Naïve Bayes

LING 572

Fei Xia

Naïve Bayes Model

- Naïve Assumption
- Training & Decoding
- Variants
- Issues

ML Questions

- Modeling:
  - What is the model structure?
  - Why is it called Naïve Bayes?
  - What assumption does it make?
  - What type of parameters are learned?
  - How many parameters?

- Training:
  - How are model parameters learned from data?

- Decoding:
  - How is model used to classify new data?

Probabilistic Model

- Given an instance x with features $f_1 \ldots f_k$,
  - Find the class with highest probability
  - Formally, $x = (f_1, f_2, \ldots, f_k)$
    - Find $c^* = \text{argmax}_c \, P(c \mid x)$
  - Applying Bayes' Rule:
    - $c^* = \text{argmax}_c \, P(x \mid c) \, P(c) / P(x)$
  - Maximizing:
    - $c^* = \text{argmax}_c \, P(x \mid c) \, P(c)$

Naïve Bayes Model

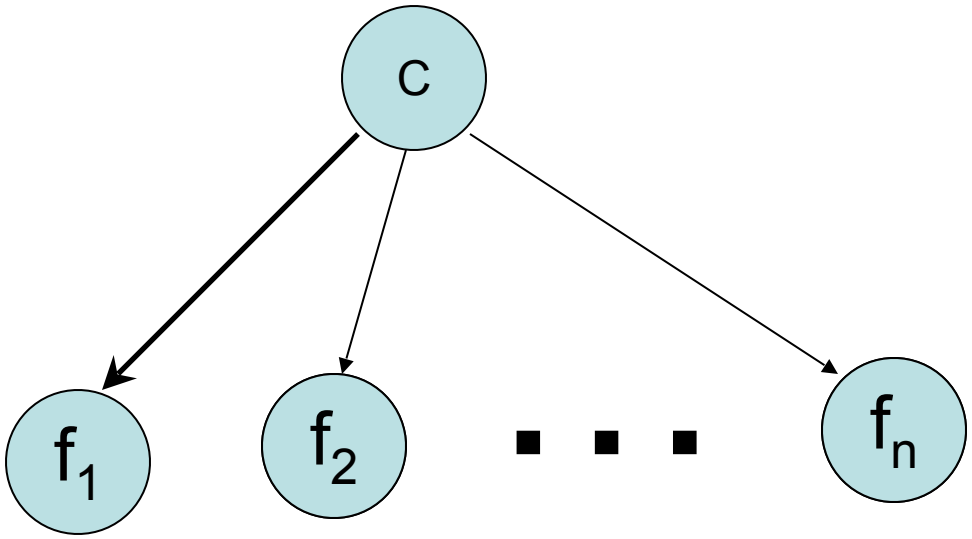
- So far just Bayes' Rule
- Key question: How do we handle/combine features?
- Consider just P(x | c)
  - $P(x \mid c) = P(f_1, f_2, \ldots, f_k \mid c)$

$$= \prod_j P(f_j | c,\ f_1^{j-1})$$

  - Can we simplify? (Remember ngrams)
  - Assume conditional independence

$$= \prod_j P(f_j | c)$$

Naïve Bayes Model



Assumption: each f_i is conditionally independent from f_j given C.

Model parameters

- Choose
  $c^* = \arg\max_c P(c \mid x)$
  $= \arg\max_c P(c)\, P(x \mid c) / P(x)$
  $= \arg\max_c P(c)\, P(x \mid c)$
  $= \arg\max_c P(c) \prod_k P(f_k \mid c)$

- Two types of model parameters:
  - Class prior: P(c)
  - Conditional probability: $P(f_k \mid c)$

- The number of model parameters:
  - |priors|+|conditional probabilities|
  - |C| + |F|*|C|
  - |C|+|V|*|C|, if features are words in vocabulary V

  |C| is the number of classes, |F| is the number of features, |V| is the number of features.

Training stage: estimating parameters θ

- Maximum likelihood estimation (ML):

  $\theta^* = \arg\max_\theta P(\text{training Data} \mid \theta)$

- Class prior: $P(c_i) = \dfrac{Count(c_i)}{\sum_j Count(c_j)}$

- Conditional prob: $P(f_j \mid c_i) = \dfrac{Count(f_j, c_i)}{Count(c_i)}$

Training

- MLE issues?
  - What's the probability of a feature not seen with a c_i?
    - 0?
    - What happens then?
- Solutions?
  - Smoothing
    - Laplace smoothing, Good-Turing, Witten-Bell
    - Interpolation, Backoff….

What are Zero Counts?

- Some of those zeros are really zeros...
  - Things that really can't or shouldn't happen.

