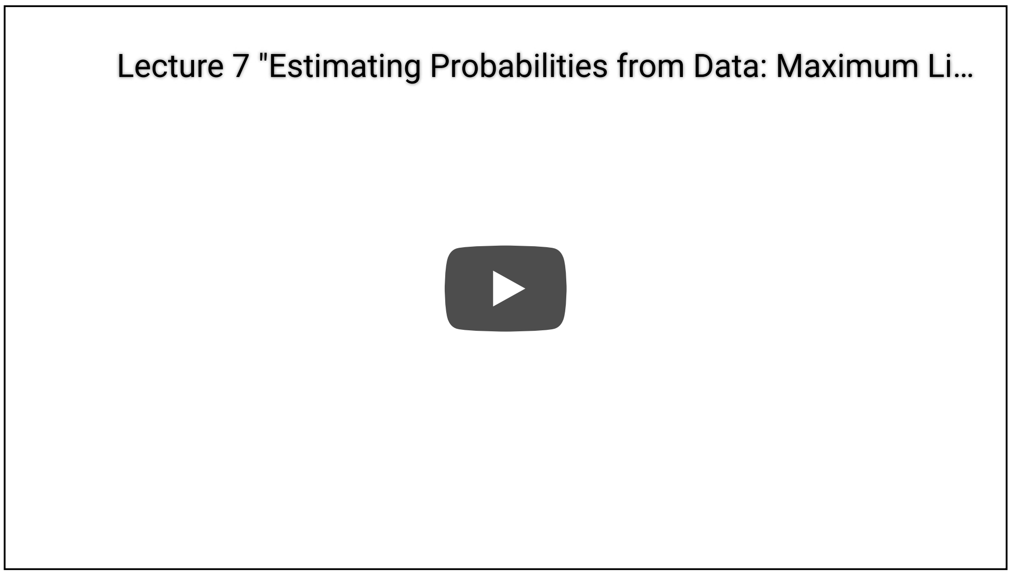
Estimating Probabilities from data

previous back next


Lecture 7 "Estimating Probabilities from Data: Maximum Li...

continued in videos (Lecture 7) (Lecture 8)

Remember the Bayes Optimal classifier: If we are provided with $P(X,Y)$ we can predict the most likely label for $\mathbf{x}$, formally $\operatorname{argmax}_y P(y|\mathbf{x})$. It is therefore worth considering if we can estimate $P(X,Y)$ directly from the training data. If this is possible (to a good approximation) we could then use the Bayes Optimal classifier in practice on our estimate of $P(X,Y)$.

In fact, many supervised learning can be viewed as estimating $P(X,Y)$. Generally, they fall into two categories:

- When we estimate $P(X,Y)=P(X|Y)P(Y)$, then we call it *generative learning.*
- When we only estimate $P(Y|X)$ directly, then we call it *discriminative learning.*

So how can we estimated probability distributions from samples?

Simple scenario: coin toss

Suppose you find a coin and it's ancient and very valuable. **Naturally**, you ask yourself, "What is the probability that this coin comes up heads when I toss it?" You toss it $n=10$ times and obtain the following sequence of outcomes: $D=\{H,T,T,H,H,H,T,T,T,T\}$. Based on these samples, how would you estimate $P(H)$?

We observed $n_H=4$ heads and $n_T=6$ tails. So, intuitively,

$$P(H)\approx\frac{n_H}{n_H+n_T}=\frac{4}{10}=0.4$$

Can we derive this more formally?

Maximum Likelihood Estimation (MLE)

The estimator we just mentioned is the Maximum Likelihood Estimate (MLE). For MLE you typically proceed in two steps: First, you make an explicit modeling assumption about what type of distribution your data was sampled from. Second, you set the parameters of this distribution so that the data you observed is as likely as possible.

