Classification in NLP

- **Text categorization:-** the task of classifying an entire text by assigning it a categorization label drawn from some set of labels.
- **Sentiment analysis:-** the extraction of sentiment, the positive or negative orientation that a writer expresses toward some object.
  - A review of a movie, book, or product on the web expresses the author’s sentiment toward the product, while an editorial or political text expresses sentiment toward a candidate or political action.
  - Automatically extracting consumer sentiment is important for marketing of any sort of product.
  - Measuring public sentiment is important for politics and also for market prediction.

- **Spam Detection :- The binary classification** task of assigning an email to one of the two classes spam or not-spam. Many lexical and other features can be used to perform this classification.
  - For example , an email containing phrases like “online pharmaceutical” or “WITHOUT ANY COST” or “Dear Winner”.

- **Authorship Attribution:-** Determining authorship a text’s author, authorship attribution, and author characteristics like gender, age, attribution and native language are text classification tasks that are relevant to the digital humanities, social sciences, and forensics as well as natural language processing.

- **Language modeling can be viewed as classification,** each word can be thought of as a class, and so predicting the next word is classifying the context-so-far into a class for each next word.

- **Goal of Classification** :- To take a single observation, extract some useful features, and thereby classify the observation into one of a set of discrete classes.

- **Supervised Learning**
- Formally, the task of classification is to take an input x and a fixed set of output classes Y = y1, y2,.....ym and return a predicted class y ε Y.

- For text classification, we'll sometimes talk about c (for "class") instead of y variable, and d (for "document") instead of x .
- In the supervised situation we have a training set of N documents that have each been hand-labeled with a class: (d1,c1)...... (dN,cN).
- Our goal is to learn a classifier that is capable of mapping from a new document d to its correct class c ε C.

- **Types of Classification:-**

- **Generative classifiers** like naive Bayes build a model of each class. Given an observation, they return the class most likely to have generated the observation.

- **Discriminative classifiers** like logistic regression instead learn what features from the input are most useful to discriminate between the different possible classes.

- **While discriminative systems are often more accurate and hence more commonly used, generative classifiers still have a role.**

Naive Bayes Classifiers

- Bag-of-Words :- A text document as if it were a **bag-of-words**, that is, an unordered set of words with their position ignored, keeping only their frequency in the document.

Naive Bayes :- A probabilistic classifier.

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

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

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \overbrace{P(d|c)}^{\text{likelihood}} \overbrace{P(c)}^{\text{prior}}$$

Without loss of generalization, we can represent a document d as a set of features $f_1, f_2, \ldots, f_n$:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \overbrace{P(f_1, f_2, \ldots, f_n | c)}^{\text{likelihood}} \overbrace{P(c)}^{\text{prior}} \tag{6.6}$$

- The features f1, f2,...... ; fn only encode word identity and not position.
- Bag-of-Word Assumption.

$$P(f_1, f_2, \ldots, f_n | c) = P(f_1|c) \cdot P(f_2|c) \cdot \ldots \cdot P(f_n|c)$$

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

$$c_{NB} = \operatorname*{argmax}_{c \in C} P(c) \prod_{i \in positions} P(w_i|c)$$

$$c_{NB} = \operatorname*{argmax}_{c \in C} \log P(c) + \sum_{i \in positions} \log P(w_i|c)$$

Training the Naive Bayes Classifier

- Learn the probabilities P(c) and P(f_i|c).
- Use the frequencies in the data.
- Document prior P(c) :- The percentage of the documents in our training set are in each class c.

- Let $\boldsymbol{N_c}$ be the number of documents in our training data with class $\boldsymbol{c}$ and $\boldsymbol{N_{doc}}$ be the total number of documents.

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

- A feature is just the existence of a word in the document's bag of words, and so ***P(w_i|c)***, which we compute as the fraction of times the word ***w_i*** appears among all words in all documents of topic **c**.

- First concatenate all documents with category **c** into one big "***category c***" text.

The frequency of ***w_i*** in this concatenated document to give a maximum likelihood estimate of the probability:

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

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

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

Laplace Smoothing (Add One)

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

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

Sentiment Analysis domain with the two classes positive (+) and negative (-).

The model predicts the class Negative for the test sentence.

Optimizing for Sentiment Analysis

- Whether a word occurs or not seems to matter more than its frequency. Thus, it often improves performance to clip the word counts in each document at 1.
- This variant is called **binary multinominal naive Bayes or binary NB.** For each document we remove all duplicate words before concatenating them into the single big document.

