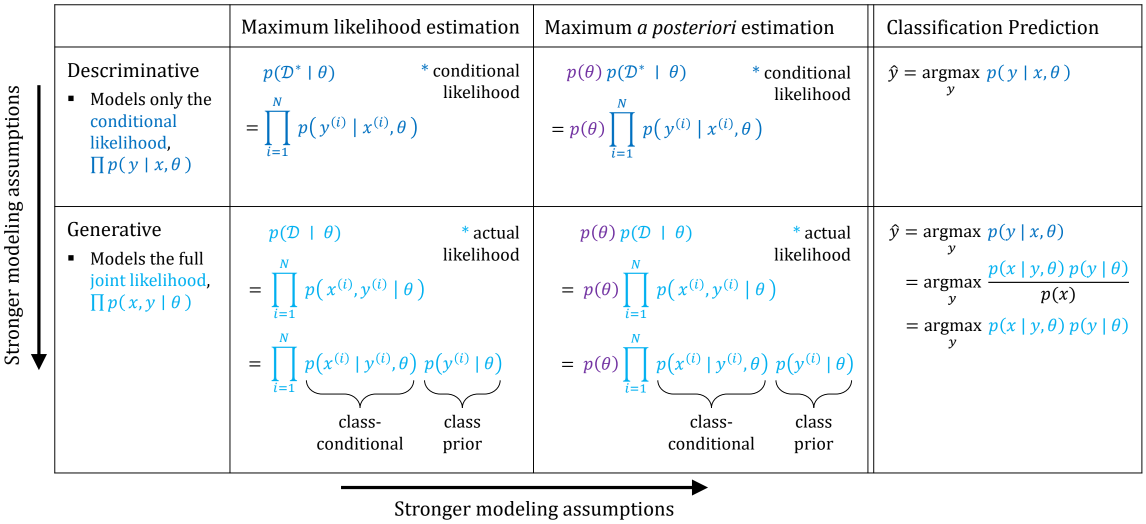
10-315 Notes
Probabilistic Generative Models

Carnegie Mellon University
Machine Learning Department

Contents

1 Summary: Generative vs Discriminative and MLE vs MAP

An overview highlighting the differences and similarities between MLE vs MAP estimation and discriminative vs generative models in one table:



	Maximum likelihood estimation	Maximum *a posteriori* estimation	Classification Prediction
Descriminative ▪ Models only the conditional likelihood, $\prod p(y \mid x, \theta)$	$p(\mathcal{D}^* \mid \theta)$ * conditional likelihood $= \prod_{i=1}^{N} p(y^{(i)} \mid x^{(i)}, \theta)$	$p(\theta)\, p(\mathcal{D}^* \mid \theta)$ * conditional likelihood $= p(\theta) \prod_{i=1}^{N} p(y^{(i)} \mid x^{(i)}, \theta)$	$\hat{y} = \underset{y}{\operatorname{argmax}}\ p(y \mid x, \theta)$
Generative ▪ Models the full joint likelihood, $\prod p(x, y \mid \theta)$	$p(\mathcal{D} \mid \theta)$ * actual likelihood $= \prod_{i=1}^{N} p(x^{(i)}, y^{(i)} \mid \theta)$ $= \prod_{i=1}^{N} \underbrace{p(x^{(i)} \mid y^{(i)}, \theta)}_{\text{class-conditional}} \underbrace{p(y^{(i)} \mid \theta)}_{\text{class prior}}$	$p(\theta)\, p(\mathcal{D} \mid \theta)$ * actual likelihood $= p(\theta) \prod_{i=1}^{N} p(x^{(i)}, y^{(i)} \mid \theta)$ $= p(\theta) \prod_{i=1}^{N} \underbrace{p(x^{(i)} \mid y^{(i)}, \theta)}_{\text{class-conditional}} \underbrace{p(y^{(i)} \mid \theta)}_{\text{class prior}}$	$\hat{y} = \underset{y}{\operatorname{argmax}}\ p(y \mid x, \theta)$ $= \underset{y}{\operatorname{argmax}} \frac{p(x \mid y, \theta)\, p(y \mid \theta)}{p(x)}$ $= \underset{y}{\operatorname{argmax}}\ p(x \mid y, \theta)\, p(y \mid \theta)$

