Mathematics of Big Data I

Lecture 3: Schur Complement,Multivariate Gaussian Distribution, Generative Learning

September 19th, 2016

Recall from last time that when dealing with big data, the closed formula involving an inverse of a huge matrix may not be fast or stable if we try to invert the matrix directly. Therefore we need to try other computational methods, such as the Cholesky decoposition, gradient descent, stochastic gradient descent, and Newton's method. Recall that there are several different ways we can go about the Cholesky decomposition. For instance, we can see this simply as elementary row operations:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ -3 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 2 \\ 0 & -2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & -2 \end{bmatrix}$$

where the final result is an LU-decomposition. In the special case that our original matrix is symmetric, we can get both the lower and upper parts to be the same. Another way of doing this is simply to use "brute force," and lastly we can also use symmetric matrices.

Theorem 1 *If A is an $n \times n$ real, symmetric positive-definite matrix, then there exists a unique lower triangular matrix G with positive diagonal elements such that $A = GG^T$.*

Proof: Observe that if we can uniquely decompose $A = LDU$, then we have that $A^T = A = U^T DL^T$ proving that $L = U^T$ and therefore $A = LDL^T$. Also if A is positive-definite, we can take the square-root of the diagonal matrix D by simply taking the square-root of all of the diagonal entries. Then we can write $A = L\sqrt{D}\sqrt{D}L^T = GG^T$ where we let $G = L\sqrt{D}$. □

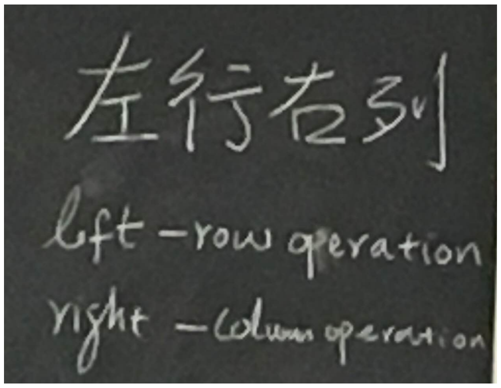
Note that if $Ax = b$, we can solve this by back-substitution, but by using the Cholesky decomposition we can see back substitution as a faster way to do it.

To solve the Cholesky decomposition by brute force, we simply assume that

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & u_{nn} \end{bmatrix} = LU.$$

In regression it is often the case that $X^T X$ is positive-definite.

1 The Schur Complement


左行右列

left - row operation

right - column operation

This is related to how we triage data and solve a smaller problem involving big data first. Let A be a symmetric positive-definite matrix. Then A can be diagonalized via an orthogonal matrix, so there exists a matrix P such that $P^{-1} = P^T$ and $A = PDP^T$ where D is diagonal, containing the eigenvalues of A. Since A is positive-definite then each eigenvalue $\lambda_i > 0$ where $i \in \{1, \dots, n\}$. We then see we can write

$$A = \underbrace{P\begin{bmatrix} \sqrt{\lambda_1} & & & \\ & \sqrt{\lambda_2} & & \\ & & \ddots & \\ & & & \sqrt{\lambda_n} \end{bmatrix}}_{=G} \underbrace{\begin{bmatrix} \sqrt{\lambda_1} & & & \\ & \sqrt{\lambda_2} & & \\ & & \ddots & \\ & & & \sqrt{\lambda_n} \end{bmatrix} P^T}_{=G^T} = GG^T.$$

This is related to the **Pfaffian decomposition**, where since we have $A = P\sqrt{D}\sqrt{D}P^T$ we can stick another copy of $I = PP^T$, so we can write

