Machine Learning - Michaelmas Term 2016
Lecture 7 : Classification: Generative Models

Lecturer: Varun Kanade

So far, we've studied the linear model for regression and some extensions. In the linear model, the output is assumed to be a linear function of the inputs with some additional noise. We studied probabilistic *discriminative models*, where the output y conditioned on the inputs $\mathbf{x}$ and the model $\mathbf{w}$ was modelled as a probability distribution, *e.g.,* $p(y \mid \mathbf{x}, \mathbf{w}) = \mathcal{N}(\mathbf{w} \cdot \mathbf{x}, \sigma^2)$. In this lecture, we'll study *generative models* for classification. In a generative model, we model both the inputs and the output as a probability distribution, with some parameters $\boldsymbol{\theta}$, *i.e.,* the entire joint distribution $p(\mathbf{x}, y \,|\boldsymbol{\theta})$.

1 Generative Models for Classification

Classification problems are also a form of *supervised learning*, where we are given data as input and output pairs. In classification problems, the output (or target) y is a class or a category; let us suppose that $y \in \{1, \ldots, C\}$, where C is the number of classes. As in the case of regression, the inputs may be of several types, such as real-valued or categorical. For regression problems, we first converted categorical inputs to real-valued ones by some encoding scheme, such as *one-hot encoding*. We can also do this for classification problems, but for now, let us leave the categorical inputs as they are. For some types of generative models, it is possible and indeed desriable to model categorical inputs directly, without converting them to some vector form.

Let us quickly summarise the distinction betwen a generative model and a discriminative model. In the *discriminative* setting, we only model the conditional distribution of the output, given the input and model parameters, *i.e.,* $p(y \mid \mathbf{x}, \boldsymbol{\theta})$.[1] On the other hand, in the *generative* framework, we model the full joint distribution $p(\mathbf{x}, y \mid \boldsymbol{\theta})$. We'll discuss a couple of different generative models in this lecture and methods to estimate their parameters, but for now let us first understand how predictions can be made by using a generative model.

1.1 Prediction Using a Generative Model

Let us suppose using some training data we've obtained an esimate for the parameters $\boldsymbol{\theta}$ of our generative model $p(\mathbf{x}, y \mid \boldsymbol{\theta})$. This model represents a joint probability distribution over the inputs and the output. Suppose we are given a new input $\mathbf{x}_{\text{new}}$ and we wish to assign a class from $\{1, \ldots, C\}$ to it. Using the entire model, we can write the conditional distribution for the output y, as follows:

For $c \in \{1, \ldots, C\}$,

$$p(y = c \mid \mathbf{x}_{\text{new}}, \boldsymbol{\theta}) = \frac{p(y = c \mid \boldsymbol{\theta}) \cdot p(\mathbf{x}_{\text{new}} | y = c, \boldsymbol{\theta})}{\sum_{c'=1}^{C} p(y = c' \mid \boldsymbol{\theta}) \cdot p(\mathbf{x}_{\text{new}} \mid y = c', \boldsymbol{\theta})} \tag{1}$$

We've simply applied Bayes' Rule to get the conditional probability distribution above. The denominator of the (1) represents the marginal distribution $p(\mathbf{x}_{\text{new}} \mid \boldsymbol{\theta})$ of observing $\mathbf{x}_{\text{new}}$ according to the generative model $p(\mathbf{x}, y \mid \boldsymbol{\theta})$, whereas the numerator represents the joint probability of the observation $p(\mathbf{x}_{\text{new}}, c|\boldsymbol{\theta})$ according to the model. In order to predict a specific class, we

[1] When we talk of parameters in generality, we'll simply use $\boldsymbol{\theta}$. Whereas when we talk about specific models, we may use different notation depending on the model. Hopefully, this should not be a cause for confusion. Unfortunately, there are many specific models for which the parameters are also denoted by $\boldsymbol{\theta}$ in the literature, increasing the chance of creating confusion.

Voted in 2012?	Annual Income	State	Candidate Choice
Y	50K	OK	Clinton
N	173K	CA	Clinton
Y	80K	NJ	Trump
Y	150K	WA	Clinton
N	25K	WV	Johnson
Y	85K	IL	Clinton
⋮	⋮	⋮	⋮
Y	1050K	NY	Trump
N	35K	CA	Trump

Table 1: Some fake data about voters in the US presidential election along with their candidate choice.

simply pick $\widehat{y} = \text{argmax}_c\, p(y = c \mid \mathbf{x}_{\text{new}}, \boldsymbol{\theta})$. The form of (1) suggests that we should model the class conditional distributions, $p(\mathbf{x} \mid y = c, \boldsymbol{\theta})$ for every $c \in \{1, \ldots, C\}$.

