Text Mining
Statistical Modeling of Textual Data
Lecture 1

Måns Magnusson, Mattias Villani

Division of Statistics and Machine Learning
Dept. of Computer and Information Science
Linköping University

Overview

Supervised text classification

Text classification techniques

Feature construction

Evaluation of classifier

Applications

Section 1

Supervised text classification

SUPERVISED CLASSIFICATION

- Predict the **class label** $s \in S$ using a set of **features**.
- Feature = Explanatory variable = Predictor = Covariate

- Binary classification: $s \in \{0, 1\}$
  - Movie reviews: $S = \{\text{pos,neg}\}$
  - E-mail spam: $S = \{\text{Spam,Ham}\}$
  - Bankruptcy: $S = \{\text{Not bankrupt, Bankrupt}\}$

- Multi-class classification: $s \in \{1, 2, ..., K\}$
  - Topic categorization of web pages:
    $S = \{'\text{News}', '\text{Sports}', '\text{Entertainment}'\}$
  - POS-tagging: $S = \{\text{VB,JJ,NN,...,DT}\}$

SUPERVISED CLASSIFICATION, CONT.

- Example data:
  - Larry Wall, born in British Columbia, Canada, is the original creator of the programming language Perl. Born in 1956, Larry went to ...
  - Bjarne Stroustrup is a 62-years old computer scientist ...

Person	Income	Age	Single	Payment remarks	Bankrupt
Larry	10	58	Yes	Yes	Yes
Bjarne	15	62	No	Yes	No
⋮	⋮	⋮	⋮	⋮	⋮
Guido	27	56	No	No	No

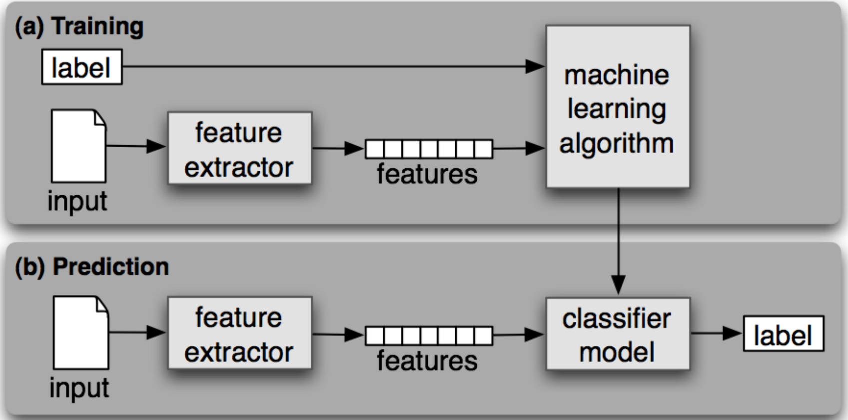
- Classification: construct prediction machine

$$\text{Features} \rightarrow \text{Class label}$$

- More generally:

$$\text{Features} \rightarrow \Pr(\text{Class label}|\text{Features})$$

SUPERVISED LEARNING FOR CLASSIFICATION



GENERATIVE AND DISCRIMINATIVE MODELS

- Generative:
  - Model (all) data $p(\mathbf{s}, \mathbf{w})$
  - Example: Naive Bayes
- Discriminative:
  - Model data conditional on features $p(\mathbf{s}|\mathbf{w})$
  - Example: Logistic regression

Section 2

Text classification techniques

THE NAIVE BAYES CLASSIFIER

- Generative classification

$$\underset{s \in S}{\operatorname{argmax}}\, p(s|\mathbf{x})$$

where $\mathbf{x} = (x_1, \ldots, x_n)$ is a feature vector.

