Text Classification & Linear Models

CMSC 723 / LING 723 / INST 725

Marine Carpuat

Slides credit: Dan Jurafsky & James Martin, Jacob Eisenstein

Logistics/Reminders

- Homework 1 – due Thursday Sep 7 by 12pm.
- Project 1 coming up
- Thursday lecture time: project set-up office hour in CSIC 1121

Recap: Word Meaning

2 core issues from an NLP perspective

- **Semantic similarity**: given two words, how similar are they in meaning?
- Key concepts: vector semantics, PPMI and its variants, cosine similarity

- **Word sense disambiguation**: given a word that has more than one meaning, which one is used in a specific context?
- Key concepts: word sense, WordNet and sense inventories, unsupervised disambiguation (Lesk), supervised disambiguation

Today

- Text classification problems
  - and their evaluation
- Linear classifiers
  - Features & Weights
  - Bag of words
  - Naïve Bayes

Text classification

Is this spam?

From: "Fabian Starr“
<Patrick_Freeman@pamietaniepeerelu.pl>

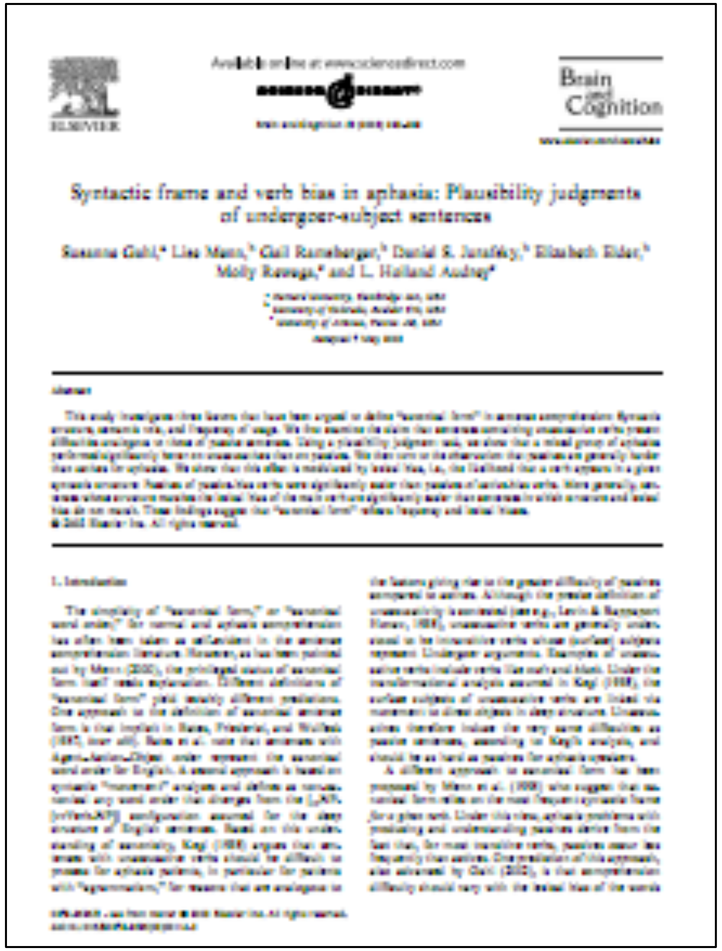
Subject: Hey! Sofware for the funny prices!

Get the great discounts on popular software today
for PC and Macintosh
http://iiled.org/Cj4Lmx

70-90% Discounts from retail price!!!
All sofware is instantly available to download - No
Need Wait!

What is the subject of this article?

MEDLINE Article


Available online at www.sciencedirect.com

Brain and Cognition

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

?

MeSH Subject Category Hierarchy

- 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 $Y = \{y_1, y_2, \ldots, y_J\}$
- *Output*: a predicted class $y \in Y$

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 $Y = \{y_1, y_2, \ldots, y_J\}$
  - a training set of m hand-labeled documents $(d_1, y_1), \ldots, (d_m, y_m)$
- *Output*
  - a learned classifier $d \rightarrow y$

Aside: getting examples for supervised learning

- Human annotation
  - By experts or non-experts (crowdsourcing)
  - Found data

- How do we know how good a classifier is?
  - Compare classifier predictions with human annotation
  - On held out test examples
  - Evaluation metrics: accuracy, precision, recall

The 2-by-2 contingency table

	correct	not correct
selected	tp	fp
not selected	fn	tn

Precision and recall

- **Precision**: % of selected items that are correct
  Recall: % of correct items that are selected

	correct	not correct
selected	tp	fp
not selected	fn	tn

A combined measure: F

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

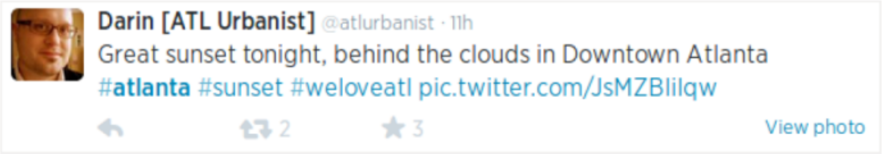
$$F = \frac{1}{\alpha \frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

