CS54701: Information Retrieval

Text Categorization (II)

3 March 2016

Prof. Chris Clifton

Announcements

- No questions about the exam yet
  - Still waiting for some distance learning exams to be uploaded
- Project 2 is out – Collaborative Filtering
  - Watch the web site for updates
    - Hoping to get some online evaluation tools running that will help in later stages of the project
  - Two weeks should be plenty of time
    - Start now, or you'll be working over spring break
    - I won't be available much over spring break (working for the National Science Foundation)

28

Text Categorization (II)

Outline

- Naïve Bayes (NB) Classification
- Logistic Regression Classification

Naïve Bayes Classification

- Naïve Bayes (NB) Classification
  - Generative Model: Model both the input data (i.e., document contents) and output data (i.e., class labels)
  - Make strong assumption of the probabilistic modeling approach
- Methodology
  - Similar with the idea of language modeling approaches for information retrieval
  - Train a language model for all the documents in one category

Naïve Bayes Classification

- Methodology
  - Train a language model for all the documents in one category

    Category 1: $\left(\vec{d}_{1,1},\vec{d}_{1,2},...,\vec{d}_{1,n_1}\right) \rightarrow$ Language model θ_1

    Category 2: $\left(\vec{d}_{2,1},\vec{d}_{2,2},...,\vec{d}_{2,n_2}\right) \rightarrow$ Language model θ_2

    ......

    Category C: $\left(\vec{d}_{C,1},\vec{d}_{C,2},...,\vec{d}_{C,n_K}\right) \rightarrow$ Language model θ_C

  - What is the language model? (Multinomial distribution)
  - How to estimate the language model for all the documents in one category?

Naïve Bayes Classification

- Representation
  - Each document is a "bag of words" with weights (e.g., TF.IDF)
  - Each category is a super "bag of words", which is composed of all words in all the documents associated with the category
  - For all the words in a specific category c, it is modeled by a multinomial distribution as

$$p(\vec{d}_{c1},..,\vec{d}_{cn_c} \mid \theta_c)$$

  - Each category (c) has a prior distribution P(c), which is the probably of choosing category c BEFORE observing the content of a document

32

Naïve Bayes Classification

Maximum Likelihood Estimation:

- Find model parameters for a category that maximizes generation likelihood:

$$\theta_c^* = \arg\max_{\theta_c} p(\vec{d}_{c1},..,\vec{d}_{cn_c} \mid \theta_c)$$

There are K words in vocabulary, $w_1 \ldots w_K$

Data: documents $\vec{d}_{c1},\ldots,\vec{d}_{cn_c}$

For $\vec{d}_{ci}$ with counts $c_i(w_1)$, …, $c_i(w_K)$, and length $|\vec{d}_c|$

Model: multinomial M with parameters $\{p(w_k)\}$

Likelihood: $\Pr(\vec{d}_{c1},\ldots,\vec{d}_{cn_c} \mid \theta)$

$$\theta_c^* = \arg\max_{\theta_c} p(\vec{d}_{c1},..,\vec{d}_{cn_c} \mid \theta_c)$$

33

Maximum Likelihood Estimation (MLE)

$$p(\vec{d}_{c1},..,\vec{d}_{cn_c} \mid \theta) = \prod_{i=1}^{n_c} \binom{|\vec{d}_{ci}|}{c_{ci}(w_1)\ldots c_{ci}(w_K)} \prod_{k=1}^{K} p_k^{c_{ci}(w_k)} \propto \prod_{i=1}^{n_c} \prod_k p_k^{c_{ci}(w_k)}$$

$$l(\vec{d}_{c1},..,\vec{d}_{cn_c} \mid \theta) = \log\ p(\vec{d}_{c1},..,\vec{d}_{cn_c} \mid \theta) = \sum_{i=1}^{n_c} \sum_k c_{ci}(w_k) \log\ p_k$$

$$l'(\vec{d}_{c1},..,\vec{d}_{cn_c} \mid \theta) = \sum_{i=1}^{n_c} \sum_k c_{ci}(w_k) \log\ \theta_k + \lambda(\sum_k p_k - 1)$$

$$\frac{\partial l'}{\partial p_k} = \frac{\sum_{i=1}^{n_c} c_{ci}(w_k)}{p_k} + \lambda = 0 \quad \Rightarrow \quad p_k = -\frac{\sum_{i=1}^{n_c} c_{ci}(w_k)}{\lambda}$$

