Table of Contents.

IN. Introduction

We will cover in short the details needed for various derivations and mathematical formulations.

Since many code pattern will have these common components, it makes sense to seperate it out.

Let's define some of the terms, before we move forward

Convention used

Symbol	description
x_4	scalar is represented as lowercase with dimension subscript letters
$\mathbf{x}$	vector is represented as a bold lowercase letters
$\mathbf{X}$	matrix is represented as bold uppercase letters
rv	random variable is represented by lowercase with no subscript letters

Symbol	description
rv	vector of random variables is represented by bold lowercase letters

LA. Linear Algebra

To represent many data points and compactly manipulate various mathematical operations linear algebra is an excellent tool. We recommend this excellent book by Gilbert Strang (https://www.amazon.com/Introduction-Linear-Algebra-Fourth-Gilbert/dp/0980232716) and/or wiki (https://en.wikipedia.org/wiki/Linear_algebra). We will cover a small portion relevant to current topic.

LA.1. Scalar variable.

When we collect data each of the measurement is scalar. We could collect N samples of each scalar. It is usually represented by non bold word with the subscript representing the index i.e x_i

LA.2. Vector variable.

During data collection, we might measure various attributes, for example age, gender etc of the test subject. to compactly represent M scalars for each subject, vector is used, it is just a tuple of M scalars and uniquely describes a point in M dimensional space.

(**Note**: We use Transpose operation to compactly represent column vector as row)

(**Note**: A vector is a point in M dimensional space)

Representation of a sample using vector

Let the original input dataset be represented by M number of dimensions i.e each sample consist of M different 1D data point and together they form a sample data point column vector represented by bold word i.e $\mathbf{x} = (x_1, x_2 \cdots x_d)^T$.

Since we will be having N different data point, we would like to suffix i for the ith sample and the ith data point vector is

$$\mathbf{x_i} = (x_{i1}, x_{i2} \cdots, x_{id})^T$$

LA.3. Matrix variable.

Representation of Whole dataset.

As we noted in previous subsection that each subject or sample can be represented by one column vector. If we take N such measurement, we can compactly represent all those N measurements across M dimension $N \times M$ matrix.

$$\mathbf{X} = (\mathbf{x_1}^T, \mathbf{x_2}^T, \cdots \mathbf{x_i}^T \cdots, \mathbf{x_n}^T)^T$$

ST. Statistics.

Statistics find applications in variety of fields including Machine Learning. It is a short representation of a lot of information. We recommend this excellent book by David Freeman (https://www.amazon.com/Statistics-4th-David-Freedman/dp/0393929728/ref=sr_1_1?s=books&ie=UTF8&qid=1539282517&sr=1-1&keywords=statistics+david+freedman+4th+edition) and/or wiki (https://en.wikipedia.org/wiki/Statistics). We will cover a small portion relevant to current topic.

ST.0.Mean of one dimensional i.e scalar variable.

If we takes N samples of a scalar variable v and we can approximate by its average i.e. mean as follows:

$$mean(v) = \mu = \overline{v} = \frac{1}{N}\sum_{i=1}^{N} v_i$$

ST.1.Variance of one dimensional i.e scalar variable.

If we takes N samples of a random variable v with zero mean, then its variance is given by

$$variance(v) = \sigma^2 = \frac{1}{N}\sum_{i=1}^{N} v_i^2$$

ST.2. Standardization of data points.

If we shift each of the dimensions by it's mean and divide by standard deviation then we would have standardized data. The advantage of this would be simpler math.

We would also like to divide it by dispersion i.e standard deviation because this will remove any bias associated with measuring units i.e measurement in inch vs cm etc.

it means new i'th sample is represented as $\mathbf{s_i} = (s_{i1}, s_{i2} \cdots, s_{id})^T$ i.e a column vector of standardized dimensions s_{ij}

Each of the i'th samples' j'th dimension is represented as follows

$$s_{ij} = \frac{x_{ij} - \overline{x_j}}{\sigma_{x_j}}$$

Where

- Average for jth dimension ⇒ $\overline{x_j} = \frac{1}{N}\Sigma x_{ij}$
- Standard Deviation for j'th dimension ⇒ $\sigma_{x_j} = \sqrt[2]{\frac{1}{N}\sum_{i=1}^{N}(x_{ij} - \overline{x_j})^2}$

and new standardized data matrix of size $N \times M$ becomes

$$\mathbf{S} = (\mathbf{s_1}^T, \mathbf{s_2}^T, \cdots \mathbf{s_i}^T \cdots, \mathbf{s_n}^T)^T$$

ST.3. Covariance of M dimensional Vector.