1.2 Toy Example

Lett us introduce a running toy example to discuss generative models. Suppose we want to predict which candidate a voter will choose in the US presidential election using some demographic information, such as whether they voted in 2012 or not, their annual income and the state where they reside. Table 1 shows some (fake) data. And finally let us suppose that our task is for some voter, who is supposedly undecided, to predict the candidate they will eventually vote for using the available data.

1.3 Defining a Generative Model

In order to fit a generative model to data, we'll express the joint distribution as follows:

$$p(\mathbf{x}, y \mid \boldsymbol{\theta}, \boldsymbol{\pi}) = p(y \mid \boldsymbol{\pi}) \cdot p(\mathbf{x} \mid y, \boldsymbol{\theta}) \tag{2}$$

Above we added an extra set of parameters $\boldsymbol{\pi}$. We express the joint distribution $p(\mathbf{x}, y \mid \boldsymbol{\theta}, \boldsymbol{\pi})$ as a product of the marginal distribution of the outputs y, parameterized using $\boldsymbol{\pi}$ and the class conditional distributions of the inputs $\mathbf{x}$ given the class label, which we parameterise using $\boldsymbol{\theta}$. Whenever we have a relatively small number of classes, we'll model the marginal distribution $p(y \mid \boldsymbol{\pi})$ over classes, by introducing a parameter π_c to denote the probability that $y = c$. Clearly, we need $\sum_{c=1}^{C} \pi_c = 1$. So we have the marginal distribution over $p(y|\boldsymbol{\pi})$ defined as:

$$p(y \mid \boldsymbol{\pi}) = \pi_c \tag{3}$$

We'll return to the question of modelling the class conditional distribution over the inputs $\mathbf{x}$ later. For now, let us suppose that we are given data $\mathcal{D} = \langle (\mathbf{x}_i, y_i) \rangle_{i=1}^{N}$. We assume that each data point is drawn independently from the same distribution, or i.i.d. for short. Let us write the likelihood of the data in terms of the model parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$.

$$p(\mathcal{D} \mid \boldsymbol{\theta}, \boldsymbol{\pi}) = \prod_{i=1}^{N} \left(\left(\prod_{i=1}^{C} \pi_c^{\mathbb{1}(y_i = c)} \right) \cdot p(\mathbf{x}_i \mid y_i, \boldsymbol{\theta}) \right)$$

Above, $\mathbb{1}$ is the indicator function, so $\mathbb{1}(y_i = c) = 1$ if $y_i = c$ and 0 otherwise. As always, it is easier to deal with the log-likelihood than the likelihood. Let N_c denote the number of input datapoints such that $y_i = c$. Then we have,

$$\log p(\mathcal{D} \mid \boldsymbol{\theta}, \boldsymbol{\pi}) = \sum_{c=1}^{C} N_c \log(\pi_c) + \sum_{i=1}^{N} \log p(\mathbf{x}_i \mid y_i, \boldsymbol{\theta}) \tag{4}$$

Based on the form of (4), we see that in order to estimate the parameter π_c, we do not need to worry about the exact form of the class conditional distributions $p(\mathbf{x} \mid y, \boldsymbol{\theta})$, as there is no dependence on $\boldsymbol{\pi}$ in the second term of the RHS of (4).

Thus, to obtain the parameters π, we can simply solve the following optimisation problem:

$$\begin{aligned} \text{maximise} \quad & \sum_{c=1}^{C} N_c \log \pi_c \\ \text{subject to :} \quad & \sum_{c=1}^{C} \pi_c = 1 \end{aligned} \tag{5}$$

