Great Place To Work. Certified MAR 2021-FEB 2022 INDIA

UPES

UNIVERSITY OF TOMORROW

nirf RANKING 2021

Ranked amongst **top 100** universities in **India**

NAAC A

Accredited **Grade** 'A' by NAAC

QS STARS RATING SYSTEM 2020

QS 5 Star Rating for Academic Development, Employability, Facilities and Program Strength

E-LEAD

Perfect score of **150/150** as a testament to exceptional E-Learning methods

University of the Year (North India) awarded by ASSOCHAM

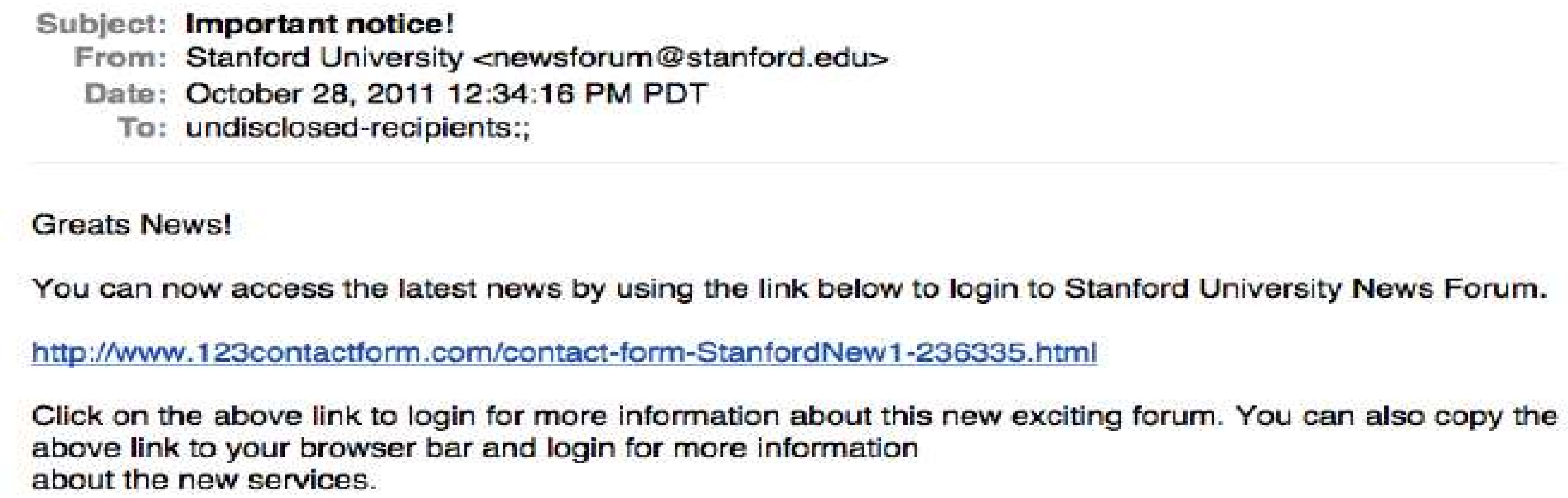
SAFE GUARD

Certified for **safety and hygiene** by Bureau Veritas

Text Classification

The Task of Text Classification

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

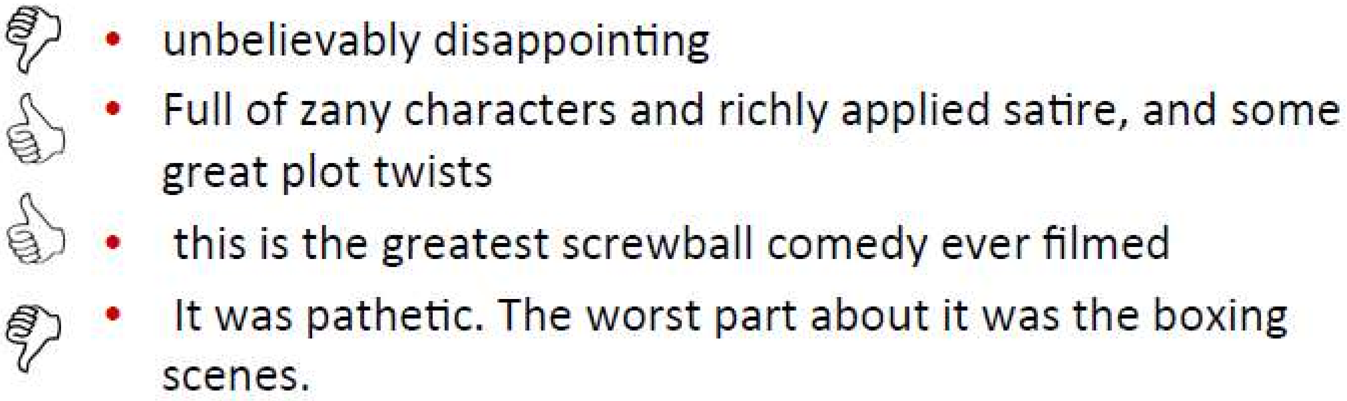
Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information
about the new services.