If we collect N samples for one dimensional data, then we know that using mean and variance we can get a good perspective on data. In short, using sufficient statistics i.e mean and variance allows us to have compression and remove the noise. The natural question is what to do when we have more than one dimensions to data i.e say M dimensions. Well how about characterizing it by M mean and M variances? This seems intuitively right, but here we are assuming that there is no relation between any two dimensions. What if we want to account for this? The approach would be to compute variances between two dimensions and it is called covariance. Since we have assumed without loss of generality that each dimension has zero mean. We can defined covariance between them and thus with about MxM parameters, we can represent our dataset.

We will assume the dataset as described in section 3.2.4 above. Where we collect N samples for each of M dimensions.

Lets also define $\mathbf{ss_k}$ as the k'th column of matrix $\mathbf{S}$ i.e $\mathbf{ss_k}$ represents all the samples we collected for the k'th dimensions and is defined as

$$\mathbf{ss_k} = (s_{1k}, s_{2k}, \cdots, s_{nk})^T$$

The Covariance Matrix (https://en.wikipedia.org/wiki/Covariance_matrix) $M \times M$ matrix and is defined as

$$\Sigma = (\Sigma_{ij}) \in \Re^{M \times M}$$

- Each Σ_{ij} is a covariance of ith and jth dimensions and is given by

$$\begin{aligned}\Sigma_{ij} &= cov(\mathbf{ss_i}, \mathbf{ss_j}) \\ &= \mathbb{E}(\mathbf{ss_i}, \mathbf{ss_j}) && \text{from the defn of covariance with zero means} \\ &= \frac{1}{N}\sum_{m=1}^{N}(ss_{mi} \times ss_{mj})\end{aligned}$$

Note that the above expression can be compactly represented in covariance matrix form (https://en.wikipedia.org/wiki/Covariance_matrix#Definition)

$$\Sigma = \mathbb{E}(\mathbf{s}, \mathbf{s}^T)$$

$$\Sigma = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{s_i}\mathbf{s_i}^T)$$

PT. Probability Theory .

We encourage user to consult this excellent book (https://www.amazon.com/Probability-Theory-Science-T-Jaynes/dp/0521592712) and wiki (https://en.wikipedia.org/wiki/Probability_theory) for indepth probability theory. Here we will discuss quick overview for the topic that are relevant to the Gaussian Mixture Model mathematical formulation.

PT.1. Random Variables.

PT.1.1 How Random Variable (RV) arises and it's measurement in terms of Probability.

In real life, we can't always pinpoints the output of the events. Event could be any action for example rolling a dice, picking a random person for survey and etc. However in all the cases, the number of possible outcome is fixed and hence instead of talking in concrete terms, we would like to think in terms of possibilities i.e if we roll a dice, what is the possibility that we will get a number 6 and we know that for unbiased dice it is $\frac{1}{6}$. Just in this case outcome X is a random variable and it's specific outcome i.e output 6 is one instance of random variable is usually denoted with lower case i.e x. It's possibilities i.e probability is measure between 0 and 1 and is denoted by

$$p(x) = p(X = x) = p(X = 6)$$

.

PT.2. Properties of Probability.

PT.2.1 Joint Probability of more than one Events.

Continuing our analogy, if we roll our dice multiple times and get two continuous sixes, we can win the game. Just we expect two events to occur one after the another. In terms of notation, if first and second events are represented by random variables X_1 and X_2 and their outputs are represented by x_1 $x_@$ respectively, then this probability is given by

$$joint\_probability\_of\_first\_rv\_is\_x_1\_and\_second\_rv\_is\_x_2 = p(x_1, x_2) = p(X_1 = x_1, X_2 = x_2) :$$

In general, if we have N events, each represented by random variables $X_i; \forall i \in (1, 2, \cdots, N)$ and their corresponding outputs as $x_i; \forall i \in (1, 2, \cdots, N)$, then their joint distribution is given by

$$p(x_1, x_2, \cdots, x_N) = p(X_1 = x_1, X_2 == x_2, \cdots, X_N)$$

PT.2.2 Probability of Conditional Event.