This problem can be solved using the method of Lagrangean multipliers. In general, for a constrained optimisation problem of the form:

$$\underset{\mathbf{z}}{\text{argmax}}\, f(\mathbf{z}), \quad \text{subject to :} \;\; g(\mathbf{z}) = 0, \tag{6}$$

the Lagrangean (dual) form is the following:

$$\Lambda(\mathbf{z}, \lambda) = f(\mathbf{z}) + \lambda g(\mathbf{z}) \tag{7}$$

A necessary condition for $\mathbf{z}$ to be a solution to (6) is that it should be a stationary point of $\Lambda(\mathbf{z}, \lambda)$. (For further details regarding this approach, please refer to (Bishop, 2006, Appendix C) or a book on multivariate calculus.) Here, let us quickly see why this should be the case: Taking the partial derivative of $\Lambda(\mathbf{z}, \lambda)$ with respect to λ and setting it to 0, gives us $g(\mathbf{z}) = 0$. The condition that the gradient of $\Lambda(\mathbf{z}, \lambda)$ with respect to $\mathbf{z}$ is 0, gives us that $\nabla_{\mathbf{z}} f = -\lambda \nabla_{\mathbf{z}} g$, *i.e.,* the gradients with respect to f and g must be parallel. This is of course necessary, as otherwise, it would be possible to increase $f(\mathbf{z})$ while going in a direction that still maintains $g(\mathbf{z}) = 0$.

Applying this to our optimization problem (5), we write the Lagrangean form:

$$\Lambda(\boldsymbol{\pi}, \lambda) = \sum_{c=1}^{C} N_c \log \pi_c + \lambda(\sum_{c=1}^{C} \pi_c - 1) \tag{8}$$

Taking the derivative with respect to each π_c and λ and setting them to 0, we get:

$$\frac{\partial \Lambda(\boldsymbol{\pi}, \lambda)}{\partial \pi_c} = \frac{N_c}{\pi_c} + \lambda = 0 \tag{9}$$

$$\frac{\partial \Lambda(\boldsymbol{\pi}, \lambda)}{\partial \lambda} = \sum_{c=1}^{C} \pi_c - 1 = 0 \tag{10}$$

Using (9), we see that it must be that $\pi_c = -\frac{N_c}{\lambda}$ for each c and (10) shows that the value of λ is given by $-\sum_c N_c = -N$. Thus, we get the very reasonably estimate that $\pi_c = \frac{N_c}{N}$.

Thus, we have seen that for generative models, we can simply use the empirical distribution over the classes as the marginal distribution $p(y \mid \boldsymbol{\pi})$. Thus, the main part of the models is choosing a suitable model for the class conditional distributions, $p(\mathbf{x} \mid y, \boldsymbol{\theta})$, and estimating the required parameters.

2 Naïve Bayes Classifier (NBC)

Let us now return to the question of modelling the class conditional distribution on the inputs given the class label, $p(\mathbf{x} \mid y = c, \boldsymbol{\theta}_c)$. For each class $c = 1, \ldots, C$, we'll use a set of parameter $\boldsymbol{\theta}_c$ to model this distribution.[2] Typically, $\mathbf{x}$ is high-dimensional, *i.e.,* the number of features D is quite large. Thus, if we assigned a probability to every possible combination of features, we would require at a minimum exponential in D parameters. There is the additional problem that some of these features may be real-valued, *e.g.,* annual income. Thus, we need to make some simplifying assumption in order to have a model with a reasonable number of parameters.[3]

In the Naïve Bayes Classifier, it is assumed that the features are conditionally independent given the class label. So in our toy example, we're assuming that given that someone is a Trump voter, their income, the state where they live and whether they voted in 2012 are conditionally independent![4] Clearly, this is far too simplistic an assumption, which will never hold in practice. Nevertheless, these classifiers do exhibit surprisingly good results in practice!

Let us now define the model $p(\mathbf{x} \mid y = c, \theta_c)$ in the Naïve Bayes Classifier:

$$p(\mathbf{x} \mid y = c, \boldsymbol{\theta}_c) = \prod_{j=1}^{D} p(x_j \mid y = c, \boldsymbol{\theta}_{jc}) \tag{11}$$

Because of the conditional independence assumption, the probability distribution $p(\mathbf{x} \mid y = c, \boldsymbol{\theta}_c)$ factorises into D terms, one for each feature. The same is the case with the overall log-likelihood from (4), which we can in the case of the NBC write as:

$$\log p(\mathcal{D} \mid \boldsymbol{\theta}, \boldsymbol{\pi}) = \sum_{c=1}^{C} N_c \log \pi_c + \sum_{c=1}^{C} \sum_{j=1}^{D} \sum_{i:y_i=c} \log p(x_{ij} \mid \boldsymbol{\theta}_{jc}) \tag{12}$$

In order to fit the parameters $\boldsymbol{\theta}_{jc}$, we only need to use datapoints where $y_i = c$ and only the j^{th} feature of those datapoints. Depending on the type of feature, we may use a different model for the individual feature and hence a different estimation procedure.

