Sentiment Classification using ML Techniques

Bo Pang, Lillian Lee, and Shivakumar Vaithyanatham

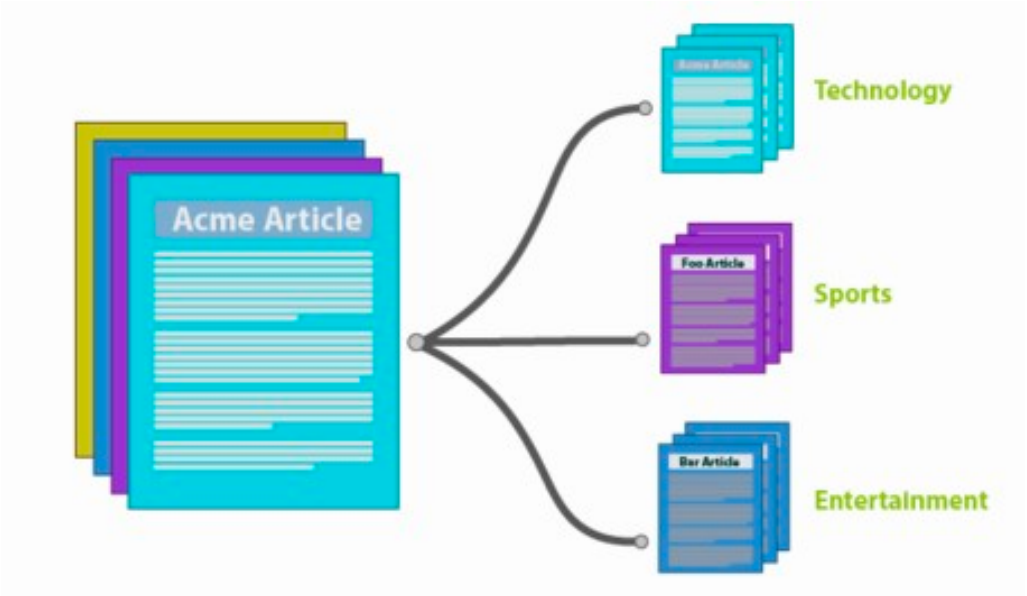
Presented by Zhengjie (Jay) Xu

10/12/2017

UNIVERSITY OF
SAN FRANCISCO

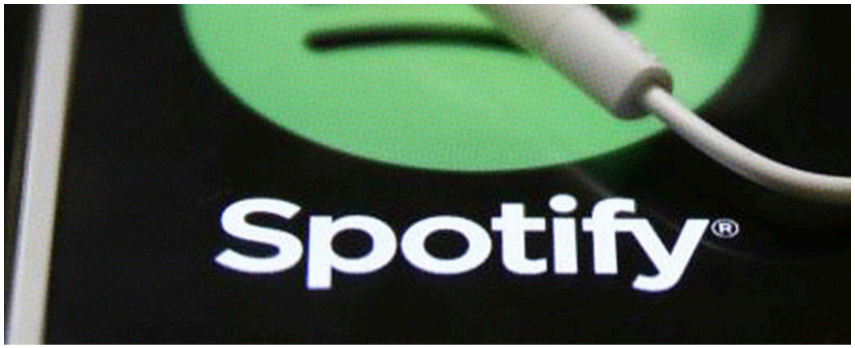
Background

- Back in 2002, when there was no Twitter
- Classify by topic vs Classify by sentiment