- By Bayes' theorem

$$p(s|\mathbf{x}) = \frac{p(\mathbf{x}|s)p(s)}{p(\mathbf{x})} \propto p(\mathbf{x}|s)p(s)$$

- Bayesian classification

$$\underset{s \in S}{\operatorname{argmax}}\, p(\mathbf{x}|s)p(s)$$

- Generative - We model both **x** and **s**: What are our generative model for $p(\mathbf{x}|s)$?
- **Naive Bayes (NB): features are assumed independent**

Naive Bayes - Multivariate Bernoulli

- We model both $\mathbf{x}$ and $\mathbf{s}$

$$p(\mathbf{x}|s) = \prod_{j=1}^{n} p(\mathbf{x}_j|s_j)$$
$$= \prod_{j=1}^{n} \prod_{v=1}^{V} p(x_{j,v}|s_j)$$

  where $p(x_{j,v}) \sim \mathit{Bernoulli}(p_{v,s})$
- Generative model
  - For all 1 to D
    - Simulate a class $s_d \sim \mathrm{MN}(\theta)$
    - For all 1 to V: $x_{d,v} \sim \mathit{Bernoulli}(p_{s,v})$
- **Naive Bayes solution:**

$$\operatorname*{argmax}_{s \in S} \left[\prod_{j=1}^{n} p(x_j|s) \right] p(s)$$

Naive Bayes - Multivariate Bernoulli

- $p(s)$ can be easily estimated from training data by relative frequencies of classes.

- With binary features, $p(x_j|s)$ can be easily estimated by

$$\hat{p}(x_j|s) = \hat{\psi}_j = \frac{C(x_j, s)}{C(s)}$$

or

$$E(\psi_j|\cdot) = \frac{C(x_j, s) + \alpha}{C(s) + 2\alpha}$$

- Example: $s =$news, $x_j =$ has('ball')

$$\hat{p}\,(\text{has(ball)}|\text{news}) = \frac{\text{Number of news articles containing the word 'ball'}}{\text{Number of news articles}}$$

Naive Bayes - Multinomial*

- We model both **x** and **s**

$$p(\mathbf{w}|s) = \prod_{j=1}^{n} p(\mathbf{w}_j|s_j)$$

  where $p(\mathbf{w}_j|s_j) \sim MN(\theta_{s_j}, n_j)$

- Generative model
  - For all 1 to D
    - Simulate a class $s_d \sim Multinomial(\theta)$
    - Simulate $x_d \sim Multinomial(\phi_s, n_d)$
- Naive Bayes solution

$$\underset{s \in S}{\operatorname{argmax}} \left[\prod_{j=1}^{n} p(x_j|s) \right] p(s)$$

NAIVE BAYES - MULTINOMIAL

- $p(s)$ can be estimated from training data by relative frequencies of classes (again).

- With frequency features, $p(w_j|s)$ can be easily estimated by

$$\hat{p}(w_j|s) = \hat{\theta}_{w,s} = \frac{C(w_j, s)}{C(s)}$$

or

$$E(\theta_{w,s}|\cdot) = \frac{C(w_j, s) + \alpha}{C(s) + \alpha V}$$

- Example: $s =$news, $w_j =$ 'ball'

$$\hat{p}\,(\text{ball}|\text{news}) = \frac{\text{Number words 'ball' in news articles}}{\text{Number of news articles}}$$

NAIVE BAYES

- **Continuous features** (e.g. lexical diversity) can be handled by:
  - Replacing continous feature with several binary features ($1 \leq$lexDiv< 2, $2 \leq$lexDiv≤ 10 and lexDiv> 10)
  - Estimating $p(x_j|s)$ by a density estimator (e.g. kernel estimator)

- Finding the **most discriminatory features**. Sort from largest to smallest

$$\frac{p(x_j|s = pos)}{p(x_j|s = neg)} \text{ for } j = 1, ..., n.$$

- **Problem with NB**: features are seldom independent $\Rightarrow$ double-counting the evidence of individual features.

- **Advantages of NB**: simple and fast, yet often surprising accurate classifications. Extendible.

LOGISTIC REGRESSION

- Discriminative classification

$$p(s|\mathbf{x}, \beta)$$

  where $p(s|\mathbf{x}, \beta)$ follow a Categorical or Bernoulli distribution.
- Logistic regression (Maximum Entropy/**MaxEnt**):

$$p(s = 1|\mathbf{x}) = \frac{\exp(\mathbf{x}'\beta)}{1 + \exp(\mathbf{x}'\beta)}$$