Authorship attributions: the task of identifying the author of a given document. (Bayesian methods)

Author gender identification from text: number of pronouns (F), noun phases, determiners (M))

Text Classification

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?

✓ automatically extract meaning from text by identifying recurrent themes or topics

Text Classification

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

Text Classification

- **Definition**

Input:

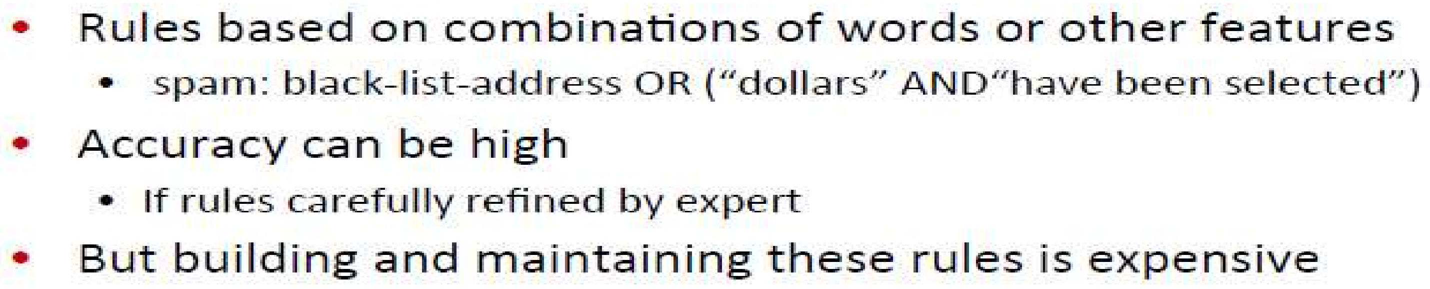
- a document d
- a fixed set of classes C = {c_1, c_2,…, c_J}

