M.L ⇒ Error or Minimization (or) Likelihood Maximization

Likelihood function ⇒ $P(x/\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$

For this to be a probability function,

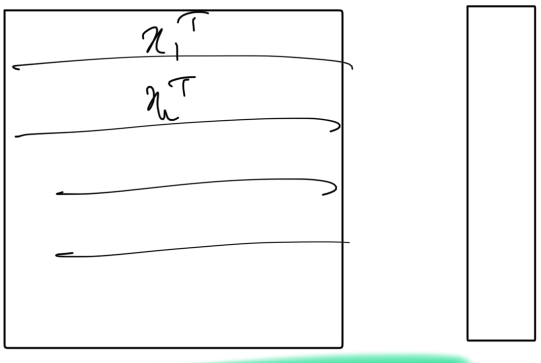
$$\int_{-\infty}^{\infty} P(x/\mu,\sigma^2) = 1$$

MLE ⇒ $P(D/\theta)$

MAP ⇒ $P(\theta/D)$ ↳ $= P(D/\theta) \times P(\theta)$

If prior is also included, it is MAP

Maximum Likelihood Estimation



⇒ learn θ such that $P(Y/X,\theta)$ is maximized

likelihood function

Coin Toss

Likelihood is like a score which tells how likely a dataset is related to a model.

Bernoulli Distribution ⇒ $n=1$

$\mu^x (1-\mu)^{(1-x)}$ ⇒ $x = 0$ (or) 1

i.i.d ⇒ independent identically distributed

Eg: n coin tosses

$P(X/\mu) = \prod_{i=1}^{n} P(x_i/\mu)$ ⇒ Joint likelihood

When many P's are multiplied ⇒ answer is small. ⇒ Underflow error

So use log likelihood.

Joint log likelihood = $\log(P(X/\mu)) = \log\left(\prod_{i=1}^{n} P(x_i/\mu)\right)$

$$= \sum_{i=1}^{N} \log P(x_i/\mu)$$

↓

Find μ such that this is maximized

$$\sum_{i=1}^{N} \log(P(x_i/\mu)) = \sum_{i=1}^{N} \log\left(\mu^{x_i}(1-\mu)^{(1-x_i)}\right)$$

$$= \sum_{i=1}^{N} \left[x_i \log\mu + (1-x_i)\log(1-\mu)\right] \Rightarrow JLL$$

$$\frac{\partial(JLL)}{\partial\mu} = \frac{\partial}{\partial\mu}\left[\left(\sum_{i=1}^{N} x_i\right)\log\mu + \left[\sum_{i=1}^{N}(1-x_i)\right]\log(1-\mu)\right]$$

$$\sum_{i=1}^{N} x_i = N_H$$

$$\sum_{i=1}^{N}(1-x_i) \rightarrow N - N_H$$

$$= \frac{\partial}{\partial\mu}\left[N_H \log\mu + (N-N_H)\log(1-\mu)\right]$$

$$= \frac{N_H}{\mu} - \frac{N-N_H}{1-\mu} = \frac{N_H - N_H\mu - N\mu + N_H\mu}{\mu(1-\mu)}$$

$$\frac{\partial(JLL)}{\partial\mu} = \frac{N_H - N\mu}{\mu(1-\mu)}$$

let $\frac{\partial(JLL)}{\partial \mu} = 0$

$$\frac{N_H - N\times\mu}{\mu(1-\mu)} = 0 \Rightarrow N_H - N\times\mu = 0$$

$$\boxed{\mu = \frac{N_H}{N}}$$

→ We know this formula. But it was formally derived like this

Dice

"K" faces

x = K-dimensional vector where if outcome is 3 index 3 is 1 & others are zeros. (1-indexed)

(or)

x = a no. b/w $1 \ldots K$.

$P(x/\mu) \Rightarrow$ Here μ is a K-dimensional vector where $\mu[i]$ represents probability of dice showing i.

$$P(x/\mu) = \mu_1^{I(x=1)}\ \mu_2^{I(x=2)}\ \mu_3^{I(x=3)} \ldots\ \mu_K^{I(x=K)}$$

$$P(x/\mu) = \prod_{i=1}^{K} \mu_i^{I(x=i)}$$ → i^{th} element of vector "x".

(or) $\prod_{i=1}^{K} \mu_i^{x^i}$

If $x = [0\ 1\ 0\ 0] \Rightarrow I(x=1) = I(x=3) = I(x=4) = 0$

$\hookrightarrow I(x=2) = 1$

Let there be N samples like these,

$x_1, x_2, x_3, \ldots\ldots, x_N$

$x_n = n^{th}$ sample.

$$J.L. = P(X/\mu) = \prod_{n=1}^{N} P(x_n/\mu)$$

$$J.L.L = \log\left(P(X/\mu)\right) = \sum_{n=1}^{N} \log\left(P(x_n/\mu)\right)$$

$$= \sum_{n=1}^{N} \log\left[\prod_{i=1}^{K} \mu_i^{x_{ni}}\right] = \sum_{n=1}^{N} \sum_{i=1}^{K} x_{ni} \times \log \mu_i$$

$$- \lambda\left[\sum_{i=1}^{K} \mu_i - 1\right]$$

(Lagrangian ↵ ↳ Constraint

As, sum of probs. = 1.

$$\frac{\partial (JLL)}{\partial \mu_i} = \sum_{n=1}^{N} \sum_{i=1}^{K} \frac{x_{ni}}{\mu_i} - \lambda = 0 \Rightarrow \frac{N_i}{\mu_i} = \lambda$$

$$\lambda = \frac{N_i}{\mu_i} \quad (\text{or}) \quad \mu_i = \frac{N_i}{\lambda}$$

$$\sum_{n=1}^{N} \sum_{i=1}^{K} x_{ni} = ?$$

$$\sum_{i=1}^{K} \mu_i = 1 \Rightarrow \frac{\sum_{i=1}^{K} N_i}{\lambda} = 1 \Rightarrow \lambda = \sum_{i=1}^{K} N_i \Rightarrow \boxed{\lambda = N}$$

Posterior $\propto$ likelihood * Prior

$P(\theta/X,\alpha) \propto P(X/\theta)\ P(\theta/\alpha)$

↳ hyperparameters for θ

For coin toss,

$$P(X/\theta) = P(X/\mu) = \prod_{i=1}^{N} \mu^{n_i} (1-\mu)^{(1-n_i)}$$

$$\beta(\theta/a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \times \theta^{a-1} \times [1-\theta]^{b-1} \Rightarrow \text{Beta function}$$

For bernoulli,

$$\beta(\mu/a,b) = \frac{\Gamma(a+b)}{\Gamma a + \Gamma b} \times \mu^{N_1+a-1} \times (1-\mu)^{N-N_1+b-1}$$

Conjugate prior ⇒ when prior doesn't change its functional form when converted to posterior

For dice game,

Dirichlet distribution ⇒ Distribution of distributions
- Distribution of prob. vectors over discrete space.

Read

$$P(\theta/X,\alpha) \propto P(X/\theta) \times P(\theta/\alpha)$$

$$\propto \prod_{i=1}^{N} P(n_i/\mu)$$

$$\propto \prod_{n=1}^{N} \prod_{i=1}^{K} \mu_i^{x_{n_i}}$$

$\Gamma \Rightarrow$ gamma function
↳ extension of factorial function for real nos.

K faces ⇒ $\sum_{i=1}^{K} \mu_i = 1$

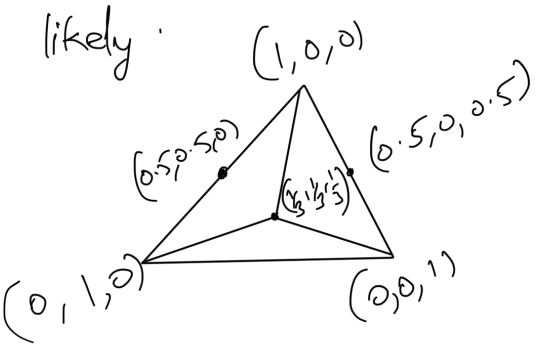
& $\mu_1, \mu_2, \mu_3, \cdots \mu_K \geq 0$

$[\alpha_1 \alpha_2 \alpha_3 \cdots \alpha_K] \rightsquigarrow \sum \alpha_i$

$$Dir(\mu|\alpha) = \frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)} \times \prod_{i=1}^{K}\mu_i^{\alpha_i - 1}$$

$$\text{Posterior} \Rightarrow \frac{\Gamma\left(\sum_{i=1}^{K}(\alpha_i + N_i)\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i + N_i)} \times \prod_{i=1}^{K}\mu_i^{N+\alpha_i - 1}$$

$\alpha = 1 \Rightarrow$ Every point in the simplex of $n-D$ data are equally likely.



$\alpha \uparrow \Rightarrow$ More uniform distribution

$\alpha \downarrow \Rightarrow$ Some events less likely, while, some are more likely.

↳ prior. Decides which kind of samples generated.

Naive Bayes ⇒ Probabilistic classifier

SVM ⇒ Non-probabilistic classifier

↓

feature conditional probabilities are independent of each other.

$$\hat{P}(y_i|x,\theta) \Rightarrow \hat{y} = \max_{i=1\ldots L}\left(\hat{P}(y_i|x,\theta)\right) \Rightarrow \text{Probabilistic classifier}$$

- Generative classifier (approach) } ⇒ Read
- Discriminative approach } ⇒ Read
  - ↳ Directly modelling conditional probabilities

1) Works on joint distribution ⇒ by product is classifier

$$P(y, X) = P(y) \times P(X/y)$$

$P(y)$ → prior, $P(X/y)$ → likelihood

$$P(X|y) = \prod_{i=1}^{k} P(x_i/y)$$

Generative models are easy to implement · work well on smaller data sets.

$$J.L. = \prod_{n=1}^{D} P(y_n, x_n) = \prod_{n=1}^{D} P(y_n) \times P(x_n/y_n)$$

If $W = [0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0 \ldots]$ → no. of words in corpus

(word-0 absent, word-1 present)

& $x = [0\ 1\ 0\ 0\ 3\ 0\ 2\ 0\ 1\ 5\ 9\ 0 \ldots]$

(word-2 absent; no. of times this occurred)

$$P(x|y) = \prod_{i=1}^{k} P(w_i/y)^{x_{ni}}$$

So, if $|V| = 30{,}000$, then you need 30,000 parameters to model it. May overfit.

$$J.L. = \prod_{n=1}^{D} P(y_n) \times \prod_{i=1}^{k} P(w_i/y_n)^{x_{ni}}$$

(k → no. of words)

→ Assume words are conditionally independent given y.

$$P(y=l) = \frac{\sum_{i=1}^{n} I(y^{(i)}=l)}{n}$$

$L \times k$ ⇒ model size

L^{th} row & i^{th} column ⇒ $P(w_i | y=L)$

$$P(w_i/y=l) = \frac{N_{li}}{N_l}$$

→ No. of times i^{th} feature occurs in class l

→ Total no. of occurrences of all features in class l.

Generative

- Naive Bayes
- Builds seperate models to characterize each class.
- Learns $P(x/y)$ (features / class) Also $P(y)$.

like try to build a model of what malignant tumors look like & another model of what benign tumors look like

⇒ What are the features like when $y=0$.

→ Then Compute $P(y=k|x)$ by

$$P(y=k|x) = \frac{P(x|y=k)P(y=k)}{P(x)}$$

$$P(x) = \sum_{k} P(x|y=l)\, P(y=l)$$

Discriminative

- logistic regression
- Builds a model to seperate the classes.
- Learns $P(y/x)$ directly.

Logistic Regression

Binary classification

i.e. estimates the probability distribution of the dataset.
> calculates the probability by country.

$p(y|x,\theta)$

$y = \theta^T x$

$\hookrightarrow (-\infty, \infty)$

$\Downarrow$ map to

$[0,1] \Rightarrow$ (use sigmoid function)

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$y = \sigma(\theta^T x) \Rightarrow p(y|x,\theta) = \frac{1}{1+e^{-\theta^T x}}$$

- Doesn't assume conditional independence of conditional probs. of features.

$\hat{y}_i \Rightarrow$ predicted.

$$E_p = (y_i - \hat{y}_i)^2$$

$$E = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

$\min_\theta E \Rightarrow$ to fit the model

$$\min_\theta E = \min_\theta \frac{1}{N} \sum_{n=1}^{N} [y_n - \sigma(\theta^T x_n)]^2$$

$$\rightarrow \frac{\partial E}{\partial \theta} = \frac{1}{N} \sum_{n=1}^{N} 2[y_n - \sigma(\theta^T x_n)] \times \sigma(\theta^T x_n) \times [1 - \sigma(\theta^T x_n)] \times x_n$$

$x_n \hookrightarrow$ vector

$$\frac{\partial}{\partial a}(a^T b) = \begin{bmatrix} \frac{\partial (a^T b)}{\partial a_1} \\ \frac{\partial a^T b}{\partial a_2} \\ \vdots \end{bmatrix}$$

$$a^T b = a_1 b_1 + a_2 b_2 + a_3 b_3 + \dots + a_k b_k$$

$$\frac{\partial (a^T b)}{\partial a_1} = b_1 \quad , \quad \frac{\partial (a^T b)}{\partial a_2} = b_2 \quad \dots$$

$$\therefore \begin{bmatrix} \frac{\partial (a^T b)}{a_1} \\ \frac{\partial (a^T b)}{\partial a_2} \\ \vdots \\ \frac{\partial (a^T b)}{\partial a_k} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix} = b$$

Derive $\Rightarrow \frac{\partial}{\partial x}(x^T A x)$ (vectors)

possible only if A is a matrix; if x is a vector.

Eg :-

y	x_0	x_1	x_2
1	1	1.5	2.8
0	1	5	6
1	1	3.4	6
0	1	7.6	8

$\Rightarrow$ D

w^T

-1	1.5	0.5
θ_0	θ_1	θ_2

$$w = w - \alpha \frac{\partial E}{\partial w}$$

Demonstrate gradient descent for one step. Take $\alpha = 1$.

$x_0 = 1$

loss function $= L = \frac{1}{N} \sum_{n=1}^{N} (y_i - \hat{y}_i)^2$

$$w^{t+1} = w^t - \alpha \times \frac{\partial L}{\partial w}$$

$$L = \frac{1}{4}\left[(1-\sigma(2.65))^2 + (0-\sigma(9.5))^2 + (1-\sigma(7.1))^2 + (0-\sigma(14.4))^2\right]$$

$$L = \frac{1}{4}\left[(1-0.934)^2 + (0-0.999)^2 + (1-0.999)^2 + (0-0.9999)^2\right] = 0.5$$

$$\frac{\partial L}{\partial w} = \frac{1}{N}\sum_{n=1}^{N} 2\left[y_n - \sigma(w^T x_n)\right] \times \sigma(w^T x_n) \times \left[1-\sigma(w^T x_n)\right] \times x_n$$

$$\frac{\partial L}{\partial w} = \frac{1}{4} \times 2 \times \Big[\left((1-\sigma(-1+1.5\times1.5+0.5\times2.8)\right) \times x_1 \; \sigma(-1+1.5\times1.5+0.5\times2.8) \times x_2 \times \left(1-\sigma(-1+1.5\times1.5+0.5\times2.8)\right) \times x_3 + (0 - \ldots -) \times x_4\Big]$$

$\sigma(w^T x_1) = \sigma(2.65) = 0.934$

$\sigma(w^T x_2) = \sigma(9.5) = 0.999$

$\sigma(w^T x_3) = \sigma(7.1) = 0.999$

$\sigma(w^T x_4) = \sigma(14.4) = 0.999$

$$\frac{\partial L}{\partial w} = \frac{1}{2}\times\Big[((1-0.934)\times 0.934\times(1-0.934)\times\begin{bmatrix}1\\1.5\\2.8\end{bmatrix}$$
$$+\Big((0-0.999)\times 0.999\times(1-0.999)\times\begin{bmatrix}1\\5\\6\end{bmatrix}$$
$$+\Big((1-0.999)\times 0.999\times(1-0.999)\times\begin{bmatrix}1\\3.4\\6\end{bmatrix}$$
$$+\Big((0-0.999)\times 0.999\times(1-0.999)\times\begin{bmatrix}1\\7.6\\8\end{bmatrix}\Big]$$

$$= \frac{1}{2}\times\Big[0.004068\times\begin{bmatrix}1\\1.5\\2.8\end{bmatrix} - 0.000998\begin{bmatrix}1\\5\\6\end{bmatrix}$$
$$+\ 0.999\times10^{-6}\begin{bmatrix}1\\3.4\\6\end{bmatrix}$$
$$-\ 0.000998\begin{bmatrix}1\\7.6\\8\end{bmatrix}\Big)$$

$$= \frac{1}{2}\times\Big(\begin{bmatrix}0.006\\0.0114\end{bmatrix} - \begin{bmatrix}0.005\\0.006\end{bmatrix} + \begin{bmatrix}3.4\times10^{-6}\\6\times10^{-6}\end{bmatrix}$$
$$-\begin{bmatrix}0.007584\\0.008\end{bmatrix}$$

$$= \frac{1}{2}\times\begin{bmatrix}0.0020721\\-0.0065\\-0.0026\end{bmatrix} = \begin{bmatrix}0.00103649\\-0.00325\\-0.0013\end{bmatrix}$$

$$w = w - \alpha\times\frac{\partial L}{\partial w} \Rightarrow w = \begin{bmatrix}-1\\1.5\\0.5\end{bmatrix} - \begin{bmatrix}0.00103649\\-0.00325\\-0.0013\end{bmatrix}$$

$$w = \begin{bmatrix}-1.0108649\\1.50325\\0.5013\end{bmatrix}$$

MSE is not used as optimization objective of logistic regression. Instead we compute KL divergence b/w y & $\hat{y}$.

$$KL(\vec{p} \,||\, \vec{q}) = \sum_{i=1}^{L} p_i \log \frac{p_i}{q_i}$$

$$L = \frac{1}{N} \sum_{n=1}^{N} KL(y_n \,||\, \hat{y}_n) = \frac{1}{N} \sum_{n=1}^{N} \sum_{l=1}^{L} y_{nl} \log \frac{y_{nl}}{\hat{y}_{nl}}$$

$$= \frac{1}{N} \sum_{n=1}^{N} -\log \left[P(y_n = l / x_n, w) \right]$$

$$P(y_n = 1 / x_n, w) = \sigma(w^T x_n)$$

$$\text{Softmax}(l, x, w) = \frac{e^{w_l^T x}}{\sum_{i=1}^{L} e^{w_i^T x}}$$

$$\begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_L^T \end{bmatrix}$$

$$P(y = l / x) \hookrightarrow P_l$$

$\exp(\text{Score } 1) \Rightarrow w_1^T x$
$\exp(\text{Score } 2) \Rightarrow w_2^T x$
$\exp(\text{Score } 3) \Rightarrow w_3^T x$
$\exp(\text{Score } l) \Rightarrow w_l^T x$

Score may be negative.
So apply exp

↳ This gives $[P_1 \; P_2 \; \ldots \; P_L]$

$$KL\big((y_1 \; y_2 \; \ldots \; y_L) \,||\, (P_1 \; P_2 \; \ldots \; P_L)\big)$$

$$= \sum_{l=1}^{L} y_l \log \frac{y_l}{P_l} = -\log P(y / x, w)$$

$$= \sum_{l=1}^{L} y_l \log y_l - \sum_{l=1}^{L} y_l \log P_l$$

↳ Minimize this $\Rightarrow \log P(y / x, w)$ increases

$$\frac{d}{dx}\left(\frac{u}{v}\right) = \frac{v\,du - u\,dv}{v^2}$$

$$\frac{\partial p_l}{\partial w_{ck}} = \frac{\partial [\text{softmax}(l, n, w)]}{\partial w_{ck}} = \frac{\partial}{\partial w_{ck}} \cdot \frac{e^{w_l^T n}}{\sum_{i=1}^{L} e^{w_i^T n}} = (\Sigma)^{-1} \times \frac{\partial}{\partial w_{ck}} \left(e^{w_l^T n}\right) + \frac{-e^{w_l^T n} \times \frac{\partial}{\partial w_{ck}} (\Sigma)}{(\Sigma)^2}$$

$$= -1 \times e^{w_l^T n} \times e^{w_c^T n} \times n_k + \frac{I(l=c) \times e^{w_l^T n} \times n_k}{\sum_{i=1}^{L} e^{w_i^T n}}$$

$$= -p_l \times p_c \times n_k + I(l=c) \times p_l \times n_k$$

$$\frac{\partial p_l}{\partial w_{ck}} = p_l \left(-p_c + I(l=c)\right) n_k$$

$$\boxed{\frac{\partial p_l}{\partial w_c} = p_l \left(I(l=c) - p_c\right) n}$$

$$E = KL\left((y_1\ y_2 \ldots\ y_n) \,||\, (p_1\ p_2 \ldots\ p_n)\right)$$

$$= \sum_{l=1}^{L} y_l \log \frac{y_l}{p_l} = -\log\left(P(y/n, w)\right)$$

$$E = \frac{1}{N} \sum_{n=1}^{N} KL(y_n || p_n)$$

one hot vector

$y_{nl} \log y_{nl} - y_{nl} \log p_{nl}$ — Here no w_{ck}, so not considered

$$w_{ck}^{t+1} = w_{ck}^{t} - \alpha \frac{\partial E}{\partial w_{ck}} \Rightarrow \text{Gradient Descent.}$$

$$\frac{\partial E}{\partial w_{ck}} = \frac{1}{N} \sum_{n=1}^{N} \frac{\partial}{\partial w_{ck}} \left[\sum_{l=1}^{L} y_{nl} \times \log \frac{y_{nl}}{p_{nl}}\right] = -\frac{1}{N} \sum_{n=1}^{N} \sum_{l=1}^{L} I(y_{nl}=1) \frac{\partial}{\partial w_{ck}} \left(\log (p_{nl})\right)$$

$$= -\frac{1}{N} \sum_{n=1}^{N} \sum_{l=1}^{L} I(y_{nl}=1) \times \frac{1}{p_{nl}} \times \frac{\partial}{\partial w_{ck}} (p_{nl})$$

$$= -\frac{1}{N} \sum_{n=1}^{N} \sum_{l=1}^{L} I(y_{nl}=1) \times \frac{1}{p_{nl}} \times p_{nl} \times \left(I(c=l) - p_{nc}\right) n_{nk}$$

$$\frac{\partial E}{\partial w_{ck}} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{l=1}^{L} I(y_{nl}=1)\left(I(c=l) - P_{nc}\right) \times x_{nk}$$

$$I(c=l) = 1, \text{ if } c==l$$

No. of parameters in logistic $= L + L \times K$
↳ bias

No. of parameters in naive bayes $= L + L \times K$
↳ prior probs.

changing the gradient as

$$w_{ck}^{t+1} = w_{ck}^{t} - \alpha\left(\frac{\partial E}{\partial w_{ck}} H^{-1}\right)$$

$H = \frac{\partial^2 E}{\partial w_{ck}^2} \Rightarrow$ Hessian

↓ Helps in faster convergence
↓ Called IRLS

→ Naive bayes assumes conditional probs. to be independent.
So logistic fares better than naive bayes.
Indirect way of solving the classification problem by naive bayes also makes it less suitable to classification when compared to logistic regression.