- If $x_j \in \mathbb{R}$, *e.g.,* annual income, we can use a Gaussian model for x_j. The parameters $\boldsymbol{\theta}_{jc}$ in this instance are simply the mean and variance, $\boldsymbol{\theta}_{jc} = (\mu_{jc}, \sigma^2_{jc})$. (Exercise: Show that the maximum likelihood estimates are given by the empirical mean and variance.) There is no specific reason to use a Gaussian distribution; we could equally well have chosen a Laplace distribution or any other. The maximum likelihood estimating procedure depends on the choice of distribution, but because we are solving a one-dimensional problem, this is usually quite easy.

- If x_j is categorical, taking one of K values, $\{1, \ldots, K\}$, we may use what Murphy (2012) calls the mulitnoulli distribution. Essentially, there is a probability $\mu_{jc,l}$ associated with each of the K possible categories, so that $\sum_{l=1}^{K} \mu_{jc,l} = 1$. And the model is simply, $p(x_j = l \mid y = c, \boldsymbol{\mu}_{jc}) = \mu_{jc,l}$. We've already seen how to obtain the maximum likelihood estimates for this distribution, when we obtained estimates for π_c. The estimate for $\mu_{jc,l}$ is given by the number of datapoints where $y_i = c$ and $x_{ij} = l$ and dividing by the number of datapoints where $y_i = c$.

[2] In this section, we'll assume that the $\boldsymbol{\theta}_c$ are all distinct for $c = 1, \ldots, C$. Later, we'll make the case for sharing some of the parameters between the different classes.

[3] Note that having too many parameters is a problem both for statistical (overfitting) and computational (memory and time) reasons.

[4] This is very different from assuming that the features are independent. The features are indeed dependent, but what this assumption states is that this dependence comes *solely* from the label, *i.e.,* in this case who they are voting for. Once we have conditioned on this, the features become conditionally independent.

- If x_j is binary, *i.e.,* taking values in $\{0, 1\}$, this is just a special case of the above. In this case, we can use a Bernoulli distribution with just a single parameter $\theta_{jc} \in [0, 1]$.

Of course, other distributions can be selected. One of the advantages of using a Naïve Bayes classifier is that it allows us to mix and match feature types rather easily, *e.g.,* there is no need to convert categorical variables to vector form. The model fitting algorithm is very easy as each feature is treated separately.

Discussion

Let us remind ourselves that our final goal is *classification* and not necessarily to build model of the data. Modelling the data using a generative model is ultimately only a tool to build a classifier.[5] Thus, even though the conditional independence assumption is too naïve and unlikely to ever hold, it may still be the case that the resulting classifiers have quite good performance; indeed, this is the observation in practice. The advantages of this simplification are both data and computational efficiency. Let us suppose that all our features are binary. If we tried to model an arbitrary joint distribution over $\{0, 1\}^D$, then we would require 2^D parameters. Given that we have C classes, the total number of parameters in the model is $C \cdot 2^D$. Even though a model this general is able to fit any distribution on the data, it will invariably overfit on the available data. Unless of course the amount of data we have is $\gg 2^D$, in which there is prohibitive computational cost. On the other hand, the Naïve Bayes classifier only has $O(C \cdot D)$ parameters (in the case where all features are binary). Thus, it is very unlikely to overfit as long as we have a reasonable amount of data. This is yet another manifestation of the principle that a simpler model may have more predictive power given that we have limited amounts of data.

3 Gaussian Discriminant Analysis

Let us now study a different type of generative model. We use the same form for the joint distribution as in (2):

$$p(\mathbf{x}, y \mid \boldsymbol{\theta}, \boldsymbol{\pi}) = p(y \mid \boldsymbol{\pi}) \cdot p(\mathbf{x} \mid y, \boldsymbol{\theta}) \tag{13}$$

