

Lecture 9: 21 February, 2022

Madhavan Mukund

<https://www.cmi.ac.in/~madhavan>

Data Mining and Machine Learning
January–May 2022

Bayesian classifiers

- As before
 - Attributes $\{A_1, A_2, \dots, A_k\}$ and
 - Classes $C = \{c_1, c_2, \dots, c_\ell\}$
- Each class c_i defines a probabilistic model for attributes
 - $Pr(A_1 = a_1, \dots, A_k = a_k \mid C = c_i)$
- Given a data item $d = (a_1, a_2, \dots, a_k)$, identify the best class c for d
- Maximize $Pr(C = c_i \mid A_1 = a_1, \dots, A_k = a_k)$

Generative models

- To use probabilities, need to describe how data is randomly generated
 - Generative model
- Typically, assume a random instance is created as follows
 - Choose a class c_j with probability $Pr(c_j)$
 - Choose attributes a_1, \dots, a_k with probability $Pr(a_1, \dots, a_k | c_j)$
- Generative model has associated parameters $\theta = (\theta_1, \dots, \theta_m)$
 - Each class probability $Pr(c_j)$ is a parameter
 - Each conditional probability $Pr(a_1, \dots, a_k | c_j)$ is a parameter
- We need to estimate these parameters

Maximum Likelihood Estimators

- Our goal is to estimate parameters (probabilities) $\theta = (\theta_1, \dots, \theta_m)$
- Law of large numbers allows us to estimate probabilities by counting frequencies
- Example: Tossing a biased coin, single parameter $\theta = \text{Pr}(\text{heads})$
 - N coin tosses, H heads and T tails
 - Why is $\hat{\theta} = H/N$ the best estimate?
- Likelihood
 - Actual coin toss sequence is $\tau = t_1 t_2 \dots t_N$
 - Given an estimate of θ , compute $\text{Pr}(\tau | \theta)$ — likelihood $L(\theta)$
- $\hat{\theta} = H/N$ maximizes this likelihood — $\arg \max_{\theta} L(\theta) = \hat{\theta} = H/N$
 - Maximum Likelihood Estimator (MLE)

Bayesian classification

- Maximize $Pr(C = c_i | A_1 = a_1, \dots, A_k = a_k)$
- By Bayes' rule,

$$\begin{aligned} & Pr(C = c_i | A_1 = a_1, \dots, A_k = a_k) \\ &= \frac{Pr(A_1 = a_1, \dots, A_k = a_k | C = c_i) \cdot Pr(C = c_i)}{Pr(A_1 = a_1, \dots, A_k = a_k)} \\ &= \frac{Pr(A_1 = a_1, \dots, A_k = a_k | C = c_i) \cdot Pr(C = c_i)}{\sum_{j=1}^{\ell} Pr(A_1 = a_1, \dots, A_k = a_k | C = c_j) \cdot Pr(C = c_j)} \end{aligned}$$

- Denominator is the same for all c_i , so sufficient to maximize

$$Pr(A_1 = a_1, \dots, A_k = a_k | C = c_i) \cdot Pr(C = c_i)$$

Example

- To classify $A = g, B = q$
- $Pr(C = t) = 5/10 = 1/2$
- $Pr(A = g, B = q \mid C = t) = 2/5$
- $Pr(A = g, B = q \mid C = t) \cdot Pr(C = t) = 1/5$
- $Pr(C = f) = 5/10 = 1/2$
- $Pr(A = g, B = q \mid C = f) = 1/5$
- $Pr(A = g, B = q \mid C = f) \cdot Pr(C = f) = 1/10$
- Hence, predict $C = t$

A	B	C
m	b	t
m	s	t
g	q	t
h	s	t
g	q	t
g	q	f
g	s	f
h	b	f
h	q	f
m	b	f

Example . . .