***Output*:**

A predicted class $c \in C$

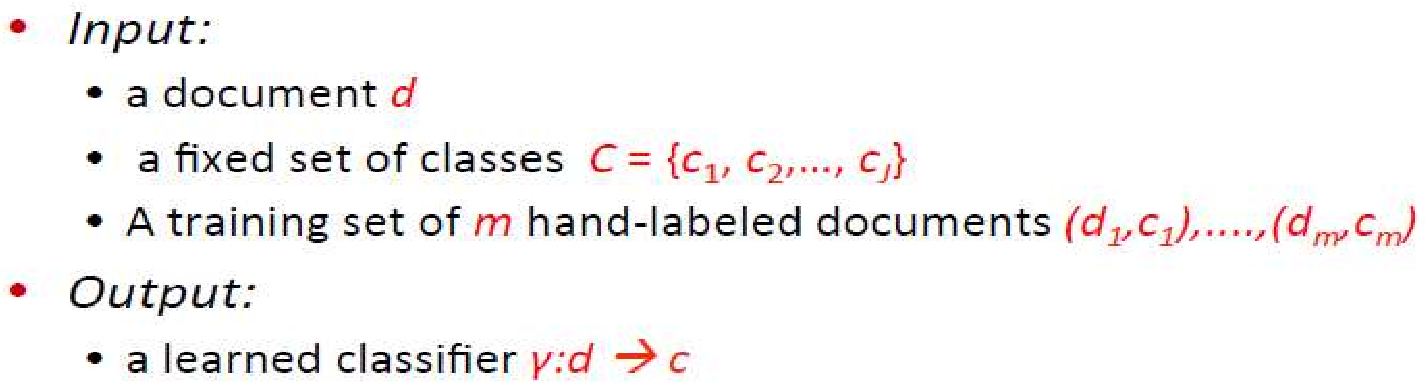
Text Classification

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

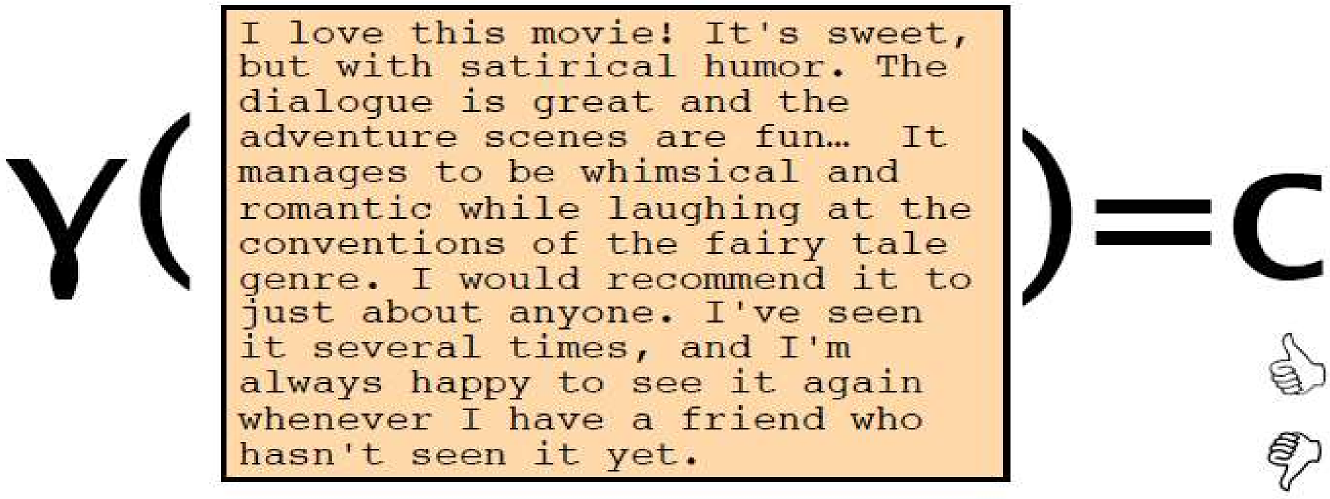


- *Input:*
  - a document d
  - a fixed set of classes $C = \{c_1, c_2,\ldots, c_J\}$
  - A training set of m hand-labeled documents $(d_1,c_1),\ldots.,(d_m,c_m)$
- *Output:*
  - a learned classifier $\gamma{:}d \rightarrow c$

Text Classification and Naïve Bayes

- Simple (“naïve”) classification method based on Bayes rule
- Relies on very simple representation of document
  - Bag of words (A bag of words is a representation of text that describes the occurrence of words within a document.)

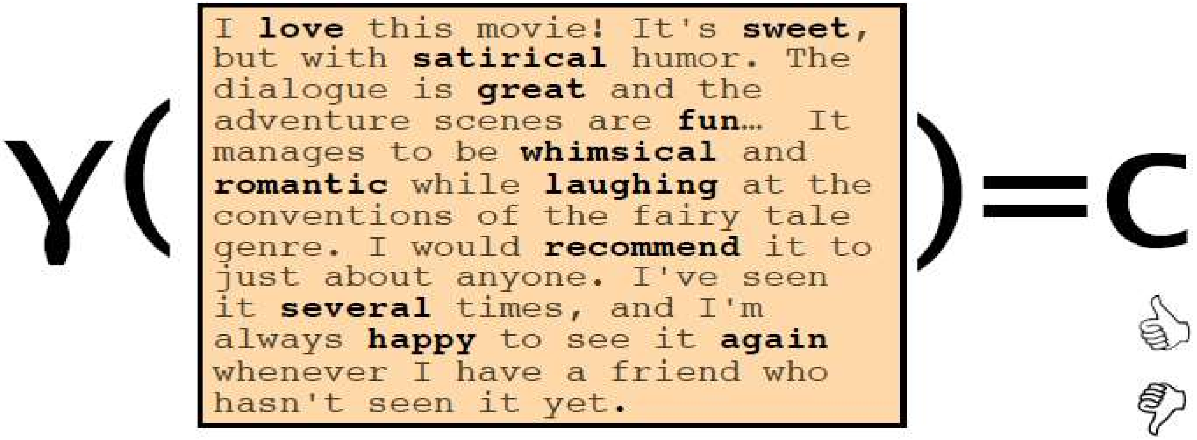
The bag of words representation



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.

