Introduction

References

http://cs229.stanford.edu/notes/cs229-notes2.pdf (http://cs229.stanford.edu/notes/cs229-notes2.pdf)
http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf (http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf)
https://stats.stackexchange.com/questions/12421/generative-vs-discriminative
(https://stats.stackexchange.com/questions/12421/generative-vs-discriminative)

General definitions

- Discriminative models learn the (hard or soft) boundary between classes (not necessarily in a probabilistic manner).
- Generative models model the distribution of individual classes.

With these general definitions, discriminative models include all other models which are not generative models. Support vector machine (SVM) and decision trees (DTs) are discriminative because they learn explicit boundaries between classes, even though are not probabilistic models at all. SVM and is a maximal margin classifier, meaning that it learns a decision boundary that maximizes the distance between samples of the two classes, given a kernel. The distance between a sample and the learned decision boundary can be used to make the SVM a "soft" classifier. Perceptron is also a discriminative algorithm. DTs learn the decision boundary by recursively partitioning the space in a manner that maximizes the information gain (or another criterion). However, there are no model of generating new data points for these discriminative algorithms.

Probabilistic definitions

A Discriminative model learns the conditional probability distribution $P(Y \mid X)$ **directly**. Here are the key points:

- Assume some functional form for $P(Y \mid X)$.
- Estimate parameters of $P(Y \mid X)$ directly from training data, e.g., by maximum likelihood estimation (MLE)

Unlike discriminative learning algorithm directly learns conditional probability, a generative learning algorithm learns the joint probability distribution $P(X, Y)$. It predicts (not learns by optimization) the conditional probability $P(Y \mid X)$ with the help of Bayes Theorem. Moreover, a generative model explicitly models the actual distribution of each class, while discriminative algorithm models the decision boundary between the classes. In other words, a generative algorithm models how the data was generated in order to categorize a signal.

- Assume some functional forms for $P(Y)$ and $P(X \mid Y)$.
- Estimate parameters of $P(Y)$ and $P(X \mid Y)$ directly from training data. This is equivalent to estimate $P(X, Y)$.
- Use Bayes rule to calculate $P(Y \mid X)$. For classification we often just compare $P(X \mid Y)P(Y)$ for different classes without calculating $P(Y \mid X)$.

Note that generative algorithms have discriminative properties, since we can get $P(Y \mid X)$ once we have $P(Y)$ and $P(X \mid Y)$. However, discriminative algorithms don't really have generative properties.

Examples on how to determine decision boundary

- We are going to understand how discriminative models learn the decision boundary (or learns the conditional probability distribution, or decision boundary) **directly** with a logistic regression example. We assume a sigmoid hypothesis function $P(y|x,\theta) = h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$. We learn this conditional probability (or decision boundary, or just θ...) directly as we learn the parameter θ DIRECTLY from maximizing the likelihood. Once we know the θ, we can assume for example $y = 1$ for $h_\theta(x) \geq 0.5$. From this, we obtain $\theta^T x \geq 0$ and thus obtain a decision boundary given by $\theta^T x = 0$.
- Generative algorithms do not learn the decision boundary. Instead they learn the joint probability $P(x, y)$ directly from data. Then it calculates (not learns) decision boundary using Bayes rule. In other words, it calculates $p(y|x,\theta)$ through the Bayes rule, rather than learns it directly.

Discriminative learning algorithm

As mentioned earlier, all algorithms that do not belong to generative algorithm can be categorized as discriminative algorithms. Here we take a big family of probabilistic models as examples to explain discriminative algorithm.

Generalized Linear Models (GLMs)

http://cs229.stanford.edu/notes/cs229-notes1.pdf (http://cs229.stanford.edu/notes/cs229-notes1.pdf)
https://en.wikipedia.org/wiki/Generalized_linear_model#General_linear_models
(https://en.wikipedia.org/wiki/Generalized_linear_model#General_linear_models).