Generative algorithms make more assumptions, whereas discriminative learning algos make lesser assumptions. If assumptions are wrong, then D.A's fair better.

G.A ⇒ computationally efficient than D.A's
But more data present ⇒ D.A's preferred

E - M Algorithm

Suppose there are documents from three classes Cricket, Hockey & Football. But they aren't labelled.

[ ] T_1
[ ] T^2
⋮
[ ] T_n

E: Derive posterior probs. for each document

$$\begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1K} \\ P_{21} & P_{22} & \cdots & P_{2K} \\ P_{31} & P_{32} & \cdots & P_{3K} \end{bmatrix}$$

M: Use posterior probs. to derive updated parameters.

Solve AI's EM example

Eg: Let there be two dice with three faces (A, B, C)

	A	B	C
Dice 1 ⇒	0.3	0.4	0.3
Dice 2 ⇒	0.1	0.4	0.5

⇒ Update P(A/Dice 1), ...

C C A B B C ⇒ T1
A B A A A B ⇒ T2
A C C B C A ⇒ T3
A A B B C B ⇒ T4
A C B C B C ⇒ T5

If sequence of throws is given, then probability is the multiplication of probs.

If sequence is not given, then apply binomial distribution, multinomial distribution, etc.

	P(D1)	P(D2)	No. of A's attr. to D1	No. of B's attr. to D1	No. of C's attr. to D1	No. of A's attr. to D2	B's to D2	C's to D2
T1	0.3932	0.607	0.3932	0.7864	1.1796	0.607	1.214	1.821
T2	0.988	0.0122	3.952	1.976	0	0.0488	0.024	0
T3	0.66	0.34	1.32	0.66	1.98	0.68	0.34	1.02
T4	0.84	0.16	1.68	2.52	0.84	0.32	0.48	0.16
T5	0.39	0.61	0.39	0.78	1.17	0.61	1.22	1.83

$$P(A/D1) = \frac{\text{No. of A's attr. to D1}}{\text{Total no. of rolls with D1}} = \frac{7.7352}{19.6272} = 0.394$$

$$P(A/D2) = \frac{\text{No. of A's attr. to D2}}{\text{Total no. of rolls with D2}} = \frac{2.2658}{10.3752} = 0.2183$$

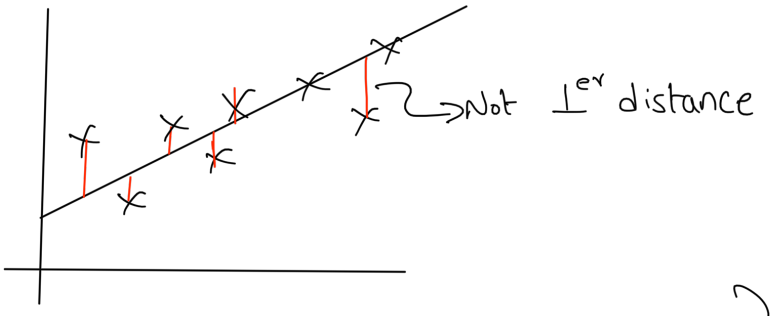
Linear Regression

y = Real-valued variable

$X \Rightarrow N \times K$ data matrix

$$\hat{y} = \alpha_1 x_1 + \alpha_2 x_2 + \dots + \alpha_K x_K = \alpha^T x$$

$$E = \frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2$$



2-dimensional

$$E = \frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (y_n - m x_n - c)^2$$

$$= d_1 m^2 + d_2 mc + d_3 c^2 + d_4 m + d_5 c + d_6$$

$d_1 \hookrightarrow \sum_{n=1}^{N} x_n^2$

$d_6 \hookrightarrow \sum_{n=1}^{N} y_n^2$

$$\frac{\partial E}{\partial m} = 2d_1 m + d_2 c + d_4 = 0$$

$$\frac{\partial E}{\partial c} = d_2 m + 2d_3 c + d_5 = 0$$

$$d_2 \times (2d_1 m + d_2 c + d_4) = 0$$

$$2d_1 \times (d_2 m + 2d_3 c + d_5) = 0$$

$$\Rightarrow \frac{2d_1 d_2 m + d_2^2 c + d_2 d_4}{2d_1 d_2 m + 4d_1 d_3 c + 2d_1 d_5} = 0$$

$$\Rightarrow c = \frac{2d_5 d_1 - d_4 d_2}{d_2^2 - 4d_1 d_3}$$

$$m = \frac{d_2 d_5 - 2 d_1 d_4}{4 d_1 d_3 - d_2^2}$$

$$m = \frac{\left(\frac{2}{N}\sum x_n\right)\left(\frac{2}{N}\sum y_n\right) - \frac{2}{N}\left[-\frac{2}{N}\sum x_n y_n\right]}{\frac{4}{N^2}\left(\sum x_n^2\right) N - \frac{4}{N^2}\left(\sum x_n\right)^2}$$

$$m = \frac{N\sum x_n y_n - \left(\sum x_n\right)\left(\sum y_n\right)}{N\left(\sum x_n^2\right) - \left(\sum x_n\right)^2} \Longrightarrow \text{2-d only}$$

K-d $\Rightarrow$ $E = \frac{1}{N}\sum_{n=1}^{N}\left(y_n - \hat{y}_n\right)^2 = \frac{1}{N}(y - X\alpha)^T(y - X\alpha)$

$$E = \frac{1}{N}\left(y^T y - (X\alpha)^T y - y^T(X\alpha) + (X\alpha)^T(X\alpha)\right)$$

$$E = \frac{1}{N}\left(y^T y - 2\, y^T(X\alpha) + \alpha^T(X^T X)\alpha\right)$$

$\frac{d}{dx}(x^T A x) = 2Ax$

$\frac{\partial E}{\partial \alpha} = \frac{1}{N}\left[2(X^T X\alpha) - 2(y^T X)^T\right] = 0$

$\frac{\partial(\alpha^T n)}{\partial \alpha} = n$ & $\frac{\partial}{\partial \alpha}(n\alpha) = n^T$

$$\alpha = (X^T X)^{-1} X^T y$$

Regularizer $\Rightarrow$ $\beta \alpha^T \alpha$

$\frac{\partial(\beta \alpha^T \alpha)}{\partial \alpha} = \frac{\partial}{\partial \alpha}(\beta \alpha^T I \alpha)$

$= \beta \times 2 I \alpha$

$= 2\beta\alpha$

$\frac{\partial E}{\partial \alpha}$ with regularizer $\Rightarrow$ $2X^T X\alpha - 2(y^T X)^T + 2\beta\alpha = 0$

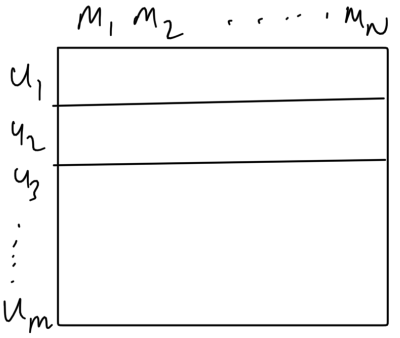
$$(X^T X + \beta I)\alpha = X^T y$$

$$\alpha = (X^T X + \beta I)^{-1} X^T y$$

Entropy

Recommender Systems

large dataset, but sparse.



① Latent factor model

↳ Each user vector to a lower dimension

→ Recommender Systems are based on one of two strategies ⇒ content filtering & collaborative filtering.

Read Netflix paper. Read eigen vectors, def^n, etc.

Read Recommender System with SVD

Problem

	M_1	M_2	M_3	M_4	M_5
U_1	1	0	4	1	0
U_2	0	0	4	4	2
U_3	1	2	0	0	1
U_4	0	0	5	2	5
U_5	0	4	0	2	5

0 ⇒ indicates no rating

P ⇒ weights for users

Q ⇒ Movie features

Q

$$\begin{bmatrix} -0.5 & -0.91 \\ 0.04 & -0.9 \\ 0.23 & -0.5 \\ 0.89 & -0.53 \\ -0.92 & -0.91 \end{bmatrix}$$

P

$$\begin{bmatrix} 0.57 & -0.48 \\ -0.35 & -0.27 \\ 0.17 & -0.84 \\ 0.49 & -0.37 \\ 0.59 & -0.67 \end{bmatrix}$$

User1 & User3 have rated Movie 1.

First fix P & find Q.
Then fix Q & find P
} ⇒ ALS (Use $\alpha = (x^T x)^{-1} x^T y$

$$P = \begin{bmatrix} 0.57 & -0.48 \\ 0.17 & -0.84 \end{bmatrix}, \quad R = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$Q = (P^T P)^{-1} P^T R$$

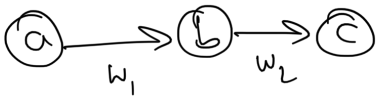
$$Q = \begin{bmatrix} \frac{5551}{10000} & \frac{5001}{10000} \\ \frac{5001}{10000} & \frac{1469}{10000} \end{bmatrix}^{-1} P^T R$$

$$Q = \begin{bmatrix} \frac{15045275}{1971098} \\ -\frac{4596125}{657366} \end{bmatrix} = \begin{bmatrix} 7.63 \\ -6.99 \end{bmatrix}$$

↳ q_1

After updating Q, use the updated (new) Q to solve for P.

Neural Networks



t = target

$E = (c-t)^2$

$b = f(w_1 a)$

$c = g(w_2 b)$

f = Tanh

g = sigmoid

$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial c} \times \frac{\partial c}{\partial g} \times \frac{\partial (w_2 b)}{w_2}$$

$$= 2(c-t) \times g(w_2 b)(1-g(w_2 b)) \times b$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial c} \times \frac{\partial c}{\partial g} \times \frac{\partial (w_2 b)}{\partial b} \times \frac{\partial b}{\partial f} \times \frac{\partial (w_1 a)}{\partial w_1}$$

$$= 2(c-t) \times g(w_2 b)(1-g(w_2 b)) \times w_2 \times (1-\tanh^2(w_1 a)) \times a$$

- Not convex optimization
- So at last we get (local optima only.
- $y = f(x)$, $y \sim \hat{y}$, To approximate that function "f", we use a complex neural network; that adjusts the weights to make $y \sim \hat{y}$.

- Binary classification ⇒ Sigmoid function ⇒ Near to o/p layer

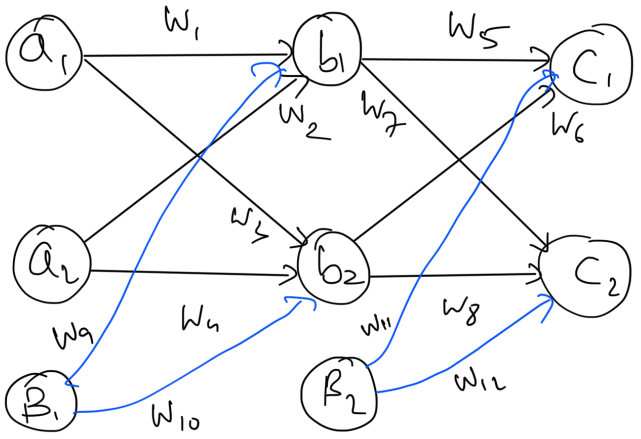
tanh ⇒ Middle layers.

↳ -1 to 1

Symmetric about 0.

$$W_2^{t+1} = W_2^t - \alpha \left(\frac{\partial E}{\partial W_2} \right)$$

Example



0.01 => y_1

0.99 => y_2

Use KL-divergence

$$E = \sum_{i=1}^{2} y_i \log \frac{y_i}{c_i}$$

$$b_1 = \tanh\left((w_1 a_1 + w_2 a_2) + w_9 B_1\right)$$

$$b_2 = \tanh\left((w_3 a_1 + w_4 a_2) + w_{10} B_1\right)$$

$$[C_1 \; C_2] = \text{Softmax}\left(\left(w_5 b_1 + w_6 b_2\right) + w_{11} B_2,\; \left(w_7 b_1 + w_8 b_2\right) + w_{12} B_2\right)$$

$$E = y_1 \log \frac{y_1}{c_1} + y_2 \log \frac{y_2}{c_2}$$

$$E = y_1 \log y_1 - y_1 \log c_1 + y_2 \log y_2 - y_2 \log c_2$$

$$C_1 = \frac{e^{w_5 b_1 + w_6 b_2 + w_{11} B_2}}{e^{w_5 b_1 + w_6 b_2 + w_{11} B_2} + e^{w_7 b_1 + w_8 b_2 + w_{12} B_2}}$$

$$C_2 = \frac{e^{w_7 b_1 + w_8 b_2 + w_{12} B_2}}{e^{w_5 b_1 + w_6 b_2 + w_{11} B_2} + e^{w_7 b_1 + w_8 b_2 + w_{12} B_2}}$$

$$\frac{\partial E}{\partial w_5} = \frac{\partial E}{C_1} \times \frac{\partial C_1}{\partial w_5} + \frac{\partial E}{\partial C_2} \times \frac{\partial C_2}{\partial w_5}$$

$$= -\frac{y_1}{C_1} \times \frac{\left(e^{\cdots} + e^{\cdots}\right) \times \left(e^{w_5 \cdots} \times b_1\right) - \left(e^{w_5 \cdots}\right)\left(e^{w_5 \cdots} \times b_1\right)}{\left(e^{\cdots} + e^{\cdots}\right)^2}$$

$$-\frac{y_2}{C_2} \times \frac{\left(e^{\cdots} + e^{\cdots}\right) \times 0 - \left(e^{w_7 \cdots}\right)\left(e^{w_5 \cdots} \times b_1\right)}{\left(e^{\cdots} + e^{\cdots}\right)^2}$$

$$\frac{\partial E}{\partial w_5} = -\frac{y_1}{c_1} \times \left(c_1 - c_1^2 b_1\right) + \frac{y_2}{c_2} \times c_1 c_2 b_1$$

$$\frac{\partial E}{\partial w_5} = -y_1 \times b_1 \times (1 - c_1) + y_2 c_1 b_1$$

$$\frac{\partial E}{\partial w_6} = \frac{\partial E}{c_1} \times \frac{\partial c_1}{\partial w_6} + \frac{\partial E}{\partial c_2} \times \frac{\partial c_2}{\partial w_6}$$

$$= -\frac{y_1}{c_1} \times \frac{\left(e^{\sim} + e^{\sim}\right)\left(e^{w_5 \cdots} \times b_2\right) - \left(e^{w_5 \cdots}\right)\left(e^{w_5 \cdots} \times b_2\right)}{\left(e^{\sim} + e^{\sim}\right)^2}$$

$$-\frac{y_2}{c_2} \times \frac{\left(e^{\sim} + e^{\sim}\right) \times 0 - \left(e^{w_7 \cdots}\right)\left(e^{w_8 \cdots} \times b_2\right)}{\left(e^{\sim} + e^{\sim}\right)^2}$$

$$\frac{\partial E}{\partial w_6} = -\frac{y_1}{c_1} \times \left(c_1 b_2 - c_1^2 b_2\right) + \frac{y_2}{c_2} \times c_1 c_2 b_2$$

$$\frac{\partial E}{\partial w_6} = -y_1 b_2 (1 - c_1) + y_2 c_1 b_2$$

$$\frac{\partial E}{\partial w_7} = \frac{\partial E}{\partial c_1} \times \frac{\partial c_1}{\partial w_7} + \frac{\partial E}{\partial c_2} \times \frac{\partial c_2}{\partial w_7}$$

$$= -\frac{y_1}{c_1} \times \frac{\left(e^{\sim} + e^{\sim}\right) \times 0 - \left(e^{w_5}\right)\left(e^{w_7 \cdots} \times b_1\right)}{\left(e^{\sim} + e^{\sim}\right)^2}$$

$$-\frac{y_2}{c_2} \times \frac{\left(e^{\sim} + e^{\sim}\right)\left(e^{w_7 \cdots} \times b_1\right) - \left(e^{w_7}\right)\left(e^{w_7} \times b_1\right)}{\left(e^{\sim} + e^{\sim}\right)^2}$$

$$\frac{\partial E}{\partial w_7} = +\frac{y_1}{c_1} \times c_1 c_2 b_1 - \frac{y_2}{c_2} \times \left(c_2 b_1 - c_2^2 b_1\right)$$

$$\frac{\partial E}{\partial w_7} = y_1 c_2 b_1 - y_2 b_1 (1 - c_2)$$

$$\frac{\partial E}{\partial w_8} = \frac{\partial E}{\partial c_1} \times \frac{\partial c_1}{\partial w_8} + \frac{\partial E}{\partial c_2} \times \frac{\partial c_2}{\partial w_8}$$

$$= -\frac{y_1}{c_1} \times \frac{(e^{\sim} + e^{\sim}) \times 0 - (e^{w_5 \cdots})(e^{w_7 \cdots} \times b_2)}{(e^{\sim} + e^{\sim})^2}$$

$$-\frac{y_2}{c_2} \times \frac{(e^{\sim} + e^{\sim})(e^{w_7 \cdots} \times b_2) - (e^{w_7 \cdots})(e^{w_7 \cdots} \times b_2)}{(e^{\sim} + e^{\sim})^2}$$

$$= +\frac{y_1}{c_1} c_1 c_2 b_2 - \frac{y_2}{c_2} \times \left(c_2 b_2 - c_2^2 b_2\right)$$

$$\frac{\partial E}{\partial w_8} = y_1 c_2 b_2 - y_2 b_2 (1 - c_2)$$

$$\frac{\partial E}{\partial w_{11}} = \frac{\partial E}{\partial c_1} \times \frac{\partial c_1}{\partial w_{11}} + \frac{\partial E}{\partial c_2} \times \frac{\partial c_2}{\partial w_{11}}$$

$$= -\frac{y_1}{c_1} \times \frac{(e^{\sim} + e^{\sim})(e^{w_5 \cdots} \times \beta_2) - (e^{w_5 \cdots})(e^{w_5 \cdots} \times \beta_2)}{(e^{\sim} + e^{\sim})^2}$$

$$-\frac{y_2}{c_2} \times \frac{(e^{\sim} + e^{\sim}) \times (0) - (e^{w_7 \cdots})(e^{w_5 \cdots} \times \beta_2)}{(e^{\sim} + e^{\sim})^2}$$

$$= -\frac{y_1}{c_1} \times \left(c_1 \beta_2 - c_1^2 \beta_2\right) + \frac{y_2}{c_2} \times c_1 c_2 \beta_2$$

$$\frac{\partial E}{\partial w_{11}} = -y_1 \beta_2 (1 - c_1) + y_2 c_1 \beta_2$$

$$\frac{\partial E}{\partial w_{12}} = \frac{\partial E}{\partial c_1} \times \frac{\partial c_1}{\partial w_{12}} + \frac{\partial E}{\partial c_2} \times \frac{\partial c_2}{\partial w_{12}}$$

$$= -\frac{y_1}{c_1} \times \frac{(e^{\cdots}+e^{\cdots})(0) - (e^{w_5\cdots})(e^{w_3\cdots} \times \beta_2)}{(e^{\cdots}+e^{\cdots})^2}$$

$$-\frac{y_2}{c_2} \times \frac{(e^{\cdots}+e^{\cdots})(e^{w_3\cdots} \times \beta_2) - (e^{w_4\cdots})(e^{w_3\cdots} \times \beta_2)}{(e^{\cdots}+e^{\cdots})^2}$$

$$= +\frac{y_1}{c_1} \times c_1 c_2 \beta_2 - \frac{y_2}{c_2} \times (c_2\beta_2 - c_2^2\beta_2)$$

$$\frac{\partial E}{\partial w_{12}} = y_1 c_2 \beta_2 - y_2 \beta_2 (1 - c_2)$$

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{c_1} \times \frac{\partial c_1}{\partial b_1} \times \frac{\partial b_1}{\partial w_1} + \frac{\partial E}{\partial c_2} \times \frac{\partial c_2}{\partial b_1} \times \frac{\partial b_1}{\partial w_1}$$

$$\frac{\partial c_1}{\partial b_1} = \frac{(e^{\cdots}+e^{\cdots})(e^{w_5\cdots} \times w_5) - (e^{w_5\cdots})(e^{w_5\cdots} \times w_5 + e^{w_3\cdots} \times w_3)}{(e^{\cdots}+e^{\cdots})^2}$$

$$\frac{\partial c_1}{\partial b_1} = c_1 w_5 - c_1 (c_1 w_5 + c_2 w_3)$$

$$\frac{\partial c_2}{\partial b_1} = \frac{(e^{\cdots}+e^{\cdots})(e^{w_3\cdots} \times w_3) - (e^{w_3\cdots})(e^{w_5\cdots} \times w_5 + e^{w_3\cdots} \times w_3)}{(e^{\cdots}+e^{\cdots})^2}$$

$$\frac{\partial c_2}{\partial b_1} = c_2 w_3 - c_2 (c_1 w_5 + c_2 w_3)$$

$$\frac{\partial b_1}{\partial w_1} = (1 - \tanh^2(w_1 a_1 + w_2 a_2 + w_3 \beta_1)) \times a_1$$

$$\frac{\partial E}{\partial w_1} = -\frac{y_1}{c_1} \times (c_1 w_5 - c_1 (c_1 w_5 + c_2 w_3)) \times \frac{\partial b_1}{\partial w_1} - \frac{y_2}{c_2} \times (c_1 w_3 - c_2 (c_1 w_5 + c_1 w_3)) \times \frac{\partial b_1}{\partial w_1}$$

$$\frac{\partial E}{\partial w_1} = \Big(-y_1\,(w_5 - c_1 w_5 - c_2 w_7) - y_2\,(w_7 - c_1 w_5 - c_2 w_7)\Big) \times \Big(1 - \tanh^2(w_1 a_1 + w_2 a_2 + w_9 \beta_1) \times a_1\Big)$$