- What if we want to classify $A = m, B = q$?
- $Pr(A = m, B = q | C = t) = 0$
- Also $Pr(A = m, B = q | C = f) = 0!$
- To estimate joint probabilities across all combinations of attributes, we need a much larger set of training data

<i>A</i>	<i>B</i>	<i>C</i>
<i>m</i>	<i>b</i>	<i>t</i>
<i>m</i>	<i>s</i>	<i>t</i>
<i>g</i>	<i>q</i>	<i>t</i>
<i>h</i>	<i>s</i>	<i>t</i>
<i>g</i>	<i>q</i>	<i>t</i>
<i>g</i>	<i>q</i>	<i>f</i>
<i>g</i>	<i>s</i>	<i>f</i>
<i>h</i>	<i>b</i>	<i>f</i>
<i>h</i>	<i>q</i>	<i>f</i>
<i>m</i>	<i>b</i>	<i>f</i>

Naïve Bayes classifier

- Strong simplifying assumption: attributes are pairwise independent

$$Pr(A_1 = a_1, \dots, A_k = a_k \mid C = c_i) = \prod_{j=1}^k Pr(A_j = a_j \mid C = c_i)$$

- $Pr(C = c_i)$ is fraction of training data with class c_i
 - $Pr(A_j = a_j \mid C = c_i)$ is fraction of training data labelled c_i for which $A_j = a_j$
- Final classification is

$$\arg \max_{c_i} Pr(C = c_i) \prod_{j=1}^k Pr(A_j = a_j \mid C = c_i)$$

- Conditional independence is not theoretically justified
- For instance, text classification
 - Items are documents, attributes are words (absent or present)
 - Classes are topics
 - Conditional independence says that a document is a set of words: ignores sequence of words
 - Meaning of words is clearly affected by relative position, ordering
- However, naive Bayes classifiers work well in practice, even for text classification!
 - Many spam filters are built using this model

Example revisited

- Want to classify $A = m, B = q$
- $Pr(A = m, B = q \mid C = t) = Pr(A = m, B = q \mid C = f) = 0$
- $Pr(A = m \mid C = t) = 2/5$
- $Pr(B = q \mid C = t) = 2/5$
- $Pr(A = m \mid C = f) = 1/5$
- $Pr(B = q \mid C = f) = 2/5$
- $Pr(A = m \mid C = t) \cdot Pr(B = q \mid C = t) \cdot Pr(C = t) = 2/25$
- $Pr(A = m \mid C = f) \cdot Pr(B = q \mid C = f) \cdot Pr(C = f) = 1/25$
- Hence predict $C = t$

A	B	C
m	b	t
m	s	t
g	q	t
h	s	t
g	q	t
g	q	f
g	s	f
h	b	f
h	q	f
m	b	f

- Suppose $A = a$ never occurs in the test set with $C = c$
- Setting $Pr(A = a | C = c) = 0$ wipes out any product $\prod_{i=1}^k Pr(A_i = a_i | C = c)$ in which this term appears
- Assume A_i takes m_i values $\{a_{i1}, \dots, a_{im_i}\}$
- “Pad” training data with one sample for each value a_j — m_i extra data items
- Adjust $Pr(A_i = a_i | C = c_j)$ to $\frac{n_{ij} + 1}{n_j + m_i}$ where
 - n_{ij} is number of samples with $A_i = a_i, C = c_j$
 - n_j is number of samples with $C = c_j$

- Laplace's law of succession

$$Pr(A_i = a_i | C = c_j) = \frac{n_{ij} + 1}{n_j + m_i}$$

- More generally, Lidstone's law of succession, or smoothing

$$Pr(A_i = a_i | C = c_j) = \frac{n_{ij} + \lambda}{n_j + \lambda m_i}$$

- $\lambda = 1$ is Laplace's law of succession

Text classification

- Classify text documents using topics
- Useful for automatic segregation of newsfeeds, other internet content
- Training data has a unique topic label per document — e.g., Sports, Politics, Entertainment
- Want to use a naïve Bayes classifier
- Need to define a generative model
- How do we represent documents?