Introduction

Many machine learning algorithms such as linear regression, logistic regression (or its generalized multi-class softmax regression) etc., belong to the same family of GLMs. Based on the GLMs, the above algorithms can be derived from normal, Bernoulli and multinomial distributions respectively. More algorithms can also be derived from binomial, Poisson, gamma, exponential, beta and Dirichlet etc. Although this family of algorithms are called linear models, they can be used to efficiently handle nonlinear problems with Kernel tricks, even though not as efficient as the support vector machine algorithm.

The exponential family

An exponential family of distribution is defined as

$$p(y;\eta) = b(y)\exp(\eta^T T(y) - a(\eta)) \tag{1}$$

Here, η is called the natural parameter or canonical parameter of the distribution; $T(y)$ is the sufficient statistic. For the distributions considered here, it will often be the case that $T(y) = y$. $a(\eta)$ is the log partition function. The quantity $e^{-a\eta}$ essentially plays the role of a normalization constant for the distribution.

Consider Bernoulli distribution

$$\begin{aligned} p(y;\phi) &= \phi^y(1-\phi)^{1-y} \\ &= \exp\left(y\log\phi + (1-y)\log(1-\phi)\right) \\ &= \exp\left(\left(\log\frac{\phi}{1-\phi}\right)y + log(1-\phi)\right) \end{aligned}$$

Compare with the general family distribution Eq.(1), we know that the natural parameter is given by $\eta = \log\left(\frac{\phi}{1-\phi}\right)$.If we solve this equation for ϕ, we obtain $\phi = \frac{1}{1+e^{-\eta}}$, which is the familiar sigmoid function. We also have $T(y) = y$, $a(\eta) = -\log(1-\phi) = \log(1+e^{\eta})$, $b(y) = 1$. This shows that the Bernoulli distribution can be written in the form of Equation (1), using an appropriate choice of T, a and b.

Constructing GLMs

Suppose you would like to build a model to estimate the number y of customers arriving in your store (or number of page-views on your website) in any given hour, based on certain features x such as store promotions, recent advertising, weather, day-of-week, etc. We know that the Poisson distribution usually gives a good model for numbers of visitors. Knowing this, how can we come up with a model for our problem? Fortunately, the Poisson is an exponential family distribution, so we can apply a Generalized Linear Model (GLM). More generally, consider a classification or regression problem where we would like to predict the value of some random variable y as a function of x. To derive a GLM for this problem, we will make the following three assumptions about the conditional distribution of y given x and about our model:
variable y as a function of x. To derive a GLM for this problem, we will make the following three assumptions about the conditional distribution of y given x and about our model:

- (1) $y \mid x; \theta \sim$ ExponentialFamily(η). That is, given x and θ, the distribution of y follows some exponential family distribution, with parameter η.
- (2) Given x, our goal is to predict the expected value of $T(y)$ given x. In most of our examples, we will have $T(y) = y$, so this means we would like the prediction $h(x)$ output by our learned hypothesis h to satisfy $h(x) = E[T(y) \mid x] = E[y \mid x]$. **Note this expectation version of hypothesis is an assumption but not derived from something rigorous**.
- (3) The natural parameter η and the inputs x are related linearly: $\eta = \theta^T x$. (Or, if η is vector-valued, then $\eta_i = \theta_i^T x$)

The third of these assumptions might seem the least well justified of the above, and it might be better thought of as a 'design choice' in our recipe for designing GLMs, rather than as an assumption per se. These three assumptions/design choices will allow us to derive a very elegant class of learning algorithms, namely GLMs, that have many desirable properties such as ease of learning.

