AI511-ML: Week 2: Naive Bayes

aniruddh.kishore@iiitb.ac.in

14 September 2021

1 1D Maximum Likelihood Estimation for Gaussian Bayes Models

1.1 Introduction

We leave behind regression and move firmly into the territory of classification.

Specifically, our aim here is, given some data points of the form $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$, we wish to be able to predict y using x as the input, **where the ys come from a small number of discrete variables**. For now, let x and y both be scalars.

1.2 Formulation as a Maximum Likelihood Estimation problem

Let's say that our model takes in some x and for each unique class y, it returns a probability score –that is, it tells us, given x, what the probability is that y is the class label, for every class.
Clearly, if we have some (x_i, y_i) in our dataset, we want

$$P(y_i|x_i)$$

to be high.

Recall Bayes' Theorem: If we have two event a and b, then:

$$P(a|b) = \frac{P(b|a)P(a)}{P(b)}$$

Therefore, here, we can write:

$$P(y_i|x_i) = \frac{P(x_i|y_i)P(y_i)}{P(x_i)}$$

Now, we wish to maximize this value over the entire dataset i.e. we wish to maximize

$$P(y_1|x_1)P(y_2|x_2)...P(y_n|x_n)$$

This looks like it could get pretty complicated when we plug in Bayes' rule. Let's break it down a bit. In the original equation we can see that $P(y_i)$ is a constant – it's just $\frac{n(y_i)}{n}$. So we can ignore it for the purposes of our maximization (but not at the inference time!).

Another thing we can do is look at $P(x_i)$. Suppose we have three output classes c_1, c_2 and c_3. Then during inference time, we would be comparing (using Bayes' rule)

$$\frac{P(x_i|y_i = c_1)P(y_i = c_1)}{P(x_i)}$$

$$\frac{P(x_i|y_i = c_2)P(y_i = c_2)}{P(x_i)}$$

and

$$\frac{P(x_i|y_i = c_3)P(y_i = c_3)}{P(x_i)}$$

and choosing that c_i for which the calculated probability is maximum.
Do you see something common here? In all three cases, we have a $P(x_i)$ term common – thus making it redundant. Therefore, we can ignore it – both during optimization AND during inference.

Putting all this together (deep breath, now!), we realize we wish to maximize:

$$P(x_1|y_1)P(x_2|y_2)P(x_3|y_3)...P(x_n|y_n)$$

Now here's where the generative modeling part comes in. We assume that our model is a set of probability distributions. For each class label, we have a distribution, and we can **sample** from the distributions to get x_is.

Now, we wish to choose the parameters of these model distributions such that the above expression is maximum. Put it another way, if θ is the vector of all the model parameters, we wish to choose θ such that, if we were to sample from these distributions, it is very likely that we would get the x_is. Formally, we define a new function, called the **likelihood**:

$$\mathcal{L}_X(\theta) = P(x_1|y_1, \theta)P(x_2|y_2, \theta)...P(x_n|y_n, theta)$$

We wish to find θ such that $\mathcal{L}$ is maximized. Which brings us to the topic of **maximum likelihood estimation**.

1.3 Solving Maximum Likelihood Estimation for 1D Data Using Gaussians

Recall that the Gaussian or the Normal Distribution is:

$$N(\mu, \sigma) \sim \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2} \right)$$

Where μ is the mean and σ is the variance. The distribution is centered around the mean and if you increase the variance, the distribution becomes flatter, and if you decrease the variance, the distribution becomes more "pointy". You can explore the Gaussian further on `desmos.com/calculator`.

Now, for each class, we assume that the distribution of x is a Gaussian. Specifically, if $y_i = c_j$, we assume that

$$P(x_i|y_i = c_j) = N(\mu_{c_j}, \sigma_{c_j})$$

Therefore, in the expression of the likelihood, we can plug in the functional form of the respective Gaussians[1] respectively evaluated at the matching x_i.

We can rewrite the likelihood as

$$\mathcal{L}_X(\theta) = \prod_{c_j} \prod_{y_i = c_j} P(x_i|y_i = c_j)$$

where

$$P(x_i|y_i = c_j) = N(\mu_{c_j}, \sigma_{c_j})$$

Don't think too hard about the above expression for $\mathcal{L}$. All I'm saying there is that it is more convenient to group data points as per their $y_i = c_j$. We take the product over all classes, and within that, we take the product of all terms where y_i belongs to that class. Therefore,

$$\mathcal{L}_X(\theta) = \prod_{c_j} \prod_{y_i = c_j} \frac{1}{\sigma_{c_j}\sqrt{2\pi}} \exp\left(-\frac{1}{2} \frac{(x_i - \mu_{c_j})^2}{\sigma_{c_j}^2} \right)$$

Now, if we maximize $\mathcal{L}$ with respect to every μ_{c_j} and σ_{c_j} we're done!

If you think that's too hard, don't worry – there's a modification we do first. We can instead optimize the **log** of the likelihood. The reason we can do this is because, firstly, the likelihood is always positive (so the log is always defined) and second, the log function is monotonically strictly increasing – so, maximizing the log-likelihood is equivalent to maximizing the likelihood.

What happens if we use the log-likelihood? First, the product of terms in the likelihood becomes a sum. Another neat part of using the log-likelihood is that if you observe the Gaussian, the bulk is in the exponential. So the log and the exponential cancel out, leaving you with a bunch of quadratic terms.