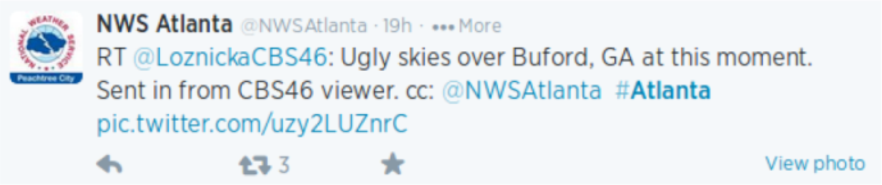
- People usually use balanced F1 measure
  - i.e., with $\beta = 1$ (that is, $\alpha = ½$):

    $F = 2PR/(P+R)$

Linear Classifiers

Bag of words





$\mathbf{w}_1 = \{\text{great}, \text{sunset}, \text{tonight}, \ldots\}$

$\mathbf{w}_2 = \{\text{ugly}, \text{skies}, \text{buford}, \ldots\}$

$$
\begin{array}{lcccccccccccccc}
 & \text{aardvark} & \text{abacus} & \ldots & \text{behind} & \ldots & \text{buford} & \ldots & \text{clouds} & \ldots & \text{great} & \ldots & \text{ugly} & \ldots \\
\hline
\mathbf{x}_1^\top = & 0 & 0 & 0\ldots0 & 1 & 0\ldots0 & 0 & 0\ldots0 & 0 & 0\ldots0 & 1 & 0\ldots0 & 0 & 0\ldots \\
\mathbf{x}_2^\top = & 0 & 0 & 0\ldots0 & 0 & 0\ldots0 & 1 & 0\ldots0 & 0 & 0\ldots0 & 0 & 0\ldots0 & 1 & 0\ldots
\end{array}
$$

$$\mathbf{x}_1 = \{\text{great} : 1, \text{sunset} : 1, \text{tonight} : 1, \ldots\}$$
$$\mathbf{x}_2 = \{\text{ugly} : 1, \text{skies} : 1, \text{buford} : 1, \ldots\}$$

Defining features

Suppose $y \in \mathcal{Y} = \{\text{pos}, \text{neg}, \text{neut}\}$. Then,

$$\begin{aligned}
\mathbf{f}(\mathbf{x}, y = \text{pos}) =& [\mathbf{x}^\top, \mathbf{0}^\top, \mathbf{0}^\top, 1]^\top \\
\mathbf{f}(\mathbf{x}, y = \text{neg}) =& [\mathbf{0}^\top, \mathbf{x}^\top, \mathbf{0}^\top, 1]^\top \\
\mathbf{f}(\mathbf{x}, y = \text{neut}) =& [\mathbf{0}^\top, \mathbf{0}^\top, \mathbf{x}^\top, 1]^\top
\end{aligned}$$

Defining features

Suppose $y \in \mathcal{Y} = \{\text{pos}, \text{neg}, \text{neut}\}$. Then,

$$\begin{aligned}
\mathbf{f}(\mathbf{x}, y = \text{pos}) =& [\mathbf{x}^\top, \mathbf{0}^\top, \mathbf{0}^\top, 1]^\top \\
\mathbf{f}(\mathbf{x}, y = \text{neg}) =& [\mathbf{0}^\top, \mathbf{x}^\top, \mathbf{0}^\top, 1]^\top \\
\mathbf{f}(\mathbf{x}, y = \text{neut}) =& [\mathbf{0}^\top, \mathbf{0}^\top, \mathbf{x}^\top, 1]^\top
\end{aligned}$$

The feature vector is composed of individual feature functions, e.g.:

$$\begin{aligned}
f_{176}(\mathbf{x}, y) :=& x_{176} \times \delta(y = \text{pos}) \\
=& \delta(\mathrm{great} \in \mathbf{w} \wedge y = \text{pos}) \\
f_{177}(\mathbf{x}, y) :=& x_{177} \times \delta(y = \text{pos}) \\
f_{10176}(\mathbf{x}, y) :=& x_{176} \times \delta(y = \text{neg}) \ldots
\end{aligned}$$

We usually add an "offset" feature at the end of each vector.

Linear classification

We can then define **weights** for each feature:

$$
\begin{aligned}
\boldsymbol{\theta} = \{&\langle \text{great}, \mathsf{pos} \rangle = 1, \langle \text{great}, \mathsf{neg} \rangle = -1, \langle \text{great}, \mathsf{neut} \rangle = 0, \\
&\langle \text{ugly}, \mathsf{pos} \rangle = -1, \langle \text{ugly}, \mathsf{neg} \rangle = 1, \langle \text{ugly}, \mathsf{neut} \rangle = 0, \\
&\langle \text{buford}, \mathsf{pos} \rangle = 0, \langle \text{buford}, \mathsf{neg} \rangle = 0, \langle \text{buford}, \mathsf{neut} \rangle = 0, \\
&\ldots\}
\end{aligned}
$$

We can arrange these weights into a vector.

