COS 424: Interacting with Data

Lecturer: David Blei
Scribes: Wei Ho, Michael Ye

Lecture #4
February 14, 2008

1 Maximum likelihood estimation

1.1 MLE of a Bernoulli random variable (coin flips)

Given N flips of the coin, the MLE of the bias of the coin is

$$\widehat{\pi} = \frac{\text{number of heads}}{\text{N}} \tag{1}$$

One of the reasons that we like to use MLE is because it is consistent. In the example above, as the number of flipped coins N approaches infinity, our the MLE of the bias $\hat{\pi}$ approaches the true bias π^*, as we can see from the graph below.

1.2 MLE of a Gaussian random variable

The parameters of a Gaussian distribution are the mean (μ) and variance (σ^2). Given observations $x_1, \ldots, x_N$, the likelihood of those observations for a certain μ and σ^2 (assuming that the observations came from a Gaussian distribution) is

$$p(x_1, \ldots, x_N | \mu, \sigma^2) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ \frac{-(x_n - \mu)^2}{2\sigma^2} \right\} \tag{2}$$

and the log likelihood is

$$\mathcal{L}(\mu, \sigma) = -\frac{1}{2} N \log(2\pi\sigma^2) - \sum_{n=1}^{N} \frac{(x_n - \mu)^2}{2\sigma^2} \tag{3}$$

We can then find the values of μ and σ^2 that maximize the log likelihood by taking derivative with respect to the desired variable and solving the equation obtained. By doing so, we find that the MLE of the mean is

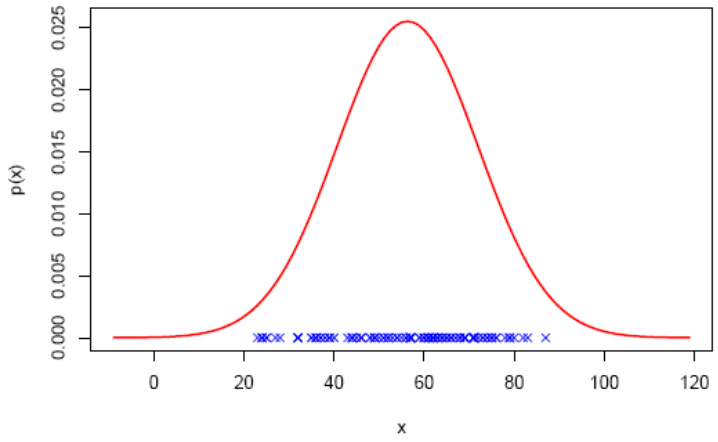
$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n \tag{4}$$

and the MLE of the variance is

$$\hat{\sigma^2} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \hat{\mu})^2 \tag{5}$$

1.3 Gaussian MLE case study

In the graph above, we have plotted the annual presidential approval ratings along with the Gaussian distribution fitted to the sample mean and variance. However, there are three main problems in using a Gaussian model to analyze this data. First, the approval ratings are restricted between 0 and 100, while the Gaussian density function is over all real numbers. Secondly, simply looking at the approval ratings as separate data points ignore the sequential nature of the data. Lastly, using the Gaussian distribution assumes that the data points are independent and identically distributed, when in fact we need to take into account of the fact that the term of a presidency is four or eight years, so there are years when the ratings are correlated. All models are merely cartoons of the data that we're analyzing, and it's hard to find a model that describe all the data. As the statistician George Box once said, "All models are wrong, but some models are useful."



2 Naive Bayes

2.1 Classification

Classification algorithms work on labeled training data. A learning algorithm takes the data produces a classifier, which is a function that takes in new unlabeled test examples and outputs predictions about the labels of those test examples based on the patterns in the training data. An example would be spam classification, where the data are e-mails and the labels are whether the e-mails are spam or not (also called ham).