Let us return to the coin example. A natural assumption about a coin toss is that the distribution of the observed outcomes is a binomial distribution. The binomial distribution has two parameters n and θ and it captures the distribution of n independent Bernoulli (i.e. binary) random events that have a positive outcome with probability θ. In our case n is the number of coin tosses, and θ could be the probability of the coin coming up heads (e.g. $P(H)=\theta$). Formally, the binomial distribution is defined as

$$P(D \mid \theta) = \binom{n_H + n_T}{n_H} \theta^{n_H} (1-\theta)^{n_T},$$

and it computes the probability that we would observe exactly n_H heads, n_T tails, if a coin was tossed $n = n_H + n_T$ times and its probability of coming up heads is θ.

MLE Principle: Find $\hat{\theta}$ to maximize the likelihood of the data, $P(D; \theta)$:

$$\hat{\theta}_{MLE} = \underset{\theta}{\operatorname{argmax}} \; P(D; \theta)$$

Often we can solve this maximization problem with a simple two step procedure: 1. plug in all the terms for the distribution, and take the $\log$ of the function. 2. Compute its derivative, and equate it with zero. Taking the log of the likelihood (often referred to as the log-likelihood) does not change its maximum (as the log is a monotonic function, and the likelihood positive), but it turns all products into sums which are much easier to deal with when you differentiate. Equating the derivative with zero is a standard way to find an extreme point. (To be precise you should verify that it really is a maximum and not a minimum, by verifying that the second derivative is negative.)

Returning to our binomial distribution, we can now plug in the definition and compute the log-likelihood:

$$\begin{aligned}
\hat{\theta}_{MLE} &= \underset{\theta}{\operatorname{argmax}} \; P(D; \theta) \\
&= \underset{\theta}{\operatorname{argmax}} \binom{n_H + n_T}{n_H} \theta^{n_H} (1-\theta)^{n_T} \\
&= \underset{\theta}{\operatorname{argmax}} \; \log \binom{n_H + n_T}{n_H} + n_H \cdot \log(\theta) + n_T \cdot \log(1-\theta) \\
&= \underset{\theta}{\operatorname{argmax}} \; n_H \cdot \log(\theta) + n_T \cdot \log(1-\theta)
\end{aligned}$$

We can then solve for θ by taking the derivative and equating it with zero. This results in

$$\frac{n_H}{\theta} = \frac{n_T}{1-\theta} \Longrightarrow n_H - n_H\theta = n_T\theta \Longrightarrow \theta = \frac{n_H}{n_H + n_T}$$

A nice sanity check is that $\theta \in [0, 1]$.

- MLE gives the explanation of the data you observed.
- If n is large and your model/distribution is correct (that is $\mathcal{H}$ includes the true model), then MLE finds the **true** parameters.
- But the MLE can overfit the data if n is small. It works well when n is large.
- If you do not have the correct model (and n is small) then MLE can be terribly wrong!

For example, suppose you observe H,H,H,H,H. What is $\hat{\theta}_{MLE}$?

Simple scenario: coin toss with prior knowledge

Assume you have a hunch that θ is close to 0.5. But your sample size is small, so you don't trust your estimate.

Simple fix: Add m imaginery throws that would result in θ' (e.g. $\theta = 0.5$). Add m Heads and m Tails to your data.

$$\hat{\theta} = \frac{n_H + m}{n_H + n_T + 2m}$$

For large n, this is an insignificant change. For small n, it incorporates your "prior belief" about what θ should be. Can we derive this formally?

The Bayesian Way

Model θ as a **random variable**, drawn from a distribution $P(\theta)$. Note that θ is **not** a random variable associated with an event in a sample space. In frequentist statistics, this is forbidden. In Bayesian statistics, this is allowed and you can specify a prior belief $P(\theta)$ defining what values you believe θ is likely to take on.

Now, we can look at $P(\theta \mid D) = \frac{P(D|\theta)P(\theta)}{P(D)}$ (recall Bayes Rule!), where

- $P(\theta)$ is the **prior** distribution over the parameter(s) θ, before we see any data.