As described in Section 1.3, we will continue to represent the marginal distribution $p(y \mid \boldsymbol{\pi})$ as, $p(y = c \mid \boldsymbol{\pi}) = \pi_c$, thus the estimates $\pi_c = N_c/N$, where N_c is the number of datapoints with $y_i = c$. The difference is in the manner in which we model the class-conditional densities. Let us suppose that all the features are real-valued, then we can model the inputs $\mathbf{x}$ given the class label $y = c$ as being generated from a multivariate Gaussian distribution with mean $\boldsymbol{\mu}_c$ and covariance matrix $\boldsymbol{\Sigma}_c$. Formally,

$$p(\mathbf{x} \mid y = c, \boldsymbol{\theta}_c = (\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)) = \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \tag{14}$$

3.1 Estimating the parameters $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$

As in the case of the Naïve Bayes model, we can write the log likelihood for the data and then estimate the parameters $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$ for the model. The calculations involved require manipulating matrix expressions, including determinants. We'll omit the calculations; the interested

[5] It is possible that building a generative model for the data is a goal in itself. This is usually more common in the unsupervised setting, where we want to a build a model that produces *new* data that "looks" like the data we have at our disposal.

student should refer to (Murphy, 2012, Section 4.1). We'll simply state the maximum likelihood estimates here:

$$\widehat{\boldsymbol{\mu}}_c = \frac{1}{N_c} \sum_{i:y_i=c} \mathbf{x}_i$$
$$\widehat{\boldsymbol{\Sigma}}_c = \frac{1}{N_c} \sum_{i:y_i=c} (\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_c)(\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_c)^\mathsf{T}$$

3.2 Quadratic Discriminant Analysis

Let us now consider the prediction rule using the Gaussian model. Recall the prediction rule for generative models given by (1):

$$p(y = c \mid \mathbf{x}_{\text{new}}, \boldsymbol{\theta}) = \frac{p(y = c \mid \boldsymbol{\theta}) \cdot p(\mathbf{x}_{\text{new}} | y = c, \boldsymbol{\theta})}{\sum_{c'=1}^{C} p(y = c' \mid \boldsymbol{\theta}) \cdot p(\mathbf{x}_{\text{new}} \mid y = c', \boldsymbol{\theta})} \tag{15}$$

When the densities are multivariate Gaussian, the above expression takes the following form (where we've dropped the subscript on $\mathbf{x}_{\text{new}}$ for notational convenience):

$$p(y = c \mid \mathbf{x}, \boldsymbol{\theta}) = \frac{\pi_c |2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_c)^\mathsf{T}\boldsymbol{\Sigma}_c^{-1}(\mathbf{x} - \boldsymbol{\mu}_c)\right)}{\sum_{c'=1}^{C} \pi_{c'} |2\pi\boldsymbol{\Sigma}_{c'}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{c'})^\mathsf{T}\boldsymbol{\Sigma}_{c'}^{-1}(\mathbf{x} - \boldsymbol{\mu}_{c'})\right)} \tag{16}$$

The decision boundary between two classes c and c' will be given by $\mathbf{x}$ that satisfy, $p(y = c \mid \mathbf{x}, \boldsymbol{\theta}) = p(y = c' \mid \mathbf{x}, \boldsymbol{\theta})$, or alternatively when their ratio is 1. This is given by:

$$\frac{\pi_c |2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_c)^\mathsf{T}\boldsymbol{\Sigma}_c^{-1}(\mathbf{x} - \boldsymbol{\mu}_c)\right)}{\pi_{c'} |2\pi\boldsymbol{\Sigma}_{c'}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{c'})^\mathsf{T}\boldsymbol{\Sigma}_{c'}^{-1}(\mathbf{x} - \boldsymbol{\mu}_{c'})\right)} = 1$$

$$\exp\left(\frac{1}{2}\left((\mathbf{x} - \boldsymbol{\mu}_{c'})^\mathsf{T}\boldsymbol{\Sigma}_{c'}^{-1}(\mathbf{x} - \boldsymbol{\mu}_{c'}) - (\mathbf{x} - \boldsymbol{\mu}_c)^\mathsf{T}\boldsymbol{\Sigma}_c^{-1}(\mathbf{x} - \boldsymbol{\mu}_c)\right)\right) = \frac{\pi_{c'} |2\pi\boldsymbol{\Sigma}_{c'}|^{-\frac{1}{2}}}{\pi_c |2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}}}$$

$$\frac{1}{2}\left((\mathbf{x} - \boldsymbol{\mu}_{c'})^\mathsf{T}\boldsymbol{\Sigma}_{c'}^{-1}(\mathbf{x} - \boldsymbol{\mu}_{c'}) - (\mathbf{x} - \boldsymbol{\mu}_c)^\mathsf{T}\boldsymbol{\Sigma}_c^{-1}(\mathbf{x} - \boldsymbol{\mu}_c)\right) = \log\left(\frac{\pi_{c'} |2\pi\boldsymbol{\Sigma}_{c'}|^{-\frac{1}{2}}}{\pi_c |2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}}}\right) \tag{17}$$