2 Bayes Rule (again)

2.1 Bayes rule: General version

Recall the we can write Bayes rule in terms of generic variables a and b, without assigning particular meaning to them:

$$p(a \mid b) = \frac{p(b \mid a)\ p(a)}{p(b)}$$
$$p(a \mid b) \propto p(b \mid a)\ p(a)$$

2.2 Bayes rule: Data and parameters

When formulating a MAP estimate with the dataset $\mathcal{D} = \{y^{(i)}\}_{i=1}^{N}$ and generic model parameter(s) θ, we used Bayes rule to convert the **likelihood** and **prior** into the **posterior**:

$$\begin{aligned} p(\theta \mid \mathcal{D}) &= \frac{p(\theta)\ p(\mathcal{D} \mid \theta)}{p(\mathcal{D})} \\ p(\theta \mid \mathcal{D}) &\propto p(\theta)\ p(\mathcal{D} \mid \theta) \\ &= p(\theta)\ \prod_{i=1}^{N} p(y^{(i)} \mid \theta) \end{aligned}$$

where:

- $p(\mathcal{D} \mid \theta)$ is the **likelihood**,
- $p(\theta)$ is the **prior** on the parameters, and
- $p(\theta \mid \mathcal{D})$ is the **posterior**

Additonally, if we had a dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$, we defined the conditional likelihood as follows $p(\mathcal{D} \mid \theta) = \prod_{i=1}^{N} p(y^{(i)} \mid x^{(i)}, \theta)$. We can then similarly formulate a MAP estimate with this dataset over generic model parameter(s), θ by using the Bayes rule to convert the **conditional likelihood** and **prior** into the **posterior**:

$$\begin{aligned} p(\theta \mid \mathcal{D}) &= \frac{p(\theta)\ p(\mathcal{D} \mid \theta)}{p(\mathcal{D})} \\ p(\theta \mid \mathcal{D}) &\propto p(\theta)\ p(\mathcal{D} \mid \theta) \\ &= p(\theta)\ \prod_{i=1}^{N} p(y^{(i)} \mid x^{(i)}, \theta) \end{aligned}$$

where:

- $p(\mathcal{D} \mid \theta)$ is the **conditional likelihood**,
- $p(\theta)$ is the **prior** on the parameters, and
- $p(\theta \mid \mathcal{D})$ is the **posterior**

2.3 Bayes rule: ML input and output data

What if we take the generic Bayes rule formula and use variables x and y, where x and y represent the input and output of a machine learning prediction problem:

$$p(y \mid x) = \frac{p(x \mid y)\ p(y)}{p(x)}$$

$$p(y \mid x) \propto p(x \mid y)\ p(y)$$

Now this takes on a different meaning than the use of Bayes rule in the MAP formulation above.

Here, we have that

- $p(x \mid y)$ is defined as the class conditional distribution
- $p(y)$ is the label class prior distribution
- $p(y \mid x)$ is the conditional likelihood

3 Discriminative vs Generative Models

First, when we define a machine learning model, it falls into either one of two categories: discriminative or generative models. We define the key difference between the two types of models below.

A **discriminative model** is when we directly model the $p(y \mid x, \theta)$ distribution, i.e. the conditional likelihood (the output data given the input data and the parameters).

A **generative model** is a more in-depth model that directly or indirectly models the $p(x, y \mid \theta)$ distribution, i.e. the likelihood (the joint distribution of input and output data given the parameters).