Ordinary Least Squares

To show that ordinary least squares is a special case of the GLM family of models, consider the setting where the target variable y (also called the response variable in GLM terminology) is continuous, and we model the conditional distribution of y given x as as a Gaussian $N(\mu, \sigma^2)$. Here, μ may depend x. So, we let the ExponentialFamily(η) distribution above be the Gaussian distribution. As we saw previously, in the formulation of the Gaussian as an exponential family distribution. In this case, we may find that we had $\mu = \eta$. So, we have $h_\theta(x) = E[y \mid x; \theta] = \mu = \eta = \theta^T x$
The first equality follows from Assumption 2, above; the second equality follows from the fact that $y|x; \theta \in N(\mu, \sigma^2)$, and so its expected value is given by μ; the third equality follows from Assumption 1 (we may find $\mu = \eta$ when mapping Gaussian to the exponential family distribution); and the last equality follows from Assumption 3.

Comments:

- In the linear regression case, $y \mid x$ is a Gaussian and $h_\theta(x) = E[y \mid x; \theta] = \theta^T x$ This means the **hypothesis function** $h_\theta(x)$ is the expectation of random variable y conditioned at x. For normal distribution, this expectation is just $\theta^T x$ which give a line for 2D case. Imagine how the result at each x is the expectation of y. Because the expectation is calculated, if there are big outliers there for y, then the results will be significantly skewed.
- From $h_\theta(x) = \theta^T x$, we know the **response function** g is just identity function. Compare hypothesis function $h_\theta(x)$, $E[y \mid x; \theta]$, and response function. The former two are same.

Logistic Regression

Here we are interested in binary classification, so $y \in \{0, 1\}$. Given that y is binary-valued, it therefore seems natural to choose the Bernoulli family of distributions to model the conditional distribution of y given x. In our formulation of the Bernoulli distribution as an exponential family distribution, we had $\phi = \frac{1}{1+e^{-\eta}}$. Furthermore, note that if $y|x; \theta \in \text{Bernoulli}(\phi)$, then $E[y|x; \theta] = \phi$. So, following a

similar derivation as the one for ordinary least squares, we get: $h_\theta(x) = E[y \mid x; \theta] = \phi = \frac{1}{1+e^{-\eta}} = \frac{1}{1+e^{-\theta^T x}}$. Once we assume that y conditioned on x is Bernoulli, then the above sigmoid hypothesis function arises as a consequence of the definition of GLMs and exponential family distributions.
To introduce a little more terminology, the function g giving the distribution's mean as a function of the natural parameter ($g(\eta) = E[T(y); \eta]$) is called the canonical response function. Its inverse, g^{-1}, is called the canonical link function. Thus, the canonical response function for the Gaussian family is just the identify function, as $y = E(y \mid x) = g(\eta) = \eta = \theta^T x$ So in Gaussian family y and x are really in linear relation. The canonical response function for the Bernoulli is the logistic function, $y = g(\eta) = \frac{1}{1+e^{-\eta}}$. Here y is linear with $g(\eta)$ but not η.

Comments:

- In the logistic regression case, $y \mid x$ is a Bernoulli and $h_\theta(x) = E[y \mid x; \theta] = \frac{1}{1+e^{-\theta^T x}}$. Due to the different population distribution, here the **response function** is no longer the identity function as in linear regression, but a nonlinear logistic function. As the **hypothesis function** $h_\theta(x)$ is still the expectation of random variable y conditioned at x, each value we obtained in the usual output in logistic regression at each x is just the expectation value of y conditioned at x. The only difference here from linear regression is the population distribution is no longer normally distributed, but Bernoulli distributed.
- Unlike linear regression case, we can say that hypothesis function, response function, and $E[y \mid x; \theta]$ are all same.

Softmax regression

While Bernoulli distribution gives logistic regression from constructing GLM, the general multinomial distribution will give Softmax regression, which is a generalized version of logistic regression.
Consider a classification problem in which the response variable y can take on any one of k values (multi-class), so $y \in \{1, 2, \ldots, k\}$. Use k parameters $\phi_1, \phi_2, \ldots \phi_k$ to specify the probability of each of k outcomes, where only $k-1$ of them are independent. We thus parameterize the multinomial with only $k-1$ parameters, $\phi_1, \phi_2, \ldots \phi_{k-1}$, where $\phi_i = p(y = i; \phi)$, and $p(y = k; \phi) = 1 - \sum_{i=1}^{k-1} \phi_i$. For notational convenience, we will also let $\phi_k = 1 - \sum_{i=1}^{k-1} \phi_i$, but we should keep in mind that this is not a parameter because it can be fully specified by other parameters $\phi_1, \phi_2, \ldots \phi_{k-1}$. To express the multinomial as an exponential family distribution, we will define $T(y) \in \mathbb{R}^{k-1}$ as follows:

$$T(1) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, T(2) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, T(3) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \ldots T(k-1) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}, T(k) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Here we do not have $T(y) = y$ and $T(y)$ is now a $k-1$ dimensional vector, rather than a real number. We will write $(T(y))_i$ to denote the ith element of the vector $T(y)$. We further introduce indicator function $1\{.\}$ which takes on a value of 1 if its argument is true, and 0 otherwise. That is, $1\{True\} = 1, 1\{False\} = 0$. So, we can also write the relationship between $T(y)$ and y as $(T(y))_i = 1\{y = i\}$. Further, we have that $E[(T(y))_i] = P(y = i) = \phi_i$. **The symbol $(T(y))_i$ is just like a random varible?**

We now show the multinomial belongs to the exponential family

$$\begin{aligned}
p(y,\phi) &= \phi_1^{1\{y=1\}}\phi_2^{1\{y=2\}}\dots\phi_k^{1\{y=k\}} \\
&= \phi_1^{1\{y=1\}}\phi_2^{1\{y=2\}}\dots\phi_k^{1-\sum_{i=1}^{k-1}1\{y=i\}} \\
&= \exp\left[(T(y))_1\log\phi_1 + (T(y))_2\log\phi_2 + \dots + \left(1 - \sum_{i=1}^{k-1}(T(y))_i\right)\log\phi_k\right] \\
&= \exp\left[(T(y))_1\log\frac{\phi_1}{\phi_k} + (T(y))_2\log\frac{\phi_1}{\phi_k} + \dots + (T(y))_{k-1}\log\frac{\phi_{k-1}}{\phi_k} + \log\phi_k\right] \\
&= b(y)\exp(\eta^T T(y) - a(\eta))
\end{aligned}$$

where

$$\begin{aligned}
\eta &= \begin{bmatrix} \log\frac{\phi_1}{\phi_k} \\ \log\frac{\phi_2}{\phi_k} \\ \vdots \\ \log\frac{\phi_{k-1}}{\phi_k} \end{bmatrix}, \\
a(\eta) &= \log(\phi_k), \\
b(y) &= 1
\end{aligned}$$

Thus the link function is given by

$$\eta_i = \log\frac{\phi_i}{\phi_k}, i = 1, 2, \dots k$$

For convenience we have also defined $\eta_k = \log\frac{\phi_k}{\phi_k} = 0$. To invert the link function and derive the response function, we therefore have

$$\begin{aligned}
e^{\eta_i} &= \frac{\phi_i}{\phi_k} \\
\phi_k e^{\eta_i} &= \phi_i \\
\phi_k\sum_{i=1}^{k} e^{\eta_i} &= \sum_{i=1}^{k}\phi_i = 1
\end{aligned} \tag{2}$$

This implies that $\phi_k = \frac{1}{\sum_{i=1}^{k} e^{\eta_i}}$, which can be substituted back into Eq.(2) to give the response function

$$\phi_i = \frac{\eta_i}{\sum_{i=1}^{k} e^{\eta_i}}$$

This function mapping from the η to the ϕ is called the softmax function. **Sigmoid is a special case of this softmax function**.