Use Lagrange multiplier approach
Set partial derivatives to zero
Get maximum likelihood estimate

$$Since \sum_k p_k = 1,\ \lambda = -\sum_k \sum_{i=1}^{n_c} c_{ci}(w_k) = -\sum_{i=1}^{n_c} |\vec{d}_{ci}| \qquad So,\ p_k = p(w_k) = \frac{\sum_{i=1}^{n_c} c_{ci}(w_k)}{\sum_{i=1}^{n_c} |\vec{d}_{ci}|}$$

Naïve Bayes Classification

- **MLE Estimator: Normalization by simple counting**
  - Train a language model for all the documents in one category

$$p(w \mid \theta_c^*) = \frac{\sum_{i=1}^{n_c} c_{ci}(w)}{\sum_{i=1}^{n_c} \left| \vec{d}_{ci} \right|}$$

- **Category Prior:**
  - Number of documents in the category divided by the total number of documents

$$p(c) = \frac{n_c}{\sum_{c'} n_{c'}}$$

Naïve Bayes Classification

- **Smoothed Estimator:**
  - Laplace Smoothing

$$p(w \mid \theta_c^*) = \frac{1 + \sum_{i=1}^{n_c} c_{ci}(w)}{K + \sum_{i=1}^{n_c} \left| \vec{d}_{ci} \right|}$$

Number of Words in Vocabulary

  - Hierarchical Smoothing

$$p(w \mid \theta_c^*) = \lambda_1 P(w \mid \theta_c^*) + \lambda_2 P(w \mid \theta_{c^{up1}}^*) .... + \lambda_m P(w \mid \theta_{c^{root}}^*)$$

  - Dirichlet Smoothing

Naïve Bayes Classification

- **Prediction:**

$$
\begin{aligned}
c^* &= \arg\max_c p(c \mid \vec{d}_i) \\
&= \arg\max_c \left\{ \frac{p(c)\,p(\vec{d}_i \mid c)}{p(\vec{d}_i)} \right\} \\
&= \arg\max_c \left\{ p(c)\,p(\vec{d}_i \mid c) \right\} \quad (\textit{Bayes Rule}) \\
&= \arg\max_c \left\{ p(c) \prod_k p(w_k \mid c)^{c_i(w_k)} \right\} \quad (\textit{Multinomail Dist}) \\
&= \arg\max_c \left\{ \log(p(c)) + \sum_k c_i(w_k) \log p(w_k \mid c) \right\}
\end{aligned}
$$

Plug in the estimator

Naïve Bayes Classification

- **Example of Binary Classification**

  Two classes

$$
c^* = \arg\max_{l \in [-,+]} p(c_l \mid \vec{d}_i) \rightarrow \frac{p(c_+ \mid \vec{d}_i)}{p(c_- \mid \vec{d}_i)}
$$

$$
p(c_+ \mid \vec{d}_i) \propto \prod_k \left[p(w_k \mid c_+) \right]^{c_i(w_k)} \frac{n_+}{n_+ + n_-}
$$

$$
p(c_- \mid \vec{d}_i) \propto \prod_k \left[p(w_k \mid c_-) \right]^{c_i(w_k)} \frac{n_-}{n_+ + n_-}
$$