Background

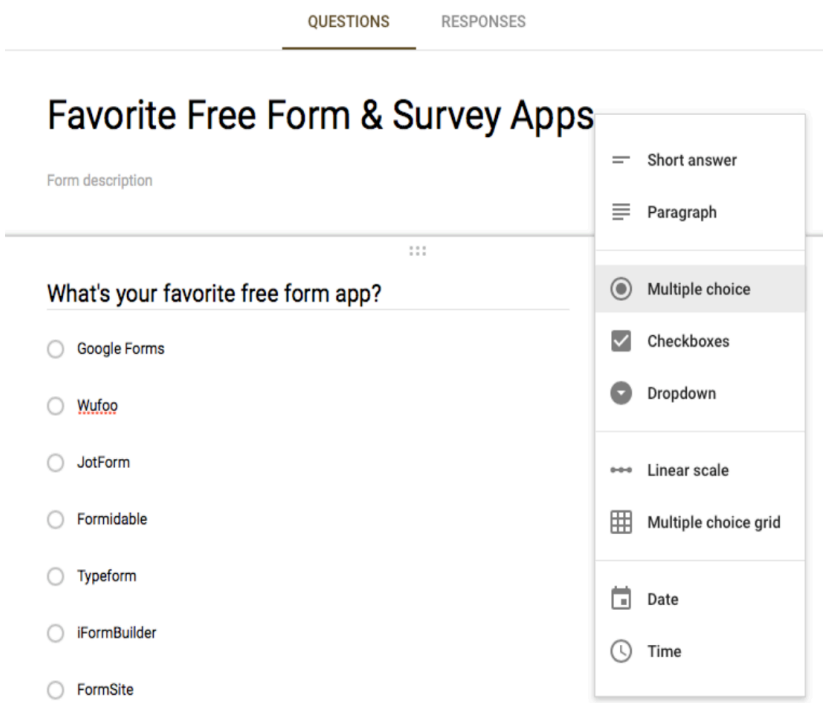
- Classify by sentiment



Spotify's Discover Weekly: How machine learning finds your new music

The science behind personalized music recommendations

This Monday—just like every Monday— over 100 million Spotify users found a fresh new playlist waiting for them. It's a custom mixtape of 30 songs they've never listened to before but will probably love. It's called Discover Weekly, and it's pretty much magic.



UNIVERSITY OF SAN FRANCISCO

Goal of the paper

Examine the effectiveness of applying ML to sentiment classification problem, in order to understand the inherent difficulty of the task.

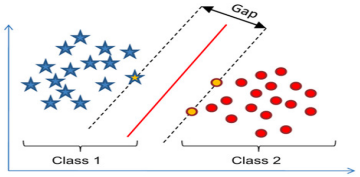
Naïve Bayes

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Maximum Entropy

$$H(p) = -\sum p(a,b) log p(a,b)$$

SVM



Human

UNIVERSITY OF SAN FRANCISCO

Challenge

- **express negation without using negative word**

  "How could anyone sit through this presentation?"

  Contains no single word that is obviously negative.

- **Thwarted expectations**

  "Okay, I'm really ashamed of it, but I enjoyed it. I mean, I admit it's a really awful movie ...the ninth floor of hell...The plot is such a mess that it's terrible. But I loved it."

  Sets up a deliberate contrast to earlier discussion.

Sentiment requires more understanding than the usual topic-based classifications

Model

Data Source

- Internet Movie Database(IMDb)
- Select only reviews where the author rating was expressed either with stars or some numerical value. Suitable for our supervised machine learning model
- Ratings are extracted and converted into one of three categories: positive, negative or neutral
- Use corpus of 752 negative and 1301 positive reviews as training set, and 700 positive and 700 negative reviews as test set
- Data set available on http://www.cs.cornell.edu/people/pabo/movie-review-data/

Model

Baseline model

Baseline 1: random choose

- Accuracy 50%

Baseline 2: human classifier

- Ask 2 students to independently choose good indicator words for positive and negative sentiments in movie reviews
- Count the number of the proposed positive and negative words in a given document, to determine the class of the document

	Proposed word lists	Accuracy	Ties
Human 1	positive: *dazzling, brilliant, phenomenal, excellent, fantastic* negative: *suck, terrible, awful, unwatchable, hideous*	58%	75%
Human 2	positive: *gripping, mesmerizing, riveting, spectacular, cool, awesome, thrilling, badass, excellent, moving, exciting* negative: *bad, cliched, sucks, boring, stupid, slow*	64%	39%

Model

Naïve Bayes

Bag-of-feature framework

- $\{f_1,...,f_m\}$ a predefined set of m features; unigram like “still”, bigram like “really stinks”
- $n_i(d)$ the number of times f_i occurs in document d
- $\vec{d} = (n_1(d), n_2(d), ..., n_m(d))$ document vector

Bayes framework

- Given a document, choose the class that can maximize the posterior probability $p(c|d)$
- Bayes formula: $p(c|d) = \frac{p(c)p(d|c)}{p(d)}$
  where