$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial c_1} \times \frac{\partial c_1}{\partial b_1} \times \frac{\partial b_1}{\partial w_2} + \frac{\partial E}{\partial c_2} \times \frac{\partial c_2}{\partial b_1} \times \frac{\partial b_1}{\partial w_2}$$

$$\frac{\partial E}{\partial w_2} = \Big(-y_1\,(w_5 - c_1 w_5 - c_2 w_7) - y_2\,(w_7 - c_1 w_5 - c_2 w_7)\Big) \times \Big(1 - \tanh^2(w_1 a_1 + w_2 a_2 + w_9 \beta_1) \times a_2\Big)$$

$$\frac{\partial E}{\partial w_9} = \Big(-y_1\,(w_5 - c_1 w_5 - c_2 w_7) - y_2\,(w_7 - c_1 w_5 - c_2 w_7)\Big) \times \Big(1 - \tanh^2(w_1 a_1 + w_2 a_2 + w_9 \beta_1) \times \beta_1\Big)$$

$$\frac{\partial E}{\partial w_3} = \frac{\partial E}{\partial c_1} \times \frac{\partial c_1}{\partial b_2} \times \frac{\partial b_2}{\partial w_3} + \frac{\partial E}{\partial c_2} \times \frac{\partial c_2}{\partial b_2} \times \frac{\partial b_2}{\partial w_3}$$

$$\frac{\partial c_1}{\partial b_2} = \frac{\left(e^{\sim} + e^{\sim}\right)\left(e^{w_5 \cdots} \times w_6\right) - \left(e^{w_5 \cdots}\right)\left(e^{w_5 \cdots} \times w_6 + e^{w_7 \cdots} \times w_8\right)}{\left(e^{\sim} + e^{\sim}\right)^2}$$

$$\frac{\partial c_1}{\partial b_2} = c_1 w_6 - c_1\left(c_1 w_6 + c_2 w_8\right)$$

$$\frac{\partial c_2}{\partial b_2} = \frac{\left(e^{\sim} + e^{\sim}\right)\left(e^{w_7 \cdots} \times w_8\right) - \left(e^{w_7 \cdots}\right)\left(e^{w_5 \cdots} \times w_6 + e^{w_7 \cdots} \times w_8\right)}{\left(e^{\sim} + e^{\sim}\right)^2}$$

$$\frac{\partial c_2}{\partial b_2} = c_2 w_8 - c_2\left(c_1 w_6 + c_2 w_8\right)$$

$$\frac{\partial b_2}{\partial w_3} = \left(1 - \tanh^2\left(w_3 a_1 + w_4 a_2 + w_{10} \beta_1\right)\right) \times a_1$$

$$\frac{\partial E}{\partial w_3} = -\frac{y_1}{c_1} \times \left(c_1 w_6 - c_1\left(c_1 w_6 + c_2 w_8\right)\right) \times \frac{\partial b_2}{\partial w_3} - \frac{y_2}{c_2} \times \left(c_2 w_8 - c_2\left(c_1 w_6 + c_2 w_8\right)\right) \times \frac{\partial b_2}{\partial w_3}$$

$$\frac{\partial E}{\partial w_3} = \left(-y_1\left(w_6 - c_1 w_6 - c_2 w_8\right) - y_2\left(w_8 - c_1 w_6 - c_2 w_8\right)\right) \times \left(1 - \tanh^2\left(w_3 a_1 + w_4 a_2 + w_{10}\beta_1\right) \times a_1\right)$$

$$\frac{\partial E}{\partial w_4} = \left(-y_1\left(w_6 - c_1 w_6 - c_2 w_8\right) - y_2\left(w_8 - c_1 w_6 - c_2 w_8\right)\right) \times \left(1 - \tanh^2\left(w_3 a_1 + w_4 a_2 + w_{10}\beta_1\right) \times a_2\right)$$

$$\frac{\partial E}{\partial w_{10}} = \left(-y_1\left(w_6 - c_1 w_6 - c_2 w_8\right) - y_2\left(w_8 - c_1 w_6 - c_2 w_8\right)\right) \times \left(1 - \tanh^2\left(w_3 a_1 + w_4 a_2 + w_{10}\beta_1\right) \times \beta_1\right)$$

Gaussian Distribution

$x_1, x_2, \ldots x_N$, $x \sim p(x|\mu, \sigma^2)$

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \quad , \quad \sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$$

$$p(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\,\sigma} \times e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

- x: random variable
- μ: location of the peak
- σ: width of the curve
- $(x-\mu)$: how far x is from μ [how likely x is]

x's closer to μ have higher probability.

Multivariate

$[x_1 \quad x_2 \quad \ldots \quad x_D]^T \rightarrow x$ in D-dimensional space

↳ If this is a mobile rating for each feature.

Eg: [3 4 2 1]
- 3 → display
- 4 → battery
- 2 → camera
- 1 → price

Model these vectors as multivariate gaussian dist. as features may be correlated. → Price depends on display, camera, etc.

- Multivariate gaussian dist. is useful when modelling numerical data with multiple dimensions & all dimensions are continuous valued.

$$P(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

(x: $D \times 1$, $(x-\mu)^T$: $1 \times D$, Σ: $D \times D$)

Trace of a matrix $= \sum_{i=1}^{N} \lambda_i$

$|\Sigma|$ → multiplication of eigen values of Σ.

μ → also a D-dimensional vector

Σ → covariance matrix of size $D \times D$.

$$\Delta^2 = (x-\mu)^T \Sigma^{-1} (x-\mu)$$

Mahalanobi's distance → is a measure of the distance between a point and a distribution. It is a generalization of the Euclidian distance, taking into account the correlations b/w the variables. It is calculated as the square root of the difference b/w the vector representation of an observation & the mean of the distribution, scaled by the covariance matrix of the distribution.

Commonly used in ML to detect outliers, classification & quantify the similarity b/w two points. The distance can be interpreted as the no. of standard deviations separating the point from the mean of the distribution, taking into account, the correlation b/w the variables.

If A is invertible $\Rightarrow |A| \neq 0$, so, no $\lambda_i = 0$.

When $\Sigma^{-1} = I \Rightarrow$ then $\Delta^2 = (x-\mu)^T (x-\mu)$ → Euclidean Distance

→ Usually in guassian dist. Σ^{-1} is symmetric (As Σ is symmetric & real-valued.

→ If A is a real-valued matrix,

$$A = \frac{1}{2}(A + A^T) + \frac{1}{2}(A - A^T)$$

$(A + A^T)$ → Symmetric (B) $\Rightarrow M^T = M$

$(A - A^T)$ → Anti-symmetric (C) $\Rightarrow M^T = -M$

$B_{ij} = A_{ij} + A_{ji}$ → '+', so B_{ij} is also same

$C_{ij} = A_{ij} - A_{ji}$ → '-', so, $C_{ji} = -C_{ij}$

→ If A is anti-symmetric (or skew-symmetric), $x^T A x = 0$, for any x

$A_{ij} = -A_{ji}$ & all diagonal elements are 0

$$x^T A x = \sum_{i,j \in D} x_i \cdot x_j \cdot A_{ij}$$

$A_{ji} = -A_{ij}$, so $x_i \cdot x_j \cdot A_{ij} + x_j \cdot x_i \cdot A_{ji} = 0$

So, we assume Σ^{-1} is symmetric in $(x-\mu)^T \Sigma^{-1} (x-\mu)$.

↳ For this not to be 0.

$\underset{D\times D}{\Sigma} \times \underset{D\times 1}{u_i} = \lambda_i u_i$

λ_i ↳ eigen value; u_i ↳ eigen vector

[Σ = full-rank matrix]

eigen vector of a matrix is a vector which when multiplied by that matrix, will not change its direction, but only gets multiplied by a scalar.

$U = [u_1 \; u_2 \; \cdots \; u_p]$ ← eigen vectors

↳ only if u_i & u_j correspond to different eigen values.

↳ i.e. u_i & u_j are orthonormal → orthogonal + unit vector

For any u_i & u_j, $u_i^T u_j = 0$

& $UU^T = U^T U = I$

& $u_i^T u_i = 1$

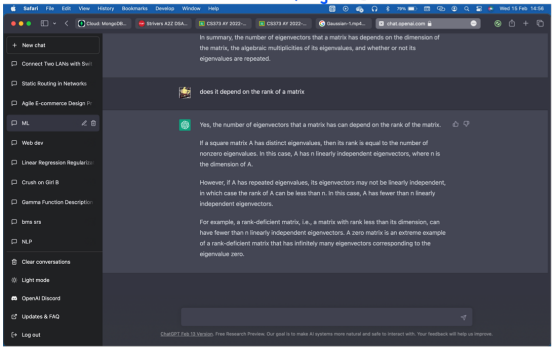
↳ If u_i & u_j are from same eigen values, dot product can be non-zero

$\Sigma = U \Lambda U^T$ ← eigen vector matrix

↳ because diagonal elements represent $u_i^T u_i = 1$ & off-diagonal represent $u_i^T u_j = 0$

↓ eigen value diagonal matrix

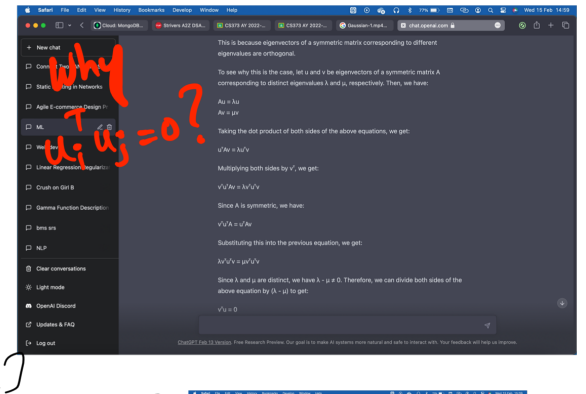
$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \\ 0 & 0 & \cdots & & \lambda_p \end{bmatrix}$



$\Sigma = \sum_{i=1}^{D} \lambda_i u_i u_i^T$

$D\times 1$ $1\times D$ ⇒ $D\times D$

u_i ↳ column vector; u_i^T ↳ row vector

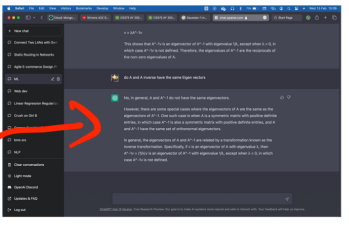


$u_i = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$

$u_i^T = [a_1 \; a_2]$

$u_i u_i^T = \begin{bmatrix} a_1^2 & a_1 a_2 \\ a_2 a_1 & a_2^2 \end{bmatrix}$

$\Sigma^{-1} = U \Lambda^{-1} U^T = \sum_{i=1}^{D} \frac{1}{\lambda_i} u_i u_i^T$



$$\Delta^2 = (x-\mu)^T \Sigma^{-1} (x-\mu) \rightarrow \text{Mahalanobi's Distance}$$

$$= (x-\mu)^T \left[\sum_{i=1}^{D} \frac{1}{\lambda_i} u_i u_i^T\right](x-\mu) = z^T \left[\sum_{i=1}^{D} \frac{1}{\lambda_i} u_i u_i^T\right] z$$

$z = x - \mu$

$$= \sum_{i=1}^{D} \frac{1}{\lambda_i} z^T u_i u_i^T z$$

(1×D, D×1, 1×D, D×1)

↳ Scalar = y

$$= \sum_{i=1}^{D} \frac{y_i^2}{\lambda_i}$$ ↳ indexed by eigen vector i

$$p(x/\mu,\Sigma) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}\sum_{i=1}^{D} \frac{y_i^2}{\lambda_i}}$$

→ product of eigen vectors

$$|\Sigma| = \prod_{i=1}^{D} \lambda_i$$

$$|\Sigma|^{1/2} = \prod_{i=1}^{D} \lambda_i^{1/2}$$

$y = U(x-\mu)$ → acting as a transformation matrix

↳ D×1 vector

$$= \prod_{i=1}^{D} \frac{1}{(2\pi)^{1/2}(\lambda_i)^{1/2}} \times e^{-\frac{1}{2} \times (y_i^2/\lambda_i)}, \quad y_i \sim N(y/0, \lambda_i)$$

↳ also a gaussian

Eg: $x = [3 \quad 4 \quad 2 \quad 1]$

let correspond $y = [0.2 \quad 0.8 \quad -0.2 \quad 0.6]$

let $\mu = [2 \quad 3 \quad 0.5 \quad 0.5]$

$(x-\mu) = [1 \quad 1 \quad 1.5 \quad 0.5]$

$\therefore U(x-\mu) = y =$ ——

→ there is some correlation b/w features of x,

↳ 0.2 doesn't correspond to only one dimension, but encapsulates correlation of all features. Each y_i is independent.

$$p(y/0,\lambda) = \prod_{i=1}^{D} \frac{1}{\sqrt{2\pi}\sqrt{\lambda_i}} e^{-\frac{1}{2}\frac{y_i^2}{\lambda_i}}$$

$\lambda \rightarrow \sigma^2$

$$\mu = \int_{-\infty}^{\infty} x\, P(x)\, dx$$ → for univariate

$$\sigma^2 = E\left[(x-\mu)^2\right] = E\left(x^2 + \mu^2 - 2\mu x\right) = E[x^2] + \mu^2 - 2\mu E[x]$$

$$= E[x^2] + \mu^2 - 2\mu^2$$

$$\sigma^2 = E[x^2] - \mu^2 = E[x^2] - (E[x])^2$$

$$U = \begin{bmatrix} u_1^T \\ u_2^T \\ \vdots \\ u_p^T \end{bmatrix}, \quad y = U\,[x-\mu]$$

$D\times 1 \quad D\times D \quad D\times 1$

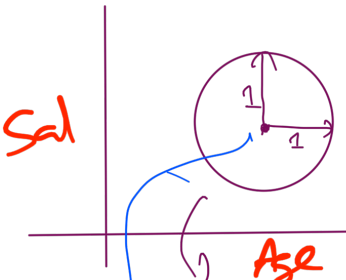
- Transformation preserves length of any vector
- And also angle b/w any two vectors.

→ These properties make this transformation rigid.

Eg: Two-variate space.

$$\Sigma = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

→ Salary & age



→ All points on this surface have equal probability.

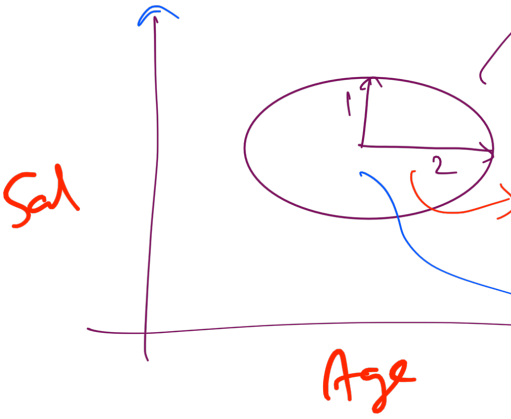
i.e; Same value of $P(x)$

Contour

Reason

Now, $\Sigma = I = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$

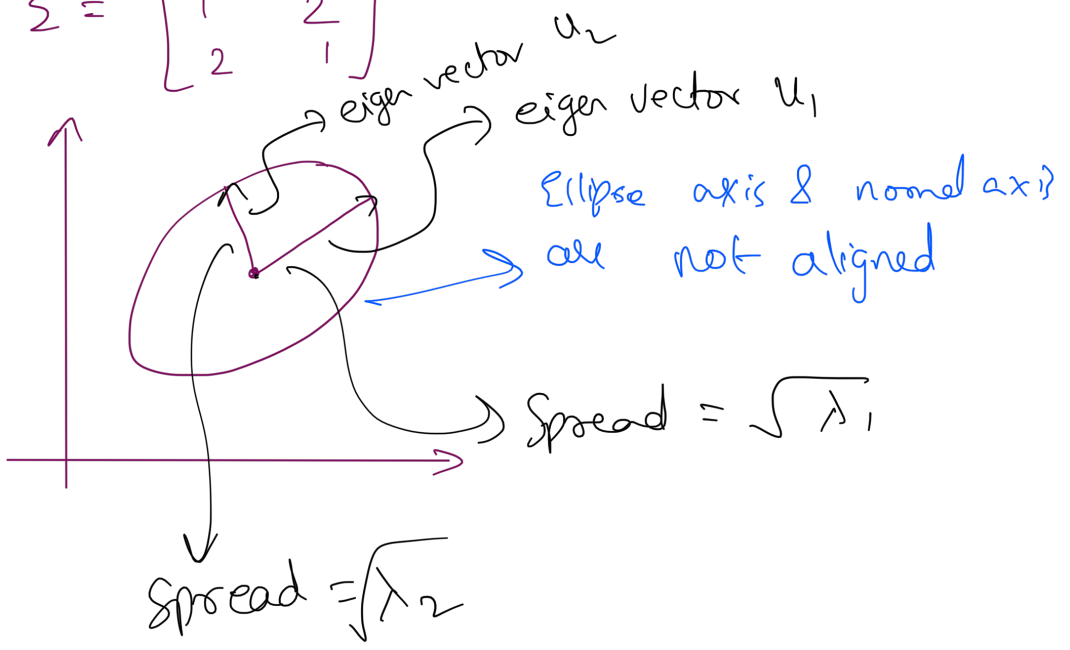
Ellipse axis & normal axis are aligned



→ Spread on x-axis is twice that of y-axis.

Ellipse axis & normal axis are aligned

Eg: $\Sigma = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$



Multivariate $\Rightarrow$ $p(x) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \times e^{-1/2 (x-\mu)^T \Sigma^{-1} (x-\mu)}$

$$E(x) = \int_{-\infty}^{\infty} x\, p(x)\, dx = \int_{-\infty}^{\infty} (d+\mu) \times \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} e^{-\frac{1}{2} d^T \Sigma^{-1} d}\, dd$$

$\rightarrow$ differentiate w.r.t. d.