Equation (17) shows that the decision boundaries are given by quadratic curves, it is for this reason that this is called as quadratic discriminant analysis. Note that not every point satisfying (17) lies on the decision boundary between the classes c and c'; when there are more than two classes, there may be some third class $\widetilde{c}$ which has a higher value of $p(y = \widetilde{c} \mid \mathbf{x}, \boldsymbol{\theta})$ for some or all of the $\mathbf{x}$ satisfying (17). The boundaries between classes are therefore given by piecewise quadratic curves. Figure 1(a) shows the decision boundary when there are only two classes.

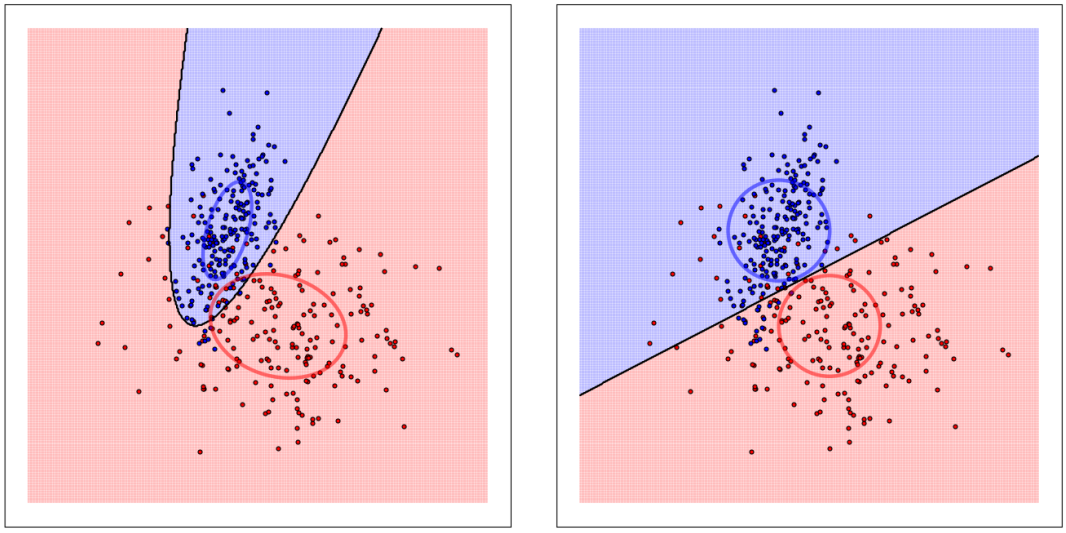
3.3 Linear Discriminant Analysis

A special case is when the model assumes that the covariance matrix is the same for all the classes; however, the means are obviously disctinct in each case. This is referred to as *weight typing* or *parameter sharing*.

Let us write the probability that $y = c$ for some new datapoint $\mathbf{x}$ in this cse:

$$p(y = c \mid \mathbf{x}, \boldsymbol{\theta}) \propto \pi_c \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_c)^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}_c)\right) \tag{18}$$

$$= \exp\left(\boldsymbol{\mu}_c^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_c^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c\right) \cdot \exp\left(-\frac{1}{2}\mathbf{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{x}\right) \tag{19}$$



(a) (b)

Figure 1: Class boundaries for (a) QDA and (b) LDA

Above we've replaced an equality by proportionality in (16). The demonimator of (16) does not depend on the specific class c and the term $|2\pi\boldsymbol{\Sigma}_c|^{-\frac{1}{2}}$ can be dropped as we are using the same $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}$ for all the classes. Note that the last term ofthe RHS of (19) does not depend on the any class c, so in this case the class boundaries are actually linear, hence the name (see Fig. 1(b)). If we let,

$$\gamma_c = -\frac{1}{2}\boldsymbol{\mu}_c^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c + \log \pi_c, \qquad\qquad \boldsymbol{\beta}_c = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c$$

we can rewrite (19) as:

$$p(y = c \mid \mathbf{x}_{\text{new}}, \boldsymbol{\theta}) \propto \exp\left(\boldsymbol{\beta}_c^\mathsf{T}\mathbf{x} + \gamma_c\right) \tag{20}$$