To complete our model, we use Assumption 3, given earlier, that the η are linearly related to the x. So, have $\eta_i = \theta_i^T x, i = 1, \ldots, k-1$. For notational convenience, we can also define $\theta_k = 0$, so that $\eta_k = \theta_k^T x = 0$, as given previously. Hence, our model assumes that the conditional distribution of y given x is given by

$$\begin{aligned} p(y = i \mid x, \theta) &= \phi_i \\ &= \frac{\eta_i}{\sum_{i=1}^{k} e^{\eta_i}} \\ &= \frac{\theta_i^T x}{\sum_{i=1}^{k} e^{\theta_i^T x}} \end{aligned}$$

This model, which applies to classification problems where $y \in \{1, \ldots, k\}$, is called softmax regression. It is a generalization of logistic regression. Our hypothesis will output

$$\begin{aligned} h_\theta(x) &= E[T(y) \mid x; \theta] \\ &= E\begin{bmatrix} 1\{y = 1\} \\ 1\{y = 2\} \\ \vdots \\ 1\{y = k-1\} \end{bmatrix} \Bigg| \; x; \theta \Bigg] \\ &= \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{k-1} \end{bmatrix} \end{aligned}$$

where ϕ_i are given earlier. Note we only calculate the first $k-1$ of $\phi's$, and ϕ_k is calculated from other $\phi's$. Once we obtain this hypothesis function, then we can use MLE to estimate parameters.

For multi-class classification problems, we can use one-versus-all approach, or softmax regression.

Linear vs nonlinear algorithm

- In the GLMs introduced above, it is because we assume that the natural parameter η and the inputs x are related linearly: $\eta = \theta^T x$, that we call this family of models generalized LINEAR models. As shown in the earlier example of logistic regression, the decision boundary is determined by $\eta = \theta^T x = \text{constant}$. So in the GLMs, although the **response function** g might be NONLINEAR function (for logistic regression g is sigmoid/logistic function, while for linear regression g is identity function), the decision boundary is linear.
- For the generative learning algorithm (see next section), the linearity or nonlinearity of the decision is a little bit complicated. See the link https://stats.stackexchange.com/questions/142215/how-is-naive-bayes-a-linear-classifier (https://stats.stackexchange.com/questions/142215/how-is-naive-bayes-a-linear-classifier).
  - In general the naive Bayes classifier is not linear, but if the likelihood factors $p(x_i \mid c)$ are from exponential families, the naive Bayes classifier corresponds to a linear classifier in a particular feature space.
  - **Comments:** However, when the likelihood is Gaussian, which belongs to exponential family, the decision boundary should not be linear as shown in many posts. This might be related to whether the Gaussians share same covariance matrix. As one comment from the link: "It is linear only if the class conditional variance matrices are the same for both classes. To see this write down the ratio of the log posteriors and you'll only get a linear function out of it if the corresponding variances are the same. Otherwise it is quadratic." This might also be related to the difference of linear discriminant analysis (with linear decision boundary) and quadratic discriminant analysis (with quadratic decision boundary).
- There are a few ways to handle nonlinear problems.
  - In the $\eta = \theta^T x$, if we include e.g. $x^2, x^3, \ldots$, then $\theta^T x$ will be a nonlinear function. This will make all the algorithms in GLMs capable of handling nonlinear problems. For example, we may have nonlinear (such as circular, or curved) decision boundaries.
  - Another way of making GLMs nonlinear is to use the kernel trick (from SVM). For example, we can have kernel regression methods. In fact, the same trick can also turn another family of algorithms called Generalized discriminant analysis (will be introduced later) into its nonlinear counterparts. Linear SVM plus kernel trick will also make SVM very powerful in handling very strong nonlinear problems.
  - Some intrinsic nonlinear algorithms include most non-parametric algorithms such as decision tree models, instance based models, KNN, neural networks, etc.

