Lecture 14: Probabilistic Classification

William Webber (william@williamwebber.com)

COMP90042, 2014, Semester 1, Lecture 14

What we'll learn in this lecture

- Probabilistic classification
- Naive Bayes classification using
  - Multinomial method
  - Bernoulli method
- An (ad-hoc) fix to NB classification
- The general log-linear model
- The concept of maximum entropy classifiers

Probabilistic IR, Classification

In probabilistic IR, we ask:

$$P(r|d,q) \tag{1}$$

In LM IR, we ask:

$$P(d|q) \tag{2}$$

In probabilistic classification, we ask:

$$P(c|d) \tag{3}$$

that is: what probability that document d belongs to class c.

Bayes rule

Invoke Bayes' rule again:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)} \tag{4}$$

Now, we estimate $P(c|d)$, and classify d into class c' s.t.:

$$c' = \underset{c \in \mathcal{C}}{\text{argmax}}\, \widehat{P}(c|d) \tag{5}$$

$P(d)$ is independent of c, only scales $P(c|d)$. So:

$$P(c|d) \propto P(c)P(d|c) \tag{6}$$

which is all we need for $\text{argmax}_{c \in \mathcal{C}}$

Naive Bayes: naive assumptions

Represent document d by list of words $\{w_1, \ldots, w_n\}$. Then:

$$P(c|w_1, \ldots, w_n) \propto P(c)P(w_1, \ldots, w_n|c) \tag{7}$$

Assume (naively) that each w_i is cond. independent given c:

$$P(c|w_1, \ldots, w_n) \propto P(c) \prod_i P(w_i|c) \tag{8}$$

Note the formal relation to unigram language models:

$$P(d|q_1, \ldots, q_n) \propto P(d) \prod_i P(q_i|M_d) \tag{9}$$

Naive Bayes estimation: MLE for $P(c)$

$$\widehat{P}(c|w_1, \ldots, w_n) \propto \widehat{P}(c)\widehat{P}(w_1, \ldots, w_n|c)$$

We need estimates $\widehat{P}(c)$, $\widehat{P}(w_i|c)$
MLE of $P(c)$ given training data:

$$\widehat{P}(c) = \frac{N_c}{N} \tag{10}$$

where:

- N number of training documents
- N_c number of train docs in class c

- Our "prior" on class membership
- (In practice, gets overweighted by term contributions)

Naive Bayes estimation: MLE for $\widehat{P}(w_i|c)$

$$\widehat{P}(c|w_1,\ldots,w_n) \propto \widehat{P}(c)\widehat{P}(w_1,\ldots,w_n|c) \tag{11}$$

Similarly:

$$\widehat{P}_{\text{mle}}(w_i|c) = \frac{F_{ci}}{F_c} \tag{12}$$

where:

- F_{ci} total occurrences of term t_i in documents of class c training set (note: t_i is the term that word w_i is)
- F_c number of words in documents of class c training set

Naive Bayes: smoothing

If term t_i never appears in class c in training data, then:

$$\widehat{P}(w_i|c) = 0 \tag{13}$$

- documents containing t have 0 probability of being in class c.
- Overfitting to training data.

To avoid, (Laplace) smooth the counts:

$$\widehat{P}(w_i|c) = \frac{F_{ci} + 1}{F_c + |V|} \tag{14}$$

In Bayesian sense:

- $1/|V|$ is our prior
  - each term equally likely to appear given class
- and F_{ci}/F_c is our update
  - evidence we see from [sampled] training documents

Multinomial model

Instead of $P(w|c)$, treat d as feature vector $f = \{f_1, \ldots, f_n\}$

- n is number of terms in vocabulary
- $f_i = f_{d,t_i}$

Then (writing $p_i = P(w_i|c)$):

$$P(f|c) = \frac{(\sum_i f_i)!}{\prod_i f_i!} \prod_i p_i^{f_i} \tag{15}$$

- Multinomial prob. dist. for drawing $|d|$ items with given prob.
- Easy to show that:

$$P(f|c) \propto P(w|c) \tag{16}$$

Bernoulli Naive Bayes

Alternative model, treat d as feature vector $e = \{e_1, \ldots, e_n\}$.