- On the other hand, some of them are just rare events.
  - If the training corpus had been a little bigger, they would have had a count (probably a count of 1!).

- Zipf's Law (long tail phenomenon):
  - A small number of events occur with high frequency
  - A large number of events occur with low frequency
  - You can quickly collect statistics on the high frequency events
  - You might have to wait an arbitrarily long time to get valid statistics on low frequency events

Laplace Smoothing (add-one smoothing)

- Pretend you saw outcome one more than you actually did.
- Suppose X has K possible outcomes, and the counts for them are n_1, …, n_K , which sum to N.
  - Without smoothing: $P(X=i) = n_i / N$
  - With Laplace smoothing: $P(X=i) = (n_i + 1) / (N+K)$

Testing stage

- MAP (maximum a posteriori) decision rule:

- Given our model and an instance $x=< f_1, .., f_d>$

classify (x)

= classify (f_1, .., f_d)

= argmax_c P(c|x)

= argmax_c P(x|c) P(c)

= argmax_c P(c) $\prod_k$ P(f_k | c)

Naïve Bayes for the text classification task

Features

- Features: bag of words (word order information is lost)

- Number of feature types: 1
- Number of features: |V|
- Features: w_t, where $t \in \{1, 2, \ldots, |V|\}$

Issues

- Is w_t a binary feature?
- Are absent features used for calculating $P(d_i|c_j)$

Two Naive Bayes Models (McCallum and Nigram, 1998)

- Multivariate Bernoulli event model

  (a.k.a. binary independence model)
  - All features are binary: the number of times a feature occurs in an instance is ignored.
  - When calculating p(d | c), all features are used, including the absent features.

- Multinomial event model: “unigram LM”

Multivariate Bernoulli event model

Bernoulli distribution

- Bernoulli trial: a statistical experiment having exactly two mutually exclusive outcomes, each with a constant probability of occurrence:
  - Ex: toss a coin

- Bernoulli distribution: has exactly two mutually exclusive outcomes: $P(X=1)=p$ and $P(X=0)=1-p$.

Multivariate Bernoulli Model

- Each document:
  - Result of |V| independent Bernoulli experiments
  - i.e., for each word in the vocabulary: does this word appear in the document?

- Another way to look at this: (to be consistent with the general NB model)
  - Each word in the voc corresponds to two features:

$$w_k \text{ and } \bar{w}_k$$

  - In any document, either w_k or $\bar{w}_k$ is present; that is, it is always the case that exactly |V| features will be present in any document.

Training stage

ML estimate:

$$P(w_t|c_i) = \frac{Cnt(w_t,c_i)}{Cnt(c_i)}$$

$$P(c_i) = \frac{Cnt(c_i)}{\sum_i Cnt(c_i)}$$

With add-one smoothing:

$$P(w_t|c_i) = \frac{1+Cnt(w_t,c_i)}{2+Cnt(c_i)}$$

$$P(c_i) = \frac{1+Cnt(c_i)}{|C|+\sum_i Cnt(c_i)}$$

Notation used in the paper

$$P(w_t \mid c_j) = \frac{1 + \text{count}(w_t, c_j)}{2 + \text{count}(c_j)}$$

$$B_{it} = \begin{cases} 1 & w_t \text{ appears in } d_i \\ 0 & \text{otherwise} \end{cases}$$

$$P(c_j \mid d_i) = \begin{cases} 1 & d_i \text{ has label } c_j \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{\theta}_{w_t|c_j} = P(w_t \mid c_j; \theta) = \frac{1 + \sum_{i=1}^{|D|} B_{it} P(c_j \mid d_i)}{2 + \sum_{i=1}^{|D|} P(c_j \mid d_i)}$$

Testing stage

$$classify(d_i) = argmax_c P(c)P(d_i|c)$$

$$P(d_i \mid c)$$

$$= \prod_k P(f_k \mid c)$$

Each word $w_k \in V$ corresponds to two features: w_k and $\bar{w}_k$.