- $P(D \mid \theta)$ is the **likelihood** of the data given the parameter(s) θ.
- $P(\theta \mid D)$ is the **posterior** distribution over the parameter(s) θ **after** we have observed the data.

A natural choice for the prior $P(\theta)$ is the Beta distribution:

$$P(\theta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha,\beta)}$$

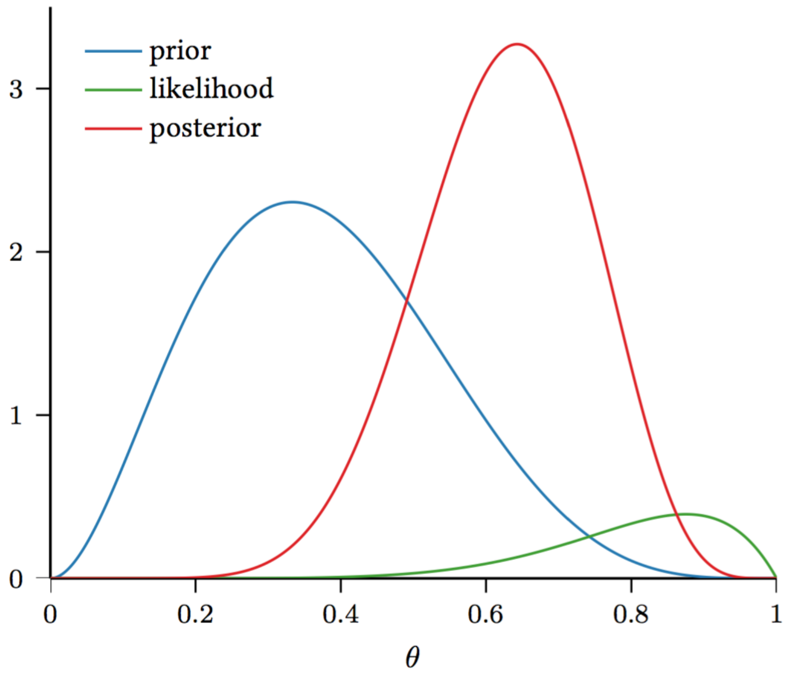
where $B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ is the normalization constant (if this looks scary don't worry about it, it is just there to make sure everything sums to 1 and to scare children at Halloween). Note that here we only need a distribution over a singly binary random variable θ. (The multivariate generalization of the Beta distribution is the Dirichlet distribution.)

Why is the Beta distribution a good fit?

- it models probabilities (θ lives on $[0,1]$)
- it is of the same distributional family as the binomial distribution (**conjugate prior**) $\rightarrow$ the math will turn out nicely:

$$P(\theta \mid D) \propto P(D \mid \theta)P(\theta) \propto \theta^{n_H+\alpha-1}(1-\theta)^{n_T+\beta-1}$$

So far, we have a distribution over θ. How can we get an estimate for θ?



Maximum a Posteriori Probability Estimation (MAP)

For example, we can choose $\hat{\theta}$ to be the most likely θ given the data.
MAP Principle: Find $\hat{\theta}$ that maximizes the posterior distribution $P(\theta \mid D)$:

$$\begin{aligned}\hat{\theta}_{MAP} &= \underset{\theta}{\operatorname{argmax}}\ P(\theta \mid D) \\ &= \underset{\theta}{\operatorname{argmax}}\ \log P(D \mid \theta) + \log P(\theta)\end{aligned}$$

