Posterior probability = highest product of two probabilities: the prior probability of the class P(c) and the likelihood of the document P(d|c):

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} P(f_1, f_2, \ldots, f_n \mid c) P(c)$$

Document d represented as features f1,f2,…,fn

the probability of every possible combination of features (for example, every possible set of words and positions)

Multinomial Naïve Bayes Independence Assumptions

Naive Bayes classifiers therefore make two simplifying assumptions.

$$P(f_1, f_2, \ldots, f_n \mid c)$$

- **Bag of Words assumption**: Assume position doesn't matter.
  - Ex. the word "love" has the same effect on classificationwhether it occurs as the 1st, 20th, or last word in the document
- **Conditional Independence**: Assume the feature probabilities $P(x_i|c_j)$ are independent given the class c and can be multiplied.

$$P(f_1, \ldots, f_n \mid c) = P(f_1 \mid c) \bullet P(f_2 \mid c) \bullet P(f_3 \mid c) \bullet \ldots \bullet P(f_n \mid c)$$

Multinomial Naïve Bayes Classifier

The final equation for the class chosen by a naive Bayes classifier is thus:

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}}\, P(f_1, f_2, \square\ , f_n \mid c) P(c)$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}}\, P(c) \prod_{f \in F} P(f \mid c)$$

Multinomial Naive Bayes Classifier

To apply the naive Bayes classifier to text, we need to consider word positions, by simply walking an index through every word position in the document

positions ← all word positions in test document

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{i \in positions} P(w_i \mid c)$$

Naïve Bayes: Learning

Learning the Multinomial Naïve Bayes Model

- How can we learn the probabilities P(c) and P(fi/c)?
  - first consider the maximum likelihood estimate.
  - use the frequencies in the data.
  - For the document prior P(c)
    - we ask what percentage of the documents in our training set are in each class c.
  - Let N_c be the number of documents in our training data with class c and N_{doc} be the total number of documents.

$$\hat{P}(c) = \frac{N_c}{N_{doc}}$$

Learning the Multinomial Naïve Bayes Model

- To learn the probability P(fi/c), we'll assume a feature is just the existence of a word in the document's bag of words
  - P(wi/c), which we compute as the fraction of times the word w_i appears among all words in all documents of topic c.
  - We first concatenate all documents with category c into one big "category c" text.

Learning the Multinomial Naïve Bayes Model

- Then we use the frequency of w_i in this concatenated document to give a maximum likelihood estimate of the probability:

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

  - Here the vocabulary V consists of the union of all the word types in all classes, not just the words in one class c.

Parameter estimation

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

fraction of times word w_i appears among all words in documents of topic c_j

Problem with Maximum Likelihood

What if we have seen no training documents with the word ***fantastic*** and classified in the topic **positive** (***thumbs-up)***?

$$\hat{P}(\text{"fantastic"} \mid \text{positive}) = \frac{count(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} count(w, \text{positive})} = 0$$

- since naive Bayes naively multiplies all the feature likelihoods together.
- Zero probabilities cannot be conditioned away, no matter the other evidence!

Laplace (add-1) smoothing for Naïve Bayes

- The simplest solution is the add-one (Laplace) smoothing

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} (count(w, c) + 1)}$$

$$= \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c) \right) + |V|}$$

unknown words

- What do we do about words that occur in our test data but are not in our vocabulary at all because they did not occur in any training document in any class?
  - standard solution for such unknown words is to ignore such words—remove them from the test document and not include any probability for them at all.

Stop words

- Some systems choose to completely ignore another class of words: stop words, very frequent words like the and a.
  - Sort the vocabulary by frequency in the training set, and defining the top 10–100 vocabulary entries as stop words, or alternatively by using one of the many pre-defined stop word list
- Every instance of stop words are removed from both training and test documents as if they had never occurred.
  - using a stop word list doesn't improve performance,
- Common to make use of the entire vocabulary and not use a stop word list.