2.1.1 Binary text classification

In binary text classification, we are given documents and their classes $\{w_{d,1:N}, c_d\}$, where $w_{d,1:N}$ are the N words in document d, and c_d is the class of the document. For example, suppose the documents are e-mails; then the categories are whether the e-mail is spam or ham. Our goal is to build a classifier that can predict the class of a new document. To do so, we would divide the data into a training set and testing set. The purpose of holding out a portion of the data as testing set is to allow for cross validation. After all, it would be cheating if we were to both train and test our classifier using the same data set. Thus, we use the labeled testing set that has not been seen by the classifier during training to

evaluate the classifier. One good evaluation metric is accuracy, which is defined as

$$\frac{\text{\# correctly predicted labels in the test set}}{\text{\# of instances in the test set}} \tag{6}$$

2.2 Naive Bayes model

The naive Bayes model defines a joint distribution over words and classes–that is, the probability that a sequence of words belongs to some class c.

$$p(c, w_{1:N}|\pi, \theta) = p(c|\pi) \prod_{n=1}^{N} p(w_n|\theta_c) \tag{7}$$

π is the probability distribution over the classes (in this case, the probability of seeing spam e-mails and of seeing ham e-mails). It's basically a discrete probability distribution over the classes that sums to 1. θ_c is the class conditional probability distribution–that is, given a certain class, the probability distribution of the words in your vocabulary. Each class has a probability for each word in the vocabulary (in this case, there is a set of probabilities for the spam class and one for the ham class).
Given a new test example, we can classify it using the model by calculating the conditional probability of the classes given the words in the e-mail p(c | w_n) and see which class is most likely. There is an implicit independence assumption behind the model. Since we're multiplying the p(w_n | θ_c) terms together, it is assumed that the words in a document are conditionally independent given the class.

2.3 Class prediction with naive Bayes

We classify using the posterior distribution of the classes given the words, which is proportional to the joint distribution since $P(X|Y)$ = P(X, Y)/P(Y) $\propto$ P(X, Y). The posterior distribution is

$$p(c|w_{1:N}, \pi, \theta) \propto p(c|\pi) \prod_{n=1}^{N} p(w_n|\theta_c) \tag{8}$$

Note that we don't need a normalizing constant because we only care about which probability is bigger. The classifier assigns the e-mail to the class which has highest probability.

2.4 Fitting a naive Bayes model with maximum likelihood

We find the parameters in the posterior distribution used in class prediction by learning on the labeled training data (in this case, the ham and spam e-mails). We use the maximum likelihood method method in finding parameters that maximize the likelihood of the observed data set. Given data $\{w_{d,1:N}, c_d\}_{d=1}^{D}$, the likelihood under a certain model with parameters ($\theta_{1:C}$, π) is

$$p(\mathcal{D}|\theta_{1:C}, \pi) \quad = \quad \prod_{d=1}^{D} \left(p(c_d|\pi) \prod_{n=1}^{N} p(w_n|\theta_{c_d}) \right)$$

$$= \prod_{d=1}^{D} \prod_{c=1}^{C} \left(\pi_c \prod_{n=1}^{N} \prod_{v=1}^{V} \theta_{c,v}^{1(w_{d,n}=v)} \right)^{1(c_d=c)}$$

which is the product over each email (D), each category (C), each word in the e-mail (N), and each word in the vocabulary (V). The $1(c_d = c)$ term serves to filter out e-mails that are not under the specific category, and the $1(w_{d,n} = v)$ term serves to filter out the words in the vocabulary that don't appear in the e-mail. Then, taking the logarithm of each side gives

$$\begin{aligned} \mathcal{L}(\pi, \theta_{1:C}; \mathcal{D}) = & \sum_{d=1}^{D} \sum_{c=1}^{C} 1(c_d = c) \log \pi_c + \\ & \sum_{d=1}^{D} \sum_{c=1}^{C} \sum_{n=1}^{N} \sum_{v=1}^{V} 1(c_d = c) 1(w_{d,n} = v) \log \theta_{c,v} \end{aligned}$$

and since the logarithm is a monotonic function, maximizing the log likelihood is the same as maximizing the likelihood of the data. Taking the log allows you to decompose the likelihood into the two separate parts of your model–one term contains only π, and the other term contains only $\theta_{c,v}$. Thus, taking the derivative with respect to one of the parameters eliminates the other one. To maximize the values of π and $\theta_{c,v}$, take the derivative with respect to each and solve for zero. This leads to the maximum likelihood estimates:

$$\hat{\pi}_c = \frac{n_c}{D} \tag{9}$$

and

$$\hat{\theta}_{c,v} = \frac{n_{c,v}}{\sum_{v'} n_{c,v'}} \tag{10}$$

where n_c is the number of times you see class c and $n_{c,v}$ is the number of times you see word v in class c.