For out coin flipping scenario, we get:

$$\begin{aligned}
\hat{\theta}_{MAP} &= \underset{\theta}{\operatorname{argmax}}\ P(\theta|Data) \\
&= \underset{\theta}{\operatorname{argmax}}\ \frac{P(Data|\theta)P(\theta)}{P(Data)} && \text{(By Bayes rule)} \\
&= \underset{\theta}{\operatorname{argmax}}\ \log(P(Data|\theta)) + \log(P(\theta)) \\
&= \underset{\theta}{\operatorname{argmax}}\ n_H \cdot \log(\theta) + n_T \cdot \log(1-\theta) + (\alpha-1)\cdot\log(\theta) + (\beta-1)\cdot\log(1-\theta) \\
&= \underset{\theta}{\operatorname{argmax}}\ (n_H+\alpha-1)\cdot\log(\theta) + (n_T+\beta-1)\cdot\log(1-\theta) \\
&\Longrightarrow \hat{\theta}_{MAP} = \frac{n_H+\alpha-1}{n_H+n_T+\beta+\alpha-2}
\end{aligned}$$

A few comments:

- The MAP estimate is identical to MLE with $\alpha-1$ hallucinated *heads* and $\beta-1$ hallucinated *tails*
- As $n \rightarrow \infty$, $\hat{\theta}_{MAP} \rightarrow \hat{\theta}_{MLE}$ as $\alpha-1$ and $\beta-1$ become irrelevant compared to very large n_H, n_T.
- MAP is a great estimator if an accurate prior belief is available (and mathematically tractable).
- If n is small, MAP can be very wrong if prior belief is wrong!

"True" Bayesian approach

Note that MAP is only one way to get an estimator. There is much more information in $P(\theta \mid D)$, and it seems like a shame to simply compute the mode and throw away all other information. A true Bayesian approach is to use the posterior predictive distribution directly to make prediction about the label Y of a test sample with features X:

$$P(Y \mid D, X) = \int_\theta P(Y, \theta \mid D, X) d\theta = \int_\theta P(Y \mid \theta, D, X) P(\theta|D) d\theta$$

Unfortunately, the above is generally *intractable* in closed form and sampling techniques, such as Monte Carlo approximations, are used to approximate the distribution. A pleasant exception are Gaussian Processes, which we will cover later in this course.

Another exception is actually our coin toss example. To make *predictions* using θ in our coin tossing example, we can use

$$\begin{aligned}
P(heads \mid D) &= \int_\theta P(heads, \theta \mid D) d\theta \\
&= \int_\theta P(heads \mid \theta, D) P(\theta \mid D) d\theta \qquad \text{(Chain rule: } P(A,B|C) = P(A|B,C)P(B|C).\text{)} \\
&= \int_\theta \theta P(\theta \mid D) d\theta \\
&= E\left[\theta|D\right] \\
&= \frac{n_H+\alpha}{n_H+\alpha+n_T+\beta}
\end{aligned}$$

Here, we used the fact that we defined $P(heads \mid D, \theta) = P(heads \mid \theta) = \theta$ (this is only the case because we assumed that our data is drawn from a binomial distribution - in general this would not hold).

Machine Learning and estimation

In supervised Machine learning you are provided with training data D. You use this data to train a model, represented by its parameters θ. With this model you want to make predictions on a test point x_t.

- **MLE** Prediction: $P(y|x_t;\theta)$ Learning: $\theta = \operatorname{argmax}_\theta P(D;\theta)$. Here θ is purely a model parameter.
- **MAP** Prediction: $P(y|x_t,\theta)$ Learning: $\theta = \operatorname{argmax}_\theta P(\theta|D) \propto P(D \mid \theta)P(\theta)$. Here θ is a random variable.
- **"True Bayesian"** Prediction: $P(y|x_t, D) = \int_\theta P(y|\theta)P(\theta|D)d\theta$. Here θ is integrated out - our prediction takes all possible models into account.

As always the differences are subtle. In MLE we maximize $\log[P(D;\theta)]$ in MAP we maximize $\log[P(D|\theta)] + \log[P(\theta)]$. So essentially in MAP we only add the term $\log[P(\theta)]$ to our optimization. This term is independent of the data and penalizes if the parameters, θ deviate too much from what we believe is reasonable. We will later revisit this as a form of regularization, where $\log[P(\theta)]$ will be interpreted as a measure of classifier complexity.