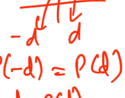
$$= \int_{-\infty}^{\infty} \mu\, p(x)\, dx + \int_{-\infty}^{\infty} d\, p(d)\, dd$$

$= \mu \times 1 \Rightarrow$ as $\int_{-\infty}^{\infty} p(x)\, dx = 1$

$= 0$

This is an odd function, so $\int_{-\infty}^{\infty} \text{odd func} = 0$

as $p(d)$ is gaussian so,



So, $p(-d) = p(d)$

So, $d \times p(d) = -(-d) \times p(-d)$

$$E(x) = \mu$$

$$E(x^2) = E(xx^T)$$

if $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix}$

All possible multiplications

$$xx^T = \begin{bmatrix} x_1^2 & x_1 x_2 & \cdots & x_1 x_D \\ x_2 x_1 & x_2^2 & \cdots & x_2 x_D \\ \vdots & & & \\ x_D x_1 & \cdots & \cdots & x_D^2 \end{bmatrix}$$

$$E(xx^T) = \int_{-\infty}^{\infty} x x^T p(x)\, dx = \int_{-\infty}^{\infty} [d+\mu][d+\mu]^T p(d)\, dd$$

$$= \int_{-\infty}^{\infty} dd^T p(d)\, dd + 2\int_{-\infty}^{\infty} d\mu^T p(d)\, dd + \mu\mu^T \int_{-\infty}^{\infty} p(d)\, dd$$

0 ⇒ as odd function

1

as $\int_{-\infty}^{\infty} P(x)\, dx = 1$

$$E(xx^T) = \int_{-\infty}^{\infty} dd^T p(d)\, dd + \mu\mu^T$$

W.K.T. $y = U[x-\mu] = Ud \Rightarrow U^T y = d$

⇒ $U^T y = U^T U d$

$U^T y = Id = d$

$$U^T = \begin{bmatrix} u_{11} & u_{21} & \cdots & u_{D1} \\ u_{12} & u_{22} & \cdots & u_{D2} \\ \vdots & \vdots & & \vdots \\ u_{1D} & u_{2D} & & u_{DD} \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_D \end{bmatrix}$$

(columns: u_1, u_2, …)

$$d = U^T y = \sum_{i=1}^{D} u_i y_i$$

(u_i → vector, y_i → scalar)

$$dd^T = \left[\sum_{i=1}^{D} u_i y_i\right]\left[\sum_{j=1}^{D} u_j y_j\right]^T = \sum_{i=1}^{D} u_i y_i \sum_{j=1}^{D} u_j^T y_j$$

$$dd^T = \sum_{i=1}^{D}\sum_{j=1}^{D} y_i y_j u_i u_j^T$$

$$E(xx^T) = \int_{-\infty}^{\infty} dd^T p(d)\, dd + \mu\mu^T$$

$$= \int_{-\infty}^{\infty} \sum_{i=1}^{D}\sum_{j=1}^{D} y_i y_j u_i u_j^T\, p(d)\, dd + \mu\mu^T$$

$$= \sum_{i=1}^{D}\sum_{j=1}^{D} u_i u_j^T \int_{-\infty}^{\infty} y_i y_j\, p(y)\, dy$$

$u_i^T u_j = 0$, but $u_i u_j^T \neq 0$, as $u_i = u_j = D \times 1$ & $u_i^T = u_j^T = 1 \times D$

$$\prod_{i=1}^{D} \frac{1}{\sqrt{2\pi}\sqrt{\lambda_i}} e^{-\frac{1}{2}\frac{y_i^2}{\lambda_i}}$$

$$E(xx^T) = \sum_{i=1}^{D}\sum_{j=1}^{D} u_i u_j^T \int_{-\infty}^{\infty} y_i y_j \left[\prod_{k=1}^{D} p(y_k)\, dy_k\right]$$

when $i \neq j$,

$$\int_{-\infty}^{\infty} y_i p(y_i) dy_i \times \int_{-\infty}^{\infty} y_j p(y_j) dy_j \times \prod_{\substack{k=1 \\ k\neq i, k\neq j}}^{D} \int p(y_k) dy_k$$

↳ Bring out $p(y_i)dy_i$ & $p(y_j)dy_j$ as it is multiplication

↳ 0 ↳ 0 ↳ 1

when $i = j$,

$$E(xx^T) = \sum_{i=1}^{D} u_i u_i^T \int_{-\infty}^{\infty} y_i^2 p(y) dy$$

$$= \sum_{i=1}^{D} u_i u_i^T \int_{-\infty}^{\infty} y_i^2 p(y_i) dy_i \times \prod_{k=1}^{D} \int_{-\infty}^{\infty} p(y_k) dy_k$$

↳ 1

$$= \sum_{i=1}^{D} u_i u_i^T \lambda_i + \mu\mu^T$$

↳ In univariate case, $\sigma^2 = E[y_i^2] - (E[y_i])^2$

$\lambda_i = \int_{-\infty}^{\infty} y_i^2 p(y_i) dy_i$ — 0

↳ As, $p(y_i)$ has $\sqrt{\lambda_i}$ & standard deviation and 0 as mean

$$E(xx^T) = \Sigma + \mu\mu^T$$

$$\Sigma = E(xx^T) - \mu\mu^T$$

Deriving conditional & marginal distribution of a gaussian distributed normal variable

Eg- random variable (gaussian dist.)

$= [\text{Battery}, \text{Camera}, \text{Display}, \text{pricing}, \text{hardware}]$

↳ Assume quality of all these features is available in numerical form

Example ⇒ $x = [5, 4, 3, 3, 3]$ ⇒ multivariate gaussian distributed variable

$$x \sim p(x(\mu, \Sigma) = N(x|\mu, \Sigma) = \frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}} \times e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

let $P = \Sigma^{-1}$ ↳ Precision Matrix

If these features are joint distributed according to gaussian distribution, what is the form of conditional probability of some features, given other features

↳ $p(x_r/x_c)$, where x_c = constant/evidence features

↳ conditional distribution given joint distribution is gaussian

x_r = random / o/p features

Here, $x = [\underbrace{5, 4}_{x_r}, \underbrace{3, 3, 3}_{x_c}]$, $P = \begin{matrix} & 2 & 3 \\ 2 & P_{rr} & P_{rc} \\ 3 & P_{cr} & P_{cc} \end{matrix}$, $P_{rc} = P_{cr}^T$

↳ Precision Matrix

Now, $-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)$

$$= -\frac{1}{2} \underbrace{[(x_r - \mu_r)^T}_{1\times 2}, \underbrace{(x_c - \mu_c)^T}_{1\times 3}] \begin{bmatrix} P_{rr} & P_{rc} \\ P_{cr} & P_{cc} \end{bmatrix} \begin{bmatrix} (x_r - \mu_r) \to 2\times 1 \\ (x_c - \mu_c) \to 3\times 1 \end{bmatrix}$$

$$= -\frac{1}{2}\left[\underbrace{(x_r-\mu_r)^T P_{rr} + (x_c-\mu_c)^T P_{cr}}_{1\times 2}, \underbrace{(x_r-\mu_r)^T P_{rc} + (x_c-\mu_c)^T P_{cc}}_{1\times 3}\right] \times \begin{bmatrix}(x_r-\mu_r) \to 2\times 1\\ (x_c-\mu_c) \to 3\times 1\end{bmatrix}$$

① involves quadratic term of x_r ② involves linear term of x_r

$$= -\frac{1}{2}\Big[(x_r-\mu_r)^T P_{rr}(x_r-\mu_r) + (x_c-\mu_c)^T P_{cr}(x_r-\mu_r) + (x_r-\mu_r)^T P_{rc}\times(x_c-\mu_c) + (x_c-\mu_c)^T \times P_{cc}\times(x_c-\mu_c)\Big]$$

② ④ ④ doesn't involve x_r term (constant)

$$-\frac{1}{2}(x-\mu)^T P (x-\mu) = -\frac{1}{2}x^T P x + \cancel{2}\, x^T P\mu + \mu^T P\mu$$

↓ Gaussian Dist. form. So we can say that $P(x_r/x_c)$ is of gaussian dist. form only.

$$= -\frac{1}{2}\Big[\underbrace{(x_r^T P_{rr} x_r)}_{\text{quadratic in } x_r} - 2^{*}\underbrace{x_r^T P_{rr}\mu_r}_{\text{linear in } x_r} + \underbrace{\mu_r^T P_{rr}\mu_r}_{\text{constant}} + (x_c-\mu_c)^T P_{cr} x_r + x_r^T P_{rc}(x_c-\mu_c) + (x_c-\mu_c)^T P_{cr}\,*\,(-\mu_r) + (-\mu_r)^T P_{rc}(x_c-\mu_c) + (x_c-\mu_c)^T P_{cc}(x_c-\mu_c)\Big]$$

(the last five terms → Constant)

Aim: To find μ & Σ for $P(x_r/x_c)$

↓

As when we talk about a gaussian dist, we have to know about its μ & Σ.

$$= -\frac{1}{2}\Big[(x_r^T P_{rr} x_r) - 2^{**} x_r^T P_{rr}\mu_r + 2^{**} x_r^T P_{rc}(x_c-\mu_c)\Big] + \text{const} + \text{const} + \text{const}.$$

→ As here no P_{rr}, so $P_{rr}\times P_{rr}^{-1} = I$

$$= -\frac{1}{2}\Big[x_r^T P_{rr} x_r\Big] + x_r^T P_{rr}\Big[\mu_r - P_{rr}^{-1}P_{rc}(x_c-\mu_c)\Big] + \text{const}.$$

$$\therefore P(x_r/x_c) = \mathcal{N}(x_r, \mu_{r/c}, \Sigma_{r/c})$$

Here, 2×1; $\mu_{r/c}$: 2×1; $\Sigma_{r/c}$: 2×2 matrix

$$\Sigma_{r|c} = P_{rr}^{-1}$$

$$\mu_{r|c} = \mu_r - P_{rr}^{-1} P_{rc} (x_c - \mu_c)$$

Cause, it is of the form $(x-\mu)^T \Sigma^{-1} (x-\mu)$

x_r^T P_{rr} x_r

→ Cause, μ is whatever is folled by a linear term in "x_r".

$$\Sigma = \begin{bmatrix} \Sigma_{rr} & \Sigma_{rc} \\ \Sigma_{cr} & \Sigma_{cc} \end{bmatrix}$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

$$A^{-1} = A - BD^{-1}C$$

So, $P_{rr}^{-1} = P_{rr} - P_{rc} P_{cc}^{-1} P_{cr}$

$$P = \Sigma^{-1}$$

$$P = \begin{bmatrix} \Sigma_{rr} & \Sigma_{rc} \\ \Sigma_{cr} & \Sigma_{cc} \end{bmatrix}^{-1} = \begin{bmatrix} P_{rr} & P_{rc} \\ P_{cr} & P_{cc} \end{bmatrix}$$

Generally,

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} M & -MBD^{-1} \\ -D^{-1}CM & D^{-1} + D^{-1}CMBD^{-1} \end{bmatrix}$$

$$M = (A - BD^{-1}C)^{-1}$$

$$\therefore P_{rr} = (P_{rr} - P_{rc} P_{cc}^{-1} P_{cr})^{-1}$$

$$\hookrightarrow P_{rr}^{-1} = P_{rr} - P_{rc} P_{cc}^{-1} P_{cr}$$

Marginal Gaussian Distribution

If joint distribution, $p(x_a, x_c)$ is gaussian, then the conditional distribution, $p(x_a | x_c)$ is also gaussian.

Now, find,

$$p(x_a) = \int p(x_a, x_c)\, dx_c$$

↳ also Gaussian

As we have to differentiate w.r.t. x_c, we need to consider the terms containing x_c from the same eqn. used in the conditional distribution

$$-\frac{1}{2}\Big[\underbrace{(x_a-\mu_a)^T \Lambda_{aa}(x_a-\mu_a)}_{\to \text{const}} + (x_c-\mu_c)^T \Lambda_{ca}(x_a-\mu_a) + (x_a-\mu_a)^T \Lambda_{ac}\times(x_c-\mu_c) + (x_c-\mu_c)^T \times \Lambda_{cc}\times(x_c-\mu_c)\Big]$$

$$= -\frac{1}{2}\Big[x_c^T \Lambda_{ca} x_a - x_c^T \Lambda_{ca}\mu_a + x_a^T \Lambda_{ac} x_c - \mu_a^T \Lambda_{ac} x_c + x_c^T \Lambda_{cc} x_c - \mu_c^T \Lambda_{cc} x_c + \text{const}\Big]$$

$$= -\frac{1}{2}\Big[x_c^T \Lambda_{ca} x_a - 2 x_c^T \Lambda_{ca}\mu_a + x_c^T \Lambda_{ac} x_a + x_c^T \Lambda_{cc} x_c - x_c^T \Lambda_{cc}\mu_c - \mu_c^T \Lambda_{ca} x_c\Big]$$

$$= -\frac{1}{2}\Big[2 x_c^T \Lambda_{ca} x_a + x_c^T \Lambda_{cc} x_c - 2 x_c^T \Lambda_{ca}\mu_a - 2 x_c^T \Lambda_{cc}\mu_c\Big)$$

$$= -\frac{1}{2} x_c^T \Lambda_{cc} x_c - x_c^T \Lambda_{ca} x_a + x_c^T \Lambda_{ca}\mu_a + x_c^T \Lambda_{cc}\mu_c$$

$$= -\frac{1}{2} x_c^T P_{cc} x_c + x_c^T \left(P_{cc} \mu_c - P_{cr} (x_r - \mu_r) \right)$$

let $m = \left[P_{cc} \mu_c - P_{cr} (x_r - \mu_r) \right]$

$$= -\frac{1}{2} x_c^T P_{cc} x_c + x_c^T m$$

$$= -\frac{1}{2} \left(x_c - P_{cc}^{-1} m \right)^T P_{cc} \left(x_c - P_{cc}^{-1} m \right) + \frac{1}{2} m^T P_{cc}^{-1} m$$

↳ standard quadratic form of a Gaussian dist.

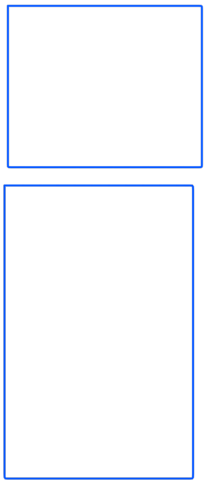
∴ When integrating over $p(c)$, that integral over x_c will take the form,

$$\int e^{-1/2 \left(x_c - P_{cc}^{-1} m \right)^T P_{cc} \left(x_c - P_{cc}^{-1} m \right)} dx_c$$

↳ integral over unnormalize gaussian.

So, result = reciprocal of the normalization coeff.

Semi-Supervised Learning



This is also a version of EM-algorithm

→ Some labelled data, remaining unlabelled data. Build
→ Build a model using the labelled data. ⇒ Sub-optimal (θ^0)
Then build an optimal classifier using the entire data
↳ θ^1
→ Then relabel D_U using θ^1, ⇒ you get θ^2

Do this till changes are not significant.

why does this work? → Find out.

↳ as some features that are not in D_L are in D_U. How do these features get weights? → Brief: These unknown features co-occur with some features that are already in DL.

→ This will work well only if Naive bayes classifier is used. Not so good when logistic regression, SVM, Gaussian neural nets are used as they are discriminative.

$I(f_i \in x_n)$ ⇒ gets all x_i such that f_i is in x_i

$KL(P(y|x_n) \| P_\theta(y|x_n))$ → model outputs

ground truth ⟹ [0.95, 0.05] (→ true, → -ve)
↳ for a all documents containing feature f_i

$$\sum_{i=1}^{F} \sum_{n=1}^{|D_L|} KL\left(P_i(y|x_n) \,\|\, P_\theta(y|x_n)\right)$$

→ Labelled Features

Labelled feature learning is weak-supervised

⇒ Use if you think the presence of some features indicates a particular class.

→ let there be a feature vector matrix X containing 2000 documents of which not all are labelled.

- let the word excellent be present in 100 documents & we assume excellent to be a positive label indicator ⇒ [0.95 0.05] ⇒ Labelled feature
  y=+ve y=-ve

  → x contains excellent as a word

  $P(y \mid x \in \text{"Excellent"}) = [0.95,\ 0.05]$

  → So, the loss function for this specific feature will contain only these 100 documents.

Assume there are 10 positive labelled features, 10 negative labelled features, & 10 neutral labelled features.

If no. of words in vocab is 1 lakh,

then model size ⇒ 3×1 lakh ⇒ $L \times K$ → vocab. size

3 → +ve, -ve, neutral

L → no. of labels

Shouldn't be 3×30 as to include only the labelled features, as, other features also contribute to the label given to the document.

$$P(y/x \in \text{"Excellent"}) = [\underset{y=2}{0.90} \quad \underset{y=1}{0.05} \quad \underset{y=0}{0.05}] \Rightarrow \tilde{P}_k$$

↳ k

↳ probability vector

$KL(y \,||\, \hat{P}_\theta(y/x)) \Rightarrow$ Loss function for supervised classification

↳ one-hot vector

$$\hat{P}_\theta(y/I(f_k \in x)) = \frac{1}{\sum_{n=1}^{N} 1 \times I(f_k \in x_n)} \sum_{n=1}^{N} I(f_k \in x_n)\, P_\theta(y/x_n)$$

↳ parameter matrix of size $L \times K$

↳ Count of docs. containing f_k

Eg:

0.8	0.1	0.1
0.9	0.05	0.05
0.7	0.2	0.1

⇒ let these be the labels given by our model to docs containing f_k

$[0.8 \quad 0.11 \quad 0.09] \Rightarrow$ average the 3 vectors.

↳ We aren't working on per document classification here, as the ground truth for labelled features. Say that all docs containing it must have prob. dist. similar to it. So we work on avg. of all docs containing that feature

Now, if ground truth is $[0.90 \quad 0.05 \quad 0.05]$

Then for feature $f_k \Rightarrow KL([0.90 \quad 0.05 \quad 0.05], [0.8 \quad 0.11 \quad 0.09])$

Using labelled features, we reduce the time taken to get gold labels

$$\text{loss} \Rightarrow \sum_{k=1}^{K} KL\left(\tilde{P}_k(y \mid I(f_k \in n)) \,\|\, \hat{P}_{k\theta}(y \mid I(f_k \in n))\right)$$

Gradient update equations

$$L = \sum_{k=1}^{K} \sum_{y} \tilde{P}_k(y \mid I(f_k \in n)) \log \frac{\tilde{P}_k(y \mid I(f_k \in n))}{\hat{P}_{k\theta}(y \mid I(f_k \in n))}$$

$$L = \sum_{k=1}^{K} \sum_{y} \left[\tilde{P}_k(y \mid I(f_k \in n)) \log \tilde{P}_k(y \mid I(f_k \in n)) - \tilde{P}_k(y \mid I(f_k \in n)) \log \hat{P}_{k\theta}(y \mid I(f_k \in n))\right]$$

$$L = \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{l=1}^{|y|} \left[\tilde{P}_k(y=l \mid I(f_k \in n_n)) \log \tilde{P}_k(y=l \mid I(f_k \in n_n)) - \tilde{P}_k(y=l \mid I(f_k \in n_n)) \log \hat{P}_{k\theta}(y=l \mid I(f_k \in n_n))\right]$$

$$\frac{\partial P_\theta^{(y=l/n)}}{\partial \theta_{ij}} = P_l\left(-P_i + I(l=i)\right) n_j \quad \Rightarrow P_l = \frac{e^{w_l^T n}}{\sum_y e^{w_y^T n}}$$

$$\frac{\partial (L)}{\partial \theta_{ij}} = -\sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{l=1}^{|y|} \frac{\tilde{P}_k(y=l \mid I(f_k \in n_n)}{\hat{P}_{k\theta}(y=l \mid I(f_k \in n_n))} \times \hat{P}_{k\theta}(y=l \mid I(f_k \in n_n)) \times \left(-\hat{P}_{k\theta}(y=i \mid I(f_k \in n_n) + I(l=i)\right) \times n_{nj}$$

$$\frac{\partial}{\partial \theta_{ij}}(L) = -\sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{l=1}^{|y|} I(f_k \in n_n) \times \tilde{P}_k(y=l/n_n) \times \left(-\hat{P}_{k\theta}(y=i/n_n) + I(l=i)\right) \times n_{nj}$$

K-means Clustering

Optimization Objective ⇒ $\sum_{c=1}^{C} \sum_{i=1}^{N_c} \|x_i - \mu_c\|_2^2$

- N_c → no. of datapoints in cluster "c"
- $\|x_i - \mu_c\|_2^2$ → within cluster sum of square
- $\|\cdot\|_2$ → L2-norm

Algorithm

(1) Initialize K random points in a D-dimensional space randomly. $(\mu_1, \mu_2, \ldots, \mu_K)$ ⇒ Cluster centroids.

(2) For each data point $x_1, \ldots, x_N$, assign it to nearest cluster center.

(3) Calculate new cluster centers.

$\mu_{1\ldots K}^{t+1}$ = mean of data points in respective cluster.

Probabilistic K-means

While assigning cluster to a data point, assign probabilities of this data point belonging to each cluster.

Model real valued ⇒ Gaussian dist.
Model text ⇒ Multinomial dist.
Model arrival times ⇒ Poisson

→ What kind of distribution to use for a particular problem?

→ Assum mixture of that particular distribution in real world.

Gaussian Mixture Models

→ Expectation - Maximization
↳ used when something is unobserved.

Assume "C" mixtures. Mixture model has $\mu_1, \Sigma_1, \mu_2, \Sigma_2, \cdots, \mu_C, \Sigma_C$

$$x \sim p(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

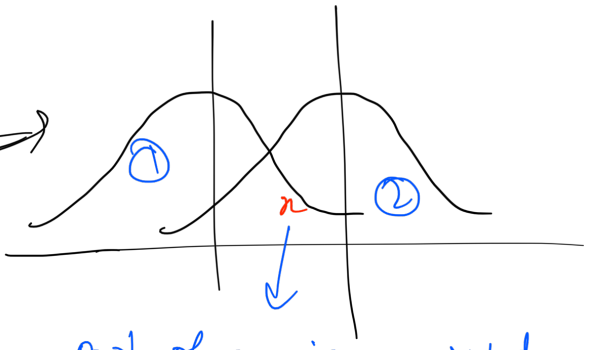
↳ for single gaussian

↳ $\mathcal{N}(x \mid \mu, \Sigma)$

$x \sim$ Mixture of gaussians

$$= \sum_{i=1}^{C} \pi_i \times \mathcal{N}(x \mid \mu_i, \Sigma_i)$$

↳ weight of gaussian "i"



prob of x is weighted sum of likelihoods that x come from ① & ②.

→ No. of parameters if C=2 & dimensions of data points is 5 (D=5),

$\mu_1 \Rightarrow 5 \times 1$

$\mu_2 \Rightarrow 5 \times 1$

$\Sigma_1 \Rightarrow 5 \times 5 \quad \Rightarrow$ Max $\Rightarrow$ 62 parameters

$\Sigma_2 \Rightarrow 5 \times 5$

$\pi_1 \Rightarrow 1$

$\pi_2 \Rightarrow 1$ (or) $(1 - \pi_1)$

E-M steps

① Initialization $\Rightarrow \pi = [\pi_1, \pi_2, \pi_3, \ldots, \pi_K] \rightarrow K \times 1$

$\mu = [\mu_1, \mu_2, \ldots, \mu_K] \rightarrow K \times D$

Each mean is a D-dimensional vector

$\Sigma_1, \Sigma_2, \ldots, \Sigma_K \Rightarrow K \times D \times D$

② E-step $\Rightarrow$ Calculate the probability of each point x_i to be from each of the mixture components. O/P is a K-dimensional probability vector, for each point.