In previous section, we saw that we can represent joint probability distribution of more than one random variable. Now, let's consider another case, where we have been playing game of dice and on first throw, we got number six. Since we only win the game, if we get two consecutive six. We ask ourselves, what is the probability that we will get another six in second throw, given we got a six in first throw? This is what the conditional probability is used for and is extremely useful in machine learning, since we are always interested in finding the probability of an event given that some other events has occurred.

The above is represented notationaly as follows:

$$\begin{aligned} probability\_that\_second\_rv\_is\_x_2\_given\_that\_first\_rv\_is\_x_1 = & \\ & = p(x_2|x_1) && \text{;notation o} \\ & = \frac{p(x_1, x_2)}{p(x_1)} && \text{;see previc} \end{aligned}$$

In general, if we have N random outputs corresponding to N different random variables, and if we are interested in knowing the probability of Nth event, given that previous $N - 1$ events has occurred, it can be represented as

$$p(x_N|x_1, x_2, \cdots, x_{N-1}) = \frac{p(x_1, x_2, \cdots, x_{N-1}, x_N)}{p(x_1, x_2, \cdots, x_{N-1})}$$

PT.2.3 Representing Joint Probability in terms of Conditional Probability.

It is often mathematically convenient to represent joint probability in terms of conditional probability. We can rearrange the previous equation to represent joint probability in terms of conditional probability as:

$$\begin{aligned} joint\_probability\_distribution &= p(x_1, x_2, \cdots, x_{N-1}) \\ &= p(x_N|x_1, x_2, \cdots, x_{N-1}) \times \{p(x_1, x_2, \cdots, x_{N-1})\} \\ &= p(x_N|x_1, x_2, \cdots, x_{N-1}) \times \{p(x_{N-1}|x_1, x_2, \cdots, x_{N-2}) \times \{p(x_1, x_2, \\ &= p(x_N|x_1, x_2, \cdots, x_{N-1})p(x_{N-1}|x_1, x_2, \cdots, x_{N-2})p(x_1, x_2, \cdots, x_{N- \\ &= p(x_N|x_1, x_2, \cdots, x_{N-1})p(x_{N-1}|x_1, x_2, \cdots, x_{N-2}) \cdots p(x_{N-i}|x_1, x_2, \end{aligned}$$

$$\implies p(x_1, x_2, \cdots, x_{N-1}) = p(x_N|x_1, x_2, \cdots, x_{N-1})p(x_{N-1}|x_1, x_2, \cdots, x_{N-2}) \cdots p(x_{N-i}|x_1, x_2, \cdots, x_N$$

PT.2.4 Probability of Independent Events.

Intuition and case for two random variables.

In many applications including machine learning, samples collected are independent, i.e outcome of random variable X_i doesn't depend on X_j. We can intuitively justify it by considering rolling of unbiased dice multiple times. If we have a probability of getting six to be $\frac{1}{6}$ in i'th throw, the probability of getting six will remains the same in any other throw too. In terms of notation for two throw of dice, we can represent them as:

$$\begin{aligned} p(x_1, x_2) &= p(x_1|x_2)p(x_2) \\ &= p(x_1)p(x_2) \qquad \text{;Since both outcomes are independent of each other, there is no} \end{aligned}$$

For N random variables.

For those cases, where all the random variables are independent of each other, we can represent joint probability of N random variables as follows:

$$\begin{aligned} p(x_1, x_2, \cdots, x_N) &= p(x_N|x_1, x_2, \cdots, x_{N-1})p(x_{N-1}|x_1, x_2, \cdots, x_{N-2}) \cdots p(x_{N-i}|x_1, x_2, \cdots, x_{N-i}) \cdots p(\\ &= p(x_N)p(x_{N-1}) \cdots p(x_{N-i}) \cdots p(x_1) \\ &= \prod_{i=1}^{N} p(x_i) \end{aligned}$$

$$\implies p(x_1, x_2, \cdots, x_N) = \prod_{i=1}^{N} p(x_i)$$

PT.2.5 Expectation: Generalized Mean.