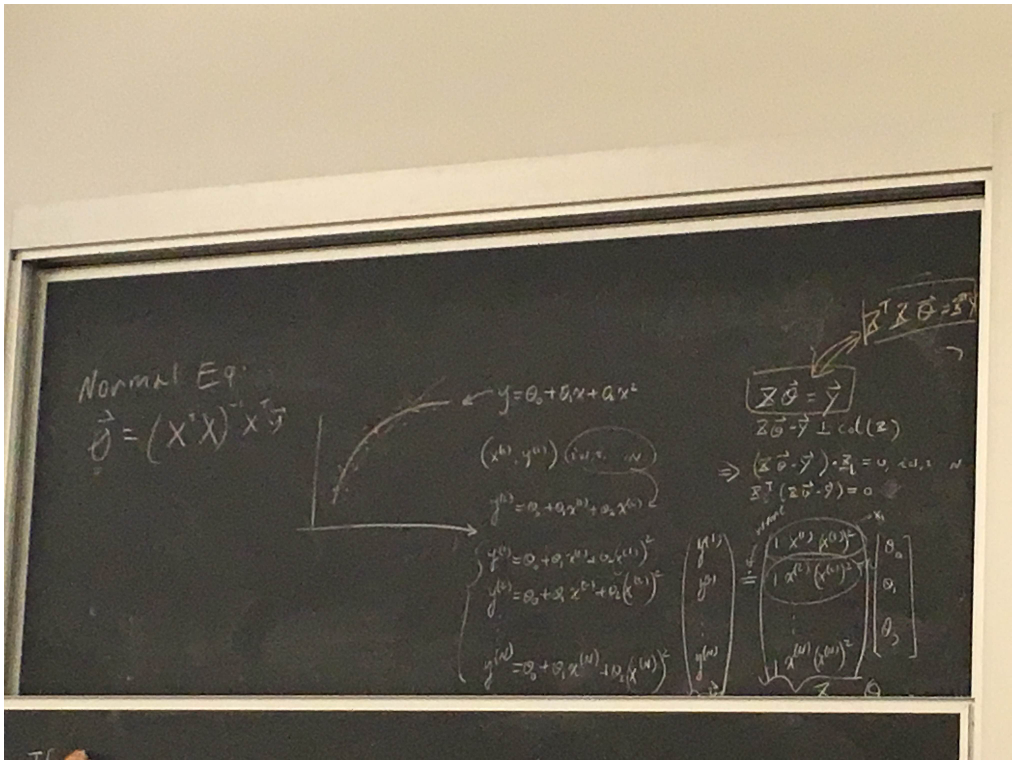
$$A = P\sqrt{D}PP^T\sqrt{D}P^T = QQ = Q^2$$

where $Q = P\sqrt{D}P^T$. The matrix Q is the Pfaffian decomposition.

Moving forward, let's discuss an example which will illustrate how we want to deal with big data. Recall the normal equation from linear regression:

$$\vec{\theta} = (X^TX)^{-1}X^T\vec{y}$$

where the vector $\vec{\theta}$ contains the parameters we want to fit. Even if we want to fit a parabola, we can still use the normal equation since we can write our equations to be linear in the parameters we want to learn, even if the equation we are modeling is nonlinear. For instance, we could be finding the best parabolic equation $y = \theta_0 + \theta_1 x + \theta_2 x^2$ given data X and targets y.



Normal Eq:
$\vec{\theta} = (X^T X)^{-1} X^T \vec{y}$
$y = \theta_0 + \theta_1 x + \theta_2 x^2$
$X\vec{\theta} = \vec{y}$
$X^T X \vec{\theta} = X^T \vec{y}$

We can arrange our various equations into a matrix so that we want to solve $X\vec{\theta} = \vec{y}$. We want a solution to this equation so that $X\vec{\theta} - \vec{y}$ is orthogonal to the column space of X, in other words we would like $X^T(X\vec{\theta} - \vec{y}) = 0$, and thus $X^T X\vec{\theta} = X^T\vec{y}$. In the case where $X^T X$ is positive-definite (which it is likely to be) we can invert it. However, inversion yields a problem if $X^T X$ is a large matrix. In this case we can reasonably not have enough memory to invert. We instead have another approach to this: suppose that $X^T X$ has a block form

$$X^T X = \begin{bmatrix} A & B \\ C & D \end{bmatrix}.$$

Here we assume that A is $p \times p$, B is $p \times q$, C is $q \times p$, and D is $q \times q$, where $p + q = N$. In solving the normal equation we can write