③ M-step $\Rightarrow$ Recalculate $\mu_{1..K}$, $\Sigma_{1..K}$ & π.

$$\mu_K = \frac{1}{N_K} \sum_{x \in X} P_{iK}\, x_i \qquad N_K = \sum_{i=1}^{N} P_{iK}$$

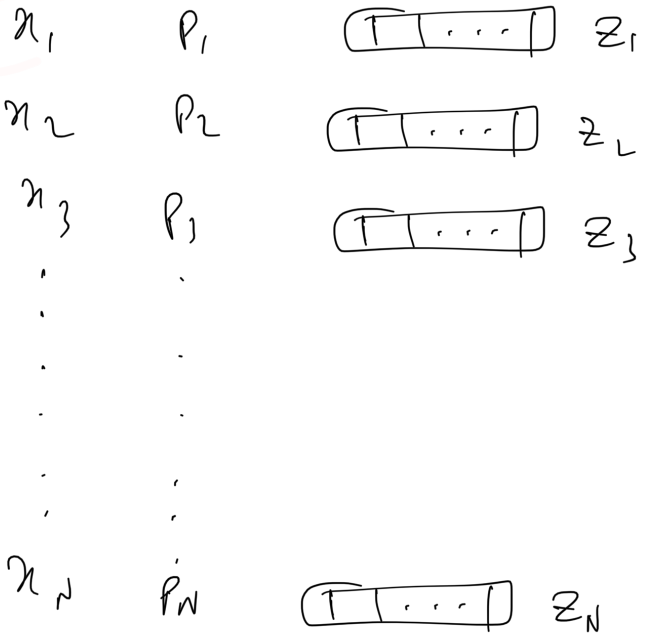
$$\Sigma_K = \frac{1}{N_K} \sum_{i=1}^{N} P_{iK} \times [x_i - \mu_K]^2$$

$$P_{iK} = \frac{\pi_K \, \mathcal{N}(x_i / \mu_K, \Sigma_K)}{\sum_{K'=1}^{K} \pi_{K'} \, \mathcal{N}(x_i / \mu_{K'}, \Sigma_{K'})}$$

↓ prob of x_i coming from k^{th} gaussian dist.

$$p(x/\theta) = \sum_{k=1}^{K} \pi_K \, \mathcal{N}(x / \mu_K, \Sigma_K)$$

↳ likelihood function for x coming from this entire mixture of gaussians



$$p(z_{ik} = 1 / \theta, X) = \text{posterior}$$

$$\ln p(x/\theta) = \ln \sum_z p(x, z/\theta)$$

$$p(x/\theta) = \sum_z p(x, z/\theta)$$

Can apply gradient descent but using E.M is advised as these type of equations (log summation over sub-components) do not have closed-form solutions

because we do not know the identity of the cluster of each x_i.
Same like in coin toss where we don't know the identity of coin A / coin B in E.M coin toss problem.

$\ln p(x, z/\theta)$ => Complete log likelihood

(observed variable, unobserved variable)

take expectation of this under posterior prob. distribution.

$$E_p \ln p(x, z/\theta)$$

log summation solution is difficult. So, do it in e-m steps & fill the unobserved variables, to form complete log likelihood function which is easier to solve. (using posterior probs.)

$$\frac{\partial}{\partial \mu_i}\left(\ln P(x/\theta)\right) = \frac{\partial}{\partial \mu_i}\left[\ln \sum_{k=1}^{K} \pi_k \mathcal{N}(x/\mu_k, \Sigma_k)\right]$$

Gradient Descent

$$= \frac{1}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x/\mu_k, \Sigma_k)} \times \pi_i \times \mathcal{N}(x/\mu_i, \Sigma_i) \times -\frac{2}{2} \times (x-\mu_i)^T \Sigma_i^{-1} (-1)$$

$\hookrightarrow p_i$

$$= p_i \times (x-\mu_i)^T \Sigma_i^{-1}$$

$$\ln P(x/\theta) = \ln \sum_{z} P(x, z/\theta)$$

$$\mathcal{L}(q, \theta) + KL(q \,||\, P(z/x, \theta))$$

$\hookrightarrow$ Lower Bound

Tries to match q with the posterior prob.

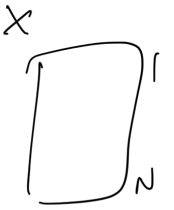
Then fixes q & tries to optimize θ.

$x_1, x_2, \ldots, x_N \Rightarrow x \sim \mathcal{N}(x|\mu, \Sigma)$

(x: D×1, μ: D×1, Σ: D×D)

Gaussian

Given N Data points, likelihood of dataset, X,



$$p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta) \qquad \theta = \mu, \Sigma$$

$$= \prod_{n=1}^{N} \mathcal{N}(x_n|\mu, \Sigma)$$

$$\text{log likelihood} = \log p(X|\theta) = \sum_{n=1}^{N} \log \frac{1}{(2\pi)^{D/2}} + \log \frac{1}{|\Sigma|^{1/2}} + \log e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

Derivative ⇒ then we get,

$$\mu = \frac{1}{N} \sum_{n=1}^{N} x_n \quad , \quad \Sigma = \frac{1}{N} \sum_{n=1}^{N} [x_n - \mu][x_n - \mu]^T$$

($[x_n-\mu]$: D×1, $[x_n-\mu]^T$: 1×D (outer product) ↪ D×D)

Multinomial

$x_1, x_2, \ldots, x_N$, $x \sim$ Multinomial $(x|\theta)$, x is K-dimensional

$$\sum_{i=1}^{K} \theta_K = 1 \quad \& \quad \theta_K \geq 0$$

Each x_i is a one-hot vector

$$p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$$

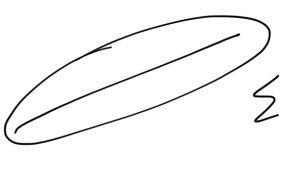
$$= \prod_{n=1}^{N} \prod_{K=1}^{K} \theta_K^{x_{nK}}$$

$$\log p(X|\theta) = \sum_{n=1}^{N} \sum_{K=1}^{K} x_{nK} \log \theta_K \quad \rightarrow \left[\left(\sum_{K=1}^{K} \theta_K \right) - 1 \right]$$

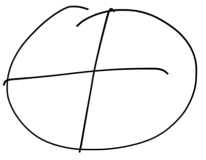
$$\theta_K = \frac{N_K}{N}$$

EM Mixture model

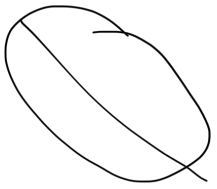
$x_1, x_2, \ldots, x_N, \quad x \sim P(x/\theta)$



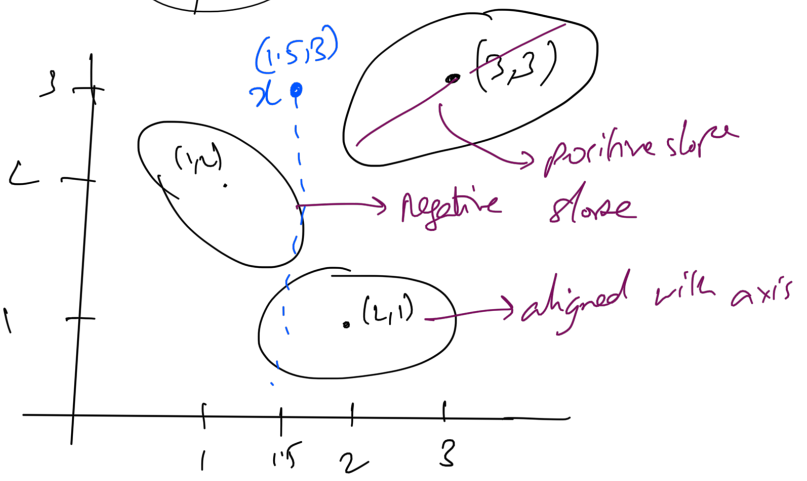
$\Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$



$\Rightarrow \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$



$\Sigma = \begin{bmatrix} 1 & -0.8 \\ 0.8 & 1 \end{bmatrix}$



Assume data is from these three gaussians.

$x_1, x_2, \ldots, x_N \Rightarrow x \sim P(x/\theta)$

These scores from each gaussian for each point x.

$\Rightarrow S_1, S_2, S_3$

Probs $\Rightarrow \dfrac{S_1}{S_1+S_2+S_3}, \dfrac{S_2}{S_1+S_2+S_3}, \dfrac{S_3}{S_1+S_2+S_3}$

$P_1 \quad P_2 \quad P_3$

↳ Prob. of x comes from GMM1.

$$\sum_{i=1}^{3} P_i = 1$$

→ We are able to calculate these probabilities as we know the parameters.

$x_1, x_2, \ldots, x_N \Rightarrow x \sim p(x|\theta)$

$z_1=2 \quad z_2=1 \qquad z_N=1$

↳ We don't have these tags.

→ If we atleast had these tags we would have calculated the parameters of each individual GMM.

So we use EM algo to solve this GMM problem.

$$x = \sum_{k=1}^{K} \pi_k \times \mathcal{N}(x \mid \mu_k, \Sigma_k)$$ → Mixture model likelihood

π_k ↓ weight of component k

$p(x|\theta)$ ↳ likelihood of a single data point.

$\theta: \pi, \mu_1, \mu_2, \ldots \mu_K, \Sigma_1, \Sigma_2, \ldots \Sigma_K$

$\sum_{i=1}^{K} \pi_k = 1$, & each $\pi_K > 0$

$$p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$$

No. of parameters $= K \times [D + D \times D] + K$

$$p(X|\theta) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n \mid \mu_k, \Sigma_k)$$

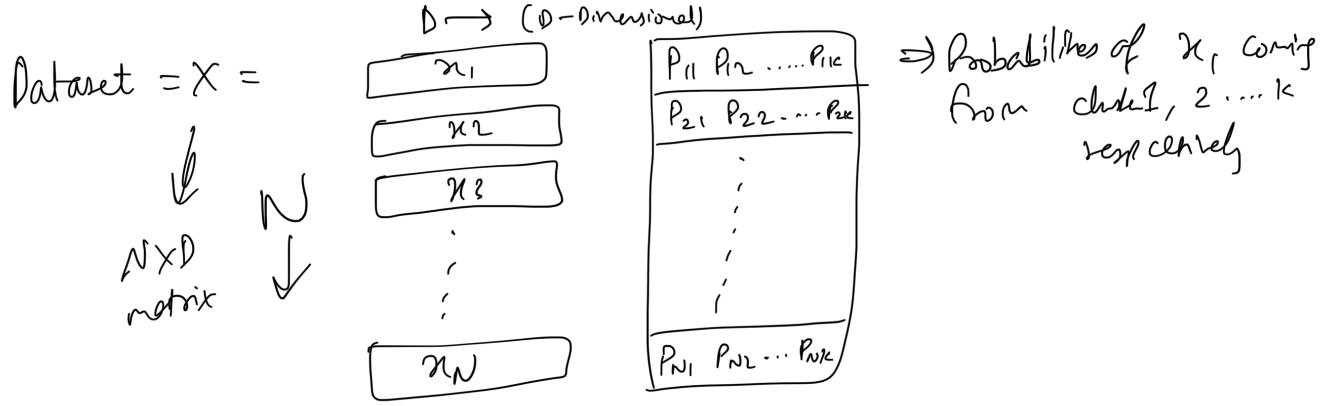
$$\log p(X|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n \mid \mu_k, \Sigma_k)$$

$$\frac{\partial \log p(x|\theta)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{\sum_{k'=1}^{K} \pi_{k'} \mathcal{N}(x_n|\mu_{k'}, \Sigma_{k'})} \times \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \times \frac{-1}{2}\left[x_n - \mu_k\right]^T \Sigma_k^{-1} (-1)$$

$\frac{\partial}{\partial \mu_k}[x_n - \mu_k]$

$\frac{\partial}{\partial x} x^T A x = 2Ax$ (or) $2x^T A$

$$= \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)}{\sum_{k'=1}^{K} \pi_{k'} \mathcal{N}(x_n|\mu_{k'}, \Sigma_{k'})} \times (x_n - \mu_k)^T \Sigma_k^{-1}$$



$$\frac{\partial \log p(x|\theta)}{\partial \mu_K} = \sum_{n=1}^{N} \underbrace{\frac{\pi_K \mathcal{N}(x_n|\mu_K, \Sigma_K)}{\sum_{K'=1}^{K} \pi_{K'} \mathcal{N}(x_n|\mu_{K'}, \Sigma_{K'})}}_{\hookrightarrow P_{nK}} \times (x_n - \mu_K)^T \Sigma_K^{-1}$$

$$0 = \sum_{n=1}^{N} P_{nK} \left[x_n - \mu_K\right]^T \Sigma_K^{-1}$$

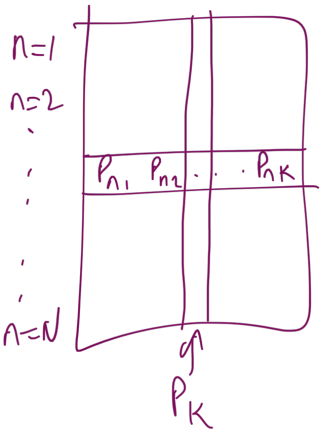
↳ neglected as it is common for every data point

$$0 = \sum_{n=1}^{N} P_{nK}\, x_n^T - \sum_{n=1}^{N} P_{nK}\, \mu_K^T$$

$$\Rightarrow \sum_{n=1}^{N} P_{nK}\, x_n^T = \sum_{n=1}^{N} P_{nK}\, \mu_K^T$$

$$\Rightarrow \sum_{n=1}^{N} P_{nK}\, x_n = \mu_K \sum_{n=1}^{N} P_{nK} \quad \text{(Transpose on both sides)}$$

$$\Rightarrow \mu_K = \frac{\sum_{n=1}^{N} P_{nK}\, x_n}{\sum_{n=1}^{N} P_{nK}}$$



let $N_K = \sum_{n=1}^{N} P_{nK}$

$$\mu_K = \frac{\sum_{n=1}^{N} P_{nK}\, x_n}{N_K}$$

$$\Sigma_K = \frac{1}{N_K} \sum_{n=1}^{N} P_{nK} \left[x_n - \mu_K\right]\left[x_n - \mu_K\right]^T$$

$$\frac{\partial \log p(x/\theta)}{\partial \pi_k} = \sum_{n=1}^{N} \log \frac{\mathcal{N}(x_n/\mu_k, \Sigma_k)}{\sum_{k'=1}^{K} \pi_{k'} \mathcal{N}(x_n/\mu_{k'}, \Sigma_{k'})} \times \frac{\pi_k}{\pi_k}$$

Multiply num. & denom by π_k

$$-\lambda\left(\sum_{i=1}^{K} \pi_i - 1\right)$$

$$\frac{\partial}{\partial \pi_k}(\pi_1 + \pi_2 + \dots + \pi_k + \dots + \pi_K) = 1$$

$$0 = \frac{1}{\pi_k} \sum_{n=1}^{N} p_{nk} - \lambda$$

$$\pi_k = \frac{\sum_{n=1}^{N} p_{nk}}{\lambda} \quad \& \quad \lambda = N$$

$$\therefore \boxed{\pi_k = \frac{N_k}{N}}$$

Mixture proportion

When we are estimating μ_k, we assume p_{nk} as, it is not a function involving μ_k; which isn't true.

So, we can use these formulas only if we fix μ_k & Σ_k. (like in the EM algo), else there is no closed-form solution.

$z \Rightarrow$ Latent variable $\Rightarrow$ indicate the component associated to x
↳ Not observed.

Dataset $\Rightarrow$ $x_1, x_2, \dots, x_N \Rightarrow$ observed data

$z_1, z_2, \dots, z_N \Rightarrow$ latent variable

MLE $\Rightarrow$ $\theta^* = \underset{\theta}{\operatorname{argmax}} \ln p(x/\theta)$

$z \Rightarrow$ K-dimension vector $\Rightarrow$ if $z_k = 1 \Rightarrow x$ is drawn from k^{th} gaussian component.

$p(z_k = 1/x, \theta) \Rightarrow$ Posterior probability of a data points comp = 1

↳ $= \frac{\pi_k \mathcal{N}(x/\mu_k, \Sigma_k)}{\sum_{k'=1}^{K} \pi_{k'} \mathcal{N}(x/\mu_{k'}, \Sigma_{k'})}$

$p(z|x,\theta) \Rightarrow$ posterior over latent variables.

$$p(z|x,\theta) = \prod_{n=1}^{N} p(z_n|x,\theta)$$

$$= \prod_{n=1}^{N} \prod_{k=1}^{K} p(z_{nk}|x,\theta)$$

$$\ln p(x|\theta) = \ln \sum_{z} p(x,z|\theta)$$ → Maximize this objective w.r.t θ.

But there is a $\sum_z$ inside log which does not lead to a closed form solution.

So we maximize $E[\ln p(x,z|\theta)]$, w.r.t. $p(z|x,\theta)$

$Q(\theta,\theta^{old})$ → $E[\ln p(x,z|\theta)]$ w.r.t $p(z|x,\theta^{old})$

① E-step consists of calculating $p(z|x,\theta^{old})$ for dataset X.

② M-step ⇒ $Q(\theta,\theta^{old}) = E_{p(z|x,\theta^{old})}[\ln p(x,z|\theta)]$

Find θ maximizing Q function

Instead of maximizing $\ln \sum_z p(x,z|\theta)$, i.e. $\ln p(x|\theta)$ we maximize $Q(\theta,\theta^{old})$

- $Q(\theta,\theta^{old})$ will keep the summation of all "z", but outside the logarithm.
- ↓ Easy to find if the components are known, i.e. summation is outside the log.

For GMM, (M-step)

$$\mu_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} p(z_{nk}=1|x_n,\theta^{old}) \times x_n$$

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} p(z_{nk}=1|x_n,\theta^{old}) \times [x_n-\mu_k][x_n-\mu_k]^T$$

$$\pi_k^{new} = \frac{N_k}{N} \quad , \quad N_k = \sum_{n=1}^{N} p(z_{nk} = 1 \mid x_n, \theta^{old})$$

$$p(x, z \mid \theta) = p(z \mid x, \theta) \times p(x \mid \theta) \quad \Rightarrow \text{Bayes rule}$$

$$\ln p(x, z \mid \theta) = \ln p(z \mid x, \theta) + \ln p(x \mid \theta)$$

$$\ln p(x \mid \theta) = \ln p(x, z \mid \theta) - \ln p(z \mid x, \theta)$$

$q(z) \Rightarrow$ arbitrary distribution over latent variables

$$z_n \rightarrow z_{n1}, z_{n2}, \ldots, z_{nk}$$
$$0.4 \quad 0.2 \quad 0.4$$

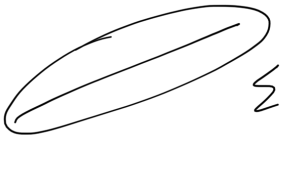
$$E_{q(z)}[\ln p(x \mid \theta)] = E_{q(z)}[\ln p(x, z \mid \theta)] - E_{q(z)}[\ln p(z \mid x, \theta)]$$

$$\ln p(x \mid \theta) = E\left[\ln_q \left[\frac{p(x, z \mid \theta)}{q(z)}\right]\right] + E_q\left[\ln \frac{q(z)}{p(z \mid x, \theta)}\right]$$

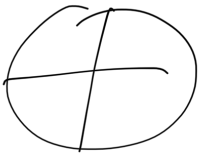
$$\ln(p(x \mid \theta)) = \mathcal{L}(q, \theta) + KL(q \| p)$$

$\mathcal{L}(q, \theta)$: lower bound

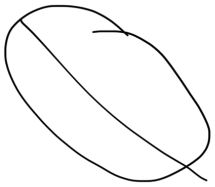
$KL(q \| p) \geq 0$ (always)



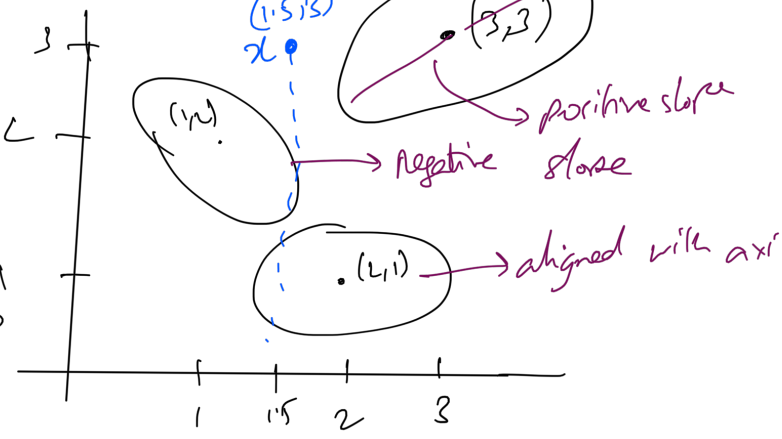
$$\Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$



$$\Rightarrow \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



$$\Sigma = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}$$



Taking a point $x = [1.5, 3]$, we get the probability density function as $[0.011, 0.015, 0.019]$ (Gaussian 1, Gaussian 2, Gaussian 3) & after normalization, $[0.25, 0.33, 0.41]$

$$\ln \sum_{k=1}^{K} \pi_k \times P(x / \mu_k, \Sigma_k)$$

Let $\pi = [1/2, 1/3, 1/3]$ ⇒ initial

$\ln \frac{1}{3}(0.011 + 0.015 + 0.019)$ ⇒ Original objective

↳ Not maximizable due to the Σ inside log

$P(x, z_1 = 1/\theta)$ $P(x, z_2 = 1/\theta)$ $P(x, z_3 = 1/\theta)$

$$E_{P(z/x,\theta}\left[\ln 0.011\right] \times 0.25 + \ln[0.015]\, 0.33 + \ln[0.019] \times 0.41$$

Represents joint dist. of observed & latent variable weighted by corresponding posterior probability

$$P(X, z(\theta) = P(Z / X, \theta) \times P(X / \theta)$$

$$\ln P(X, z/\theta) = \ln P(z/x, \theta) + \ln P(x/\theta)$$

$$\ln P(X/\theta) = \ln P(X, z/\theta) - \ln P(z/x, \theta)$$

↳ posterior

$q(z)$ ⇒ Arbitrary probability dist. over latent variable

$z_n \rightarrow z_{n1}, z_{n2}, \ldots, z_{nk}$

0.4 0.2 0.4

$$E_{q(z)}\left[\ln P(X/\theta)\right] = E_{q(z)}\left[\ln P(X,z/\theta)\right] - E_{q(z)}\left[\ln P(z/X,\theta)\right]$$

No, z in this term. So remains same.

$$\ln P(X/\theta) = E\left[\ln_q\left(\frac{P(X,z/\theta)}{q(z)}\right)\right] - E\left[\ln \frac{q(z)}{P(z/X,\theta)}\right]$$

The effect of this division, is that we have added entropy of $q(z)$ ($+H(q)$). So subtracting entropy from second term will keep it same.

$$\ln P(X/\theta) = \mathcal{L}(q,\theta) + KL(q\|p)$$

$\mathcal{L}(q,\theta)$: lower bound

$KL(q\|p) \geq 0$ always

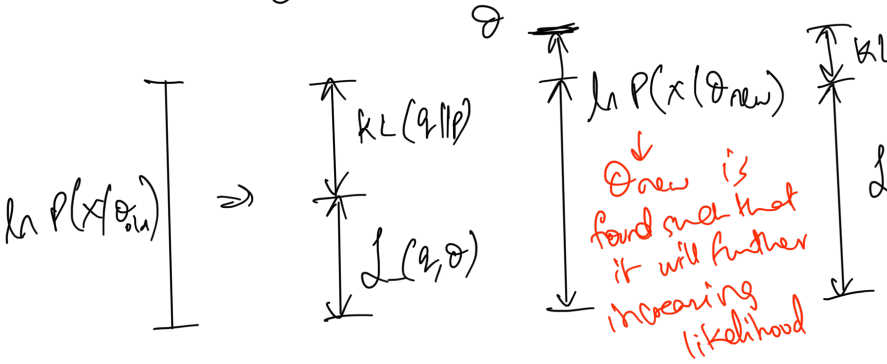
optimized w.r.t q in E-step & w.r.t. θ in M-step

→ When $KL(q\|p)$ is more, then $\ln P(X/\theta)$ & $\mathcal{L}(q,\theta)$ are more distant with respect to each other. & vice-versa.