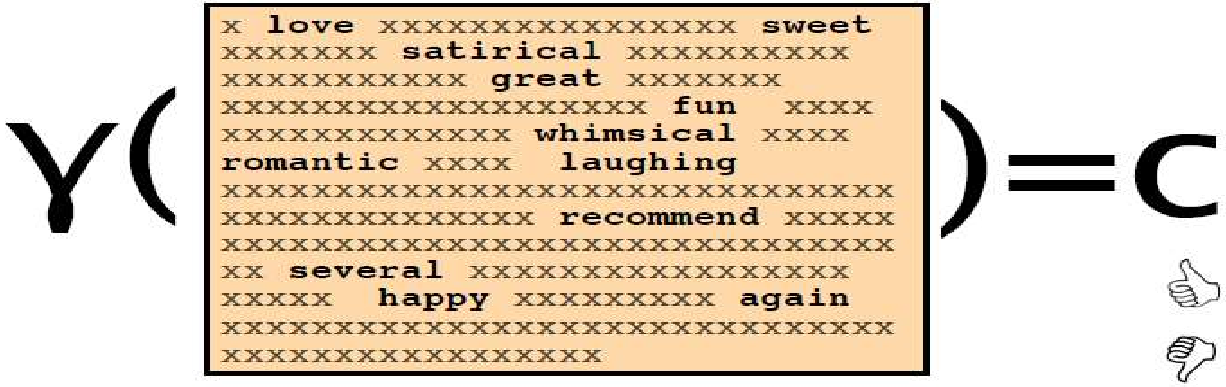
Text Classification and Naïve Bayes

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.

The bag of words representation: using a subset of words



x **love** xxxxxxxxxxxxxxxxx **sweet**
xxxxxxx **satirical** xxxxxxxxxx
xxxxxxxxxxx **great** xxxxxxx
xxxxxxxxxxxxxxxxxxx **fun** xxxx
xxxxxxxxxxxxx **whimsical** xxxx
romantic xxxx **laughing**
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxx **recommend** xxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx **several** xxxxxxxxxxxxxxxxx
xxxxx **happy** xxxxxxxxx **again**
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxx

Text Classification and Naïve Bayes

Understanding Bag of Words with an example

Example(1) **without preprocessing:**

- Sentence 1: ”Welcome to Great Learning, Now start learning”
- Sentence 2: “Learning is a good practice”

Sentence 1	**Sentence 2**
Welcome	Learning
to	is
Great	a
Learning	good
,	practice
Now	
start	
learning	

Text Classification and Naïve Bayes

Understanding Bag of Words with an example

Go through all the words in the above text and make a list of all of the words in our model vocabulary.

- Welcome
- To
- Great
- **Learning**
- ,
- Now
- start
- **learning**
- is
- a
- good
- practice

Understanding Bag of Words with an example

The scoring of sentence 1 would look as follows:

Word	Frequency
Welcome	1
to	1
Great	1
Learning	1
,	1
Now	1
start	1
learning	1
is	0
a	0
good	0
practice	0

Writing the above frequencies in the vector

Sentence 1 → **[1,1,1,1,1,1,1,1,0,0,0]**

Understanding Bag of Words with an example

Now for sentence 2, the scoring would like

Word	Frequency
Welcome	0
to	0
Great	0
Learning	1
,	0
Now	0
start	0
learning	0
is	1
a	1
good	1
practice	1

Similarly, writing the above frequencies in the vector form
Sentence 2 → **[0,0,0,1,0,0,0,1,1,1,1,1]**

Sentence	Welcome	to	Great	Learning	,	Now	start	learning	is	a	good	practice
Sentence1	1	1	1	1	1	1	1	1	0	0	0	0
Sentence2	0	0	0	1	0	0	0	1	1	1	1	1

Understanding Bag of Words with an example

Example(2) **with preprocessing**:

Sentence 1: ”Welcome to Great Learning, Now start learning”

Sentence 2: “Learning is a good practice”

Step 1: Convert the above sentences in lower case as the case of the word does not hold any information.

Step 2: Remove special characters and stopwords from the text. Stopwords are the words that do not contain much information about text like ‘is’, ‘a’,’the and many more’.

After applying the above steps, the sentences are changed to

Sentence 1: ”welcome great learning now start learning”

Sentence 2: “learning good practice”

Understanding Bag of Words with an example

Go through all the words in the above text and make a list of all of the words in our model vocabulary.