We saw in ST.0 that we can approximate N samples of scalar variable with it's mean. Mean gives equal weightage to each of N samples and that is the best one can do if we don't know anything about the distribution of the scalar variables. But in the case of random variable x, we know it's probability and we can define the generalized mean as expectation wrt to probability of random variable as the weighted mean as follows:

$$\begin{aligned}\text{Expectation of random variable } x &= \text{Weighted mean of } x \text{ wrt } p(x)\\ &= \mathbb{E}_{p(x)}(x) = \mathbb{E}(x) = \sum_{i=1}^{N} x_i p(x_i)\end{aligned}$$

PT.2.6 Probability as Expectation of Indicator function

Indicator Function:

Intuitively, it says whether I am interested in some part of the whole. If the whole universe is represented by a set U and If we are interested in a subset S of U (i.e $S \subset U$), our interest can be represented by indicator function $\mathbb{I}$ defined on set U indicating our interest in the subset S. Obviously, if an element u of U is in our interest i.e u belongs to S also it maps it to 1 otherwise 0.

Formally we can write above as:

$$\begin{aligned}&\textbf{Assumption:}\\ &\qquad U = \{u_1, u_2, \cdots u_n\}\\ &\qquad S \subset U\\ &\textbf{Indicator Function:}\\ &\qquad \mathbb{I}(u) = \begin{cases} 1 \text{ if } u \in S\\ 0 \text{ if } u \notin S\end{cases}\end{aligned}$$

Relationship between probability and expectation of indicator function.

In many analysis, we collect samples i.e observe a part of the whole dataset and we would like to consider the best estimate of our samples. Since certain attribute of samples we collected are of our interest and we represent it using indicator function. Now all the samples though similar will be somewhat different from each other and in such scenario, we usually use expectation to represent whole samples by one single sample. It turns out that expectation of the indicator function of a set S is just the probability of observing that set S and it can be formalize as:

$$\begin{aligned}&\textbf{Assumption:}\\ &\qquad U = \{u_1, u_2, \cdots u_n\}\\ &\qquad S \subset U\\ &\textbf{Expectation of Indicator Function:}\\ &\qquad \mathbb{E}_{p(u)}\{\mathbb{I}(u)\} = \sum_{i=0}^{N} \mathbb{I}(u_i)p(u_i) && \text{;From PT.2.5}\\ &\qquad \mathbb{E}_{p(u)}\{\mathbb{I}(u)\} = \mathbb{I}(u \in S)p(u \in S) + \mathbb{I}(u \notin S)p(u \notin S) && \text{;Split u into } S \text{ and } \overline{S}\\ &\qquad \mathbb{E}_{p(u)}\{\mathbb{I}(u)\} = 1.p(u \in S) + 0.p(u \notin S) && \text{;Definition of Indicator function}\\ &\qquad \mathbb{E}_{p(u)}\{\mathbb{I}(u)\} = p(u \in S) && \text{;QED}\end{aligned}$$

PT.3 Parametric Distributions.

It is mathematically convenient to use probability distributions characterized by parameters. There are some common probability distributions, that we will use often and we will review it here.

PT.3.1 Distributions: Multinomial Distribution.

Multinomial distribution (https://en.wikipedia.org/wiki/Multinomial_distribution) arises in various practical scenario.

For example, let's assume we have a biased dice where the expected value of getting 1 is θ_1 and getting 2 is θ_2 and so on. Note that since when we roll a dice, we may or maynot get out number of interests i.e say number 2 and hence we present each of the events occuring with random variables Z_1, Z_2 and so on. We now roll the dice, N times and, we would like to know what is the probability that we exactly get n1 ones and n2 twos and so on. The above is given by multinomial distribution.

$$p(Z_1 = n_1, Z_2 = n_2, \cdots, Z_K = n_K) = \frac{(\sum_{j=1}^{K} n_j)!}{\prod_{j=1}^{K} n_j!} \prod_{j=1}^{K} \theta_j^{n_j}$$

$$Where \sum_{j=1}^{K} n_j = N \qquad \text{;Since sum of all the sub-events is}$$

$$Where \sum_{j=1}^{K} \theta_j = 1 \qquad \text{;Since } \theta_j \text{ spans the probability spa}$$

PT.3.2 Multivariate Gaussian Distribution.