  $p(c)$ frequency of each class in the training set; prior probability

  $p(d|c) = p(f_1,...,f_m|c)$ given a class, the probability to have features

  Naïve Bayes assumption: $p(d|c) = p(f_1|c) \bullet p(f_2|c) \bullet ... \bullet p(f_m|c)$

  $p(d|c)$ measure the contribution of each feature to the class; likelihood

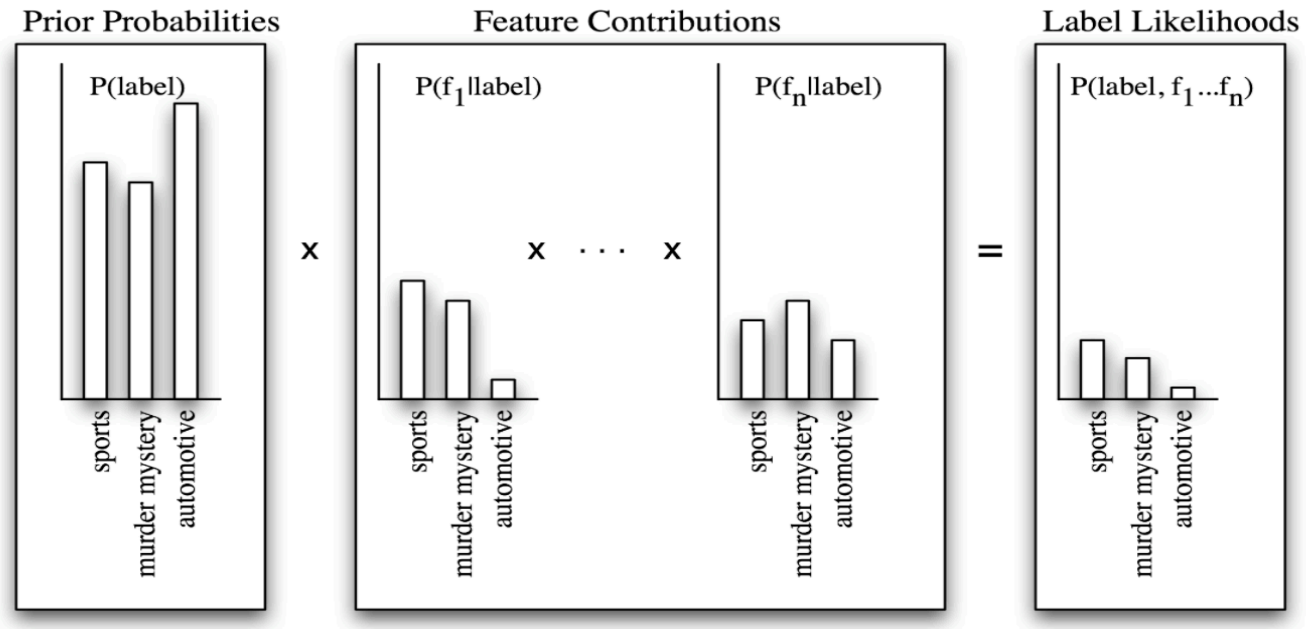
  $p(d)$ the same for every choice of class, does not affect decision; ignore it

Model

Naïve Bayes

- The problem reduces to find the class that maximize

$$p(c) \bullet p(d \mid c)$$
$$= p(c) \bullet p(f_1 \mid c) \bullet p(f_2 \mid c) \bullet \ldots \bullet p(f_m \mid c)$$



Model

Naïve Bayes: a simple example

Let's say we have counted the number of words in a set of labeled training documents. In this set each text document has been labeled as either Positive, Neutral or as Negative. The result will then look like:

Word	Positive Class	Neutral Class	Negative Class	Total
lame	10	20	70	**100**
awesome	70	20	10	**100**
this	50	500	50	**600**
Is	100	600	100	**800**
blog-post	10	90	10	**100**
Total	**240**	**1.230**	**240**	**1.700**

"This blog-post is awesome."

Test set

Model

Naïve Bayes: a simple example

Word	Positive Class	Neutral Class	Negative Class	Total
lame	10	20	70	**100**
awesome	70	20	10	**100**
this	50	500	50	**600**
Is	100	600	100	**800**
blog-post	10	90	10	**100**
Total	**240**	**1.230**	**240**	**1.700**

"This blog-post is awesome."

