Text Categorization (I)

- Introduction to the task of text categorization
  - Manual vs. automatic text categorization
- Text categorization applications
- Evaluation of text categorization
- K nearest neighbor text categorization method
  - Lazy learning: no training
  - Local-continuity assumption: find your language by what language your neighbors speak
Problem: Weighting of Terms

• K-NN treats all terms equally
  – Frequent but unimportant terms may dominate

• Which terms are more important?
  – TF.IDF?
  – ...

• Solution – machine learning
  – We have training data

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: $(d_{1,1}, d_{1,2}, \ldots, d_{1,n_1}) \rightarrow \text{Language model } \theta_1$
    Category 2: $(d_{2,1}, d_{2,2}, \ldots, d_{2,n_2}) \rightarrow \text{Language model } \theta_2$
    .......
    Category C: $(d_{C,1}, d_{C,2}, \ldots, d_{C,n_C}) \rightarrow \text{Language model } \theta_C$
  – What is the language model? (Multinomial distribution)
  – How to estimate the language model for all the documents in one category?

• 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
    $$p(d_{c1}, \ldots, d_{cn_c} | \theta_c)$$
  – Each category $(c)$ has a prior distribution $P(c)$, which is the probably of choosing category $c$ BEFORE observing the content of a document
Naïve Bayes Classification

Maximum Likelihood Estimation:
• Find model parameters for a category that maximizes generation likelihood:
\[ \theta_c^* = \arg \max_{\theta_c} p(\tilde{d}_{c1}, ..., \tilde{d}_{cn} \mid \theta_c) \]

There are K words in vocabulary, \( w_1 \ldots w_K \)
Data: documents \( \tilde{d}_{c1}, ..., \tilde{d}_{cn} \)
For \( \tilde{d}_c \) with counts \( c_i(w_1), \ldots, c_i(w_K) \), and length \( |\tilde{d}_c| \)
Model: multinomial \( M \) with parameters \( \{p(w_k)\} \)
Likelihood: \( \Pr(\tilde{d}_{c1}, ..., \tilde{d}_{cn} \mid \theta) \)
\[ \theta_c^* = \arg \max_{\theta_c} p(\tilde{d}_{c1}, ..., \tilde{d}_{cn} \mid \theta_c) \]

Maximum Likelihood Estimation (MLE)

\[ p(\tilde{d}_{c1}, ..., \tilde{d}_{cn} \mid \theta) = \prod_{i=1}^{n_c} \left( c_i(w_1) \ldots c_i(w_K) \right)^{\frac{|\tilde{d}_{ci}|}{n_c}} \prod_{k=1}^{K} p_i(w_k)^{n_i} \prod_{k=1}^{K} p_i(w_k) \]

\[ l(\tilde{d}_{c1}, ..., \tilde{d}_{cn} \mid \theta) = \log p(\tilde{d}_{c1}, ..., \tilde{d}_{cn} \mid \theta) = \sum_{i=1}^{n_c} \sum_{k=1}^{K} c_i(w_k) \log p_k \]

\[ l'(\tilde{d}_{c1}, ..., \tilde{d}_{cn} \mid \theta) = \sum_{i=1}^{n_c} \sum_{k=1}^{K} c_i(w_k) \log \theta_k + \lambda(\sum_{k=1}^{K} p_k - 1) \]

\[ \frac{\partial l'}{\partial p_k} = \frac{\sum_{i=1}^{n_c} c_i(w_k)}{p_k} + \lambda = 0 \quad \Rightarrow \quad p_k = -\frac{\sum_{i=1}^{n_c} c_i(w_k)}{\lambda} \]

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

Since \( \sum_{k=1}^{K} p_k = 1 \), \( \lambda = -\sum_{i=1}^{n_c} \sum_{k=1}^{K} c_i(w_k) = -\sum_{i=1}^{n_c} |\tilde{d}_{ci}| \)
So, \( p_k = p(w_k) = \frac{\sum_{i=1}^{n_c} c_i(w_k)}{\sum_{i=1}^{n_c} |\tilde{d}_{ci}|} \)
Naïve Bayes Classification

- **MLE Estimator:** Normalization by simple counting
  - Train a language model for all the documents in one category
  
  \[
p(w | \theta_c^*) = \frac{\sum_{i=1}^{n_c} c_{ci}(w)}{\sum_{i=1}^{n_t} |d_{ci}|}
  \]

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

---

- **Smoothed Estimator:**
  - Laplace Smoothing
    
    \[
p(w | \theta_c^*) = \frac{1 + \sum_{i=1}^{n_c} c_{ci}(w)}{K + \sum_{i=1}^{n_t} |d_{ci}|}
    \]
  