- Generative model:
  - For all 1 to D
    - We have x (this is not probabilistically modeled)
    - Simulate a class $s_d \sim \text{Bernoulli}\left(\frac{\exp(\mathbf{x}'\beta)}{1+\exp(\mathbf{x}'\beta)}\right)$ (if binary)
- The likelihood function

$$LogLik(\beta) = \log\left(\prod^{n} \pi_i^{s}(1 - \pi_i)^{(1-s)}\right)$$

where

$$\exp(\mathbf{x}'\beta)$$

LOGISTIC REGRESSION

- Classification rule: Choose $s = 0$ if $p(s|\mathbf{x}) < 0.5$ otherwise choose $s = 1$.
- ... at least when consequences of different choices of s are the same. Loss/Utility function.
- Multinomial regression for multi-class data with K classes

$$p(s = s_j|\mathbf{x}) = \frac{\exp(\mathbf{x}'\beta_j)}{\sum_{k=1}^{K} \exp(\mathbf{x}'\beta_k)}$$

- Classification

$$\underset{s \in \{s_1, \ldots s_K\}}{\operatorname{argmax}}\ p(s|\mathbf{x})$$

- $P \times (S - 1)$ number of coefficients
- Classification with text data is like any multi-class regression problem ... but with hundred or thousands of covariates! **Wide data**.
- **Similar to genomics**

SHRINKAGE

- Keep all covariates, but **shrink** their β-coefficient to zero.
- **Penalized likelihood**

$$L_{Ridge}(\beta) = LogLik(\beta) - \lambda\beta'\beta$$

  where λ is the **penalty parameter**.

- Maximize $L_{Ridge}(\beta)$ with respect to β. Trade-off of fit ($LogLik(\beta)$) against complexity penalty $\beta'\beta$.
- **Ridge regression** if regression is linear.
- The penalty can be motivated as a **Bayesian prior** $\beta_i \overset{iid}{\sim} N(0, \lambda^{-1})$.
- λ can be estimated by cross-validation or Bayesian methods.

Lasso/ElasticNet - Shrinkage and variable selection

- Replace Ridge penalty

$$L_{Ridge}(\beta) = LogLik(\beta) - \lambda \sum_{j=1}^{n} \beta_j^2$$

by

$$L_{Lasso}(\beta) = LogLik(\beta) - \lambda \sum_{j=1}^{n} |\beta_j|$$

- The β that maximizes $L_{Lasso}(\beta)$ is called the **Lasso estimator**.
- Some parameters are shrunked exactly to zero $\Rightarrow$ Lasso does **both shrinkage AND variable selection**.
- Lasso penalty is equivalent to a double exponential prior

$$p(\beta_i) = \frac{\lambda}{2} \exp\left(\lambda \left|\beta_i - 0\right|\right)$$

FASTTEXT

- **Fast** state-of-the-art text classification
- Good baseline (together with Naive Bayes)
- Similar idea as the word2vec model
  - Uses the CBOW word2vec model but instead of classifying the middle word, it classifies the class
  - Handles ngrams to use word orders

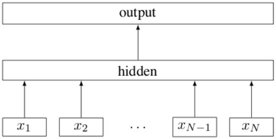


Figure 1: Model architecture of `fastText` for a sentence with N ngram features $x_1, \ldots, x_N$. The features are embedded and averaged to form the hidden variable.

FIGUR: Idea behind fastText (Joulin et. al. 2016)

FASTTEXT

Model	AG	Sogou	DBP	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
BoW (Zhang et al., 2015)	88.8	92.9	96.6	92.2	58.0	68.9	54.6	90.4
ngrams (Zhang et al., 2015)	92.0	97.1	98.6	95.6	56.3	68.5	54.3	92.0
ngrams TFIDF (Zhang et al., 2015)	92.4	97.2	98.7	95.4	54.8	68.5	52.4	91.5
char-CNN (Zhang and LeCun, 2015)	87.2	95.1	98.3	94.7	62.0	71.2	59.5	94.5
char-CRNN (Xiao and Cho, 2016)	91.4	95.2	98.6	94.5	61.8	71.7	59.2	94.1
VDCNN (Conneau et al., 2016)	91.3	96.8	98.7	95.7	64.7	73.4	63.0	95.7
`fastText`, $h = 10$	91.5	93.9	98.1	93.8	60.4	72.0	55.8	91.2
`fastText`, $h = 10$, bigram	92.5	96.8	98.6	95.7	63.9	72.3	60.2	94.6

Table 1: Test accuracy [%] on sentiment datasets. `FastText` has been run with the same parameters for all the datasets. It has 10 hidden units and we evaluate it with and without bigrams. For char-CNN, we show the best reported numbers without data augmentation.