- $e_i = 1$ if $t_i \in d$, 0 otherwise

Then:

$$P(e|c) = \prod_{i=1}^{n} (e_i P(e_i|c) + (1 - e_i)(1 - P(e_i|c))) \qquad (17)$$

where:

$\widehat{P}(e_i|c)$ is fraction of docs containing t_i

- Bernoulli model
- Explicitly accounts for missing terms
  - which multinomial does not
- Equivalent to Binary Independence Model in prob. IR

Performance

- Naive Bayes makes poor estimates of absolute probabilities.
- In particular, tends to be overconfident
- . . . due to dependence between terms
- But class predictions more reasonable

Experiment

- Binary classification of GCAT class
  - Train 1000, test 1000
  - Smoothed multinomial NB classifier
- Estimation accuracy:
  - Of 256 documents classifier is $\geq$ 99.9% sure are GCAT
  - . . . only 92.9% actually are
- Prediction accuracy:
  - 93% accuracy, 89% recall

Assumption of independence

- NB assumes evidence of terms independent, given class
- However, terms tend to co-occur (be dependent)
- Dependence not weakened (possibly reinforced) by class
- For instance:

$$P(\text{“dow jones”}|\text{FIN}) \neq P(\text{“dow”}|\text{FIN}) \times P(\text{“jones”}|\text{FIN}) \quad (18)$$

but:

$$\widehat{P}(\text{“dow jones”}|\text{FIN}) = \widehat{P}(\text{“dow”}|\text{FIN}) \times \widehat{P}(\text{“jones”}|\text{FIN}) \quad (19)$$

- This exaggerates probability estimates
- Can also harm class predictions if:
  - One class has strong co-dependence
  - Other classes do not

Comparative performance

					SVM (poly) $d=$					SVM (rbf) $\gamma=$			
	Bayes	Rocchio	C4.5	k-NN	1	2	3	4	5	0.6	0.8	1.0	1.2
earn	95.9	96.1	96.1	97.3	98.2	98.4	**98.5**	98.4	98.3	**98.5**	98.5	98.4	98.3
acq	91.5	92.1	85.3	92.0	92.6	94.6	**95.2**	95.2	95.3	95.0	95.3	95.3	**95.4**
money-fx	62.9	67.6	69.4	78.2	66.9	72.5	75.4	74.9	**76.2**	74.0	75.4	**76.3**	75.9
grain	72.5	79.5	89.1	82.2	91.3	93.1	**92.4**	91.3	89.9	**93.1**	91.9	91.9	90.6
crude	81.0	81.5	75.5	85.7	86.0	87.3	88.6	**88.9**	87.8	**88.9**	89.0	88.9	88.2
trade	50.0	77.4	59.2	77.4	69.2	75.5	76.6	77.3	**77.1**	76.9	78.0	**77.8**	76.8
interest	58.0	72.5	49.1	74.0	69.8	63.3	67.9	73.1	**76.2**	74.4	75.0	**76.2**	76.1
ship	78.7	83.1	80.9	79.2	82.0	85.4	86.0	**86.5**	86.0	**85.4**	86.5	87.6	87.1
wheat	60.6	79.4	85.5	76.6	83.1	84.5	85.2	**85.9**	83.8	**85.2**	85.9	85.9	85.9
corn	47.3	62.2	87.7	77.9	86.0	86.5	85.3	**85.7**	83.9	**85.1**	85.7	85.7	84.5
microavg.	**72.0**	**79.9**	**79.4**	**82.3**	84.2	85.1	85.9	86.2	85.9	86.4	86.5	86.3	86.2
					combined: **86.0**					combined: **86.4**			

Figure : Joachims, "Text Categorization with SVM", 1998

- Generally worse than Rocchio, *k*-NN, SVM
- Tendency to "explode": perform very poorly on certain topics
- Easy to implement, quick to train (single pass)
- Bernoulli method more robust, but requires feature selection

NB as log linear model

Consider again (feature-vector) NB calculation:

$$P(c|f_1, \ldots, f_n) \propto P(c) \prod_i P(f_i|c) \tag{20}$$

Under the multinomial model:

$$P(f_i|c) = f_i P(w_i|c) \tag{21}$$

Exchanging (21) in (20), and taking logs, gives:

$$\log P(c|f_1, \ldots, f_n) \propto \log P(c) + \sum_{i=1}^{n} f_i \log P(w_i|c) \tag{22}$$

NB as log linear model (cont.)

$$\log P(c|f_1, \ldots, f_n) \propto \log P(c) + \sum_{i=1}^{n} f_i \log P(w_i|c) \tag{23}$$

This has form:

$$\log \ell(c|f_1, \ldots, f_n) = b_c + \mathbf{w_c}^{\mathbf{T}} f_i \tag{24}$$

where:

$$\begin{aligned} b_c &= \log P(c) = \log \frac{N_c}{N} \\ w_{ci} &= \log P(w_i|c) = \log \frac{F_{ci}+1}{F_c+|V|} \end{aligned} \tag{25}$$

This is a *log linear model*:

- log of response is linear sum of weighted features plus bias
- (bias b quickly outweighed by features)

"Fixing" NB for text

$$\log \ell(c|f_1, \ldots, f_n) = b + \mathbf{w_c}^{\mathbf{T}} f_i \tag{26}$$

Rennie et al. (2003) argue that (multinomial) NB for text classification has three problems:

- *Skewed data bias:* class underrepresented in training data will be biased against
- *Weight magnitude errors:* some classes violate independence more than others
- *Poor text modelling:* text frequency violates multinomial assumptions

Fixing NB: undoing skew

Rennie et al. alleviate skew by changing weight from:

$$w_{ci} = \log \frac{F_{ci} + 1}{F_c + |V|} \tag{27}$$

to:

$$w'_{ci} = -\log \frac{F_{\bar{c}i} + 1}{F_{\bar{c}} + |V|} \tag{28}$$

where $F_{\bar{c}i}$ is number of occurences of term t_i in all classes *except* c. (I suspect this only works for multi-class classification.)