3.1 Discriminative Models

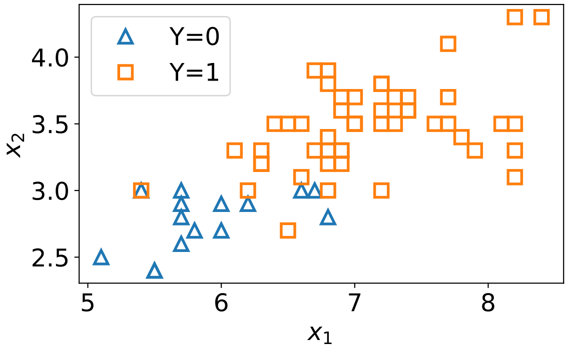
So far, we have only been using discriminative models to represent the relationship between the input, x, and the output, y, so far. For example, logistic regression is a discriminative model. The term discriminative comes from the fact that we are trying to discriminate between different classes of y, given information about x. Despite the term "discriminative," we can also use discriminative models for regression, e.g. linear regression is a discriminative model.

A discriminative model is convenient because it directly gives us the probability distributions that we are typically interested in for machine learning tasks, namely $p(y \mid x, \theta)$. However, it is not a very strong probability model. One property of a strong probability model is its ability to generate new data points. If we only have the distribution $p(y \mid x, \theta)$, we can sample a new value of $y \sim p(y \mid x, \theta)$, but only if we have the input data, x. A discriminative model doesn't provide us with a way to sample values for the input data, x.

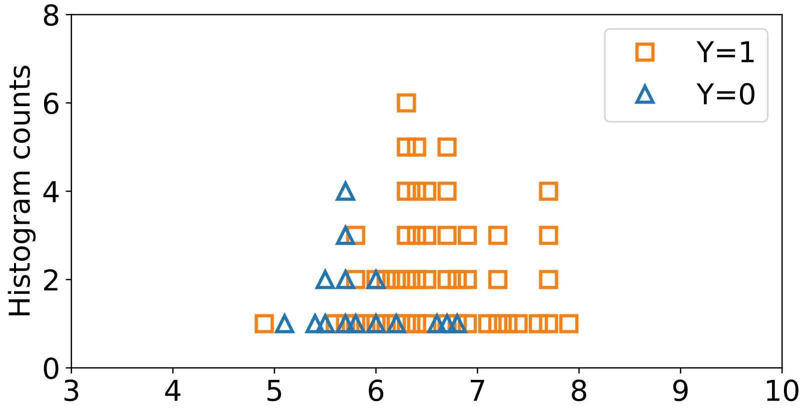
3.1.1 Example: Iris Classification

The canonical discriminative model for classification is logistic regression. Consider the case where we apply logistic regression to classify two species of Iris flowers given their petal measurements.

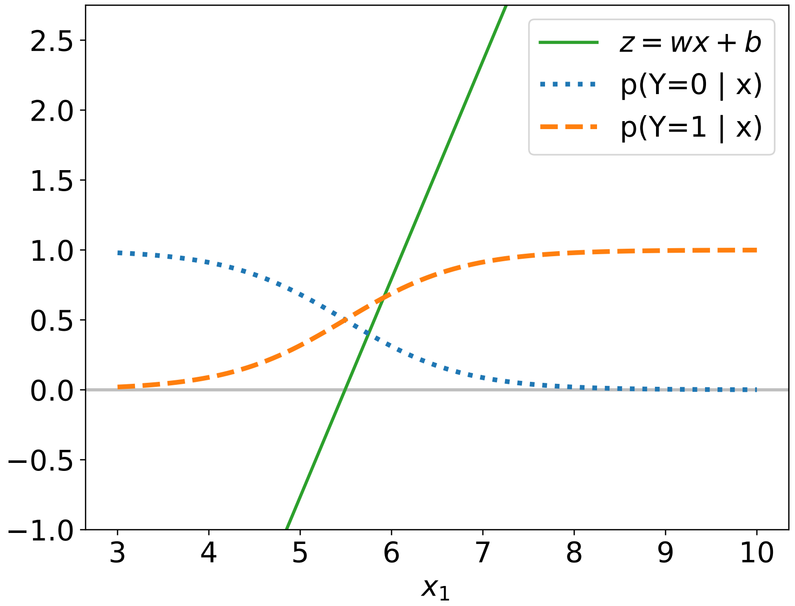
Here is a plot of the Iris data with just two species and just two of the four input parameters:



To simplify our introductory examples, let's just consider a single feature, x_1, as shown in the histogram below:



When applying logistic regression to model this classification task, we learn the slope and intercept parameters of a linear model, $z = wx_1 + b$, inside of a logistic function, $p(y \mid x_1, w, b) = \frac{1}{1+e^{-(wx_1+b)}}$:



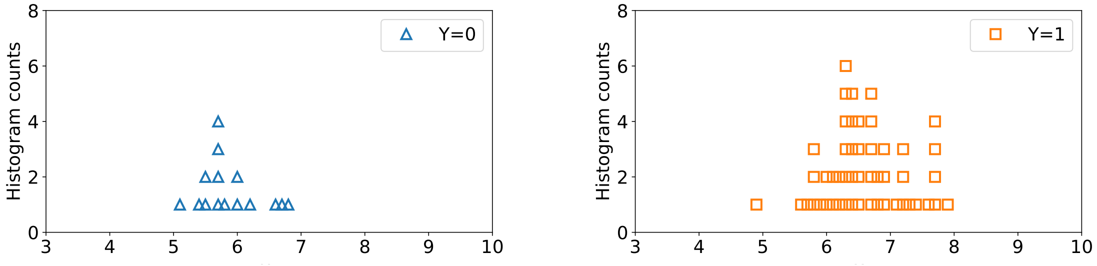
While logistic regression does seem to be doing the right thing, i.e., predicting $P(Y = 1 \mid x_1)$ values > 0.5 when x_1 is greater than 5.5. However, it seems overly simplistic to have the underlying model be based on a linear expression of the input data. Let's now turn to generative models, which may be more satisfying.

3.2 Generative Models

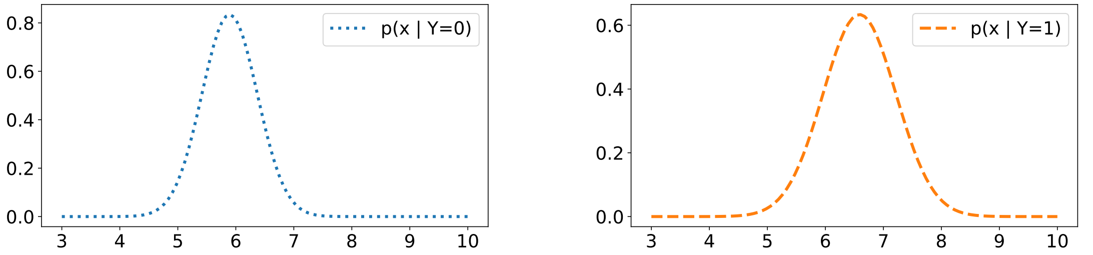
Bear with us a moment as we build up to a generative model by continuing with our iris example from above.

3.2.1 Example: Iris Classification

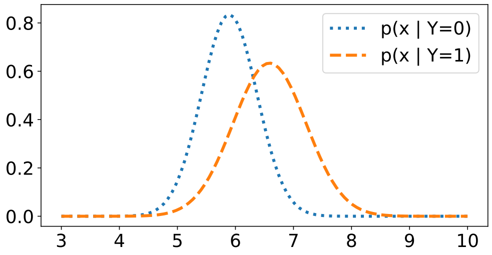
Looking at the single-feature version of the iris dataset, one might suspect that a Gaussian distribution may be helpful to model this data, especially when we visualize the two species $Y = 0$ and $Y = 1$ separately:



If we only consider x_1 values for the datapoints with $Y = 0$ (blue triangles), then we can estimate the Gaussian mean and standard deviation parameters, which gives us a model for the Gaussian density, $p(x_1 \mid Y = 0, \mu_{Y=0}, \sigma_{Y=0})$, shown on the left below. Similarly, but separately, we can estimate the Gaussian parameters for only the $Y = 1$ points in our dataset, $p(x_1 \mid Y = 1, \mu_{Y=1}, \sigma_{Y=1})$, shown on the right below.