Side notes:

- https://www.quora.com/Whats-the-difference-between-linear-and-non-linear-machine-learning-model (https://www.quora.com/Whats-the-difference-between-linear-and-non-linear-machine-learning-model) Linear models deal with modeling correlation, that is, noting what things occur together, and drawing inferences about how likely or unlikely things are to happen given how much they have happened together in the past. Second order linear modeling incorporates information about indirect relationships through a chain of co-occurrences. Some models (like LSA) incorporate inferences based on many orders of co-occurrence. However, correlation is not causality, and there are situations that cannot be modeled by simple correlation. Linear models also fail when the order in which things occur matters (i.e. where the probability of A, B is not related to the probability of B, A).
- Why unsupervised learning technique K-means is linear algorithm? See details in https://stats.stackexchange.com/questions/53305/why-is-the-decision-boundary-for-k-means-clustering-linear (https://stats.stackexchange.com/questions/53305/why-is-the-decision-boundary-for-k-means-clustering-linear)

Generative learning algorithm

Generalized Discriminant Analysis

Introduction

In analogous to the GLMs in the discriminative algorithms, in generative learning algorithms there is a big family of algorithms called generalized discriminant Analysis (GDA). However, the same Acronym GDA sometimes refers to a general term for LDA (Linear Discriminant Analysis) and QDA (Quadratic Discriminant Analysis). LDA is a case where each observation is drawn from multivariate Gaussian distribution with class-specific mean vector and common covariance matrix, while QDA is similar except the class-specific covariance matrix. Even more confusing, GDA also refers to Gaussian discriminant analysis, which is often equivalent to LDA. Moreover, the term Fisher's linear discriminant and LDA are often used interchangeably. In this write-up, we will only introduce two important generative learning algorithms: Gaussian discriminant analysis and Naïve Bayes. For a broader concept of LDA for classification, dimension reduction, and its relation to principal component analysis (PCA) and factor analysis (FA), see other write up about LDA and PCA.

Gaussian Discriminant Analysis (GDA)

- Assuming the data is generated from a Gaussian distribution with some unknown parameters.
$$P(x \mid y = 0) \sim N(\mu_0, \Sigma)$$
$$P(x \mid y = 1) \sim N(\mu_1, \Sigma)$$
Using MLE on the data set, we can thus estimate the parameters μ_0, μ_1, Σ. Thus distributions $P(x \mid y = 0)$ and $P(x \mid y = 1)$ are fully determined.
- We also know $P(y = 0)$ and $P(y = 1)$. Using the early determined $P(x \mid y = 0)$ and $P(x \mid y = 1)$, we obtain two joint probabilities:
$$P(x \mid y = 0)P(y = 0) = P(x, y = 0)$$
$$P(x \mid y = 1)P(y = 1) = P(x, y = 1)$$
- For a test data $x = x_0$, we compare $P(x_0, y = 0)$ and $P(x_0, y = 1)$ to determine whether x_0 belongs to $y = 0$ or $y = 1$ class.

Naïve Bayes (NB)

- Like GDA, NB is a generative algorithm except that we assume a different distribution to generate data. We also compare $P(x_0, y = 0)$ and $P(x_0, y = 1)$ to determine whether x_0 belongs to $y = 0$ or $y = 1$ class.

- The key approximation leading to the term naïve.

$$\begin{aligned} P(X \mid y)P(y) &= P(x_1, x_2, \ldots x_n|y)P(y) \quad \text{(joint probability. Not naïve so far)} \\ &= P(x_1 \mid y)P(x_2 \mid y)\ldots P(x_n \mid y) * P(y) \quad \text{(Conditional independence: naïve)} \end{aligned}$$

- Normally n outcomes will need n parameters. However, due to the conditional independent approximation, the probability here and those of general binomial or multinomial distributions only have one or $k-1$ independent parameters respectively.
- Depending the specific form of $P(x_i \mid y)$, we can have (https://en.wikipedia.org/wiki/Naive_Bayes_classifier (https://en.wikipedia.org/wiki/Naive_Bayes_classifier))
  - Gaussian NB
  - multinomial NB
  - Bernoulli NB
  - Semi-supervised parameter estimation.

- GDA/LDA is closely related to Gaussian NB in that both classifiers assume Gaussian within-class distributions. However, NB relies on a less flexible distributional model in that it assumes zero off-diagonal covariance.
- Some details of derivations on a Bernoulli NB, Laplace smoothing, event models can be seen in the cs229 notes and the problem set.