$$X^T X\vec{\theta} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \vec{\beta} \\ \vec{\eta} \end{bmatrix} = \begin{bmatrix} \vec{b} \\ \vec{h} \end{bmatrix}.$$

where $\vec{\theta}$ gets broken up into it's first p components contained in $\vec{\beta}$ and it's last q components in $\vec{\eta}$. Similarly, the vector $X^T\vec{y}$ gets broken up into the first p entries in $\vec{b}$ and the last q entries in $\vec{h}$. This matrix equation is equivalent to solving the system of equations

$$\begin{aligned} A\vec{\beta} + B\vec{\eta} &= \vec{b} \\ C\vec{\beta} + D\vec{\eta} &= \vec{h}. \end{aligned}$$

Suppose now that D is invertible, and let's multiply the second equation by $-BD^{-1}$ and add it to the first equation. We then obtain

$$(A - BD^{-1}C)\vec{\beta} = -BD^{-1}\vec{h} + \vec{b}.$$

Here the matrix $A - BD^{-1}C$ is called the **Schur complement** of D. We thus transformed our problem of inverting $X^T X$ into a smaller problem. First we solve the above equation for $\vec{\beta}$ and then solve for $\vec{\eta}$. There are other ways we can obtain a Schur complement. Initially suppose that X^TX is of the same block form given above. Suppose again that D is invertible. Let's perform some of the following operations: we can multiply column two on the write by $-D^{-1}C$ and add it to the first column of our original matrix. We then obtain

$$\begin{bmatrix} B \\ D \end{bmatrix} \rightarrow \begin{bmatrix} -BD^{-1}C \\ -C \end{bmatrix}$$

which is a column operation, so we obtain

$$\begin{bmatrix} A - BD^{-1}C & B \\ 0 & D \end{bmatrix}.$$

We can actually put X^TX into block diagonal form by the operation

$$\begin{bmatrix} I_p & -BD^{-1} \\ 0 & I_q \end{bmatrix}\begin{bmatrix} A & B \\ C & D \end{bmatrix}\begin{bmatrix} I_p & 0 \\ -D^{-1}C & I_q \end{bmatrix} = \begin{bmatrix} A - BD^{-1}C & 0 \\ 0 & D \end{bmatrix}$$

and therefore we can decompose X^TX as

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} I_p & BD^{-1} \\ 0 & I_q \end{bmatrix}\begin{bmatrix} A - BD^{-1}C & 0 \\ 0 & D \end{bmatrix}\begin{bmatrix} I_p & 0 \\ D^{-1}C & I_q \end{bmatrix}.$$

Similarly one can find the inverse of X^TX now using this Schur complement:

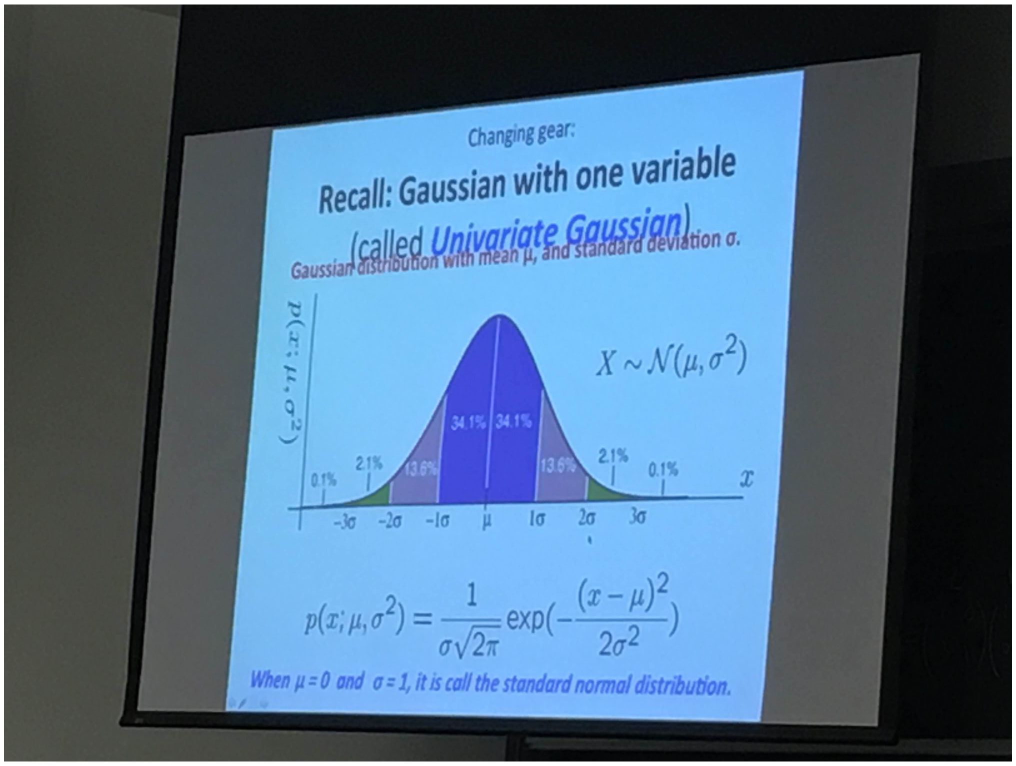
$$\begin{aligned}\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} &= \begin{bmatrix} I_p & 0 \\ -D^{-1}C & I_q \end{bmatrix}\begin{bmatrix} (A - BD^{-1}C)^{-1} & 0 \\ 0 & D^{-1} \end{bmatrix}\begin{bmatrix} I_p & -BD^{-1} \\ 0 & I_q \end{bmatrix} \\ &= \begin{bmatrix} (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1} + D^{-1}C(A - BD^{-1}C)^{-1}BD^{-1} \end{bmatrix}.\end{aligned}$$

This has many applications to probability and statistics. For example, given a covariance matrix Σ we can write

$$\Sigma = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}$$

then we can write $Cov(x|y) = A - BC^{-1}B^T$. When dealing with a cost function, we are always dealing with an optimization problem. We usually like to use a quadratic cost function, but in the case we don't have a quadratic cost function, then we can represent it locally by it's Taylor expansion in which case the Hessian gets involved.

2 Multivariate Gaussian Distribution



Changing gear:

Recall: Gaussian with one variable (called *Univariate Gaussian*)

Gaussian distribution with mean μ, and standard deviation σ.

$p(x;\mu,\sigma^2)$
$X \sim \mathcal{N}(\mu,\sigma^2)$
0.1%
2.1%
13.6%
34.1%
34.1%
13.6%
2.1%
0.1%
x
-3σ
-2σ
-1σ
μ
1σ
2σ
3σ

$$p(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(x-\mu)^2}{2\sigma^2})$$

When $\mu = 0$ and $\sigma = 1$, it is call the standard normal distribution.

The single-variable Gaussian is called the *univariate Gaussian*. Recall that the single-variable Gaussian is given by the probability density

$$p(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

There are different ways to find the expectation for a probability density function f, where the expectaiton is given by

$$E[X] = \int_{-\infty}^{\infty} x f(x) dx.$$

Therea re generally three different ways to compute the expectation:

- Argue the mean is a certain value using symmetry: for instance the standard Gaussian we want to integrate $xe^{-x^2/2}$ which is odd about the origin, hence it is odd and $E[X] = 0$.
- Directly integrate the function.
- Use the moment generating function.

