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

Material adapted from course created by Dr. Luo Si, now leading Alibaba research group
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

\[
\begin{align*}
\text{Category 1:} & \quad (\tilde{d}_{1,1}, \tilde{d}_{1,2}, \ldots, \tilde{d}_{1,n_1}) \rightarrow \text{Language model } \theta_1 \\
\text{Category 2:} & \quad (\tilde{d}_{2,1}, \tilde{d}_{2,2}, \ldots, \tilde{d}_{2,n_2}) \rightarrow \text{Language model } \theta_2 \\
& \quad \ldots \\
\text{Category C:} & \quad (\tilde{d}_{C,1}, \tilde{d}_{C,2}, \ldots, \tilde{d}_{C,n_C}) \rightarrow \text{Language model } \theta_C
\end{align*}
\]

– 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
    \[ p(\tilde{d}_{c1}, \ldots, \tilde{d}_{cn_c} | \theta_c) \]
  - Each category ($c$) has a prior distribution $P(c)$, which is the probably of choosing category $c$ BEFORE observing the content of a document

Maximum Likelihood Estimation:

- Find model parameters for a category that maximizes generation likelihood:
  \[ \theta_c^* = \arg \max_{\theta_c} p(\tilde{d}_{c1}, \ldots, \tilde{d}_{cn_c} | \theta_c) \]
  There are $K$ words in vocabulary, $w_1 \ldots w_K$
  Data: documents $\tilde{d}_{c1}, \ldots, \tilde{d}_{cn_c}$
  For $\tilde{d}_{ci}$ with counts $c_i(w_1), \ldots, c_i(w_k)$, and length $| \tilde{d}_{i} |$
  Model: multinomial $M$ with parameters $\{p(w_k)\}$
  Likelihood: $\Pr(\tilde{d}_{c1}, \ldots, \tilde{d}_{cn_c} | \theta)$
  \[ \theta_c^* = \arg \max_{\theta_c} p(\tilde{d}_{c1}, \ldots, \tilde{d}_{cn_c} | \theta_c) \]
Maximum Likelihood Estimation (MLE)

\[ p(\tilde{d}_1, \ldots, \tilde{d}_{cn} \mid \theta) = \prod_{i=1}^{n_c} \left( \prod_{j=1}^{k} \frac{1}{p_{\theta}^{c_j}(w_i)} \right) \prod_{k=1}^{k} p_{\theta}^{c_j}(w_i) \]

\[ l(\tilde{d}_1, \ldots, \tilde{d}_{cn} \mid \theta) = \log p(\tilde{d}_1, \ldots, \tilde{d}_{cn} \mid \theta) = \sum_{i=1}^{n_c} \sum_{k} c_{ci}(w_i) \log p_k \]

\[ l(\tilde{d}_1, \ldots, \tilde{d}_{cn} \mid \theta) = \sum_{i=1}^{n_c} \sum_{k} c_{ci}(w_i) \log \theta_k + \lambda (\sum_{k} p_k - 1) \]

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

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

Since \( \sum_{k} p_k = 1 \), \( \lambda = -\sum_{k} \sum_{i=1}^{n_c} c_{ci}(w_k) = -\sum_{i=1}^{n_c} |\tilde{d}_{ci}| \)

So, \( p_k = p(w_k) = \frac{\sum_{i=1}^{n_c} c_{ci}(w_k)}{\sum_{i=1}^{n_c} |\tilde{d}_{ci}|} \)

Naïve Bayes Classification

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

\[ p(w \mid \theta^*_c) = \frac{\sum_{i=1}^{n_c} c_{ci}(w)}{\sum_{i=1}^{n_c} |\tilde{d}_{ci}|} \]

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

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

- **Smoothed Estimator:**
  - Laplace Smoothing
    \[
    p(w | \theta^*_c) = \frac{1 + \sum_{i=1}^{n_c} c_{ci}(w)}{K + \sum_{i=1}^{n_i} d_{ci}}
    \]
    Number of Words in Vocabulary
  - Hierarchical Smoothing
    \[
    p(w | \theta^*_c) = \lambda_1 P(w | \theta^*_c) + \lambda_2 P(w | \theta^*_c^{ap1}) + \ldots + \lambda_m P(w | \theta^*_c^{ma})
    \]
  - Dirichlet Smoothing

Naïve Bayes Classification

- **Prediction:**
  \[
  c^* = \arg \max_c p(c | \vec{d}_i) = \arg \max_c \left\{ \frac{p(c) p(d_i | c)}{p(d_i)} \right\} \\
  = \arg \max_c \left\{ p(c) p(d_i | c) \right\} \quad (\text{Bayes Rule}) \\
  = \arg \max_c \left\{ p(c) \prod_k p(w_k | c)^{c_{ki}(w_k)} \right\} \quad (\text{Multinomial Dist}) \\
  = \arg \max_c \left\{ \log(p(c)) + \sum_k c_{ki}(w_k) \log p(w_k | c) \right\}
  \]
  Plug in the estimator
Naïve Bayes Classification

- Example of Binary Classification

  Two classes

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

  \[ p(c_+ | \tilde{d}_i) \propto \prod_k [p(w_k | c_+)] c_i^{(w_k)} \frac{n_+}{n_+ + n_-} \]

  \[ p(c_- | \tilde{d}_i) \propto \prod_k [p(w_k | c_-)] c_i^{(w_k)} \frac{n_-}{n_+ + n_-} \]

Naïve Bayes Classification

- Example of Binary Classification

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

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

  \[ = \log \left( \frac{n_+}{n_-} \right) + \sum_k c_i(w_k) \log \left( \frac{p(w_k | c_+)}{p(w_k | c_-)} \right) \]

  \[ \log \frac{p(c_+ | \tilde{d})}{p(c_- | \tilde{d})} \propto b_0 + \sum_k c_i(w_k) \times \text{weight}(w_k) \]
Naïve Bayes = Linear Classifier

\[
\begin{align*}
\log \frac{p(c_+ | \vec{d}_i)}{p(c_- | \vec{d}_i)} &= b_0 + \sum_k c_i(w_k) \times \text{weight}(w_k)
\end{align*}
\]

- **denotes +1**
- **denotes -1**

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)