These two distributions are called **class conditional** distributions, as they both require the class of the output variable Y to be given. It is important to note (and easy to forget later), that these are two different distributions with two different mean and standard deviation parameters. By plotting these together, as in the figure below, we can see that $\mu_{Y=0}$ and $\mu_{Y=1}$ are different, as are $\sigma_{Y=0}$ and $\sigma_{Y=1}$, so there are four total parameters, because there are two Gaussians, and we have two parameters for each Gaussian (a μ and a σ).

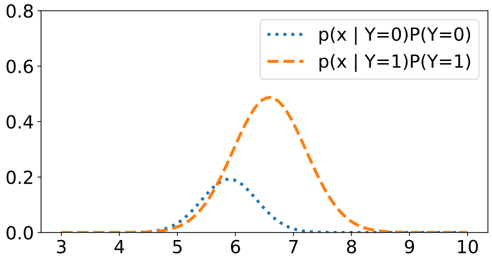


Now while these two class conditional distributions, $p(x_1 \mid y, \mu_y, \sigma_y)$ are interesting, they don't really provide enough information for our desired probability $p(y \mid x_1)$. In fact, if you look at the plot above, you might guess that the decision boundary for $p(y \mid x_1) = 0.5$ is roughly $x_1 = 6.5$, where the blue and orange Gaussian density functions intersect. However, this is incorrect! We need one more piece of the puzzle to help us convert from the $p(x_1 \mid y)$ distributions to the desired $p(y \mid x_1)$.

Enter Bayes rule:

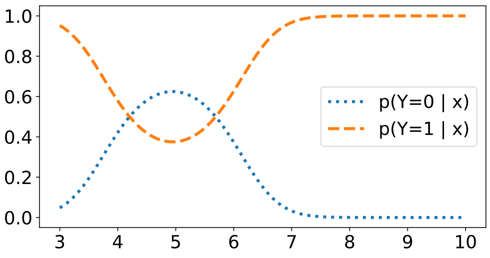
If we also have the **class prior** distributions, $p(y)$, then we have enough information to apply Bayes rule to get to the desired **conditional likelihood** $p(y \mid x_1)$.

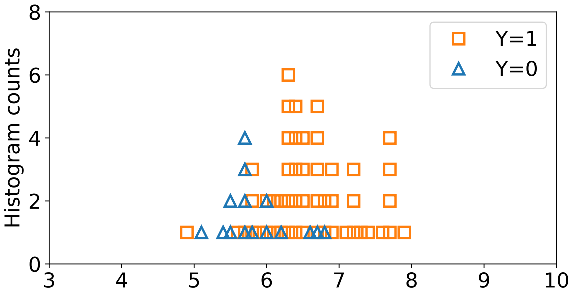
Fortunately, we can model $P(Y = 1)$ (and simultaneously $P(Y = 0)$) as a Bernoulli distribution, and estimate the Bernoulli parameter ϕ by simply calculating the fraction of occurrences of $Y = 1$ in the dataset out of the total N points. Multiplying this class prior times the class conditional, we arrive at the numerator of Bayes rule, $p(x_1 \mid y)\ p(y)$, which can be written as the joint probability $p(x_1, y)$. Below are the plots of the following two joint distributions for our iris example: $p(x_1, Y = 1)$ and $p(x_1, Y = 0)$:



Using Bayes rule to normalize these, we arrive at the desired conditional likelihood $p(y \mid x_1)$:

$$\begin{aligned} p(Y = 1 \mid x_1) &= \frac{p(x \mid Y = 1)\ P(Y = 1)}{p(x)} && (1)\\ &= \frac{p(x \mid Y = 1)\ P(Y = 1)}{\sum_y p(x, y)} && (2)\\ &= \frac{p(x \mid Y = 1)\ P(Y = 1)}{p(x, Y = 0) + p(x, Y = 1)} && (3)\\ &= \frac{p(x \mid Y = 1)\ P(Y = 1)}{p(x \mid Y = 0)P(Y = 0) + p(x \mid Y = 1)P(Y = 1)} && (4) \end{aligned}$$





This generative model of $p(y \mid x_1) \propto p(x_1 \mid y)\ p(y)$ seems to produce reasonable $P(Y = 1 \mid x_1) = 0.5$ decision boundaries at $x_1 = 5.8$ *and* at $x_1 = 4.2$. The latter decision boundary makes some sense when we consider the larger Gaussian standard deviation for $p(x_1 \mid Y = 1, \mu_{Y=1}, \sigma_{Y=1})$ scaled by the also larger $P(Y = 1, \phi)$.