[1] It is important that you realize that P for a continuous distribution in this case here, unless specified, refers to the **probability density** and NOT the actual probability. The actual probability of landing EXACTLY a particular value of x when you sample from the distribution is precisely zero. The probability of getting something between x and $x + \mathrm{d}x$, on the other hand, is equal to the probability density times $\mathrm{d}x$.

$$\log(\mathcal{L}_X(\theta)) = \sum_{c_j}\sum_{y_i=c_j}\left(-\log(\sigma_{c_j}) - \frac{1}{2}\frac{(x_i-\mu_{c_j})^2}{\sigma_{c_j}^2}\right) + \text{constant terms involving } \pi$$

Now, we minimize the log terms with respect to μ_{c_j} and σ_{c_j} the same way we did the 1D closed-form linear regression.

$$\frac{\partial\mathcal{L}}{\partial\mu_{c_j}} = \sum_{y_i=c_j}\frac{x_i-\mu_{c_j}}{\sigma_{c_j}^2} = 0$$

Rearranging a little, we get:

$$\sum_{y_i=c_j} x_i = \sum_{y_i=c_j}\mu_{c_j}$$

But $\sum_{y_i=c_j} 1 = n_{c_j}$, so

$$\sum_{y_i=c_j} x_i = n_{c_j}\mu_{c_j}$$

or,

$$\mu_{c_j} = \frac{1}{n_{c_j}}\sum_{y_i=c_j} x_i$$

Thus, the mean of the Gaussian for a particular class is equal to the sample mean for the x_is belonging to that class.
Now, to get σ_{c_j},

$$\frac{\partial\mathcal{L}}{\partial\sigma_{c_j}} = \sum_{y_i=c_j}\left(-\frac{1}{\sigma_{c_j}} + \frac{(x_i-\mu_{c_j})^2}{\sigma_{c_j}^3}\right) = 0$$

Rearranging, we get:

$$\sum_{y_i=c_j} 1 = \sum_{y_i=c_j}\frac{(x_i-\mu_{c_j})^2}{\sigma_{c_j}^2}$$

Or,

$$\sigma_{c_j}^2 = \sum_{y_i=c_j}\frac{(x_i-\mu_{c_j})^2}{n_{c_j}}$$

Thus, the Gaussian's variance is the same as the sample variance of the x_is belonging to c_j class.

With this, we have obtained the optimal parameters for all the Gaussians.

1.4 Inference

At inference time, given a x we wish to predict the $\hat{y}$ for, compute:

$$P(x|y = c_j)P(c_j)$$

For all the classes c_j and choose that class for which the above expression is maximum. The first term in that product is the value at x for the Gaussian for class c_j and the second term is the frequency of class c_j i.e n_i/n.

2 Extending to multiple dimensions: Naive Bayes

2.1 Buildup

Previously, our data overall was a vector $\vec{x} = [x_1, x_2, ..., x_n]^T$ and $\vec{y} = [y_1, y_2, ..., y_n]^T$. Now our data is a matrix $X_{n \times m}$ where every column is a feature (m features) and every row is a data point (n data points). y remains the same.

We extend our previous discussion by saying that we wish to maximize:

$$P(y_i|X[i,:]) = \frac{P(X[i,:]|y_i)P(y_i)}{P(X[i,:])}$$

Where we use Numpy 2D array slicing notation for convenience – that there means the i row and all columns.
As before, we discard the denominator term for both optimization and inference, and we discard the second numerator term during optimization.

Like before, we are left with:

$$\mathcal{L}_X(\theta) = \prod_{c_j} \prod_{y_i = c_j} P(X[i,:]|y_i = c_j)$$

What does the probability of a vector mean?

$$P(X[i,:]) = P(X[i,1] \cap X[i,2] \cap ... \cap X[i,m])$$

That is, the probability that the first feature is $X[i,1]$ AND the second feature is $X[i,2]$ AND so on.

2.2 The Naive Part

Here's the interesting part. **We assume that all features are mutually independent.** Recall that if events a and b are mutually independent, then

$$P(a \cap b) = P(a)P(b)$$

What this means is that if we have

$$\mathcal{L}_X(\theta) = \prod_{c_j} \prod_{y_i = c_j} P(X[i,:]|y_i = c_j)$$

Then, if we assume that all features are independent,

$$\mathcal{L}_X(\theta) = \prod_{c_j} \prod_{k=1}^{m} \prod_{y_i = c_j} P(X[i,k]|y_i = c_j)$$

All I've done is broken up the probability of the row vector into a product of probabilities for the features of that row.[2]

I'll spare you the Maximum Likelihood Estimation story and get to the important bit: **We fit a Gaussian for each class, for each feature column**. We do this by considering the values of x in that feature column and for that value of y. The equations for μ and σ are the same.

Now, during inference, we compare:

$$\left(\prod_{k=1}^{m} P(X[i,k]|y_i = c_j)\right) P(y_i = c_j)$$

The first term is the product of densities of the Gaussians for each feature column for the given class. The second term is the frequency of the class.

3 Looking ahead

One thing to note here is that we always assume that the features are distributed as per a Gaussian. However, that need not always be the case. We could have features like male/female (binary), or gen/sc/st/obc/km (categorical), In those cases, using Gaussians would be inappropriate.
For a binary feature, we use the Bernoulli distribution, and for a categorical feature, we first label encode it (coming up in the TA session) and use the Multinomial distribution. After this week's TA session, look up these distributions and their implementations in Scikit-Learn.

[2]Again, I mean probability density. Sorry about that, but I've commonly seen places use "probability" as a stand-in for "probability density" for the sake of convenience. I shall continue in that glorious, confusing tradition.