Four original documents:

− it was pathetic the worst part was the boxing scenes
− no plot twists or great scenes
+ and satire and great plot twists
+ great scenes great film

After per-document binarization:

− it was pathetic the worst part boxing scenes
− no plot twists or great scenes
+ and satire great plot twists
+ great scenes film

	NB Counts		Binary Counts	
	+	−	+	−
and	2	0	1	0
boxing	0	1	0	1
film	1	0	1	0
great	3	1	2	1
it	0	1	0	1
no	0	1	0	1
or	0	1	0	1
part	0	1	0	1
pathetic	0	1	0	1
plot	1	1	1	1
satire	1	0	1	0
scenes	1	2	1	2
the	0	2	0	1
twists	1	1	1	1
was	0	2	0	1
worst	0	1	0	1

- The difference between **I really like this movie** and **I didn’t like this movie.**
- During text normalization to prepend the prefix **NOT** to every word after a token of logical negation (**n’t, not, no, never**) until the next punctuation mark.
- **(didn't like this movie , but I)**
- **didn't NOT_like NOT_this NOT_movie , but I**
- **NOT_like, NOT_recommend will thus occur more often in negative document and act as cues for negative sentiment, while words like NOT_bored, NOT_dismiss will acquire positive associations.**

Evaluation: Precision, Recall, F-measure

- **Contingency Table.**

- Human defined labels as the gold labels.
- Each cell labels a table set of possible outcomes.

- In the spam detection case, for example, true positives are documents that are indeed spam (indicated by human-created gold labels) and our system said they were spam.

- False negatives are documents that are indeed spam but our system labeled as non-spam.

gold standard labels

		gold positive	gold negative	
system output labels	system positive	**true positive**	**false positive**	**precision** = $\frac{tp}{tp+fp}$
	system negative	**false negative**	**true negative**	
		recall = $\frac{tp}{tp+fn}$		**accuracy** = $\frac{tp+tn}{tp+fp+tn+fn}$

- **Precision** :-Measures the percentage of the items that the system detected (i.e., the system labeled as positive) that are in fact positive (i.e., are positive according to the human gold labels).

- **Recall**:- Measures the percentage of items actually present in the input that were correctly identified by the system.

F-Measure

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

- The β parameter differentially weights the importance of recall and precision, based perhaps on the needs of an application.
- Values of $\beta > 1$ favor recall, while values of $\beta < 1$ favor precision. When $\beta = 1$, precision and recall are equally balanced.
- This is the most frequently used metric, and is called F_β=1 or just F1.

Multi-Label and Multinominal- “More than one Class”.

- Multi-label (any-of) classification, each document or item can be assigned more than one label.
- We can solve any-of classification by building separate binary classifiers for each class c, trained on positive examples labeled c and negative examples not labeled c.
- Given a test document or item d, then each classifier makes their decision independently, and we may assign multiple labels to d.

- **One-of or multinomial classification**,multinomial classification in which the classes are mutually exclusive and each document or item appears in exactly one class.

- Build a separate binary classifier trained on positive examples from c and negative examples from all other classes.

- Now given a test document or item d, we run all the classifiers and choose the label from the classifier with the highest score.

gold labels

system output		urgent	normal	spam	
	urgent	8	10	1	$\textbf{precision}_{u} = \frac{8}{8+10+1}$
	normal	5	60	50	$\textbf{precision}_{n} = \frac{60}{5+60+50}$
	spam	3	30	200	$\textbf{precision}_{s} = \frac{200}{3+30+200}$
		$\textbf{recall}_{u} = \frac{8}{8+5+3}$	$\textbf{recall}_{n} = \frac{60}{10+60+30}$	$\textbf{recall}_{s} = \frac{200}{1+50+200}$	

Class 1: Urgent

	true urgent	true not
system urgent	8	11
system not	8	340

$$\text{precision} = \frac{8}{8+11} = .42$$

Class 2: Normal

	true normal	true not
system normal	60	55
system not	40	212

$$\text{precision} = \frac{60}{60+55} = .52$$

Class 3: Spam

	true spam	true not
system spam	200	33
system not	51	83

$$\text{precision} = \frac{200}{200+33} = .86$$

Pooled

	true yes	true no
system yes	268	99
system no	99	635

$$\text{microaverage precision} = \frac{268}{268+99} = \mathbf{.73}$$

$$\text{macroaverage precision} = \frac{.42+.52+.86}{3} = \mathbf{.60}$$