Naïve Bayes Classification

- **Example of Binary Classification**

$$c^* = \arg\max_{l \in [-,+]} p(c_l \mid \vec{d}_i) \rightarrow \frac{p(c_+ \mid \vec{d}_i)}{p(c_- \mid \vec{d}_i)}$$

$$\log \frac{p(c_+ \mid \vec{d}_i)}{p(c_- \mid \vec{d}_i)} = \log \left\{ \frac{\prod_k [p(w_k \mid c_+)]^{c_i(w_k)} \frac{n_+}{n_+ + n_-}}{\prod_k [p(w_k \mid c_-)]^{c_i(w_k)} \frac{n_-}{n_+ + n_-}} \right\}$$

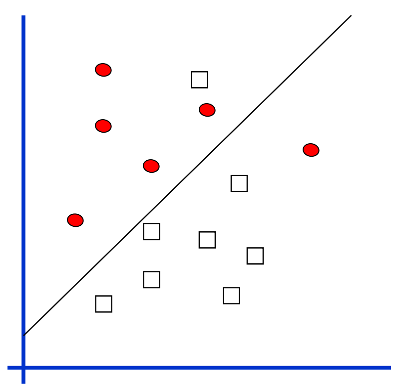
$$= \boxed{\log\left(\frac{n_+}{n_-}\right)} + \sum_k c_i(w_k) \boxed{\log\left(\frac{p(w_k \mid c_+)}{p(w_k \mid c_-)}\right)}$$

$$\log \frac{p(c_+ \mid \vec{d})}{p(c_- \mid \vec{d})} \propto \boxed{b_0} + \sum_k c_i(w_k) \times \boxed{weight(w_k)}$$

Naïve Bayes = Linear Classifier

● denotes +1

□ denotes -1



$$\log \frac{p(c_+ \mid \vec{d}_i)}{p(c_- \mid \vec{d}_i)} \propto b_0 + \sum_k c_i(w_k) \times weight(w_k)$$

Naïve Bayes Classification

- **Entropy**
  - Measuring the uncertainty

    lower entropy means easier predictions

    $$H(\vec{p}) = -\sum_k p_k \log(p_k)$$

  - KL divergence ("relative entropy")

    Distance between p and q

    Nonnegative, 0 when p and q are the same

    $$KL(\vec{p}\,\|\,\vec{q}) = \sum_k p_k \log(\frac{p_k}{q_k})$$

  - Cross entropy

    measuring the coding length based on q when true distribution is p

    $$H(\vec{p}\,\|\,\vec{q}) = -\sum_k p_k \log(q_k) = H(\vec{p}) + KL(\vec{p}\,\|\,\vec{q})$$

Naïve Bayes Classification

- **Prediction:**

$$c^* = \arg\max_c \left\{ \log(p(c)) + \sum_k c_i(w_k) \log p(w_k \mid c) \right\}$$

$$= \arg\max_c \left\{ \frac{\log(p(c))}{|\vec{d}|} + \sum_k \frac{c_i(w_k)}{|\vec{d}|} \log p(w_k \mid c) \right\} \quad (divide\ \vec{d})$$

$$= \arg\max_c \left\{ \frac{\log(p(c))}{|\vec{d}|} + \sum_k p_i(w_k) \log p(w_k \mid c) \right\} \quad (Def\ of\ Cross\ Entropy)$$

$$= \arg\min_c \left\{ H(\vec{p}_i \,\|\, \vec{p}(c)) - \frac{\log(p(c))}{|\vec{d}|} \right\}$$

Cross Entropy

Naïve Bayes Classification

- **Prediction:**

$$c^* = \arg\min_c \left\{ H(\vec{p}_i \,||\, \vec{p}(c)) - \frac{\log(p(c))}{|\vec{d}|} \right\}$$

  - Cross Entropy term selects the category with minimum cross entropy with document (i.e., class distribution that yield the best compression of the document)
  - Second term favors more common category