FIGUR: State-of-the-art classification accuracy (Joulin et. al. 2016)

Summary

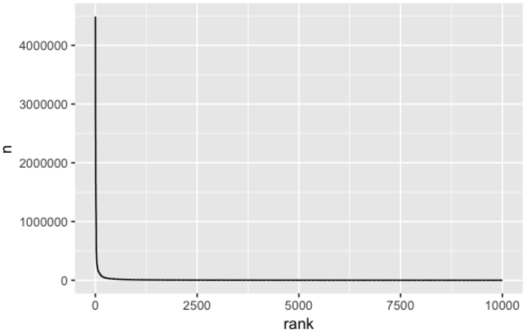
- A lot of other approaches exist (Perceptrons, Deep neural nets etc., XGBoosts etc)
- Generally linear models are close to state of the art.
- In many cases Naive Bayes works really good!

Section 3

FEATURE CONSTRUCTION

Background: Zipf law of language

- The frequency of any word is inversely proportional to its rank.



Figur: The distribution of the Riksdagen corpus

Background: Zipf law of language

token	n	rank	prop
att	4490445	1	0.0443122
det	3832592	2	0.0378204
i	2719543	3	0.0268368
och	2662884	4	0.0262776
är	2399496	5	0.0236785
som	2369072	6	0.0233783
har	1654926	7	0.0163310
för	1640451	8	0.0161882
en	1632564	9	0.0161103
vi	1572204	10	0.0155147

Tabell: The Riksdagen corpus

STANDARD CORPUS CURATION

- Removal of stop words
- Removal of rare words
- Identifying terms using TF-IDF
- Stemming/lemmatization
  - Language specific
  - May not improve much

BAG OF WORD REPRESENTATIONS

- Any quantity computed from a document can used as a **feature**
- Works generally very well (see `fastText` article).

- Document language structure
  - Lexical diversity/complexity,
  - Total number of tokens
- Individual terms
  - Presence/absence of individual words
  - Number of times an individual word is used
  - Presence/absence of individual bigrams

Document	has('ball')	has('EU')	has('political_arena')	wordlen	Lex. Div.	Topic
Article1	Yes	No	No	4.1	5.4	Sports
Article2	No	No	No	6.5	13.4	Sports
⋮	⋮	⋮		⋮	⋮	⋮
ArticleN	No	No	Yes	7.4	11.1	News

Latent features

- Preprocessing step of training models
- Using latent representations of a document
  - Topic models (for thematic predictions)
  - Word embeddings (word2vec, fastText)

Section 4

Evaluation of classifier

EVALUATING A CLASSIFIER: ACCURACY AND ERROR

- Train and test set.
- **Confusion matrix**:

		Truth	
		Spam	Not Spam
Decision	Spam	tp	fp
	Not Spam	fn	tn

- tp = true positive, fp = false positive
- fn = false negative, tn = true negative

- **Accuracy** is the proportion of correctly classified items

$$\texttt{Accuracy} = \frac{tp + tn}{tp + tn + fn + fp}$$

ACCURACY CAN BE MISLEADING

- Accuracy is problematic when tn is large. High accuracy can then be obtained by not acting at all!

		Truth	
		Spam	Not Spam
Choice	Spam	0	0
	Not Spam	100	900

- But it's what people commonly use.

Section 5

APPLICATIONS

Sentiment analysis

- We have trained sentiments - Positive/Negative/Neutral
- Commonly used to analyze corpora
- Classify different text segments (sentences) to different sentiments
- Common problem!

SPAM FILTER

- Classify e-mail as spam/not spam
- Why has Gmail is the best spam filter?

MASTER THESIS PROPOSAL: LARGE SCALE THEMA CODE CLASSIFIERS AT STORYTEL

- **Taken already!**
- Large scale hiearchical classification of books
  - Hiearchical classification is a real problem
  - State-of-the-art: Ensamble of Multinomial Naive Bayes...
- There is a large classification structure (approx 6 000 classes/thema codes)
- Classify individual books given text and meta data.

TEXT MINING PROJECTS

- **Spooky Author Identification (Kaggle)**
- *Deadline* 15 december. Cash prize of $25 000.
    - https://www.kaggle.com/c/spooky-author-identification