So, when $KL(q\|p)$ is closer to zero, $\ln(P(X/\theta))$ & $\mathcal{L}(q,\theta)$ will have similar (close values, ie. a tight lower bound. ,i.e. it is exactly equals to LHS.

$$\theta' = \underset{\theta}{\operatorname{argmax}}\, \mathcal{L}(q,\theta)$$

$q \to P(z|X,\theta)$



If the improvement is not much, then stop the procedure

Mixture of Bernoulli Distribution

GMM ⇒ Mixtures of distributions over continuous variables

→ ⇒ mixtures of distributions over discrete binary variables.

↳ Latent class analysis.

Consider a set of D binary variables, $x_i \Rightarrow i = 1$ to D, with corresponding μ_i.

$x = (x_1, \ldots, x_D)^T$ & $\mu = (\mu_1, \ldots \mu_D)^T$

$$p(x/\mu) = \prod_{i=1}^{D} \mu_i^{x_i} (1-\mu_i)^{1-x_i}$$

$$E[x] = \mu$$

$$\text{cov}[x] = \text{diag}\{\mu_i(1-\mu_i)\}$$

Now, consider a finite mixture of these distributions given by,

$$p(x/\mu, \pi) = \sum_{k=1}^{K} \pi_k \, p(x/\mu_k), \quad \& \quad p(x/\mu_k) = \prod_{i=1}^{D} \mu_{ki}^{x_i} (1-\mu_{ki})^{(1-x_i)}$$

$$E[x] = \sum_{k=1}^{K} \pi_k \mu_k$$

$$\text{cov}[x] = \sum_{k=1}^{K} \pi_k \{\Sigma_k + \mu_k \mu_k^T\} - E[x] E[x]^T$$

↳ $= \text{diag}\{\mu_{ki}(1-\mu_{ki})\}$

$$\text{log likelihood} = \ln p(X/\mu, \pi) = \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \, p(x_n/\mu_k)$$

↓

Σ inside log. Use EM.

Latent variable z ⇒ one hot K-d vector.

$$p(x/z, \mu) = \prod_{k=1}^{K} p(x/\mu_k)^{z_k}$$

$$p(z/\pi) = \prod_{k=1}^{K} \pi_k^{z_k}$$

↳ prior probs.

$$\ln p(X, Z/\mu, \pi) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left\{ \ln \pi_k + \sum_{i=1}^{D} x_{ni} \ln \mu_{ki} + (1-x_{ni}) \ln(1-\mu_{ki}) \right\}$$

$$E_z[\ln p(X, z \mid \mu, \pi)] = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk}) \left\{ \ln \pi_k + \sum_{i=1}^{D} x_{ni} \ln \mu_{ki} + (1 - x_{ni}) \ln (1 - \mu_{ki}) \right\}$$

$$, \gamma_{nk} = E[z_{nk}] = \sum_{z_{nk}} z_{nk} \left[\pi_k p(x_n \mid \mu_k) \right]^{z_{nk}}$$

$$\frac{}{\sum_{z_j} \left[\pi_j p(x_n \mid \mu_j) \right]^{z_{nj}}}$$

$$= \frac{\pi_k p(x_n \mid \mu_k)}{\sum_{j=1}^{K} \pi_j p(x_n \mid \mu_i)}$$

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

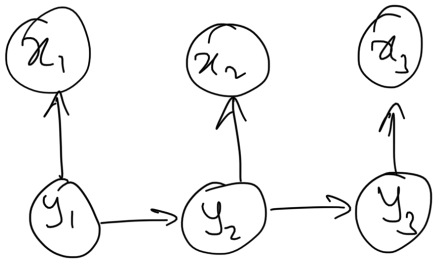
$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n$$

$$\pi_k = \frac{N_k}{N}$$

Hidden Markov Model

⇒ Used for seq2seq classification

like NER tagging, POS tagging



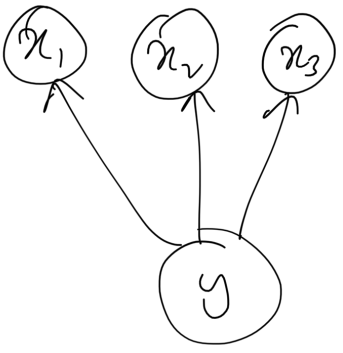
$$\Rightarrow P(x, y) = P(y_1) \times P(x_1/y_1) \times \prod_{i=2}^{T} P(x_i/y_i) \times P(y_i/y_{i-1})$$

(y_i/y_0)

$$= \prod_{i=1}^{T} P(x_i/y_i) \times P(y_i/y_{i-1})$$

↳ Size of model $= L \times K + L \times L$

In naive Bayes ⇒

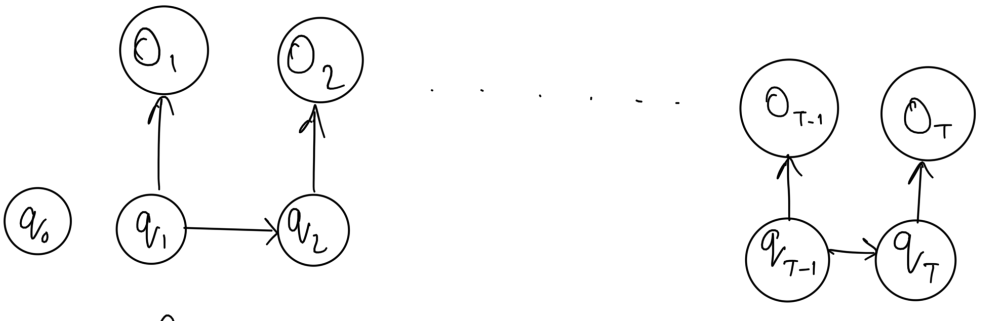


$$\Rightarrow P(x, y) = \frac{P(y) \times P(x/y)}{\prod_{i=1}^{K} P(x_i/y)}$$

↓

Model Size ⇒ $L \times K + L$

A tutorial of HMM by Rabiner



$P(q_1 = 1/q_0) = \pi_1$

$P(q_1 = 2/q_0) = \pi_2$

$\pi = [\pi_1, \pi_2, \ldots, \pi_N]$

↓ initial probabilities

$q_t = 1 \ldots N$

↳ N states

↳ represents state at time "t"

$P(O, q/\lambda)$ ⇒ To find

↳ $\lambda = \{A, B, \pi\}$

$N \Rightarrow$ no. of distinct states

M = no. of distinct observation symbols

$$A = N\left[\quad\right], \quad B = N\left[\quad\right]$$

$A \to p(i/j)$

$B \to p(i/j)$

$$P(O/\lambda) = \sum_{q \in Q} P(O, q/\lambda)$$

$Q \to$ No. of possible state sequences $= N^T$

T observations, with each having N possible states.

$\Rightarrow$ So computationally expensive

Brute force $\Rightarrow$ Compute prob. of all state sequences

Joint probability $\Rightarrow$ $P(O, q/\lambda) = \pi[q_0] \times \prod_{t=1}^{T} P(q_t/q_{t-1}) \times P(O_t/q_t)$

Forward probability

$$\alpha_t(i) = P(O_1, O_2, \ldots, O_t, q_t = i/\lambda)$$

Initialization $\Rightarrow$ $\alpha_1(i) = P(O_1, q_1 = i/\lambda)$

$$\alpha_1(i) = \pi_i \times B_i(O_1)$$

$B_i(O_1) \to i^{th}$ row & O_1^{th} column of B.

prob. of being in state i initially & observing O_1.

$\alpha_t(1)$ ○

$\alpha_t(2)$ ○

⋮

$\alpha_t(N)$ ○

(i) $\alpha_{t+1}(i)$

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j) \times A_{ji}\right] \times B_i(O_{t+1})$$

Computational effort $= TN^2$

$$P(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i) \quad , \quad \alpha_T(i) = P(O_1, O_2, \ldots, O_T, q_T = i/\lambda)$$

Backward Probability

$$\beta_t(i) = P(O_{t+1}, \ldots, O_T / q_t = i, \lambda)$$

$$\beta_T(i) = 1$$

$$\beta_t(i) = \sum_{j=1}^{N} A_{ij} \times \beta_{t+1}(j) B_j(O_{t+1})$$

$$\alpha_t(i) = P(O_1, O_2, \ldots, O_t, q_t = i/\lambda)$$

$$\beta_t(i) = P(O_{t+1}, \ldots, O_T / q_t = i, \lambda)$$

$$P(O_1, O_2, \ldots, O_T, q_t = i/\lambda) = \alpha_t(i) \times \beta_t(i)$$

$$P(q_t = i/O, \lambda) = \frac{\alpha_t(i) \times \beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j) \times \beta_t(j)} = \gamma_t(i)$$

The most probable state sequence given the observed sequences.

Viterbi Algorithm → For most probable state sequence
↳ Solves inference problem.

EM Algorithm → To train, to find parameters.

$$\delta_1(i) = \pi_i \times B_i(O_1)$$

$$\delta_{t+1}(j) = \max_{i=1\dots N} \delta_t(i) \times A_{ij} \times B_j(O_{t+1})$$

$$\psi_{t+1}(j) = \operatorname{argmax} \delta_t(i) A_{ij} \times B_j(O_{t+1})$$

HMM not class

Notations ⇒ time instants associated with state changes, $t = 1, 2, 3, \ldots$

State at $t = t \Rightarrow q_t$

$a_{ij} = P(q_t = S_j \,/\, q_{t-1} = S_i)$ ⇒ transition probability from state S_i at q_{t-1} to S_j at q_t.

N = no. of states

$a_{ij} \geq 0$ & $\sum_{j=1}^{N} a_{ij} = 1$

A = transition prob. matrix, Size = $N \times N$

Π ⇒ initial state probability vector, size = $N \times 1$

$\Pi_i = P(q_1 = S_i)$ ← say S_i

⇒ Given that the model is in a known state, what is the probability that it stays in that same state for exactly "d" days?

$O = \{S_i, S_i, S_i, \ldots, S_i, S_j \neq S_i\}$
(1, 2, 3, ..., d, d+1)

$P(O/\text{Model}) = \Pi_i \times (a_{ii})^{d-1} \times (1 - a_{ii}) = p_i(d)$

↳ 1 (as it is given that at $t=1 \Rightarrow q_t = S_i$)

⇒ discrete probability density function of duration d in state i.

Based on $p_i(d)$, we can calculate the expected no. of observations (duration) in a state, conditioned on starting in that state as,

$$\bar{d}_i = \sum_{d=1}^{\infty} d\, p_i(d)$$

$$= \sum_{d=1}^{\infty} d\,(a_{ii})^{d-1}(1-a_{ii}) = \frac{1}{1-a_{ii}}$$

∴ Expected no. of consecutive days in S_i given start is $S_i = \frac{1}{1-a_{ii}}$

→ **Elements of a HMM**

① $N \Rightarrow$ no. of states in the model.

$S = \{S_1, S_2, \ldots, S_N\}$ & state at time $= t \Rightarrow q_t$

(N can be the vocabulary size)

② $M \Rightarrow$ no. of distinct observation symbols per state.

Correspond to the physical o/p of the system being modeled.

$V = \{V_1, V_2, \ldots, V_M\}$ (M can be the no. of NER tags)

③ State transition probability dist. $A = \{a_{ij}\}$ of size $N \times N$.

$$a_{ij} = P[q_t = S_j \mid q_{t-1} = S_i]$$

④ Observation symbol prob. dist. in state j, $B = \{b_j(k)\}$ of size $N \times M$. (also called emission prob from hidden to o/p)

$$b_j(k) = P[V_k \text{ at } t \mid q_t = S_j]$$

⑤ Initial state distribution, $\pi = \{\pi_i\}$ of size $N \times 1$

$$\pi_i = P(q_1 = S_i)$$

→ Given appropriate values of N, M, A, B, π, HMM can be used as a generator to give an observation sequence

$$O = O_1, O_2 \ldots\ldots O_T,$$

where each O_t is a symbol from V, & T is the # of observations in the sequence.

Steps for generating

① Choose initial state $q_1 = S_i$ according to π.

② Set $t = 1$

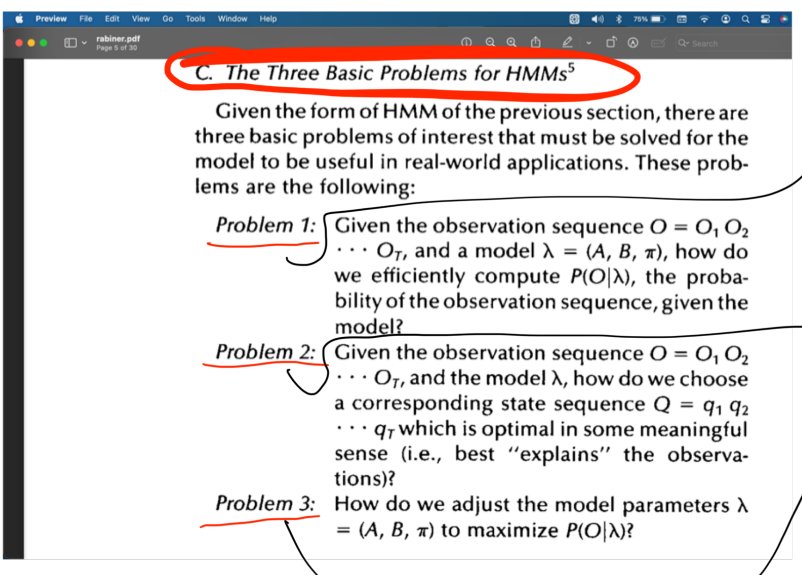
③ Choose $O_t = v_k$ acc. to symbol prob. dist in state q_i ⇒ $b_i(k)$

④ Transit to new state $q_{t+1} = S_j$ according to state prob. dist for state S_i ⇒ a_{ij}

⑤ Set $t = t+1$ & return to step 3 if $t \leq T$.
else terminate.

$$\lambda = [A, B, \pi]$$

→ parameter set of the model

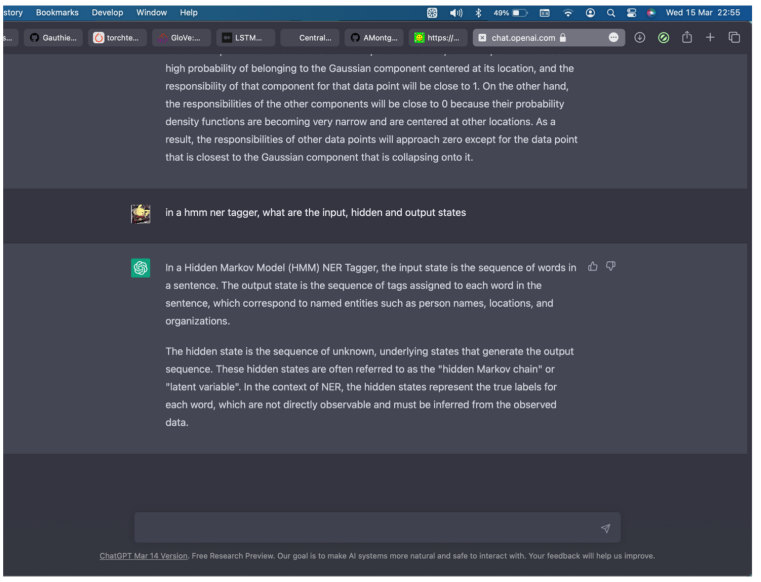


→ Evaluation problem, ie. given a model & a sequence, how likely is the sequence generated by the model.
(OR)
Scoring how well a given model suits a given obs. seq.

→ Uncovering the hidden part of the model. ie. to find correct state sequence.

→ Optimize the model parameters to best describe how a given obs. seq comes about.

To fix ideas, consider the following simple isolated word speech recognizer. For each word of a W word vocabulary, we want to design a separate N-state HMM. We represent the speech signal of a given word as a time sequence of coded spectral vectors. We assume that the coding is done using a spectral codebook with M unique spectral vectors; hence each observation is the index of the spectral vector closest (in some spectral sense) to the original speech signal. Thus, for each vocabulary word, we have a training sequence consisting of a number of repetitions of sequences of codebook indices of the word (by one or more talkers). The first task is to build individual word models. This task is done by using the solution to Problem 3 to optimally estimate model parameters for each word model. To develop an understanding of the physical meaning of the model states, we use the solution to Problem 2 to segment each of the word training sequences into states, and then study the properties of the spectral vectors that lead to the observations occurring in each state. The goal here would be to make refinements on the model (e.g., more states, different codebook size, etc.) so as to improve its capability of modeling the spoken word sequences. Finally, once the set of W HMMs has been designed and optimized and thoroughly studied, recognition of an unknown word is performed using the solution to Problem 1 to score each word model based upon the given test observation sequence, and select the word whose model score is highest (i.e., the highest likelihood).



Solution to Problem 1 $P(O|\lambda)$

Brute force ⇒ Enumerating every possible state sequence of length T (no. of observations)

$Q = q_1 q_2 \dots q_T$ ⇒ If there are "N" states, no. of such sequences $= N^T$

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda)$$

$O = O_1 O_2 \dots O_T$

we assumed statistical independence of observations.

$$P(O|Q,\lambda) = b_{q_1}(O_1) \times b_{q_2}(O_2) \times \dots \times b_{q_T}(O_T)$$

↓

emission probability of token represented by O_1 at time $t=1$ from hidden state q_1.

$$P(Q|\lambda) = \pi_{q_1} \times a_{q_1 q_2} \times a_{q_2 q_3} \times \dots \times a_{q_{T-1} q_T}$$

Joint probability of O & Q, ie. prob that O & Q occur simultaneously $\Rightarrow P(O, Q/\lambda) = P(O/Q, \lambda) \times P(Q/\lambda)$

The probability of O given λ, is obtained by summing the joint prob over all possible state sequences q giving;

$$P(O/\lambda) = \sum_{\text{all } Q} P(O/Q,\lambda) \times P(\lambda)$$

$$= \sum_{q_1 q_2 \cdots q_T} \pi_{q_1} \cdot b_{q_1}(O_1) \times a_{q_1 q_2} b_{q_2}(O_2) \times \cdots \cdots - a_{q_{T-1} q_T} b_{q_T}(O_T)$$

Interpretation of above computation & computational Complexity

The interpretation of the computation in the above equation is the following. Initially (at time $t = 1$) we are in state q_1 with probability π_{q_1}, and generate the symbol O_1 (in this state) with probability $b_{q_1}(O_1)$. The clock changes from time t to $t + 1$ ($t = 2$) and we make a transition to state q_2 from state q_1 with probability $a_{q_1 q_2}$, and generate symbol O_2 with probability $b_{q_2}(O_2)$. This process continues in this manner until we make the list transition (at time T) from state q_{T-1} to state q_T with probability $a_{q_{T-1} q_T}$ and generate symbol O_T with probability $b_{q_T}(O_T)$.