Naïve Bayes Classification

- Summary
  - Utilize multinomial distribution for modeling categories and documents
  - Use posterior distribution (posterior of category given document) to predict optimal category
- Pros
  - Solid probabilistic foundation
  - Fast online response, linear classifier for binary classification
- Cons
  - Empirical performance not very strong
  - Probabilistic model for each category is estimated to maximize the data likelihood for documents in the category (generative), not for purpose of distinguishing documents in different categories (discriminative)

Logistic Regression Classification

- Naïve Bayes and probabilistic language models for IR are generative models. Model predicts the probability that a document is generated by a category
- In binary classification, Naïve Bayes is a linear classifier, (similarly for multiple categories); how to find a classifier that best distinguish documents in different categories

Discriminative Model

Logistic Regression

Support Vector Machine

Logistic Regression Classification

•Directly model probability of generating *class* conditional on words: $p(c \mid \vec{d}_i)$

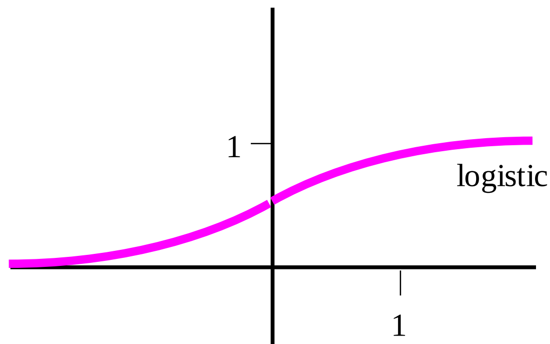
$$\log \frac{P(C_+ \mid \vec{d}_i)}{P(C_- \mid \vec{d}_i)} = \beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k)$$

$$P(C_+ \mid \vec{d}_i) = \frac{\exp\left(\beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k)\right)}{1 + \exp\left(\beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k)\right)}$$

Sigmod/logistic function: $\sigma\left(\beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k)\right)$

Logistic regression: Tune the parameters to optimize *conditional* likelihood (class probability predictions)

logistic transforms



- logistic(x) = $e^x/(1+e^x)$

Logistic Regression Classification

- Estimation Parameters

Maximum Likelihood Estimation:

- Find model parameters for a category that best distinguish its documents from other documents

$$\vec{\beta}_c^* = \arg\max_{\vec{\beta}_c} \prod_i \left[P(C \mid \vec{d}_i)^{\delta(\vec{d}_i,C)} \left[1 - P(C \mid \vec{d}_i)\right]^{1-\delta(\vec{d}_i,C)} \right]$$

$$= \arg\max_{\vec{\beta}_c} \sum_i \left[\delta(\vec{d}_i,C)\log(P(C \mid \vec{d}_i)) + \left(1-\delta(\vec{d}_i,C)\right)\log\left[1 - P(C \mid \vec{d}_i)\right]\right]$$

Logistic Regression Classification

$$\vec{\beta}_c^* = \arg\max_{\vec{\beta}_c} \prod_i \left[P(C|\vec{d}_i)^{\delta(\vec{d}_i,C)} \left[1-P(C|\vec{d}_i)\right]^{1-\delta(\vec{d}_i,C)} \right]$$

$$= \arg\max_{\vec{\beta}_c} \sum_i \left[\delta(\vec{d}_i,C)\log(P(C|\vec{d}_i)) + \left(1-\delta(\vec{d}_i,C)\right)\log\left[1-P(C|\vec{d}_i)\right]\right]$$

Two approaches:

- Newton's method, calculate first and second derivative to update the parameters; more efficient version (Quasi-Newton method) only needs first derivative (BFGS)
- Special type of expectation-maximization method, variational method, complicated to provide a lower bound of the original log-likelihood function