Let's recap what we have completed so far. We have modelled 3 different probability distributions

1. Prior: $p(y \mid \phi)$ using a Bernoulli distribution
2. Class Conditional Distributions:
   - $p(x_1 \mid Y = 0, \mu_{Y=0}, \sigma_{Y=0})$ as a Gaussian
   - $p(x_1 \mid Y = 1, \mu_{Y=1}, \sigma_{Y=1})$ as a Gaussian

Using these 3 models for the probability distribution, we were able to calculate $p(Y = 1 \mid x_1)$ using these distributions above as shown in equation (4). As a result, not only is this model generative because it models $p(x_1, y) = p(x_1 \mid y)\ p(y)$ which we have defined the two distributions on the RHS of the equality; however, it also has the capabilities of a discriminative model because it can estimate $P(Y = 1 \mid x_1)$, although with a different decision boundary.

Now, what's so powerful about this? We've defined more probability distributions (made more assumptions) than the logistic regression model to create a more complex decision boundary for $P(Y = 1 \mid x_1)$. We could create a more complex discriminative model that could perform something similar (e.g. logistic regression with feature engineering), so what sets generative models apart from discriminative models? It's their ability to generate new data.

4 The Generative Story

Briefly, we will continue with the examples that we had before. Recall that our logistic regression model had a model for the probability distribution $p(y \mid x_1)$ while we defined a generative model with a Bernoulli class prior distribution, $p(y)$ and Gaussian class conditional distributions, $p(x_1 \mid y)$.

Suppose that we wanted to sample new, unseen points, (x, y) pairs, from our model. If we had our discriminative model, which models $p(y \mid x_1)$, we would not be able to generate new (x, y) pairs. This is because $p(y \mid x_1)$ requires a known x_1 value in order to generate a probability distribution over the classes to sample from. However, requiring an x_1 sample simply assumes what we are trying to get, which is a new (x_1, y) pair.

This is where generative models come in. A discriminative model's distribution will not match a joint distribution that we can sample from $p(x_1, y) \neq p(y \mid x_1)$. However, a generative model that models the joint, $p(x_1, y)$, will be able to sample from the joint distribution because it can calculate those probabilities.

For example, we can note that for our generative model that classifies the Iris dataset, $p(x_1, y) \propto p(x_1 \mid y)\ p(y)$. This means that we can use the $p(x_1 \mid y)$ and $p(y)$ to sample from $p(x_1, y)$. The generative process is as follows

1. First, sample a class, y, from $p(y)$. Note that this must be done first because all other probability distributions to sample from require a y value to condition on. In our case, this sampling is done by flipping a biased coin with the probability matching that of the Bernoulli distribution that $p(y)$ represents.
2. Second, given the y from the prior step, sample a point, x_1, using appropriate class conditional distribution, $p(x_1 \mid Y = y)$. Recall that we can only perform this step once we have a known $Y = y$ value. Note that we can use one of the many algorithms to sample from a 1D-Gaussian distribution.

3. Finally, take the (x_1, y) pair. That is our new sample which did not occur in our original dataset $\mathcal{D}$. Instead, it was **generated from our model which was fitted to model the probability distribution that generated the dataset** $\mathcal{D}$.

As a result, we can see that any discriminative model can't necessarily generate new samples due to their limited modeling assumptions. However, by making stronger assumptions and defining the class prior and class conditional distributions, we are able to construct a generative model which then allows us to sample from it to generate new points.