Let's focus on the moment-generating function. Recall that given a random variable X it's moment generating function is given by

$$\phi(t) = E[e^{tX}].$$

For instance, for the Gaussian distribution it's moment generating function is given by $\phi(t) = e^{t^2/2}$. We then write this function into a Taylor series

$$e^{t^2/2} = 1 + \frac{t^2}{2} + \frac{1}{2}\left(\frac{t^2}{2}\right)^2 + \frac{1}{3!}\left(\frac{t^2}{2}\right)^3 + \ldots + \frac{1}{n!}\left(\frac{t^2}{2}\right)^n + \ldots$$

but let's compare this to the Taylor expansion of $E[e^{tX}]$ which is given by

$$E[e^{tX}] = 1 + E[X]t + \frac{1}{2}E[X^2]t^2 + \ldots + \frac{1}{n!}E[X^n]t^n + \ldots$$

We can compare both of these expressions and see that there is no t term in the expansion of $\phi(t)$, which proves that $E[X] = 0$. Now, while this was fine for the standard Gaussian distribution $\mathcal{N}(0,1)$, what if we have a general distribution $X \sim \mathcal{N}(\mu,\sigma^2)$? We can uncover the mean and variance by performing certain tricks. Observe for instance if we wanted to compute the mean $E[X]$ we can see that

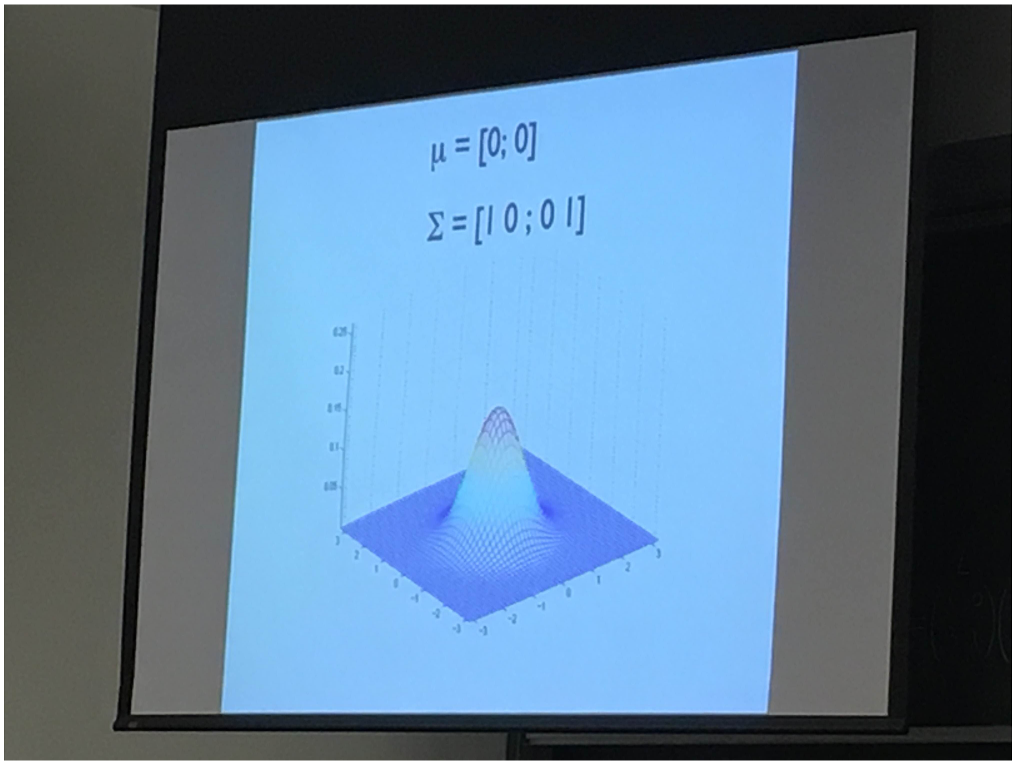
$$\begin{aligned} E[X] &= \int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} x \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \\ &= \underbrace{\int_{-\infty}^{\infty} \frac{x-\mu}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}_{=0} + \mu \underbrace{\int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}_{=1} \\ &= \mu \end{aligned}$$

where the first term is an odd function, while the second equation is just an integral over the density function. Tricks like this abound in probability, and a similar computation can be done to compute the variance of the single-variable Gaussian.