Multinomial Naïve Bayes: Algorithms

function TRAIN NAIVE BAYES(D, C) **returns** log $P(c)$ and log $P(w|c)$

for each class $c \in C$ # Calculate $P(c)$ terms
  N_{doc} = number of documents in D
  N_c = number of documents from D in class c
  $logprior[c] \leftarrow \log \frac{N_c}{N_{doc}}$
  $V \leftarrow$ vocabulary of D
  $bigdoc[c] \leftarrow$ **append**(d) **for** d $\in$ D **with** class c
  for each word w in V # Calculate $P(w|c)$ terms
    $count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$
    $loglikelihood[w,c] \leftarrow \log \frac{count(w,c) + 1}{\sum_{w' \text{ in } V} (count(w',c) + 1)}$
return *logprior*, *loglikelihood*, V

function TEST NAIVE BAYES(*testdoc*, *logprior*, *loglikelihood*, C, V) **returns** best c

for each class $c \in C$
  $sum[c] \leftarrow logprior[c]$
  for each position i in *testdoc*
    $word \leftarrow testdoc[i]$
    if $word \in V$
      $sum[c] \leftarrow sum[c] + loglikelihood[word,c]$
return $\text{argmax}_c\ sum[c]$

Multinomial Naïve Bayes: A Worked Example

	Cat	Documents
Training	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable with no fun

The prior P(c) for the two classes is computed via

$$\hat{P}(c) = \frac{N_c}{N_{doc}}$$

Worked Example

$$P(-) = \frac{3}{5} \qquad P(+) = \frac{2}{5}$$

The word with doesn't occur in the test set, so we drop it completely

The likelihoods from the training set for the remaining three words "predictable", "no", and "fun", are as

$$P(w/c) = \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + |V|}$$

$$P(\text{“predictable”}|-) = \frac{1+1}{14+20} \qquad P(\text{“predictable”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“no”}|-) = \frac{1+1}{14+20} \qquad P(\text{“no”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“fun”}|-) = \frac{0+1}{14+20} \qquad P(\text{“fun”}|+) = \frac{1+1}{9+20}$$

Worked Example

For the test sentence S = “predictable with no fun”, after removing the word ‘with’, the chosen class

$$P(-)P(S|-) = \frac{3}{5} \times \frac{2 \times 2 \times 1}{34^3} = 6.1 \times 10^{-5}$$
$$P(+)P(S|+) = \frac{2}{5} \times \frac{1 \times 1 \times 2}{29^3} = 3.2 \times 10^{-5}$$

The model thus predicts the class negative for the test sentence.

Another Example

$$\hat{P}(c) = \frac{N_c}{N}$$

$$\hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

Conditional Probabilities:

P(Chinese|c) = (5+1) / (8+6) = 6/14 = 3/7

P(Tokyo|c) = (0+1) / (8+6) = 1/14

P(Japan|c) = (0+1) / (8+6) = 1/14

P(Chinese|j) = (1+1) / (3+6) = 2/9

P(Tokyo|j) = (1+1) / (3+6) = 2/9

P(Japan|j) = (1+1) / (3+6) = 2/9

Choosing a class:

$P(c|d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 \approx 0.0003$

$P(j|d5) \propto 1/4 * (2/9)^3 * 2/9 * 2/9 \approx 0.0001$

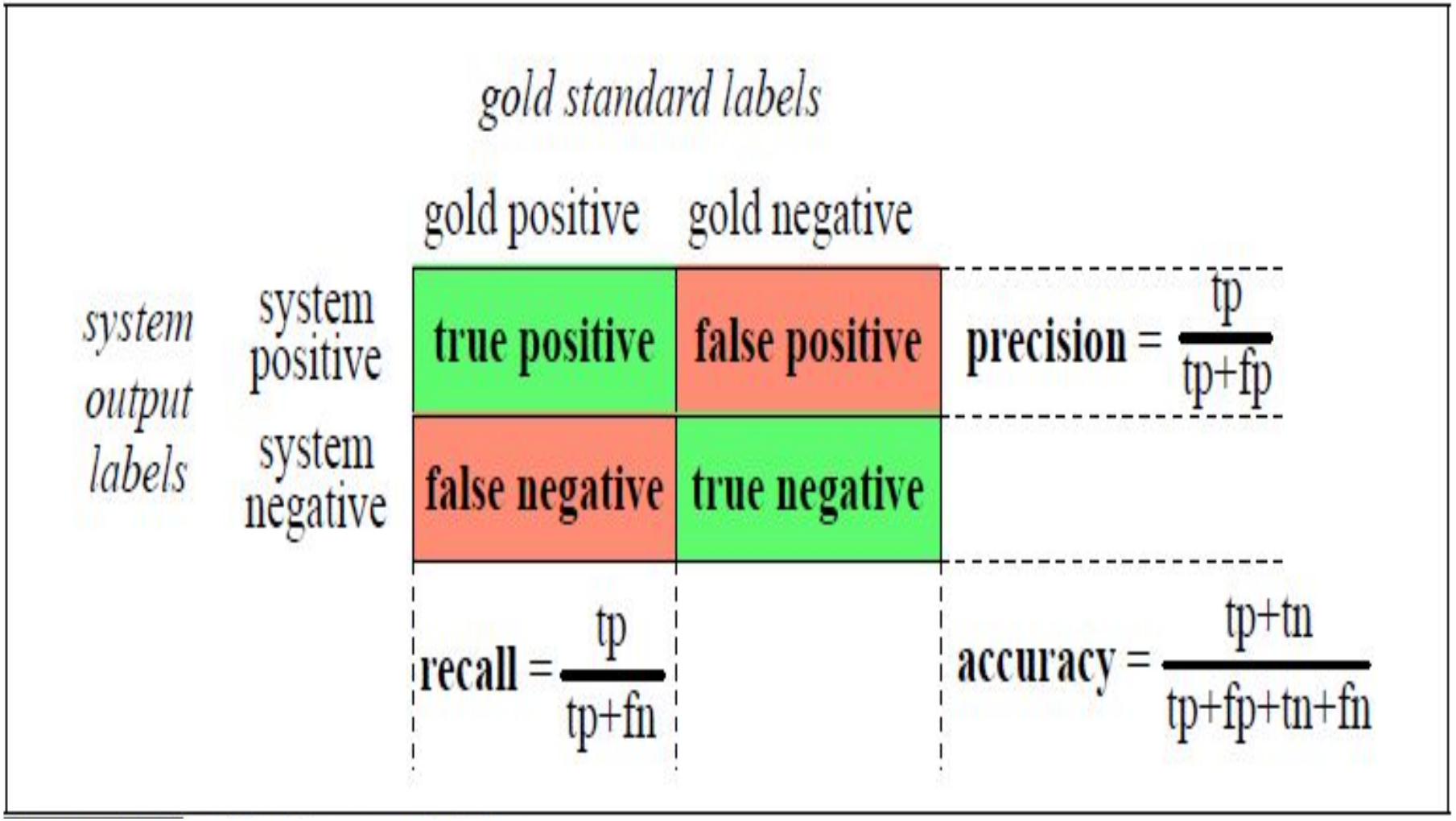
Precision, Recall, and the F measure

TEXT CLASSIFICATION: EVALUATION

Text Classification: Evaluation

- We also need to know whether an email (in spam detection) is actually spam or not, i.e the human-defined labels for each document that we are trying to match.
- We will refer to these human labels as the gold labels.
- we need a metric for knowing how well our spam detector is doing.
- To evaluate any system for detecting things, we start by building a contingency table

Contingency table



The 2-by-2 contingency table

	correct	not correct
selected	tp	fp
not selected	fn	tn

Accuracy

- Accuracy, which asks what percentage of all the observations our system labelled correctly.
  - Although accuracy might seem a natural metric, we generally don't use it.
- Accuracy is not a good metric when the goal is to discover something that is rare,
- or at least not completely balanced in frequency, which is a very common situation in the world.

Precision and recall

- **Precision**: % of selected items that are correct
  Precision P = tp/(tp + fp)
  Recall: % of correct items that are selected
  Recall R = tp/(tp + fn)

	correct	not correct
selected	tp	fp
not selected	fn	tn

- there is typically a trade-off between precision and recall
- to get high precision, be very reluctant to make guesses – but then you may have poor recall
- to get high recall, be very promiscuous in making guesses – but then you may have poor precision

A combined measure: F

- A combined measure (weighted harmonic mean between precision and recall) that assesses the P/R tradeoff is F measure :

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

- The harmonic mean is a very conservative average;
- People usually use balanced F1 measure
  - i.e., with $\beta = 1$ (that is, $\alpha = ½$): $F = 2PR/(P+R)$
- if P and R are far apart, F tends to be near lower value in order to do well on F1, need to do well on BOTH P and R
- can't beat the system by being either too reluctant or too promiscuous