And hence,

$$p(y = c \mid \mathbf{x}, \boldsymbol{\theta}) = \frac{\exp\left(\boldsymbol{\beta}_c^\mathsf{T}\mathbf{x} + \gamma_c\right)}{\sum_{c'} \exp\left(\boldsymbol{\beta}_{c'}^\mathsf{T}\mathbf{x} + \gamma_{c'}\right)} =: \text{softmax}(\boldsymbol{\eta})_c \tag{21}$$

where, $\boldsymbol{\eta} = [\boldsymbol{\beta}_1^\mathsf{T}\mathbf{x} + \gamma_1, \cdots, \boldsymbol{\beta}_C^\mathsf{T}\mathbf{x} + \gamma_C]$.

Softmax

Softmax maps a set of numbers to a probability distribution with mode at the maximum. Because of the exponentiation, it is invariant to translation of the numbers, but not to scaling. Two examples are included below to clarify this point.

$$\text{softmax}([1, 2, 3]) \approx [0.090, 0.245, 0.665]$$
$$\text{softmax}([10, 20, 30]) \approx [2 \times 10^{-9}, 4 \times 10^{-5}, 1]$$

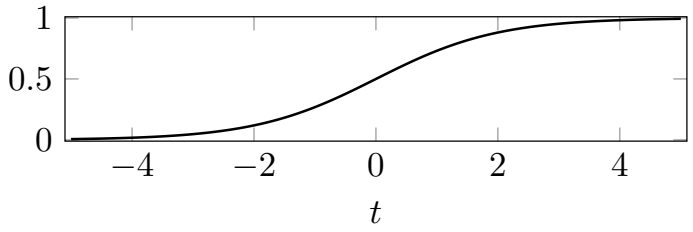


Figure 2: The sigmoid function.

3.3.1 Two-Class LDA

In the special case, where there are only two classes, we get a particularly simple form. Let us refer to these two classes as 0 and 1.

$$\begin{aligned} p(y = 1 \mid \mathbf{x}, \theta) &= \frac{\exp\left(\boldsymbol{\beta}_1^\mathsf{T}\mathbf{x} + \gamma_1\right)}{\exp\left(\boldsymbol{\beta}_1^\mathsf{T}\mathbf{x} + \gamma_1\right) + \exp\left(\boldsymbol{\beta}_0^\mathsf{T}\mathbf{x} + \gamma_0\right)} \\ &= \frac{1}{1 + \exp\left(-((\boldsymbol{\beta}_1 - \boldsymbol{\beta}_0)^\mathsf{T}\mathbf{x} + (\gamma_1 - \gamma_0))\right)} \\ &=: \text{sigmoid}((\boldsymbol{\beta}_1 - \boldsymbol{\beta}_0)^\mathsf{T}\mathbf{x} + (\gamma_1 - \gamma_0)) \end{aligned}$$

Sigmoid Function

The sigmoid function is defined as:

$$\text{sigmoid}(t) = \frac{1}{1 + e^{-t}}$$

The sigmoid function is monontone and maps every real number to a number between 0 and 1; hence it is useful as a means to map arbitrary functions to probabilities. The shape of the sigmoid is shown in Figure 2.

3.4 Discussion

The Gaussian discriminant analysis is another generative model, which may be more suitable to model data in certain cases. For example, if the classes are zebras and giraffes and measurements are height and weight of the animals, then a Gaussian distribution may be better-suited to model the class conditional distributions than the Naïve Bayes approach (clearly height and weight are not conditionally independent given the type of animal).

However, when using these models in high dimensions there may be some concern of overfitting. The number of parameters grows as roughly $C \cdot D^2$ for C classes and D features. Some approaches to reduce overfitting are (i) using a diagonal covariance matrix (this is equivalent to Naïve Bayes), (ii) using the same covariance matrix for all classes (LDA). The Bayesian approach of using priors on the parameters can also reduce overfitting, however, we will not cover this in the course.

The name 'discriminant' is a bit unfortunate in this context. These are *generative* models, not *discriminative* ones. The term simply refers to the shapes of the boundaries that discriminate between the classes when using these models. In the next lecture, we will study *discriminative* models for classification.

References

Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Kevin P. Murphy. *Machine Learning : A Probabilistic Perspective*. MIT Press, 2012.