Compare generative and discriminative algorithms

An intuitive picture connecting GDA and logistic regression

Here we see an intuitive example for connecting generative GDA to discriminative logistic regression. For a set of 1D training data set, two Gaussian like data centered on a low and high x respectively along the x-axis. The high x data correspond to, e.g. a tumor data while the small x data corresponds to benign data. Assume the tumor class is for $y=1$ and we calculate its probability with

$$p(y=1 \mid x) = \frac{p(x \mid y=1)p(y=1)}{p(x)}$$

$$p(x) = p(x \mid y=1)p(y=1) + p(x \mid y=0)p(y=0)$$

From the picture and the equation, $p(y=1 \mid x)$ will be very small in the left of x-axis. As x increases, $p(y=1 \mid x)$ will reach one and saturates when x is very big. This shape is like the sigmoid function in logistic regression. Therefore, there is strong relation between GDA and logistic regression. For example, in the far right, $p(x \mid y=1)$ can becomes much smaller. However, the $p(x)$ also becomes much smaller, so $p(y=1 \mid x)$ is still big.

A mathematical connection between GDA and logistic regression

If we assume Gaussian distribution for $p(x \mid y)$, then it is possible to write $p(y = 1 \mid x)$ in the format of sigmoid function. See details in the problem set of cs229 notes (available from internet).

Choosing generative or discriminative?

- When generative models will be more appropriate?
  - Generative algorithm learns joint probability distribution $P(x, y)$. If a data set approximately follows the assumed distributions, a generative model often outperforms a discriminative model.
  - For a small data set, while a discriminative algorithm such as logistic regression might pick up on spurious patterns that don't really exist, a generative algorithm such as Naive Bayes can act as a kind of regularizer to prevent overfitting, as the CORRECT assumptions place some structure on your model.
  - When a data set follows the assumptions of GDA (e.g. Gaussian distributions and common covariance matrix), then GDA is asymptotically more efficient than its discriminative counterpart: logistic regression. In other words, it requires less data to learn well.
- When discriminative models will be more appropriate?
  - Unlike making much stronger model assumptions such as the full probability distribution $P(x, y)$ in a generative model, a discriminative model makes significantly weaker assumptions. It only focus on splitting class variable y without knowing full information about feature x. Different x distributions such as a Gaussian or Poisson distribution can lead to the same logistic regression.
  - This less restrictive assumption makes the logistic regression more robust and less sensitive to incorrect modeling assumptions. So when a data set does not follow our distribution assumptions, logistic regression will almost always outperform generative models, especially in the limit of large datasets.
- What happens when we give correlated features?
  - One of the main Gaussian Naive Bayes (GNB) modeling assumptions is that the features are independent of each other, but we often see that it performs very well on small datasets even when some features seem correlated.
  - One explanation could be that GNB converges towards its asymptotic accuracy at a different rate than other methods (e.g. logistic regression). As Ng & Jordan show, GNB parameter estimates converge toward their asymptotic values in order $\log n$ examples, where n is the dimension of X. In contrast, Logistic Regression parameter estimates converge more slowly, requiring order n examples.
  - When features (not include target) are correlated, discriminative models such as logistic regression may have big variance. In generative algorithm, how this is overcomed?
- Which model will take less time to get trained?
  - Usually generative models converge fast.
- Which model prone to overfitting very easily and which model prone to underfitting (high bias)?

    - The Naive Bayes classifier employs **a very simple (linear) hypothesis function**. As a result it suffers from high bias. On the other hand, it exhibits low variance.
    - As a result of this attribute Naive Bayes classifier has been shown to perform surprisingly well with very small amounts of training data than most other classifiers.
- Which model fails to work well if we give a lot of features?
    - Because discriminative approach is prone to overfitting as compared to generative approach, we should apply generative models when there are a lot of features.
- Which model is more sensitive to outliers?
    - Naive Bayes is robust to isolated noise/outliers.
    - Missing values are easily handled. They are simply ignored.
    - It is robust to irrelevant attributes.