Use $P(w_k \mid c)$ when $w_k \in d_i$, and use $P(\bar{w}_k \mid c)$ when $w_k \notin d_i$

$$= \prod_{w_k \in d_i} P(w_k \mid c) \prod_{w_k \notin d_i} P(\bar{w}_k \mid c)$$

$$P(\bar{w}_k \mid c) = 1 - P(w_k \mid c)$$

$$= \prod_{w_k \in d_i} P(w_k \mid c) \prod_{w_k \notin d_i} (1 - P(w_k \mid c))$$

$$P(d_i \mid c) = \prod_k P(f_k \mid c)$$

$$= \prod_{w_k \in d_i} P(w_k \mid c) \prod_{w_k \notin d_i} P(\bar{w}_k \mid c) = \prod_{w_k \in d_i} P(w_k \mid c) \prod_{w_k \notin d_i} (1 - P(w_k \mid c))$$

$$= \prod_{w_k \in d_i} P(w_k \mid c) \frac{\prod_{w_k \in V} (1 - P(w_k \mid c))}{\prod_{w_k \in d_i} (1 - P(w_k \mid c))}$$

$$= \left(\frac{\prod_{w_k \in d_i} P(w_k \mid c)}{\prod_{w_k \in d_i} (1 - P(w_k \mid c))}\right) \prod_{w_k \in V} (1 - P(w_k \mid c))$$

$$= \prod_{w_k \in d_i} \frac{P(w_k \mid c)}{(1 - P(w_k \mid c))} \prod_{w_k \in V} (1 - P(w_k \mid c))$$

$$= Z_c * \prod_{w_k \in d_i} \frac{P(w_k \mid c)}{(1 - P(w_k \mid c))} \qquad \text{where } Z_c = \prod_{w_k \in V} (1 - P(w_k \mid c))$$

Since Z_c is a constant w.r.t. d_i (but it depends on c), Z_c can be calculated beforehand.

Multinomial event model

Multinomial distribution

- Possible outcomes = $\{w_1, w_2, \ldots, w_{|v|}\}$
- A trial for each word position:
  $P(\text{CurWord}=w_i)=p_i$ and $\sum_i p_i = 1$
- Let X_i be the number of times that the word w_i is observed in the document.

$$P(X_1 = x_1, ..., X_v = x_v) = p_1^{x_1} ... p_v^{x_v} \frac{n!}{x_1! ... x_v!}$$

$$= n! \prod_k \frac{p_k^{x_k}}{x_k!}$$

An example

- Suppose
  - the voc, V, contains only three words: a, b, and c.
  - a document, d_i, contains only 2 word tokens
  - For each position, P(w=a)=p1, P(w=b)=p2 and P(w=c)=p3.

- What is the prob that we see “a” once and “b” once in d_i?

An example (cont)

- 9 possible sequences: aa, ab, ac, ba, bb, bc, ca, cb, cc.
- The number of sequences with one “a” and one “b” (ab and ba): n!/(x_1!...x_v!)
- The prob of the sequence “ab” is p_1*$p_{2,}$

  so is the prob of the sequence “ba”.

- So the prob of seeing “a” once and “b” once is: $$n! \prod_k \frac{p_k^{x_k}}{x_k!} = 2p_1p_2$$

Multinomial event model

- A document is seen as a sequence of word events, drawn from the vocabulary V.
- N_{it}: the number of times that w_t appears in d_i
- Modeling: multinomial distribution:

$$P(d_i \mid c_j) = P(\mid d_i \mid) \mid d_i \mid ! \prod_{t=1}^{\mid V \mid} \frac{P(w_t \mid c_j)^{N_{it}}}{N_{it}!}$$

Training stage for multinomial model

$$P(c_j \mid d_i) = \begin{cases} 1 & d_i \text{ has label } c_j \\ 0 & \text{otherwise} \end{cases}$$

$$P(w_t \mid c_j) = \frac{1 + \sum_{i=1}^{|D|} N_{it} P(c_j \mid d_i)}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N_{is} P(c_j \mid d_i)}$$

Testing stage

$$\text{classify}(d_i) = \arg\max_c P(c)P(d_i \mid c)$$

$$P(d_i \mid c) = P(\mid d_i \mid)\mid d_i \mid ! \prod_{k=1}^{|V|} \frac{P(w_k \mid c)^{N_{ik}}}{N_{ik}!}$$

$$\text{classify}(d_i) = \arg\max_c P(c) \prod_{k=1}^{|V|} P(w_k \mid c)^{N_{ik}}$$

Two models

- Multi-variate Bernoulli event model: treat features as binary; each trial corresponds to a word in the voc.