A little thought should convince the reader that the calculation of $P(O|\lambda)$, according to its direct definition (17) involves on the order of $2T \cdot N^T$ calculations, since at every $t = 1, 2, \cdots, T$, there are N possible states which can be reached (i.e., there are N^T possible state sequences), and for each such state sequence about $2T$ calculations are required for each term in the sum of (17). (To be precise, we need $(2T - 1)N^T$ multiplications, and $N^T - 1$ additions.) This calculation is computationally unfeasible, even for small values of N and T; e.g., for $N = 5$ (states), $T = 100$ (observations), there are on the order of $2 \cdot 100 \cdot 5^{100} \approx 10^{72}$ computations

262

and observations are the input words as they are observed

Picture a HMM like neural network, where hidden states are analogous to neurons. Each input goes to a specific hidden state from which a specific output is generated. Hidden states are usually the outputs, like NER & POS tags. While neurons are just hidden computational units.

Efficient solution ⇒ Forward-Backward Procedure

→ to model $P(O/\lambda)$

$$\alpha_t(i) = P(O_1 O_2 \dots O_t, q_t = S_i / \lambda)$$

probability of the partial observation sequence $O_1 O_2 \dots O_t$ (until time t) & state S_i at time time t.

→ $P(O, Q/\lambda) = P(O/Q,\lambda) \times P(Q/\lambda)$

① Initialization

$$\alpha_1(i) = \pi_i \, b_i(O_1) \quad \text{for all } i, \text{ such that } 1 \le i \le N$$

$\pi_i = P(q_1 = S_i)$ → $P(O/Q,\lambda)$

This step initializes the forward probabilities as the joint probabilities of state S_i & initial observation O_1.

② Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) \times a_{ij} \right] \times b_j(O_{t+1}) \quad , \quad 1 \le t \le T-1 \;,\; 1 \le j \le N$$

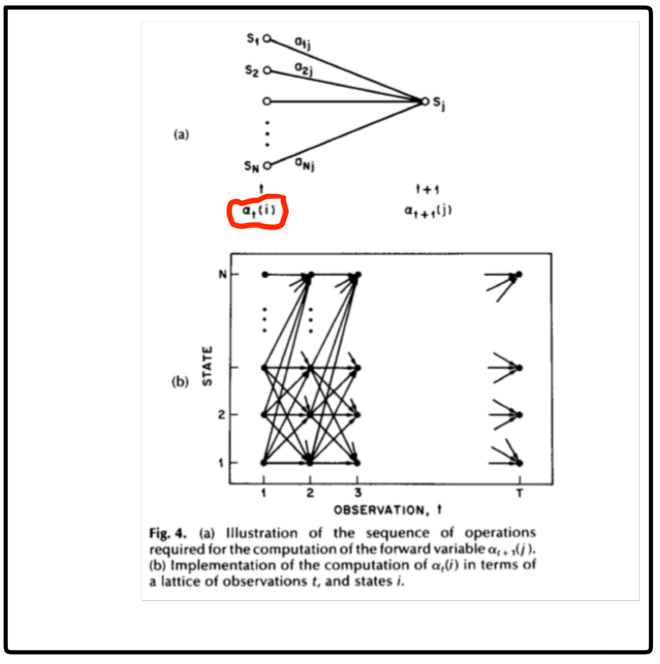


Fig. 4. (a) Illustration of the sequence of operations required for the computation of the forward variable $\alpha_{t+1}(j)$. (b) Implementation of the computation of $\alpha_t(i)$ in terms of a lattice of observations t, and states i.

* we only need $\alpha_T(i)$ for all possible i to solve problem 1, but $\beta_t(i)$ is introduced next, just because it is used in solving problem 2.

- Figure shows how state S_j can be reached at time $t+1$ from the N possible states S_i, $1 \le i \le N$ at time t.
- As $\alpha_t(i)$ is the prob. of joint event that $O_1 O_2 \dots O_t$ are observed and the state at time t is S_i, the product $\alpha_t(i) \times a_{ij}$ is then the probability of the joint event that $O_1 O_2 \dots O_t$ are observed & state S_j is reached at time $t+1$ via state S_i at time t.
- Summing this product over all "N" possible states, results in the prob. of S_j at time $t+1$ with all accompanying previous partial observations.

- Once this is done, $\alpha_{t+1}(j)$ is obtained by accounting for O_{t+1} in state j, i.e. $b_j(O_{t+1})$.
- The computation of $\alpha_{t+1}(j)$ is done for all states j, $1 \le j \le N$; for a given time t.

③ Termination

$$P(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

- Step 3 gives $P(O/\lambda)$, because by definition,

$$\alpha_T(i) = P(O_1 O_2 \dots O_T, q_T = S_i / \lambda)$$

→ Computational Complexity $\Rightarrow N^2 T$

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) \times a_{ij}\right] \times b_j(O_{t+1})$$

N multiplications + 1 multiplication

$j = 1$ to N & $t = 1$ to $T-1$

Constitutes $N(T-1)(N+1)$ Multiplications

& $\alpha_1(i) = \pi_i \times b_i(O_1)$ for all i, $1 \le i \le N$

↳ N multiplications

$\oplus$ = $N(N+1)(T-1) + N$ multiplications

If we examine the computation involved in the calculation of $\alpha_t(j)$, $1 \le t \le T$, $1 \le j \le N$, we see that it requires on the order of N^2T calculations, rather than $2TN^T$ as required by the direct calculation. (Again, to be precise, we need $N(N+1)(T-1) + N$ multiplications and $N(N-1)(T-1)$ additions.) For $N = 5$, $T = 100$, we need about 3000 computations for the forward method, versus 10^{72} computations for the direct calculation, a savings of about 69 orders of magnitude.

→ The forward probability calculation is, in effect, based upon the lattice (or trellis) structure shown in Fig. 4(b). The key is that since there are only N states (nodes at each time slot in the lattice), all the possible state sequences will remerge into these N nodes, no matter how long the observation sequence. At time $t = 1$ (the first time slot in the lattice), we need to calculate values of $\alpha_1(i)$, $1 \le i \le N$. At times $t = 2, 3, \dots, T$, we only need to calculate values of $\alpha_t(j)$, $1 \le j \le N$, where each calculation involves only N previous values of $\alpha_{t-1}(i)$ because each of the N grid points is reached

→ In the same manner, consider a backward variable $\beta_t(i)$ defined as

$$\beta_t(i) = P(O_{t+1}\ O_{t+2}\ \ldots\ldots\ O_T / q_t = S_i / \lambda)$$

↓

Probability of the partial observation sequence from $t+1$ to the end, given state S_i at time t & model λ.

① Initialization

$$\beta_T(i) = 1, \quad 1 \le i \le N$$

② Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)$$

, $t = T-1, T-2, \ldots, 1$ & $1 \le i \le N$

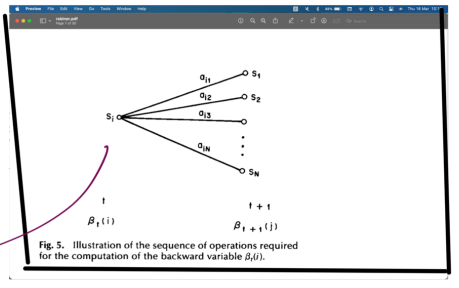


Fig. 5. Illustration of the sequence of operations required for the computation of the backward variable $\beta_t(i)$.

In order to have been at state S_i at time t, and to account for the observation sequence from time $t+1$ on, you have to consider all possible states S_j at time $t+1$, accounting for the transition from S_i to S_j as well as the observation O_{t+1} in state j & then account for the remaining partial observation from state j.

Computational Complexity ⇒ $N^2 T$

Solution to Problem 2

→ $P(Q/O,\lambda)$

- Problem 1 has an exact solution which can be given. But there are several possible ways to find the "optimal" state sequence. The difficulty lies with the definition of the optimal state sequence.

One such optimality criterion (→ a principle (or) standard by which something is judged) is to choose the states q_t which are individually most likely.

↳ This optimality criterion maximizes the expected no. of correct individual states.

To implement this solution, we define the variable,

$$\gamma_t(i) = P(q_t = S_i / O, \lambda)$$

→ probability of being in state S_i at time t, given the observation sequence O & model λ.

$$\gamma_t(i) = \frac{\alpha_t(i) \times \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \times \beta_t(i)} = \frac{P(O_1, O_2, \ldots, O_T, q_t = S_i/\lambda)}{P(O/\lambda)} =$$

→ accounts for partial observation sequence $O_1, O_2 \ldots O_t$ & state S_i at t

→ accounts for partial obs. seq. $O_{t+1}, \ldots, O_T$ given state S_i at t.

→ Normalization factor makes $\gamma_t(i)$ a probability measure such that $\sum_{i=1}^{N} \gamma_t(i) = 1$

- Using $\gamma_t(i)$, we solve for the individually most likely state q_t at time t, as,

$$q_t = \underset{1 \le i \le N}{\operatorname{argmax}} \left[\gamma_t(i)\right], \text{ for all } t,\ 1 \le t \le T$$

- Although the above optimality criterion maximizes the expected no. of correct states (by choosing the most likely state for each t), there could be some problems with the resulting state sequence.
- This is when some state transitions may have zero probability, i.e. $a_{ij} = 0$ for some $i \neq j$.
- So the "optimal" state sequence may not even be a valid state sequence, as the optimality criterion above just choose the most likely state at every instant, without regard to the probability of occurrence of sequences of states.

- So, we need to modify the criterion.
For example, we could solve for the state sequence that maximizes the expected number of correct pairs of states (q_t, q_{t+1}) or triples of states (q_t, q_{t+1}, q_{t+2}), etc.
- These criteria might be reasonable for some applications, but the most widely used criterion is to find the single best state sequence (path), i.e. to maximize $P(Q/O, \lambda)$ which is equivalent to maximizing $P(Q, O/\lambda)$.
↳ Formal technique for this is the Viterbi Algorithm based on Dynamic Programming.

Viterbi Algorithm → To find the single best sequence, $Q = \{q_1 q_2 \dots q_T\}$ for the given observation sequence $O = \{O_1 O_2 \dots O_T\}$, we define,

$$\delta_t(i) = \max_{q_1 q_2 \dots q_{t-1}} P(q_1 q_2 \dots q_t = i, O_1 O_2 \dots O_t / \lambda)$$

$\delta_t(i)$: the best score (highest prob.) along a single path, at time t, which accounts for first "t" observation & ends at state S_i.

By induction we have, $\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij}\right] \times b_j(O_{t+1})$

To actually retrieve the state sequence, we need to keep track of the argument which maximized for each t & j. Store there in the array $\psi_t(j)$

① Initialization

$$\delta_1(i) = \pi_i \, b_i(O_1), \quad 1 \le i \le N$$

$$\psi_1(i) = 0$$

② Recursion

At each instant of time, for each hidden state, calculate prob till then. i.e. you are keeping track of all hidden states possible from where S_j is reached at time "t-1" at time "t".

$$\delta_t(j) = \max_{1 \le i \le N} \left[\delta_{t-1}(i) \, a_{ij}\right] b_j(O_t), \quad 2 \le t \le T \;\&\; 1 \le j \le N$$

(max value)

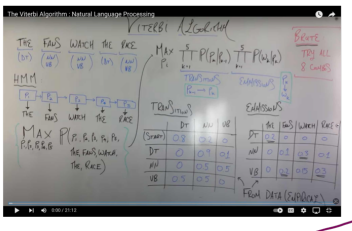
$$\psi_t(j) = \operatorname*{argmax}_{1 \le i \le N} \left[\delta_{t-1}(i) \, a_{ij}\right], \quad 2 \le t \le T \;\&\; 1 \le j \le N$$

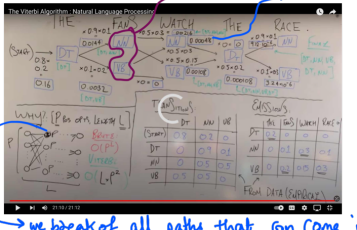
(index which gives max value; chi)

③ Termination

$$P^* = \max_{1 \le i \le N} \left[\delta_T(i)\right]$$

$$q_T^* = \operatorname*{argmax}_{1 \le i \le N} \left[\delta_T(i)\right]$$





keeps track of all hidden states possible at time "t-1".

cut this path

we break of all paths that can come into P and consider only the path that has max. prob. till this step.

④ Path (state seq.) backtracking

$$q_t^* = \psi_{t+1}\left[q_{t+1}^*\right], \quad t = T-1, T-2, \ldots, 1$$

→ Similar to forward prob. calculation. Major difference is max over summation in the forward prob. calculation.

Computational complexity of Viterbi ⇒ $O(N^2 T)$

Solution to Problem 3

- Most difficult problem of HMM's is to determine a method to adjust the model parameters (A, B, π) to maximize the prob. of observation sequence given to the model.

  Global optimum ⇒ ? ⇒ No optimal way

  local optimum ⇒ Yes ⇒ Baum – Welch method based on EM.

- In order to describe the procedure for re-estimation i.e. iterative update & improvement of HMM parameters, we first define $\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j / O, \lambda)$

The seq. of events leading to the conditions here one illustrated below.

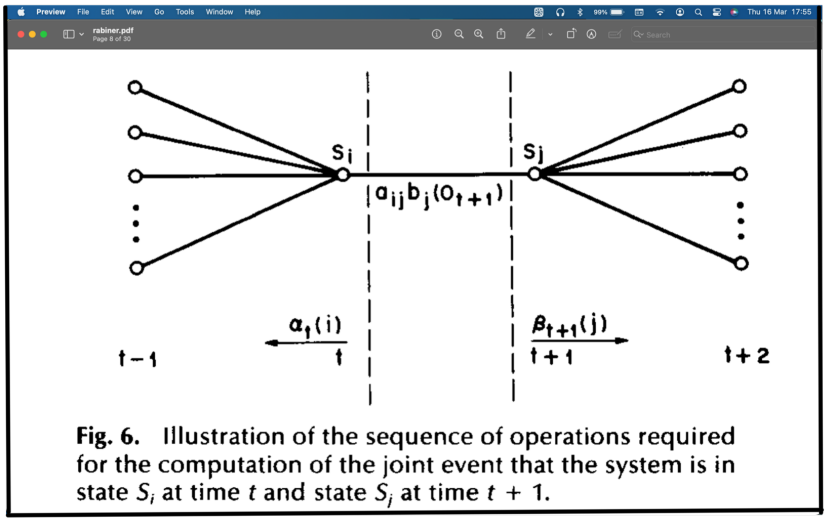


Fig. 6. Illustration of the sequence of operations required for the computation of the joint event that the system is in state S_i at time t and state S_j at time $t + 1$.

- From the definition of forward & backward probs, i.e. $\alpha_t(i)$ & $\beta_t(i)$, we can express $\xi_t(i, j)$ as,

$$\xi_t(i, j) = \frac{\alpha_t(i) \times a_{ij} \times b_j(O_{t+1}) \times \beta_{t+1}(j)}{P(O/\lambda)}$$

(psi)

$$= \frac{\alpha_t(i) \times a_{ij} \times b_j(O_{t+1}) \times \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) \times a_{ij} \times b_j(O_{t+1}) \times \beta_{t+1}(j)}$$

⇒ $P(q_t = S_i, q_{t+1} = S_j, O/\lambda)$

↳ $P(O/\lambda)$ ⇒ can also be accounted by $\sum_{i=1}^{N} \alpha_t(i)$

$\gamma_t(i)$ = Prob. of being in State S_i at time t, given O & λ.

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

- If we sum over $\gamma_t(i)$, we get a quantity which can be interpreted as the expected no. of times that State S_i is visited, or equivalently, the expected no. of transitions made from state S_i, if we exclude $t=T$ from the summation.

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected no. of transition from } S_i$$

- Similarly summation over $\xi_t(i,j)$ over t, gives expected no. of transitions from State S_i to State S_j.

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{expected no. of transitions from } S_i \text{ to } S_j$$

Reestimation formulas

$\bar{\pi}_i$ = expected frequency in state S_i at time $t=1$ = $\gamma_1(i)$

$$\bar{a}_{ij} = \frac{\text{expected no. of transitions from } S_i \text{ to } S_j}{\text{Total expected no. of transitions from } S_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\bar{b}_j(k) = \frac{\text{expected no. of times in state } S_j \text{ \& observing } V_k}{\text{Total expected no. of times in state } S_j}$$

$$= \frac{\sum_{\substack{t=1 \\ \text{such that} \\ O_t = V_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

- It has been proven that, $\bar{\lambda} = \{\bar{A}, \bar{B}, \bar{\pi}\}$ is a model such that,

$$P(O/\bar{\lambda}) \geq P(O/\lambda)$$

Equal only if $\bar{\lambda} = \lambda$, i.e λ is a critical point of the likelihood function.

→ Obs. seq. is more likely to have been produced from model $\bar{\lambda}$; wha compared to λ.

- Repeatedly doing this will eventually end in a maximum likelihood estimate of the HMM. (local maxima only)

$$Q(\lambda, \bar{\lambda}) = \sum_{Q} P(Q/O, \lambda) \log [P(O, Q/\bar{\lambda})]$$

$$\max_{\bar{\lambda}} [Q(\lambda, \bar{\lambda})] \Rightarrow P(O/\bar{\lambda}) \geq P(O/\lambda)$$

→ Important point to note is that, at every update

$$\sum_{i=1}^{N} \bar{\pi}_i = 1 \quad \& \quad \sum_{i=1}^{N} \bar{a}_{ij} = 1 \quad \& \quad \sum_{k=1}^{M} \bar{b}_i(k) = 1$$

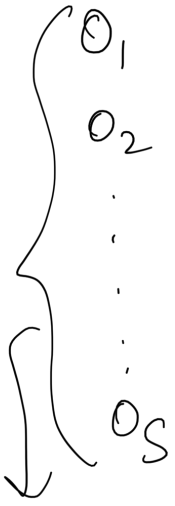
This can be achieved using lagrange multipliers.

By using lagrange multipliers, it is shown that P is maximized when,

$$\pi_i = \frac{\pi_i \frac{\partial P}{\partial \pi_i}}{\sum_{k=1}^{N} \pi_k \frac{\partial P}{\partial \pi_k}} \quad , \quad a_{ij} = \frac{a_{ij} \frac{\partial P}{\partial a_{ij}}}{\sum_{k=1}^{N} a_{ik} \frac{\partial P}{\partial a_{ik}}}$$

$$\& \quad b_i(k) = \frac{b_i(k) \frac{\partial P}{\partial b_i(k)}}{\sum_{l=1}^{M} b_i(l) \frac{\partial P}{\partial b_i(l)}}$$

Dataset



"S" no. of sequences, each of length "T"

O_s = Sentence / Sequence "s"

$$P(O, q/\lambda) = \prod_{t=1}^{T} P(O_t/q_t) \times P(q_t/q_{t-1})$$

→ parameter come from matrix B

→ parameter comes from matrix A & vector π

$$= \pi_{q_1} \times B_{q_1}(O_1) \times \prod_{t=2}^{T} P(O_t/q_t) \times P(q_t/q_{t-1})$$

$$P(O/\lambda) = \sum_{q \in Q} P(O, q/\lambda) = \sum_{q \in Q} P(O/q, \lambda) \times P(q/\lambda)$$

4_{N^T} possibilities

$$\sum_{s=1}^{S} \log P(O_s/\lambda) = \log \sum_{q \in Q} P(O_s, q/\lambda)$$

Estep $\Rightarrow$ $P(q/O, \lambda^t)$

Mstep $\Rightarrow$ $E_{P(q/O,\lambda)}[\log P(O, q/\lambda)]$

Expectation

using posterior dist. to maximize $\log P(O, q/\lambda)$

$$\log P(O, q/\lambda) = \log \pi_{q_1} + \left[\prod_{t=1}^{T} P(O_t/q_t)\right]\left[\prod_{t=2}^{T} P(q_t/q_{t-1})\right]$$

$\log \pi_{q_1}$ → A, $\left[\prod_{t=1}^{T} P(O_t/q_t)\right]$ → B

$$E_{P(q/O,\lambda^{old})}[\log \pi_{q_1}] + \sum_{t=1}^{T} E_{P(q/O,\lambda^{old})}[\log P(O_t/q_t)] + \sum_{t=2}^{T} E_{P(q/O,\lambda^{old})}[\log P(q_t/q_{t-1})]$$

$E_{P(q/O,\lambda^{old})}[\log P(O_t/q_t)] = \gamma_1(i)$

$$\sum_{i=1}^{N}[\log \pi_i] P(q_1=i/O,\lambda^{old}) - g\left(\sum_{i=1}^{N} \pi_i - 1\right)$$

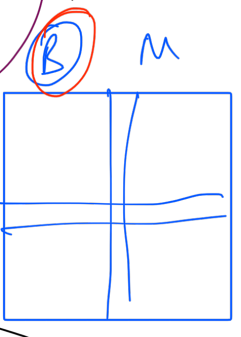
g: lagrangian multiplier

$\frac{\partial}{\partial \pi_i}$

$$\frac{\gamma_1(i)}{\pi_i} - g = 0 \Rightarrow \pi_i = \frac{\gamma_1(i)}{g} \quad \& \quad g = \sum_{i=1}^{N} \gamma_1(i)$$

$$\boxed{\pi_i = \frac{\gamma_1(i)}{\sum_{i=1}^{N} \gamma_1(i)}}$$

$B_{ij} = P(O=j/q=i)$



Each row is a probability vector

$$E_{P(q/O,\lambda^{old})}\left[\sum_{t=1}^{T} \log P(O_t/q_t)\right] - \sum_{i=1}^{N} h_i\left(\sum_{j=1}^{M} B_{ij} - 1\right)$$

$B_i(O_t)$, $\gamma_t(i)$

$$= \sum_{t=1}^{T}\sum_{i=1}^{N}\left[\log P(O_t/q_t=i)\right] P(q_t=i/O,\lambda^{old})$$

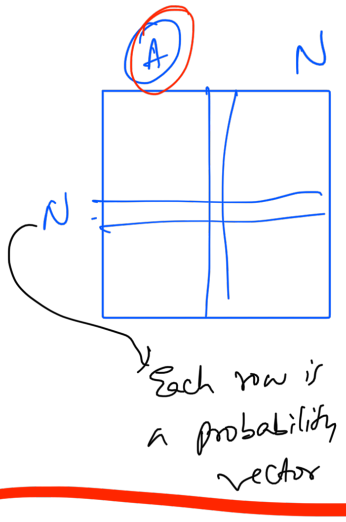
$$E_{P(q/O,\lambda^{old})}\left[\sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{i=1}^{N} \log P(O_t/q_t=i) \times I(O_t=m)\right]$$

$$- \sum_{i=1}^{N} h_i\left(\sum_{j=1}^{M} B_{ij} - 1\right)$$

$$\frac{\partial}{\partial B_{ij}} = \sum_{t=1}^{T} \frac{I(O_t=m)}{B_{im}} \gamma_t(i) - h_i = 0$$

$$B_{im} = \frac{\sum_{t=1}^{T} I(O_t = m) \times \gamma_t(i)}{h_i}$$

$$h_i = \sum_{m=1}^{M} \sum_{t=1}^{T} I(O_t = m) \times \gamma_t(i)$$



$$\frac{\partial}{\partial A_{ij}}$$

$$\sum_{t=2}^{T} E_{P(q/O,\lambda^{old})}\left[\log P(q_t/q_{t-1})\right] - \sum_{i=1}^{N} K_i\left[\sum_{j=1}^{N} A_{ij} - 1\right]$$

$A_{q_t q_{t-1}}$

$$\sum_{t=1}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \left[\log A_{q_t=i\, q_{t+1}=j}\right] \times P(q_t = i,\ q_{t+1} = j/O, \lambda)$$

$\xi_t(i,j)$

This is because summation is from $t=1$ here

$$\sum_{t=1}^{T-1} \frac{1}{A_{ij}} I(q_t = i)\ I(q_{t+1} = j)\ \xi_t(i,j) - K_i = 0$$

$$A_{ij} = \frac{\sum_{t=1}^{T-1} I(q_t = i) \times I(q_{t+1} = j)\, \xi_t(i,j)}{K_i}$$

$$\& \quad K_i = \sum_{j=1}^{N} \sum_{t=1}^{T-1} I(q_t = i)\ I(q_{t+1} = j) \times \xi_t(i,j)$$

$$= \sum_{t=1}^{T-1} \gamma_t(i)$$

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

Weak-supervised learning => learning from labelled features

K-means, GMM, general EM

HMM => parameter estimation

Transfer learning using spectral learning

SVM, PCA, SVD → EM for PLSA

Mixture models, latent Dirichlet Allocation, Gibbs sampling, variational Bayes

End Ki Sadhe

Graphical Models

V => Vocab

Docs.