PT.3.2.1 Joint PDF of Multivariate Gaussian Distribution.

As an example, let's assume we have male candidates in America, what is there heights? We know that on average american male height (lets call it μ_1) is about 5 feet and 9 inches but we will definitely have the standard deviation (σ_1) (see section 3.2.1) from the average and heights of most male candidates will fall within three standard deviation and if we take enough samples the random variable height (let's call it X_1) will be represented by a Gaussian distribution and we says that random variable X_1 is drawn from one dimensional Gaussian distribution with mean (μ_1) and standard deviation (σ_1) and is represented as follows:

$$X_1 \sim \mathcal{N}(\mu_1,\ \sigma_1^2)$$

Since a random variable will give different output for each draw of the samples, we can't exactly represent it unless we use Probability Distribution (https://en.wikipedia.org/wiki/Probability_distribution). If we call a sample drawn as x_1, it's one dimensional Gaussian Distribution, it is given by:

$$p(x_1) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x_1-\mu_1)^2/2\sigma_1^2}$$

Continuing in the above line of thinking, let's also assume that we are also asked to collect weights of the the male candidates and our Gaussian distribution will now have two random variables, one for height and another for weight. Now, since we know that in general we can assume that weight and height are independent random variable. But in reality there will be some correlation between them also. It means that if we have k attributes or dimensions, we will have to calculate, $k \times k$ matrix and is explained in section 3.2.5 . So to represent a multidimensional Gaussian distribution, we use vector as follows

$\mathbf{X} = (X_1, X_2, \cdots, X_k)^T$ => represents k dimensional random variable.

$\mathbf{x} = (x_1, x_2, \cdots, x_k)^T$ => represents k attributes or dimensions drawn for each sample.

$\mu = (\mu_1, \mu_2, \cdots, \mu_k)^T$ => represents k attributes mean.

$\mathbf{\Sigma}$ => represents covariance matrix, (see section 3.2.5)

$|\mathbf{\Sigma}|$ => represents determinant of covariance matrix

$\mathbf{\Sigma}^{-1}$ => represents inverse of covariance matrix

and followings holds true for the random vector $\mathbf{X}$ and it's probability distribution $p(x)$

$$\begin{aligned} &\mathbf{X} \sim \mathcal{N}(\mu,\ \mathbf{\Sigma}) && \text{; a vector of rv X is drawn from Normal dis} \\ &p(\mathbf{x}) = \frac{1}{\sqrt{2\pi}\sqrt{|\mathbf{\Sigma}|}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)} && \text{; vector } \mathbf{x} \text{ is a sample of vector random vari} \end{aligned}$$

see also (https://en.wikipedia.org/wiki/Multivariate_normal_distribution)

PT.3.2.2 Conditional PDF of Multivariate Gaussian Distribution.

In many application, we model dataset using Multivariate Gaussian Distribution. i.e. we assume that a vector of random variable $\mathbf{x} = (x_1, x_2, \cdots, x_k)^T$ is drawn from $\mathcal{N}(\mu,\ \mathbf{\Sigma})$.

Typically, we first train multivariate Gaussian by estimating it's parameters mean $\mu = (\mu_1, \mu_2, \cdots, \mu_k)^T$ and covariance $\mathbf{\Sigma} = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1k} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{k1} & \sigma_{k2} & \cdots & \sigma_{kk} \end{pmatrix}$

Then, we predict the output given a part of the sample vector. For example, we might be given only a subset of $\mathbf{x} = (x_1, x_2, \cdots, x_k)^T$ i.e. $\mathbf{x}_g = (x_1, x_2, \cdots, x_g)^T$ with $a < k$ as an input to our model $\mathcal{N}(\mu,\ \mathbf{\Sigma})$. And we are asked to predict remaining variables i.e. $\mathbf{x}_p = (x_1, x_2, \cdots, x_p)^T$ with $p < k$ and $g + p = k$. We can find $\mathbf{x}_p$ using conditional probability $p(\mathbf{x}_p | \mathbf{x}_g)$.