Now when we deal with the multivariate Gaussian we denote this distribution as

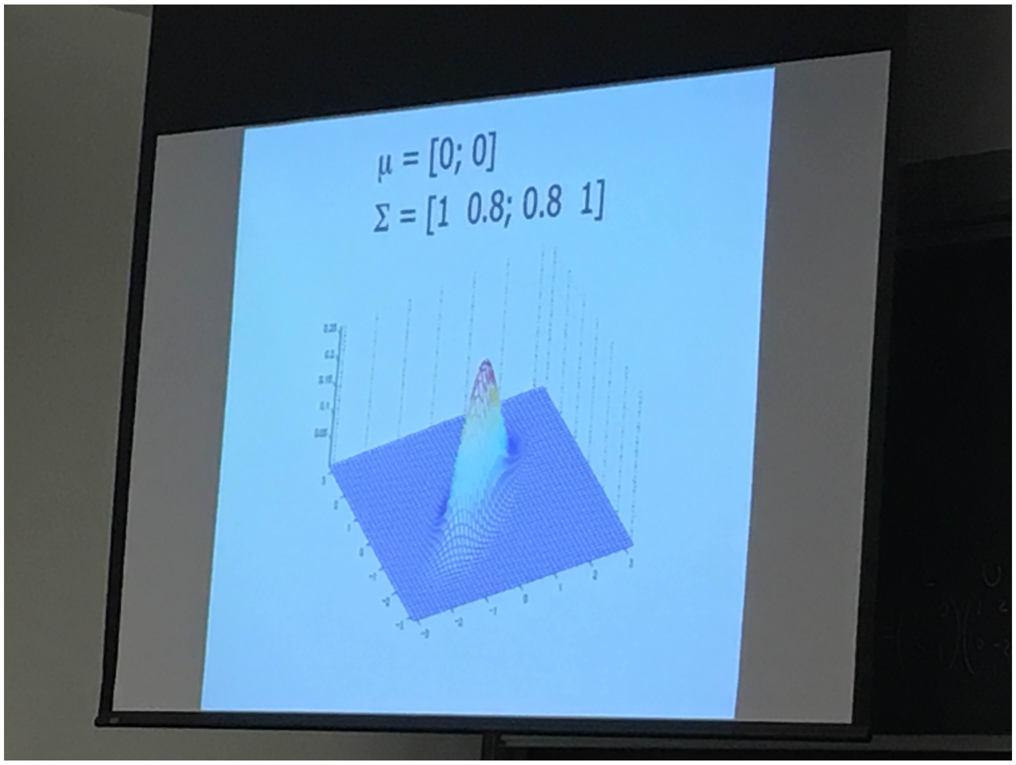
$$\mathcal{N}(x|\mu,\Sigma) \triangleq \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right]$$

where the inverse Σ^{-1} is called the **precision matrix**. Here Σ is the covariance matrix with respect to the entries in x.



$\mu = [0; 0]$

$\Sigma = [1\ 0 ; 0\ 1]$



$\mu = [0; 0]$

$\Sigma = [1\ \ 0.8; 0.8\ \ 1]$

3 Generative Learning Algorithms

We'd like to introduce the concepts behind generative learning algorithms. Let's illustrate this with a real-world problem: suppose we have a body scan of a tumor and we would like to make a reasonable prediction as to whether this tumor is malignant or benign. We have historical data for tumors of some variety, but we would like a conditional probability on classifying a new tumor. We will mainly study a distribution $p(y|x;\theta)$ and try to model this conditional distribution based on the classes y (malignant/benign), previous data x (data for tumors) and a parameter θ (a collection of variables with which we can adjust or "learn" the conditional distribution to model our problem appropriately). Our goal is to learn the true distribution $p(y|x)$ or learn a function $h_\theta(x) \in \{0,1\}$. We will model this using a logistic regression model, so that $h_\theta(x) = g(\theta)$ where $g(\theta)$ was taken as the sigmoid function.