→ w_{11} w_{12} w_{13} . . . . . w_{1N_1} (→ index of word from Vocab V)

→ w_{21} w_{22} w_{23} . . . . w_{2N_2}

→ w_{D1} w_{D2} w_{D3} . . . . . w_{DN_D}

w_{ij} = j^{th} word in i^{th} document. (index of this word)

Each doc. of diff. length

D = no. of docs.

N_D = no. of words in doc. D

Multinomial => Multi(θ)

↳ $[\theta_1 \ \theta_2 \ \ldots \ \theta_{|V|}]$

↳ probs. of all words

$$\text{Multi}(d|\theta) = \prod_{n=1}^{N_d} \theta_{w_n}$$

$$\text{Joint likelihood} = \prod_{d=1}^{D} \prod_{n=1}^{N_d} \theta_{w_{dn}}$$

$$\text{Joint log likelihood} = \sum_{d=1}^{D} \sum_{n=1}^{N_d} \log \theta_{w_{dn}} - \lambda \left[\sum_{w=1}^{|V|} \theta_w - 1 \right]$$

(lagrange multiplier)

$$\prod_{n=1}^{N_d} \theta_{w_n} = \prod_{w=1}^{|V|} \theta_w^{N_{dw}}$$

[N_{dw} = no. of times "w" occurs in doc. d]

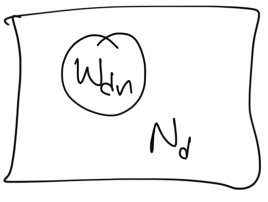
$$\therefore \text{JLL} = \sum_{d=1}^{D} \sum_{w=1}^{|V|} \log \theta_w^{N_{dw}} - \lambda \left[\sum_{w=1}^{|V|} \theta_w - 1 \right]$$

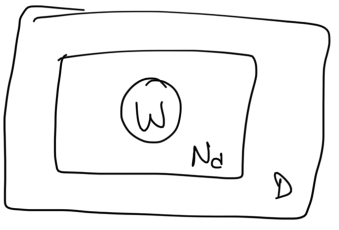
$$\frac{\partial \text{JLL}}{\partial \theta_w} = \sum_{d=1}^{D} \frac{N_{dw}}{\theta_w} - \lambda = 0$$

$$\therefore \theta_w = \frac{\sum_{d=1}^{D} N_{dw}}{\lambda} \Rightarrow \lambda = N \rightarrow \text{Total no. of tokens}$$

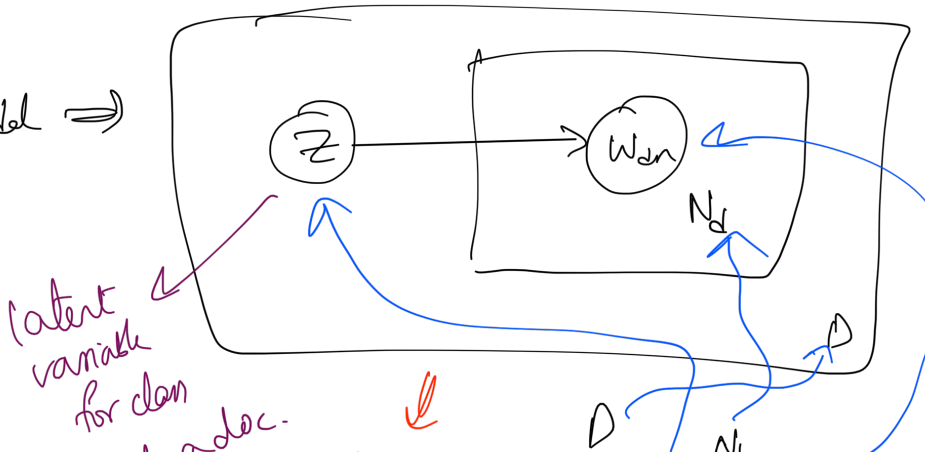
$$\therefore \theta_w = \frac{\sum_{d=1}^{D} N_{dw}}{N}$$

Read plate Notation $\Longrightarrow$





for mixture model ⇒



latent variable for class of a doc.

likelihood = $\prod_{d=1}^{D} P(z_d) \prod_{n=1}^{N_d} P(w_{dn}/z_d)$

No. of parameters = $V \times C + C$

$$\prod_{d=1}^{D} \left[\sum_{z_d=1}^{C} P(z_d) \prod_{w=1}^{|V|} P(w/z_d)^{N_{dw}} \right]$$

$$JLL = \sum_{d=1}^{D} \left[\log \sum_{z_d=1}^{C} P(z_d) \times \prod_{w=1}^{|V|} P(w/z_d)^{N_{dw}} \right]$$

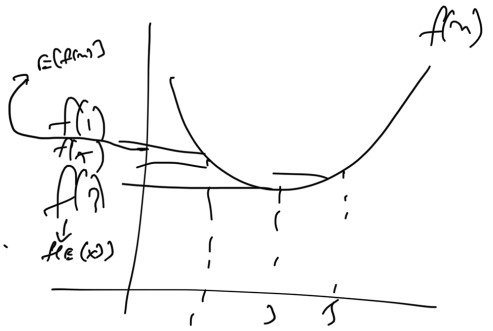
E-step : $P(z_d / w_d, \theta^{old})$

M-step : $E_{P(z_d/w_d,\theta^{old})} \left[\sum_{d=1}^{D} \log P(z_d, w_d / \theta) \right]$

Jensen's Inequality

Let f be a convex function

↳ i.e. $f''(x) \geq 0$, for all x.



Let X be a random variable,

Then $f(E[X]) \leq E[f(x)]$

if $f(x)$ is strictly convex, → $f''(x) > 0$

then $E[f(x)] = f(E[x])$

$P(x=1) = 1/2$

$P(x=5) = 1/2$

mean $= f(E[x])$

$= 3$

$E[x] = \frac{1+5}{2} = 3$

$E[f(x)] = \frac{1}{2} \times f(1) + \frac{1}{2} \times f(5)$

$= 3$

⇒ If f is a concave function

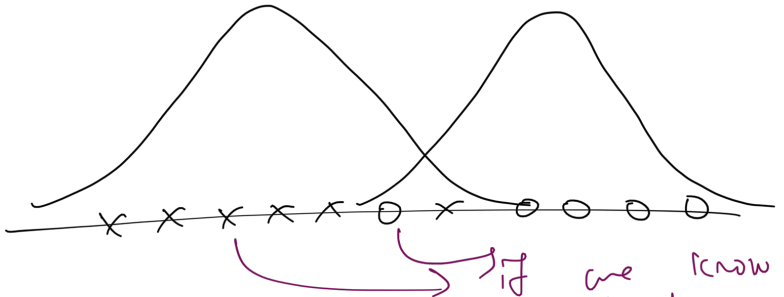
↳ $f''(x) \leq 0$, for all x

& $f(E[X]) \geq E[f(x)]$

If f is convex ⇒ $-f$ is concave

EM

1-D Example



if we know the clans then we can calculate the μ_k & Σ_k for each cluster. Else if, we know the μ_k & Σ_k, we can predict the clans

for each x.

The problem here is that we don't know both.
Classic Chicken & Egg problem.

Mixture of Gaussians Model

Suppose that a latent (hidden / unobserved) random variable z & $x^{(i)}$, $z^{(i)}$ are distributed,

$$p(x^{(i)}, z^{(i)}) = p(x^{(i)} | z^{(i)}) \times p(z^{(i)})$$

where $z^{(i)} \sim \text{Multinomial}(\phi)$ ← parameters

$$x^{(i)} | z^{(i)} = j \sim \mathcal{N}(\mu_j, \Sigma_j) \qquad z \in \{1, 2, \cdots, k\}$$

If we know the $z^{(i)}$'s we use MLE:

$$l(\phi, \mu, \Sigma) = \sum_{i=1}^{m} \log p(x^{(i)}, z^{(i)} | \phi, \mu, \Sigma)$$

$$\phi_j = \frac{1}{m} \sum_{i=1}^{m} 1 \times I(z^{(i)} = j)$$

$$\mu_j = \frac{\sum_{i=1}^{m} 1 \times I(z^{(i)} = j)\, x^{(i)}}{\sum_{i=1}^{m} 1 \times I(z^{(i)} = j)}$$

E-step (Guess value of $z^{(i)}$'s)

$w_j^{(i)}$ is how much $x^{(i)}$ is assigned to the Gaussian "j"

Set $w_j^{(i)} = p(z^{(i)} = j | x^{(i)}, \phi, \mu, \Sigma)$

$$= \frac{p(x^{(i)} | z^{(i)} = j)\, p(z^{(i)} = j)}{\sum_{l=1}^{k} p(x^{(i)} | z^{(i)} = l)\, p(z^{(i)} = l)}$$

$p(x^{(i)} | z^{(i)} = j) \sim \mathcal{N}(\mu_j, \Sigma_j)$

$p(z^{(i)} = j) \sim z \sim \text{Multinomial}(\phi)$

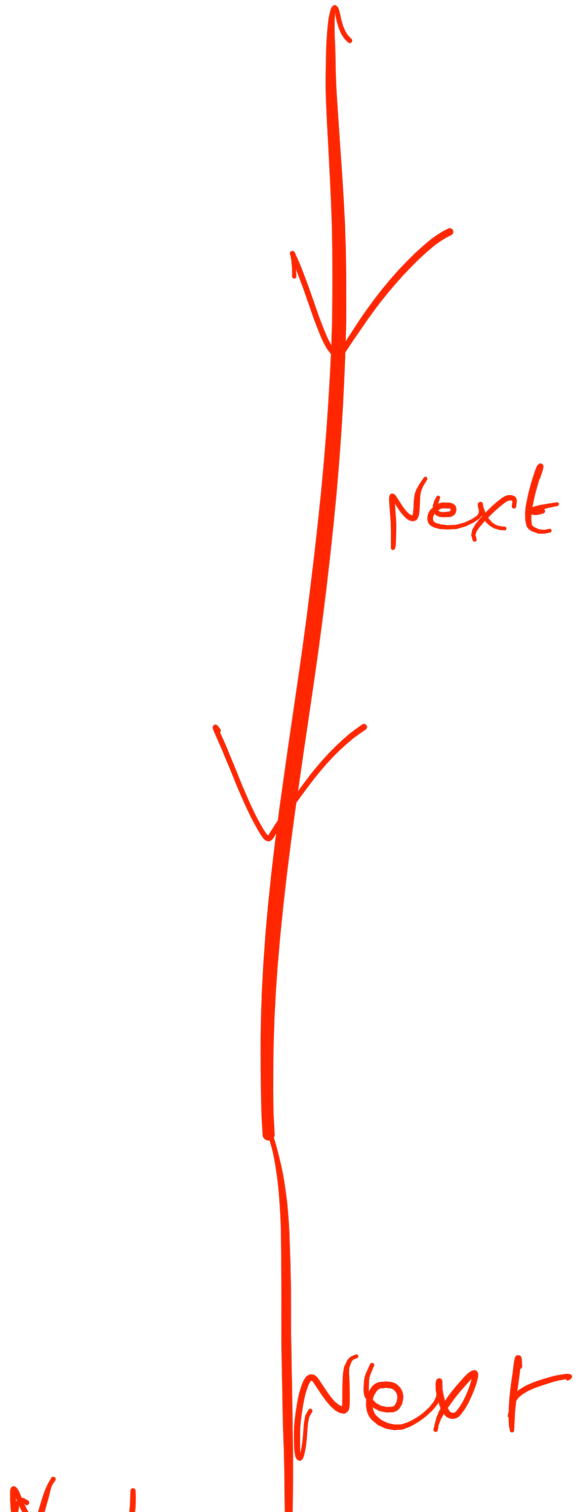
M-step

$$\phi_j = \frac{1}{m} \sum_{i=1}^{m} w_j^{(i)}$$

$$\mu_j = \frac{\sum_{i=1}^{m} w_j^{(i)} x^{(i)}}{\sum_{i=1}^{m} w_j^{(i)}}$$

$$w_j^{(i)} = E\left[1 \times I(z^{(i)} = j)\right]$$

EM ⇒ Soft clustering assignment.



EM.

Density estimation → Used in anomaly detection.

Model $p(x) \Rightarrow$ Estimate the density from which the dataset points came from.

if $p(x) < \epsilon \Rightarrow$ flag anomaly.

Mixture of gaussians for each feature.

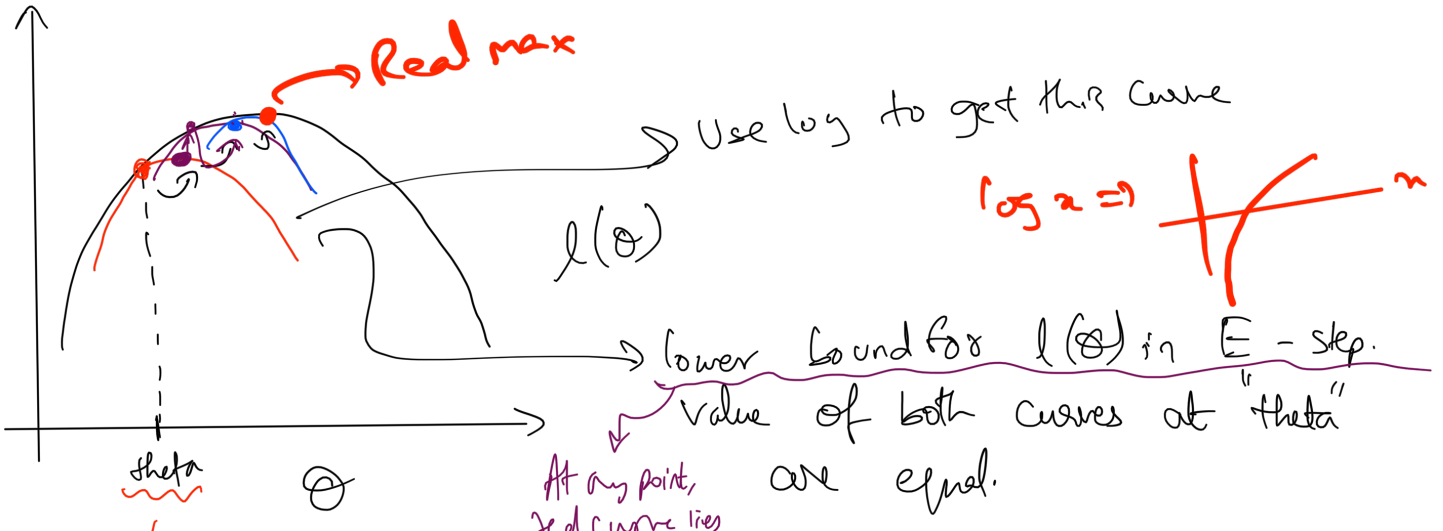
Read ML notes ⇒ for Anomaly detection

Have model for $p(x, z/\theta)$

Only observe x $\{x^{(1)}, \ldots x^{(m)}\}$

$$l(\theta) = \sum_{i=1}^{m} \log p(x^{(i)}/\theta)$$

$$= \sum_{i=1}^{m} \log \sum_{z^{(i)}} p(x^{(i)}, z^{(i)}/\theta)$$



$\log x \Rightarrow$

↳ initial value of θ.

M - step ⇒ Find maximum on the new-curve.
theta = this maximum.

$$\max_\theta \sum_i \log \sum_{z^{(i)}} p(x^{(i)}, z^{(i)} | \theta) \rightarrow \ell(\theta)$$

$$= \sum_i \log \sum_{z^{(i)}} Q_i(z^{(i)}) \frac{p(x^{(i)}, z^{(i)} | \theta)}{Q_i(z^{(i)})}$$

where $Q_i(z^{(i)})$ is a probability distribution

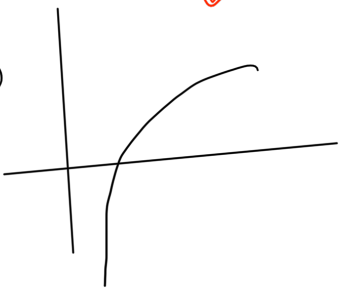
i.e. $\sum_{z^{(i)}} Q_i(z^{(i)}) = 1$

$$\geq \sum_i \log \mathop{E}_{z^{(i)} \sim Q_i} \left[\frac{p(x^{(i)}, z^{(i)} | \theta)}{Q_i(z^{(i)})} \right]$$

$$= \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)} | \theta)}{Q_i(z^{(i)})}$$

i.e., $f(E[x]) \geq E[f(x)]$

$\log x \Rightarrow$



If $z = \{1, \ldots, 10\}$

(roll a 10-sided dice

then

$$E[g(z)] = \sum_z p(z)\, g(z)$$

$$E[z] = \sum_z p(z) \cdot z$$

→ On a given iteration of EM (with params θ)

we want,

$$\log \mathop{E}_{z^{(i)} \sim Q_i} \left[\frac{p(x^{(i)}, z^{(i)} | \theta)}{Q_i(z^{(i)})} \right] = \mathop{E}_{z^{(i)} \sim Q_i} \left[\log \frac{p(x^{(i)}, z^{(i)} | \theta)}{Q_i(z^{(i)})} \right]$$

For this to hold, we need

$$\frac{P(x^{(i)}, z^{(i)})}{Q_i(z^{(i)})} = \text{Constant}$$

Set $\sum_{z^{(i)}} Q_i(z^{(i)}) = 1$

$$Q_i(z^{(i)}) = \frac{P(x^{(i)}, z^{(i)}/\theta)}{\sum_{z^{(i)}} P(x^{(i)}, z^{(i)}/\theta}$$

After derivation,

$$Q_i(z^{(i)}) = P(z^{(i)} / x^{(i)}, \theta)$$

E-step $\Rightarrow$ set $Q_i(z^{(i)}) = P(z^{(i)}, x^{(i)}/\theta)$ $\hookrightarrow w_j^{(i)}$

M-step $\Rightarrow$ $\theta = \underset{\theta}{\text{argmax}} \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{P(x^{(i)}, z^{(i)}/\theta)}{Q_i(z^{(i)})}$

General

⇒ The gamma function, denoted by Γn is a generalization of the factorial func. for non-integer values of n.

$$\Gamma n = \int_0^\infty t^{(n-1)} e^{-t}\, dt$$

⇒ A simplex is geometric object defined by a set of $n+1$ vertices in n-dimensional space.

Eg: triangle in 2-d space, defined by 3 vertices.
tetrahedron in 3-d space, defined by 4 vertices.

⇒ Dirichlet distribution is a probability dist. on the simplex. It is generalization of β-dist. in n-dimensions.

$$f(n|\alpha) = \frac{\Gamma(\alpha_0)}{\prod_{j=1}^{k} \Gamma(\alpha_i)} \prod_{i=1}^{k} M_i^{\alpha_i - 1}$$

$$\alpha_0 = \sum_{i=1}^{k} \alpha_i$$

Dirichlet is used as a prior for categorical data, where the simplex represents the set of possible categorical outcomes.

⇒ Multinomial Distribution

$$P(X = n) = \frac{n!}{n_1!\, n_2!\, n_3! \ldots n_k!} \times P_1^{n_1} P_2^{n_2} \ldots P_k^{n_k}$$