One can derive (https://stats.stackexchange.com/questions/30588/deriving-the-conditional-distributions-of-a-multivariate-normal-distribution) that the conditional probability $p(\mathbf{x}_p | \mathbf{x}_g)$ itself a Multivariate Gaussian $\mathcal{N}(\mu_{p|g}, \mathbf{\Sigma}_{p|g})$ with mean $\mu_{p|g}$ and covariance matrix $\mathbf{\Sigma}_{p|g}$

$$\begin{aligned}
&\textbf{Given :}\\
&\quad \text{Trained Model:}\mathcal{N}(\mu_{k\times 1},\, \boldsymbol{\Sigma}_{k\times k})\\
&\quad \text{Prediction Input:}\mathbf{x}_g = (x_1, x_2, \cdots, x_g)^T;\, g < k\\
&\quad \text{Prediction Output:}\mathbf{x}_p = (x_1, x_2, \cdots, x_p)^T;\, p < k \text{ and } g + p = k\\
&\textbf{Pre} - \textbf{process :}\\
&\quad\ \text{Because } \mathbf{x} = \begin{pmatrix} \mathbf{x}_g \\ \mathbf{x}_p \end{pmatrix} \sim \mathcal{N}(\mu_{k\times 1},\, \boldsymbol{\Sigma}_{k\times k})\\
&\quad\ \text{Therefore Create Blocks of } \mu \text{ and } \Sigma \text{ as:}\\
&\qquad\quad \mu = \begin{pmatrix} \mu_g \\ \mu_p \end{pmatrix}\\
&\qquad\quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{gg} & \boldsymbol{\Sigma}_{gp} \\ \boldsymbol{\Sigma}_{pg} & \boldsymbol{\Sigma}_{pp} \end{pmatrix}\\
&\textbf{Prediction :}\mathbf{x}_p \sim \mathcal{N}(\mu_{p|g},\, \boldsymbol{\Sigma}_{p|g})\\
&\quad p(\mathbf{x}_p|\mathbf{x}_g) = \frac{p(\mathbf{x}_p, \mathbf{x}_g)}{p(\mathbf{x}_g)} = \mathcal{N}(\mu_{p|g},\, \boldsymbol{\Sigma}_{p|g})\\
&\quad \text{where:}\\
&\qquad \mu_{p|g} = \mu_p + \Sigma_{pg}\Sigma_{gg}^{-1}(\mathbf{x}_g - \mu_g)\\
&\qquad \boldsymbol{\Sigma}_{p|g} = \boldsymbol{\Sigma}_{pp} - \Sigma_{pg}\Sigma_{pp}^{-1}\Sigma_{gp}
\end{aligned}$$

PT.4 Bayes Theorem.

Bayes Theorem is remarkably simple concept, which allows one to iteratively, refined the belief in light of new evidences. It is given by

$$p(\textbf{posterior}) \propto p(\textbf{likelihood}) \times p(\textbf{prior})$$

And more details is at Bayes' Theorem (https://en.wikipedia.org/wiki/Bayes%27_theorem)

NT. Numerical Techniques.

We will review, commonly used numerical techniques in Machine Learning.

NT.1 Maximizing Vector Values Functions.

NT.1.1 Maximizing a Function via Matrix Calculus.

There are many numerical algorithm for maximizing and many good book on numerical optimization talks about it.

We will explain the concept on simple using Matrix Calculus (https://en.wikipedia.org/wiki/Matrix_calculus)

Condition for maximizing

- The condition for critical points (i.e maximum or minimum) is that each component of first vector derivative should be zero i.e tangent (interpretation of first derivative) is parallel to x axis at the point of maximum or minimum
- The condition on a critical point (found by first derivative) to be maximum is that second vector derivative must be positive for each component. i.e rate of change of tangent is increasing at maximum point.

If the vector variables is represented by $\Theta = (\theta_1, \theta_2, \cdots, \theta_m)$, then functional on Θ will be represented by $\mathcal{F}(\Theta)$

What is our variables here? Since we want to find the maximum wrt to parameter vector Θ. We can differentiate wrt each of the components i.e $(\theta_1, \theta_2, \cdots, \theta_m)$ and equate each of the equations to zero. We can also verify that it is in fact maximum by finding the second derivative and making sure that it is always positive.