In contrast to this approach, we will try to understand this problem via a *generative learning algorithm*. Here we instead look at the historical data of $p(x|y)$ (the distribution of tumors given that we know if they're benign or malignant). We also look at the distribution for classifying tumors $p(y)$, but our goal still remains to predict whether a new tumor is malignant/benign hence we still want to learn $p(y|x)$. We can use Bayes' rule from probability to write this as

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

where here we have

$$p(x) \quad = \quad p(x|y=1)p(y=1) + p(x|y=0)p(y=0).$$

We therefore model our distribution using Bayes' rule. Here we need to remember certain things from probability theory. Recall that probability is a function on *events* A. For instance, $p(A)$ is the probability that event A is true. The event $\overline{A}$ is the event that A does not occur. It follows that $p(\overline{A}) = 1 - p(A)$.

In order to make a prediction, we want to maximize the likelihood so that

$$\arg\max_y p(y|x) \quad = \quad \arg\max_y \frac{p(x|y)p(y)}{p(x)} \quad = \quad \arg\max_y p(x|y)p(y)$$

where we use the fact that $p(x)$ does not depend on y, and therefore does not contribute to finding the maximum argument. We therefore want to find the variable y such that the conditional probability is maximized.

3.1 Gaussian Discriminant Analysis (GDA)

Our first look at a generative learning algorithm will be using the Gaussian discriminant analysis. We assume that in our problem above that $p(x|y)$ is distributed according to a multivariate normal distribution (MND)

$$p(x|y;\mu,\Sigma) \quad = \quad \frac{1}{(\sqrt{2\pi})^n|\Sigma|^{1/2}}\exp\left[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right] \quad \triangleq \quad \mathcal{N}(\mu,\Sigma).$$

Here the $\mu \in \mathbb{R}^n$ and $\Sigma \in \mathbb{R}^{n\times n}$. We also assume y is a Bernoulli random variable parameterized by ϕ. The GDA model is that $y \sim \mathit{Bernoulli}(\phi)$ and $x|y=0 \sim \mathcal{N}(\mu_0,\Sigma)$ and $x|y=1 \sim \mathcal{N}(\mu_1,\Sigma)$. Notice here that the normal distributions have different means μ_0 and μ_1 but their covariance matrices are the same. Observe that our densities are given by

$$\begin{aligned} p(y) &= \phi^y(1-\phi)^{1-y} \\ p(x|y=0) &= \frac{1}{(\sqrt{2\pi})^n|\Sigma|^{1/2}}\exp\left[-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)\right] \\ p(x|y=1) &= \frac{1}{(\sqrt{2\pi})^n|\Sigma|^{1/2}}\exp\left[-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right]. \end{aligned}$$

We therefore see that the parameters of our model are ϕ, μ_0, μ_1, and Σ. The log-likelihood of the data is given by

$$\begin{aligned} l(\phi,\mu_0,\mu_1,\Sigma) &= \log\left[\prod_{i=1}^{m} p\left(x^{(i)},y^{(i)};\phi,\mu_0,\mu_1,\Sigma\right)\right] \\ &= \log\left[\prod_{i=1}^{m} p\left(x^{(i)}|y^{(i)};\phi,\mu_0,\mu_1,\Sigma\right)p\left(y^{(i)};\phi\right)\right]. \end{aligned}$$

If we want to maximize l with respect to the parameters (homework problem), we find that our parameters must be

$$\begin{aligned} \phi &= \sum_{1=1}^{m} 1_{\{y^{(i)}=1\}} \\ \mu_0 &= \frac{\sum_{i=1}^{m} 1_{\{y^{(i)}=0\}}x^{(i)}}{\sum_{i=1}^{m} 1_{\{y^{(i)}=0\}}} \\ \mu_1 &= \frac{\sum_{i=1}^{m} 1_{\{y^{(i)}=1\}}x^{(i)}}{\sum_{i=1}^{m} 1_{\{y^{(i)}=1\}}} \\ \Sigma &= \frac{1}{m}\sum_{i=1}^{m}\left(x^{(i)}-\mu_{y^{(i)}}\right)\left(x^{(i)}-\mu_{y^{(i)}}\right)^T. \end{aligned}$$