  Number of Words in Vocabulary

  - Hierarchical Smoothing
    
    \[
p(w | \theta_c^*) = \lambda_1 P(w | \theta_c^*) + \lambda_2 P(w | \theta_{c^{apl}}^*) + ... + \lambda_m P(w | \theta_{c^{nou}}^*)
    \]

  - Dirichlet Smoothing
Naïve Bayes Classification

- Prediction:

\[ c^* = \arg \max_c p(c \mid d_i) \]
\[ = \arg \max_c \left\{ \frac{p(c)p(d_i \mid c)}{p(d_i)} \right\} \]
\[ = \arg \max_c \left\{ p(c)p(d_i \mid c) \right\} \text{ (Bayes Rule)} \]
\[ = \arg \max_c \left\{ p(c)[\prod_k p(w_k \mid c)^{c(w_k)}] \right\} \text{ (Multinomial Dist)} \]
\[ = \arg \max_c \left\{ \log(p(c)) + \sum_k c_i(w_k) \log p(w_k \mid c) \right\} \]

Plug in the estimator

Naïve Bayes Classification

- Example of Binary Classification

Two classes

\[ c^* = \arg \max_{l \in \{-, +\}} p(c_l \mid \tilde{d}_i) \rightarrow \frac{p(c_+ \mid \tilde{d}_i)}{p(c_- \mid \tilde{d}_i)} \]
\[ p(c_+ \mid \tilde{d}_i) \propto \prod_k [p(w_k \mid c_+)]^{c(w_k)} \frac{n_+}{n_+ + n_-} \]
\[ p(c_- \mid \tilde{d}_i) \propto \prod_k [p(w_k \mid c_-)]^{c(w_k)} \frac{n_-}{n_+ + n_-} \]
Naïve Bayes Classification

Example of Binary Classification

\[ c^* = \arg \max_{l \in \{+1,-1\}} p(c_l \mid \tilde{d}_i) \rightarrow \frac{p(c_+ \mid \tilde{d}_i)}{p(c_- \mid \tilde{d}_i)} \]

\[
\log \frac{p(c_+ \mid \tilde{d}_i)}{p(c_- \mid \tilde{d}_i)} = \log \left[ \frac{\prod_{k, i} [p(w_k \mid c_+)]^{c_i(w_k)}}{\prod_{k, i} [p(w_k \mid c_-)]^{c_i(w_k)}} \right] \frac{n_+}{n_+ + n_-} - \frac{n_-}{n_+ + n_-} \\
= \log \left( \frac{n_+}{n_-} \right) + \sum_k c_i(w_k) \log \left( \frac{p(w_k \mid c_+)}{p(w_k \mid c_-)} \right) \\
\log \frac{p(c_+ \mid \tilde{d})}{p(c_- \mid \tilde{d})} \propto b_0 + \sum_k c_i(w_k) \times \text{weight}(w_k)
\]

Naïve Bayes = Linear Classifier

● denotes +1

□ denotes -1

\[
\log \frac{p(c_+ \mid \tilde{d}_i)}{p(c_- \mid \tilde{d}_i)} \propto b_0 + \sum_k c_i(w_k) \times \text{weight}(w_k)
\]
Naïve Bayes Classification

- **Entropy**
  - Measuring the uncertainty
    - lower entropy means easier predictions
    \[ H(p) = -\sum_k p_k \log(p_k) \]
  - KL divergence ("relative entropy")
    - Distance between p and q
    \[ KL(p \parallel q) = \sum_k p_k \log\left(\frac{p_k}{q_k}\right) \]
    - Nonnegative, 0 when p and q are the same
  - Cross entropy
    - measuring the coding length based on q when true distribution is p
    \[ H(p \parallel q) = -\sum_k p_k \log(q_k) = H(p) + KL(p \parallel \tilde{q}) \]

- **Prediction:**
\[
c^* = \arg \max_c \left\{ \log(p(c)) + \sum_k c_i(w_k) \log p(w_k \mid c) \right\}
\]
\[
= \arg \max_c \left\{ \frac{\log(p(c))}{d} + \sum_k \frac{c_i(w_k)}{d} \log p(w_k \mid c) \right\} \quad (divide \ d)
\]
\[
= \arg \max_c \left\{ \frac{\log(p(c))}{d} + \sum_k p_i(w_k) \log p(w_k \mid c) \right\} \quad (Def \ of \ Cross \ Entropy)
\]
\[
= \arg \min_c \left\{ H(p_i \parallel \tilde{p}(c)) - \frac{\log(p(c))}{d} \right\}
\]

Cross Entropy
Naïve Bayes Classification

- **Prediction:**

\[
    c^* = \arg \min_c \left\{ H(p_i \| p(c)) - \frac{\log(p(c))}{|d|} \right\}
\]

- 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

- **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)