The **score** for any instance and label is equal to the sum of the weights for all features in the instance:

$$
\begin{aligned}
\psi_{y,\mathbf{x}} =& \sum_n \theta_n f_n(\mathbf{x}, y) \\
=& \boldsymbol{\theta}^\top \mathbf{f}(\mathbf{x}, y) \\
\hat{y} =& \operatorname*{arg\,max}_y \boldsymbol{\theta}^\top \mathbf{f}(\mathbf{x}, y)
\end{aligned}
$$

Linear Models for Classification

Feature function representation

$$\hat{y} = \arg\max_{y} \theta^{\top} \mathbf{f}(\mathbf{x}, y)$$

Weights

How can we learn weights?

- By hand
- Probability
  - e.g.,Naïve Bayes
- Discriminative training
  - e.g., perceptron, support vector machines

Generative Story for Multinomial Naïve Bayes

- A hypothetical stochastic process describing how training examples are generated

For each document i,

- draw the label $y_i \sim \text{Categorical}(\mu)$
- draw the vector of counts $\boldsymbol{x}_i \sim \text{Multinomial}(\phi_{y_i})$.

$$\mathrm{p}_{\text{mult}}(\boldsymbol{x}; \phi) = \frac{\left(\sum_j x_j\right)!}{\prod_j x_j!} \prod_j \phi_j^{x_j}$$

Prediction with Naïve Bayes

$$
\begin{aligned}
\text{Score(x,y)} \quad &:= \log P(\mathbf{x}, y; \phi, \mu) \\
&= \log P(\mathbf{x}|y; \phi) P(y; \mu) \\
&= \log P(\mathbf{x}|y; \phi) + \log P(y; \mu)
\end{aligned}
$$

Prediction with Naïve Bayes

$$
\begin{aligned}
\text{Score(x,y)} \quad &:= \log P(\mathbf{x}, y; \phi, \mu) \\
&= \log P(\mathbf{x}|y; \phi) P(y; \mu) \\
&= \log P(\mathbf{x}|y; \phi) + \log P(y; \mu) \\
&= \log \text{Multinomial}(\mathbf{x}; \phi_y) + \log \text{Cat}(y; \mu) \\
&= \log \frac{(\sum_n x_n)!}{\prod_n x_n!} + \log \prod_n \phi_{y,n}^{x_n} + \log \mu_y \\
&\propto \sum_n x_n \log \phi_{y,n} + \log \mu_y \\
&= \boldsymbol{\theta}^\mathsf{T} \mathbf{f}(\mathbf{x}, y)
\end{aligned}
$$

where

$$
\begin{aligned}
\boldsymbol{\theta} &= [\log \boldsymbol{\phi}_1^\mathsf{T}, \log \mu_1, \log \boldsymbol{\phi}_2^\mathsf{T}, \log \mu_2, \ldots]^\mathsf{T} \\
\mathbf{f}(\mathbf{x}, y) &= [\mathbf{0}, \ldots, \mathbf{0}, \mathbf{x}^\mathsf{T}, 1, \mathbf{0}, \ldots, \mathbf{0}]^\mathsf{T}
\end{aligned}
$$

Parameter Estimation

- "count and normalize"
- Parameters of a multinomial distribution

$$\phi_{y,j} = \frac{\sum_{i:Y_i=y} x_{i,j}}{\sum_{j'} \sum_{i:Y_i=y} x_{i,j'}} = \frac{\text{count}(y,j)}{\sum_{j'} \text{count}(y,j')}$$

  - Relative frequency estimator
  - Formally: this is the maximum likelihood estimate
    - See CIML for derivation

Smoothing (add alpha / Laplace)

$$\phi_{y,j} = \frac{\alpha + \sum_{i:Y_i=y} x_{i,j}}{\sum_{j'=1}^{V} \left(\alpha + \sum_{i:Y_i=y} x_{i,j'}\right)} = \frac{\alpha + \mathbf{count}(y, j)}{V\alpha + \sum_{j'=1}^{V} \mathbf{count}(y, j')}$$

Naïve Bayes recap

- Define $\mathrm{p}(\boldsymbol{x}, \boldsymbol{y})$ via a *generative model*
- Prediction: $\hat{y} = \arg\max_y \mathrm{p}(\boldsymbol{x}_i, y)$
- Learning:

$$\begin{aligned}
\boldsymbol{\theta} &= \arg\max_{\boldsymbol{\theta}} \mathrm{p}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) \\
\mathrm{p}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) &= \prod_i \mathrm{p}(\boldsymbol{x}_i, y_i; \boldsymbol{\theta}) = \prod_i \mathrm{p}(\boldsymbol{x}_i | y_i) \mathrm{p}(y_i) \\
\phi_{y,j} &= \frac{\sum_{i:Y_i=y} x_{ij}}{\sum_{i:Y_i=y} \sum_j x_{ij}} \\
\mu_y &= \frac{\text{count}(Y = y)}{N}
\end{aligned}$$

This gives the maximum likelihood estimator (MLE; same as relative frequency estimator)

Today

- Text classification problems
    - and their evaluation
- Linear classifiers
    - Features & Weights
    - Bag of words
    - Naïve Bayes