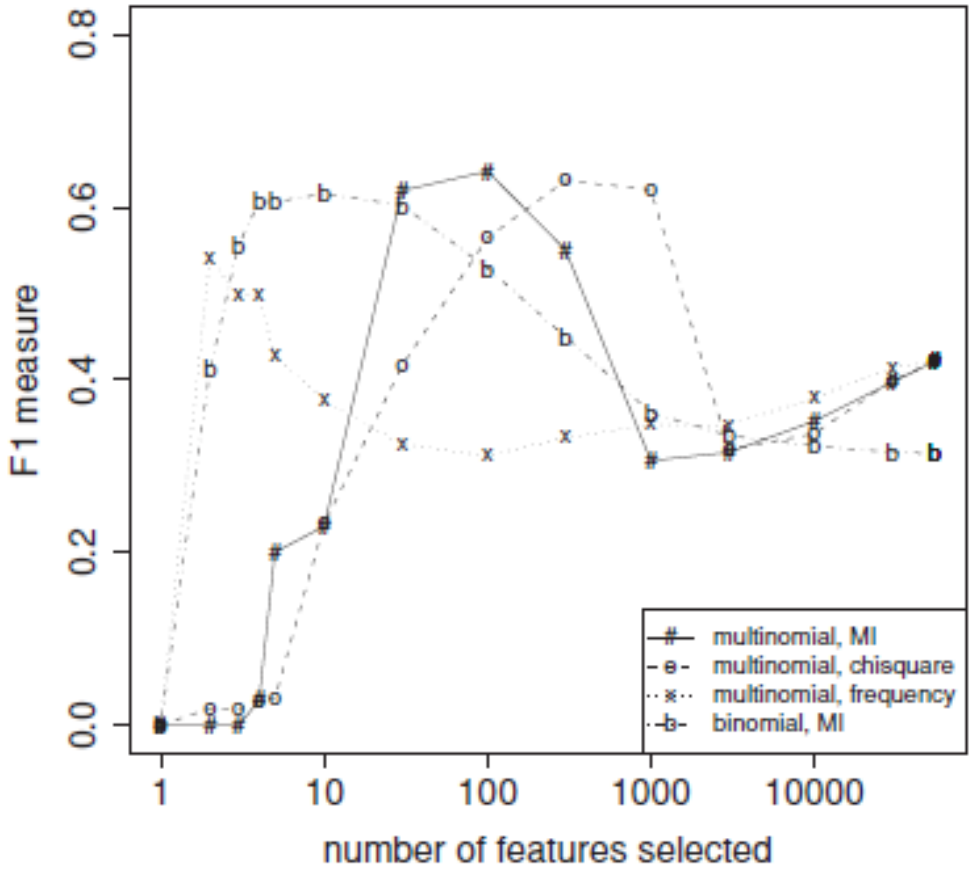
- Multinomial event model: treat features as non-binary; each trial corresponds to a word position in the document.

Which model is better?

- (McCallum and Nigram, 1998): Multinomial event model usually beats the Bernoulli event model

- Chapter 13 in (Manning et al., 2008): The Bermoulli model
  - is particularly robust w.r.t. concept shift
  - is more sensitive to noisy features (requiring feature selection)
  - peaks early for feature selection (see fig)
  - works well for shorter documents

From (Manning et al., 2008)



► **Figure 13.8** Effect of feature set size on accuracy for multinomial and Bernoulli models.

Two models (cont)

	Multivariate Bernoulli	Multinomial
Features	Binary: present or absent	Real-valued: the occurrence
Each trial	Each word in the voc	Each word position in the doc
$P(c_i)$	$\frac{1 + \text{count}(c_i)}{\lvert C \rvert + \sum_j \text{count}(c_j)}$	$\frac{1 + \text{count}(c_i)}{\lvert C \rvert + \sum_j \text{count}(c_j)}$
$P(w_t \mid c_j)$	$\frac{1 + \text{count}(w_t, c_j)}{2 + \text{count}(c_j)}$	$\frac{1 + \sum_{i=1}^{\lvert D \rvert} N_{it} P(c_j \mid d_i)}{\lvert V \rvert + \sum_{s=1}^{\lvert V \rvert} \sum_{i=1}^{\lvert D \rvert} N_{is} P(c_j \mid d_i)}$
classify(d_i)	$\arg\max_c P(c) \prod_{w_k \in d_i} P(w_k \mid c) \prod_{w_k \notin d_i} (1 - P(w_k \mid c))$	$\arg\max_c P(c) \prod_{k=1}^{\lvert V \rvert} P(w_k \mid c)^{N_{ik}}$

Summary of Naïve Bayes

- It makes a strong independence assumption: all the features are conditionally independent given the class.
- It generally works well despite the strong assumption. Why?
- “Correct estimation implies accurate prediction, but accurate prediction does not imply correct estimation.”
- Both training and testing are simple and fast.

Summary of Naïve Bayes (cont)

- Strengths:
  - Simplicity (conceptual)
  - Efficiency at training
  - Efficiency at testing time
  - Handling multi-class
  - Scalability
  - Output topN
- Weaknesses:
  - Theoretical validity: the independency assumption
  - Prediction accuracy: might not as good as MaxEnt etc.