More Than Two Classes

- Dealing with any-of or multivalued classification
  - A document can belong to 0, 1, or >1 classes.
- For each class $c \in C$
  - Build a classifier γ_c to distinguish c from all other classes $c' \in C$
- Given test doc d,
  - Evaluate it for membership in each class using each γ_c
  - d belongs to any class for which γ_c returns true

More Than Two Classes: Sets of binary classifiers

- One-of or multinomial classification
  - Classes are mutually exclusive: each document in exactly one class
- For each class $c \in C$
  - Build a classifier γ_c to distinguish c from all other classes $c' \in C$
- Given test doc d,
  - Evaluate it for membership in each class using each γ_c
  - d belongs to the one class with maximum score

Confusion matrix

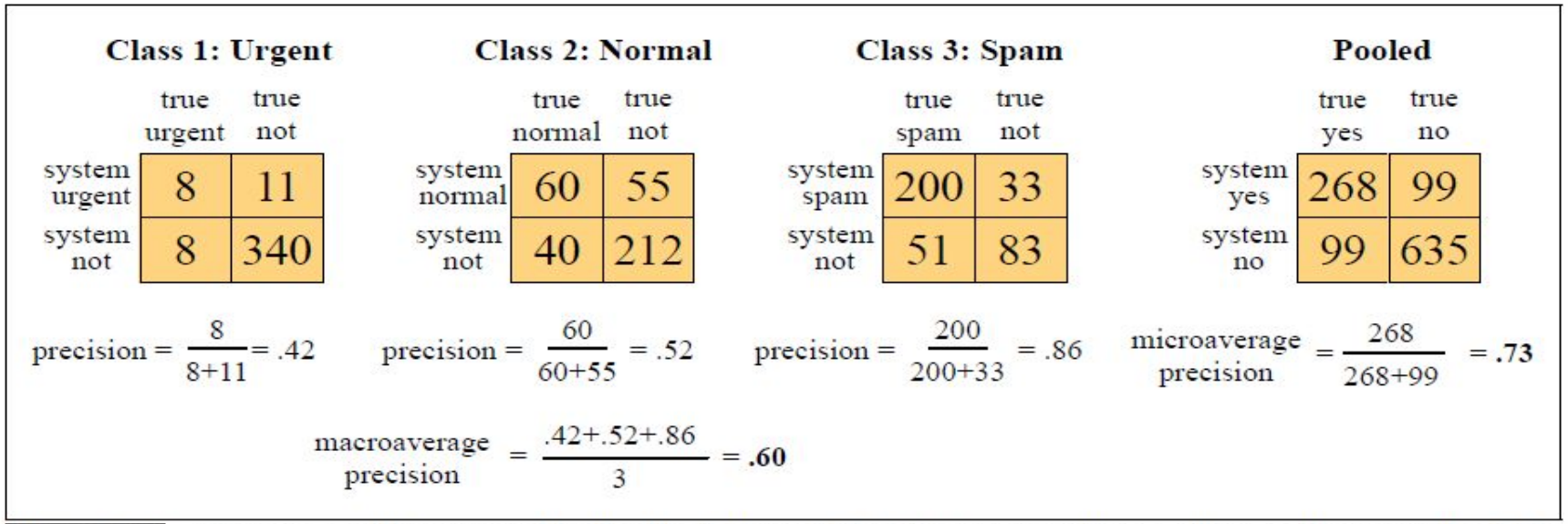
For each pair of classes <c_1,c_2> how many documents from c_1 were incorrectly assigned to c_2?

system output \ *gold labels*	urgent	normal	spam	
urgent	8	10	1	$\text{precision}_u = \frac{8}{8+10+1}$
normal	5	60	50	$\text{precision}_n = \frac{60}{5+60+50}$
spam	3	30	200	$\text{precision}_s = \frac{200}{3+30+200}$
	$\text{recall}_u = \frac{8}{8+5+3}$	$\text{recall}_n = \frac{60}{10+60+30}$	$\text{recall}_s = \frac{200}{1+50+200}$	

Micro- vs. Macro-Averaging

- In order to derive a single metric that tells us how well the system is doing, we can combine these values in two ways.
- **Macroaveraging**: Compute performance for each class, then average.
- **Microaveraging**: Collect decisions for all classes, compute contingency table, evaluate.

Micro- vs. Macro-Averaging



Microaverage is dominated by the more frequent class

Macroaverage better reflects the statistics of the smaller classes, and so is more appropriate when performance on all the classes is equally important

THANK YOU