Set of words model

- Each document is a **set** of words over a vocabulary $V = \{w_1, w_2, \dots, w_m\}$
- Topics come from a set $C = \{c_1, c_2, \dots, c_k\}$
- Each topic c has probability $Pr(c)$
- Each word $w_i \in V$ has conditional probability $Pr(w_i | c_j)$ with respect to each $c_j \in C$
- Generating a random document d
 - Choose a topic c with probability $Pr(c)$
 - For each $w \in V$, toss a coin, include w in d with probability $Pr(w | c)$
- $Pr(d | c) = \prod_{w_i \in d} Pr(w_i | c) \prod_{w_i \notin d} (1 - Pr(w_i | c))$
- $Pr(d) = \sum_{c \in C} Pr(d | c)$

Naïve Bayes classifier

- Training set $D = \{d_1, d_2, \dots, d_n\}$
 - Each $d_i \subseteq V$ is assigned a unique label from C
- $Pr(c_j)$ is fraction of D labelled c_j
- $Pr(w_i | c_j)$ is fraction of documents labelled c_j in which w_i appears
- Given a new document $d \subseteq V$, we want to compute $\arg \max_c Pr(c | d)$
- By Bayes' rule, $Pr(c | d) = \frac{Pr(d | c)Pr(c)}{Pr(d)}$
 - As usual, discard the common denominator and compute $\arg \max_c Pr(d | c)Pr(c)$
- Recall $Pr(d | c) = \prod_{w_i \in d} Pr(w_i | c) \prod_{w_i \notin d} (1 - Pr(w_i | c))$

Bag of words model

- Each document is a **multiset** or **bag** of words over a vocabulary $V = \{w_1, w_2, \dots, w_m\}$
 - Count multiplicities of each word
- As before
 - Each topic c has probability $Pr(c)$
 - Each word $w_i \in V$ has conditional probability $Pr(w_i | c_j)$ with respect to each $c_j \in C$ (but we will estimate these differently)
 - Note that $\sum_{i=1}^m Pr(w_i | c_j) = 1$
 - Assume document length is independent of the class

Bag of words model

- Generating a random document d
 - Choose a document length ℓ with $Pr(\ell)$
 - Choose a topic c with probability $Pr(c)$
 - Recall $|V| = m$.
 - To generate a single word, throw an m -sided die that displays w with probability $Pr(w | c)$
 - Repeat ℓ times
- Let n_j be the number of occurrences of w_j in d

- $Pr(d | c) = Pr(\ell) \ell! \prod_{j=1}^m \frac{Pr(w_j | c)^{n_j}}{n_j!}$

Parameter estimation

- Training set $D = \{d_1, d_2, \dots, d_n\}$
 - Each d_i is a multiset over V of size ℓ_i
- As before, $Pr(c_j)$ is fraction of D labelled c_j
- $Pr(w_i | c_j)$ — fraction of occurrences of w_i over documents $D_j \subseteq D$ labelled c_j
 - n_{id} — occurrences of w_i in d

$$\blacksquare Pr(w_i | c_j) = \frac{\sum_{d \in D_j} n_{id}}{m \sum_{t=1}^m \sum_{d \in D_j} n_{td}} = \frac{\sum_{d \in D} n_{id} Pr(c_j | d)}{m \sum_{t=1}^m \sum_{d \in D} n_{td} Pr(c_j | d)},$$

$$\text{since } Pr(c_j | d) = \begin{cases} 1 & \text{if } d \in D_j, \\ 0 & \text{otherwise} \end{cases}$$

Classification

- $Pr(c | d) = \frac{Pr(d | c) Pr(c)}{Pr(d)}$
- Want $\arg \max_c Pr(c | d)$
- As before, discard the denominator $Pr(d)$
- Recall, $Pr(d | c) = Pr(\ell) \ell! \prod_{j=1}^m \frac{Pr(w_j | c)^{n_j}}{n_j!}$, where $|d| = \ell$
- Discard $Pr(\ell), \ell!$ since they do not depend on c
- Compute $\arg \max_c Pr(c) \prod_{j=1}^m \frac{Pr(w_j | c)^{n_j}}{n_j!}$