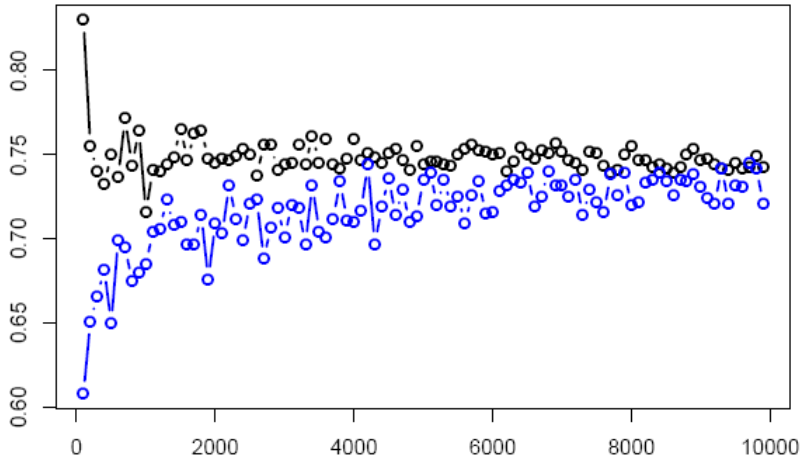
2.5 Full procedure

So, we have a general outline of how to build and use our model as follows:

- Estimate the model from the training set.
- Predict the class of each test example.
- Compute the accuracy.

2.6 Naive Bayes case study

We'll consider an example of using the naive Bayes model to do spam filtering. The e-mail data set comes from Enron's subpoenaed e-mails. The training set size is 10,000 e-mails (all labeled as spam or ham) and the test set size is 1,000 e-mails. Preprocessing has removed common words such as "the" from data set. In the graph below, the x-axis represents training size and the y-axis represents accuracy. Note the sensitivity of model performance to training data size. Test accuracy starts off bad, but gets better as training size increases and converges at around 70% accuracy. On the other hand, the training accuracy starts off very high and then drops off. This is due to overfitting, since when you're only training on a few e-mails, the word distribution can perfectly describe the training e-mails.

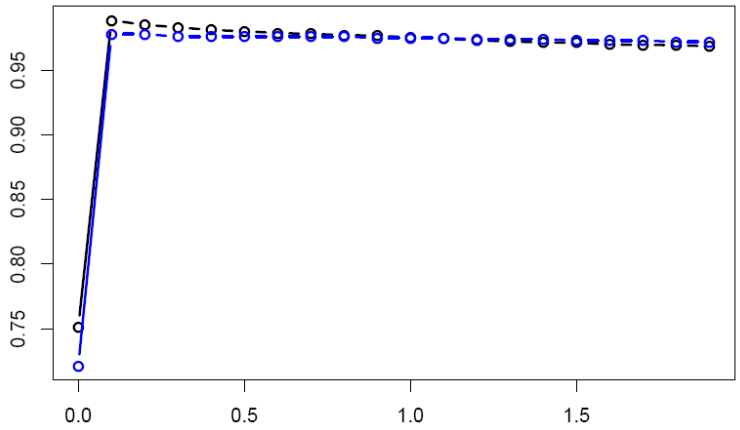


2.7 Smoothing

However, suppose in the above case study, we see a rare word like "peanut" appear in one spam e-mail and none of the ham e-mails. In that case, $\theta_{\text{spam,peanut}}$ will be some small value, but $\theta_{\text{ham,peanut}}$ will be 0. Since the probabilities are multiplied together, this leads to the problem that any e-mail containing the word "peanut" will never be classified as ham. This is clearly a ludicrous conclusion. To resolve that, we use smoothing to make the probability distribution more reasonable by adding some number λ to the per-word class counts. The result is that all words will have non-zero probabilities, while the more frequent words have slighly decreased probability, leading to a smoother overall distribution. Thus, the new equation is

$$\hat{\theta}_{c,w} = \frac{n_{c,w} + \lambda}{\sum_{w'} n_{c,w'} + V\lambda} \tag{11}$$

When $\lambda = 1$, we call it Laplace smoothing. When $\lambda = 0.5$, we call it Jeffrey's smoothing.



The x-axis is λ and the y-axis is the accuracy. We can see that just adding a little bit of smoothing results in a huge increase in accuracy. It seems that the problem of

misclassification based on the appearances of rare words is fairly significant.

2.8 Questions about naive Bayes

This model assumes that the future looks like the past - that is, the test cases come from the same distribution as the training cases. What other assumptions are made? What effect do the assumptions have on this classifier? What is strange about the NB model of text? Can you adapt NB to different data, e.g., vectors of reals? We will address these questions next time.