Fixing NB: undoing weight magnitude errors

Rennie et al. alleviate weight magnitude errors by normalizing feature weights:

$$w''_{ci} = \frac{w'_{ci}}{\sum_k |w'_{ck}|} \tag{29}$$

This adjustment means that the sum of weights for each class is the same.

- Reduces strong term dependence (co-occurrence) as side-effect

Fixing NB: modelling text better

Rennie et al. improve text modelling by (ta-da!) applying document-length normalized TF * IDF transform:

$$
\begin{aligned}
f_i' &= \log(1 + f_i) \\
f_i'' &= f_i' \log \frac{N}{\mathrm{ft}_i} \\
f_i''' &= \frac{f_i''}{\sqrt{\sum_k f_i''^2}}
\end{aligned}
$$

Fixing NB: the outcome

With these transformations, Rennie et al. report similar effectiveness for transformed NB as for SVM:

Collection	NB	TNB	SVM
Industry sector	0.582	0.923	0.934
20 Newsgroups	0.848	0.861	0.862
Reuters (micro)	0.739	0.844	0.887
Reuters (macro)	0.270	0.647	0.694

But we've departed from assumptions of model . . .

Loglinear models

Consider again the multiplicative model:

$$P(c|\vec{f}) = Z^{-1} \prod_{i=1}^{n} \alpha_i^{f_i} \tag{30}$$

and its log-linear equivalent

$$\log P(c|\vec{f}) = -\log Z + \sum_{i=1}^{n} f_i \log \alpha_i \tag{31}$$

where:

- Z is a normalizing value (to make this a probability)
- α_i is the *weight* of feature α_i

Loglinear models and Maximum Entropy

$$\log P(c|\vec{f}) = -\log Z + \sum_{i=1}^{n} f_i \log \alpha_i \tag{32}$$

- In NB, α_i calculated from empirical frequencies
- However, we are free to choose α_i to suit other criteria
- One such criterion is that of *maximum entropy*

Entropy

If X is discrete RV with distribution:

$$\Pr(X = x_k) = p_k \;\; k = 1, 2, \ldots, n \tag{33}$$

then entropy of X is:

$$H(X) = -\sum p_k \log p_k \tag{34}$$

- Entropy is 0 if there is an i s.t. $p_i = 1$
  - We are certain which even will occur
- Entropy unconditionally maximized where $p_i = p_j$ for all i, j
  - We have no idea which event will occur
- Given data, the *maximum entropy* model is the one that maximizes entropy, given data
  - The maxent model "best explains" the data

Maxent and loglinear models

$$\log P(c|\vec{f}) = -\log Z + \sum_{i=1}^{n} f_i \log \alpha_i \tag{35}$$

For loglinear models:

- Constraints are observed frequency of features (as they co-occur with classes)
- Maxent model maximizes entropy of $P(c|\vec{f})$
- . . . by selecting correct α_i
- There is a unique solution, which can be found iteratively
  - See Chapter 16 of Manning and Schutze, *Foundations of NLP*, 1999, for details

Maxent and logistic regression

- Maxent classifiers a relatively recent idea (last 20 years or so)
- Turn out to be equivalent to a much older (c. 70 years) idea:
- That of *logistic regression*, which we discuss next week

Looking back and forward

Back

- Naive Binomial a simple probabilistic classification model
- Assume independence between features (terms)
- $P(d|c) = \prod P(t|c)$
- Multinomial method analogous to unigram language models in IR
- Bernoulli method analogous to BIM

Looking back and forward

Back

- NB gives poor probability estimates
- Reasonable class predictions
- Fragile to classes with high term dependence
- Can be "fixed" to give SVM-competitive accuracy
- NB a case of general log linear models
- Maxent another way of fitting log linear models

Looking back and forward

Forward

- Logistic regression another form of log-linear model
- Try to fit transformed linear model to empirical data
- Equivalent to maxent but old-school (hence preferable)

Further reading

- Chapter 13, "Text classification and Naive Bayes"[1], of Manning, Raghavan, and Schutze, *Introduction to Information Retrieval*, CUP, 2009.
- Rennie, Shih, Teevan, and Karger, "Tackling the Poor Assumptions of Naive Bayes Text Classifiers", ICML 2003 (fixing Naive Bayes)
- Chapter 16, Manning and Schutze, *Foundations of NLP*, 1999 (maximum entropy classifiers)

[1] http://nlp.stanford.edu/IR-book/pdf/12lmodel.pdf