The above can be summarized as:

$$\begin{aligned}\frac{\partial^1 \left[\mathcal{F}(\Theta)\right]}{\partial^1 \Theta} &= \mathbf{0} && \text{; vector } \mathbf{0} \text{ means all the m components are component wise zero} \\ \frac{\partial^2 \left[\mathcal{F}(\Theta)\right]}{\partial^2 \Theta} &\succ \mathbf{0} && \text{; greater than vector } \mathbf{0} \text{ means all the m components are component wise gr}\end{aligned}$$

NT.2 MLE Techniques.

NT.2.1 Likelihood Function.

Machine learning is about modeling i.e you have seen something and you wonder can I predict those dataset in future? Model can be described by a set of parameters. Those parameters could be coefficient in linear model or it could be the parameters of the probability distribution that our data sets are generated from? So in this line of thinking, we can see that if we are interested in finding best parameters. That is by definition is called likelihood of data.

Likelihood is referring to the events that has occured in the past i.e (our observation or our datasets) and we are trying to model it using parameters. That is same as saying what if my parameters are correct, what is the probability of observing my future events i.e (our future observation or our future datasets). Thus, we have established a relationship between likelihood of the past data to the probability of future data linked by parameters.

In notations, the above discussion can be summarised as follows:

$$\mathcal{L}(parameters, model|Dataset) = p(Dataset|parameters, model)$$

Since, we represent our dataset as a matrix $\mathbf{X}$ where each row corresponds to one data point or data vector. and our model are represented as parametric probability distribution may contains many parameters i.e $(\theta_1, \theta_2, \cdots, \theta_m)$ and we can compactly represent them by vector Θ and likelihood as:

$$\begin{aligned} &\mathcal{L}(\Theta|\mathbf{X}) = p(\mathbf{X}|\Theta) && \text{; we will use definition later} \\ &Where\ \ \Theta = (\theta_1, \theta_2, \cdots, \theta_m) \end{aligned}$$

NOTE:

- In practice, we use log likelihood instead of likelihood.
- Since likelihood of past data is same as probability of future data for a set of parameters, it ranges from 0 to 1 inclusive.

NT.2.2 Likelihood to Log Likelihood.

In Machine learning, we are trying to learn model or it's parameters and to we have to choose parameters which maximizes the probability of occurrence of data. Maximization involves taking first and second vector derivative of the joint probability. The above operation can

Logarithm is mathematically convenient function because of :

- It is the monotonic increasing function, taking logarithm of both sides, won't change the coordinate of its maximum point or vector.
- Logarithm converts multiplication into summation and maximizing (i.e takings it's derivatives) is much simpler.

Notationally, log likelihood is expressed as

$$\mathcal{LL}(\Theta|\mathbf{X}) = log(\mathcal{L}(\Theta|\mathbf{X})) = log(p(\mathbf{X}|\Theta)) \qquad \text{; we will use definition later}$$

NT.2.3 Maximum Likelihood Estimation (MLE).

Since likelihood is function Θ, for every choice of $\hat{\Theta} \in \Theta$, we will get different likelihood. Which $\hat{\Theta}$ should we choose? Intuitively, we would like to have $\hat{\Theta}$ which maximizes our likelihood function. This is another way of saying that we want to choose $\hat{\Theta}$ that maximizes our confidence in the likelihood function.

The above discussion is summarized as:

$$\begin{aligned} &\hat{\Theta} = arg\ max\ \mathcal{L}(\Theta|\mathbf{X}) && \text{; } \hat{\Theta} \text{ represents best estimation of parameters vector} \\ &\hat{\Theta} = arg\ max\ \mathcal{LL}(\Theta|\mathbf{X}) && \text{; As discussed in previous section we will get same answer for nc} \end{aligned}$$

In practice, we often use empirical frequency counting to estimate probability and it can be proven that estimating probability using frequency is also MLE estimate.

For example if we would like to find probability of observing various numbers for a biased. We just roll many times (for example 10000 and we will call it $freq_{all}$) and count the number of times 1 occurs (i.e $freq_1$), 2 occurs (i.e $freq_2$) and so on.