- welcome
- great
- learning
- now
- start
- good
- practice

For sentence 1, the count of words is as follow:

Word	Frequency
welcome	1
great	1
learning	2
now	1
start	1
good	0
practice	0

Writing the above frequencies in the vector

Sentence 1 → **[1,1,2,1,1,0,0]**

Understanding Bag of Words with an example

Now for sentence 2, the scoring would be like

Word	Frequency
welcome	0
great	0
learning	1
now	0
start	0
good	1
practice	1

Writing the above frequencies in the vector form:
Sentence 2 → **[0,0,1,0,0,1,1]**

Sentence	welcome	great	learning	now	start	good	practice
Sentence1	1	1	2	1	1	0	0
Sentence2	0	0	1	0	0	1	1

How N-Grams can help?

Sentence 1: “This is a good job. I will not miss it for anything”

Sentence 2: ”This is not good at all”

For this example, let us take the vocabulary of 5 words only. The five words being-

- good
- job
- miss
- not
- all

So, the respective vectors for these sentences are:

“This is a good job. I will not miss it for anything”=**[1,1,1,1,0]**

”This is not good at all”=**[1,0,0,1,1]**

How N-Grams can help?

Sentence 2 is a negative sentence and sentence 1 is a positive sentence. Does this reflect in any way in the vectors above?

An N-gram is an N-token sequence of words: a 2-gram (more commonly called a bigram) is a two-word sequence of words like “really good”, “not good”, or “your homework”, and a 3-gram (more commonly called a trigram) is a three-word sequence of words like “not at all”, or “turn off light”.

For example, the bigrams of Sentence 2: “This is not good at all” are as follows:

- “This is”
- “is not”
- “not good”
- “good at”
- “at all”

we use bigrams (Bag-of-bigrams), this model can differentiate between sentence 1 and sentence 2. So, using bi-grams makes tokens more understandable

Term frequency-inverse document frequency

- The scoring method being used above takes the count of each word and represents the word in the vector by the number of counts of that particular word.
- We rescale the frequency of words by how often they appear in all documents so that the scores for frequent words like “the” that are also frequent across all documents are penalized.
- TF-IDF is intended to reflect how relevant a term is in a given document.
- It has many uses, most importantly in automated text analysis, and is very useful for scoring words in machine learning algorithms for NLP.
- It is a statistical measure that evaluates how relevant a word is to a document in a collection of documents.

Term frequency-inverse document frequency

- Google has already been using TF*IDF (or TF-IDF, TFIDF, TF.IDF) to rank your content for a long time, as the search engine seems to focus more on term frequency rather than on counting keywords.
- TF-IDF for a word in a document is calculated by multiplying two different metrics:
- The **term frequency (TF)** of a word in a document. There are several ways of calculating this frequency, with the simplest being a raw count of instances a word appears in a document. Then, there are other ways to adjust the frequency. For example, by dividing the raw count of instances of a word by either length of the document, or by the raw frequency of the most frequent word in the document.
- The formula to calculate Term-Frequency is: TF(i,j) = n(i,j) / Σ n(i,j)
- n(i,j)= number of times nth word occurred in a document
- Σn(i,j) = total number of words in a document.

Term frequency-inverse document frequency

- The inverse document frequency(IDF) of the word across a set of documents. This suggests how common or rare a word is in the entire document set. The closer it is to 0, the more common is the word. This metric can be calculated by taking the total number of documents, dividing it by the number of documents that contain a word, and calculating the logarithm.
- So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach 1.
- The formula to calculate Inverse document Frequency is: IDF= log(N/dN)
- Where N=Total number of documents in the dataset
- dN=total number of documents in which nth word occur

The TF-IDF is obtained by TF-IDF=TF*IDF

Term frequency-inverse document frequency

- Term frequency = no of repetition of words in a sentence/ No of words in sentence
- IDF = log(no of sentences/ no of sentences containing words)
- Sentence 1= good boy
- Sentence 2 = good girl
- Sentence 3 = boy girl good

Word	Frequency
good	3
girl	2
boy	2

TF	Sent-1	Sent-2	Sent-3
good	1/2	1/2	1/3
boy	1/2	0	1/3
girl	0	1/2	1/3

IDF	Words	IDF
	good	log(3/3)
	boy	log(3/2)
	girl	log(3/2)

Term frequency-inverse document frequency

- Goal = TF * IDF

TF	f-1	f-2	f-3
	good	boy	girl
Sen-1	½*log(3/3)	½*log(3/2)	0*log(3/2)
Sen-2	½*log(3/3)	0	½*log(3/2)
Sen-3	1/3*log(3/3)	1/3*log(3/2)	1/3*log(3/2)

UPES

UNIVERSITY OF TOMORROW

Thank You