Class probability

$$P(C_{pos}) = 0.141,$$
$$P(C_{neu}) = 0.723,$$
$$P(C_{neg}) = 0.141.$$

Posterior probability

$$P(C_{pos}) = 0.141 \cdot 50/240 \cdot 10/240 \cdot 100/240 \cdot 70/240 = 1.49 \cdot 10^{-3}$$
$$P(C_{neu}) = 0.723 \cdot 500/1230 \cdot 90/1230 \cdot 600/1230 \cdot 20/1230 = 1.71 \cdot 10^{-4}$$
$$P(C_{neg}) = 0.141 \cdot 50/240 \cdot 10/240 \cdot 100/240 \cdot 10/240 = 2.12 \cdot 10^{-4}$$

Decision

- classified in the positive category

Model

Maximum Entropy

Similar to Naïve Bayes

- Also estimate $p(c|d)$
- The probability that a document belongs to a particular class given a context must maximize the entropy of the classification system
- Distribution should be as uniform as possible; maximize uncertainty of event

ME Framework

- Entropy is measured by $H(p) = -\sum p(a,b) log p(a,b)$

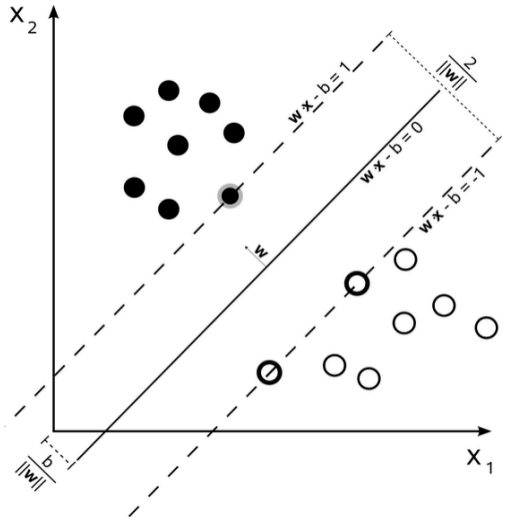
  In our document classification case, it can be shown that the probability distribution has an exponential form: $P(c|d) = \frac{1}{Z(d)} exp(\sum_i \lambda_i f_i(d,c))$

  where $f_i(d,c)$ is a feature function, λ_i is the weight parameter of the feature function and $Z(d)$ is a normalization factor given by $Z(d) = \sum_c exp(\sum_i \lambda_i f_i(d,c))$
- Use Improved Iterative Scaling algorithm to train the model; essentially a gradient descent
- Choose the class that maximize $p(c|d)$

Model

Support Vector Machine

- The SVM tries to find a hyperplane that separates the document vectors in one class from the other as optimally as possible. Here as optimally as possible means that it maximizes the margin between two classes.



- This hyperplane is given by $\langle \vec{\omega} \bullet \vec{d} \rangle + b = 0$; $\vec{\omega}$ is the weight factor defining the hyperplane
- Use the optimization to find the $\vec{\omega}$
- Test data: if $\langle \vec{\omega} \bullet \vec{d} \rangle + b \geq 0$, then the document belongs to positive class.

Compare result

	Features	NB	ME	SVM
(1)	unigrams	81.0	80.4	**82.9**
(2)	bigrams	77.3	**77.4**	77.1
(3)	unigrams + position	81.0	80.1	**81.6**

For Unigrams

- Machine learning algorithms clearly surpass the random-choice baseline of 50%
- Also beat the human-selected-unigram baselines of 58% and 64%
- The topic-based classification utilizing same ML achieve accuracy of 90%

For Bigrams

- Bigrams are not effective in our setting

Position

- Adding position of word information does not help

Conclusion

- The result produced via ML techniques are quite good in comparison to the human-generated baselines
- In terms of relative performance, Naïve Bayes tends to do the worst and SVM tend to do the best, although the differences aren't very large
- On the other hand, we were not able to achieve accuracies on the sentiment classification problem comparable to those reported for topic-based categorization
- Other models like Logistic Regression, Neural Networks and Hidden Markov Models can be used