Then the probability is given by:

$$p(event = i) = \frac{freq_i}{freq_{all}}; \forall\, i\ \in (1, 2, \cdots, 6)$$

NT.3 EM Techniques.

NT.3.1 Maximizing a Function via Expectation Maximization (EM).

Background and Intuition.

Here is an excellent short tutorial on Expectation Maximization (http://ai.stanford.edu/~chuongdo/papers/em_tutorial.pdf). Also we recommend this excellent book by Maya R Gupta (https://www.amazon.com/Theory-Algorithm-Foundations-Trends-Processing/dp/1601984308) for interested user.

In many MLE problem, we can't exactly find the closed form solution and we have to perform iterative methods. EM is one of those iterative methods.

We will cover a short detail of this algorithm here. EM algorithm is useful for the case of missing or hidden variable.

Let's go back to our favorite dice examples. Let's assume two biased dices with random variable D_A and D_B and we would like to find probability mass function (i.e probability for each of the outcomes). Notationally speaking, we would like to find

$$\begin{array}{c} p(D_A = 1) = \theta_{A1}; p(D_A = 2) = \theta_{A2}; \cdots p(D_A = 6) = \theta_{A6}; \\ p(D_B = 1) = \theta_{B1}; p(D_B = 2) = \theta_{B2}; \cdots p(D_B = 6) = \theta_{B6}; \end{array}$$

compactly, we can represent using parameter vectors as

$$\begin{array}{c} pmf(D_A) = \Theta_A = (\theta_{A1}, \theta_{A2}, \cdots, \theta_{A6}); \\ pmf(D_B) = \Theta_B = (\theta_{B1}, \theta_{B2}, \cdots, \theta_{B6}); \end{array}$$

As we explained in section 3.3.3, we can just use the frequency counting to have the MLE of probability mass function. In words, we can roll each dice many times and

Lets add a twist i.e we are given the outcome and we don't know whether it was generated from dice D_A or D_B. How can we find the MLE probability mass function? Well since we don't know which dice generated our outcome, the best we can assume that a outcome can be generated from either of dices.

In short, given an unknown die (singular of dice), we just roll it many times and estimate the posterior probability of each realization, using Bayes Theorem (see section PT.4). This will act as the weights of the original samples. This step is called **Expectation Step (E-Step)**.

Then, given our weights on data, we again compute the MLE estimate of the data using frequency count. But this time, instead of using the original frequency, we use the weighted frequency . The weights being the posterior computed from (E-step). This is called **Maximization Step(M-Step)**. Since in the maximization step, we used our better guess, it is a better estimate of the actual probability.

We can iterate over E-Step and M-Step until convergence.

Developing the pseudocode.

Above **Expectation Maximization Algorithm** can be summarized as follows

```
$\hat{\Theta}^{(t)}$ : be estimated parameter vectors for current iteration;
$\hat{\Theta}^{(t+1)}$ : be estimated parameter vectors for next iteration;
 Set initial current_parameters to some random values ;
 Repeat Until Convergence:
    Begin:
         set current_parameters values to next_parameters;
         1. Perform E-Step using current_parameters and get the new set of weighted data;
         2. using MLE compute the best parameters and call it next_parameters;
    End:
```

EM Algorithm.

Let's put the above algorithm in the more convenient form.

We have already established the relationship between likelihood and probability. Here our dataset consists of K hidden variable also, lets call it $\mathbf{Z} = (\mathbf{z_1}, \mathbf{z_1}, \cdots, \mathbf{z_K})$ and dataset as $\mathbf{X}$ (see section 3.2.3).

$$\begin{aligned}
&\mathbf{E-Step:} \\
&\quad Q(\Theta|\hat{\Theta}^{(t)}) = \mathcal{E}_{\mathbf{Z|X,\hat{\Theta}^{(t)}}}[\mathcal{LL}(\Theta|\mathbf{X,Z})] &&\text{; Expected value of log likelihood wrt } \mathbf{Z} \text{ given dat} \\
&\mathbf{M-Step:} \\
&\quad \hat{\Theta}^{(t+1)} = argmax_{\Theta}\mathcal{LL}(Q(\Theta|\hat{\Theta}^{(t)}) &&\text{; find the next iteration parameter } \hat{\Theta}^{(t+1)} \text{ by maxin}
\end{aligned}$$