Logistic Regression Classification

$$\vec{\beta}_c^* = \arg\max_{\vec{\beta}_c} \prod_i \left[P(C|\vec{d}_i)^{\delta(\vec{d}_i,C)} \left[1-P(C|\vec{d}_i)\right]^{1-\delta(\vec{d}_i,C)} \right]$$

$$= \arg\max_{\vec{\beta}_c} \sum_i \left[\delta(\vec{d}_i,C)\log(P(C|\vec{d}_i)) + \left(1-\delta(\vec{d}_i,C)\right)\log\left[1-P(C|\vec{d}_i)\right]\right]$$

Newton's method

- Simple transformation, y_{ic}=1 iff the document in category c
  y_{ic}= -1 iff the document is not in category c

$$\vec{\beta}_c^* = \arg\max_{\vec{\beta}_c} \sum_i \left[\sigma\left(y_{ic}\left(\beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k) \right)\right)\right]$$

Logistic Regression Classification

Newton method

- Simple transformation, y_{ic}=1 iff the document in category c
  y_{ic}= -1 iff the document is not in category c

$$l = \sum_i \left[\sigma \left(y_{ic} \left(\beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k) \right) \right) \right]$$

Calculate derivative

$$\partial \frac{l}{\beta_k} = \sum_i \left[y_{ic} c_i(w_k) + y_{ic} c_i(w_k) \sigma \left(y_{ic} \left(\beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k) \right) \right) \right]$$

Quasi-Newton method can utilize the first derivative to estimate second derivative and update estimated parameters

Logistic Regression Classification

Training process

- Initiate model parameters: (e.g., set to 0)
- Iterate until model converges (log-likelihood does not change or model parameters do not change much)
  - Calculate log-likelihood by current model parameters

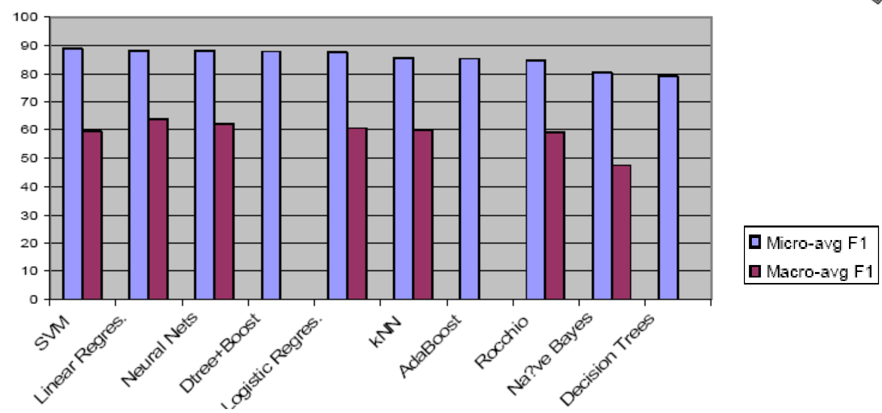
$$l = \sum_i \left[\sigma \left(y_{ic} \left(\beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k) \right) \right) \right]$$

  - Calculate first derivative

$$\partial \frac{l}{\beta_k} = \sum_i \left[y_{ic} c_i(w_k) + y_{ic} c_i(w_k) \sigma \left(y_{ic} \left(\beta_c(0) + \sum_k \beta_c(k) \times c_i(w_k) \right) \right) \right]$$

  - Send to Quasi-Newton method to update model parameters (e.g., BFGS method within Matlab)

Text Categorization: Evaluation



Performance of different algorithms on Reuters-21578 corpus: 90 categories, 7769 Training docs, 3019 test docs, (Yang, JIR 1999)

Logistic Regression Classification

Summary

- Discriminative model, only focus on how to distinguish documents in one category from documents in other categories
- It is often more effective than naïve bayes model due to the discriminative power
- Training process is more complicated than Naïve Bayes
- Can be extended to directly work with multi-class problems (i.e., predict among multiple categories)
- Used extremely common in many applications (i.e., predict a human in a picture; predict network intrusion…)