Module 4

- Introduction, **Bayes theorem**, Bayes theorem and concept learning, ML and LS error hypothesis, ML for predicting probabilities, Minimum Description Length principle, Naive Bayes classifier, Bayesian belief networks, EM algorithm

4.1 Introduction

- Probabilistic approach to inference
- **Basic assumption**:
  - Quantities of interest are governed by probability distributions
  - Optimal decisions can be made by reasoning about these probabilities together with observed training data

Relevance of Bayesian Learning

Bayesian Learning is relevant for two reasons

- **First reason**: **explicit manipulation of probabilities**
  - among the most practical approaches to certain types of learning problems
  - **e.g.** Bayes classifier is **competitive with decision tree** and neural network learning
- **Second reason: useful perspective for understanding learning methods that do not explicitly manipulate probabilities**
  - determine conditions under which algorithms output the most probable hypothesis
  - e.g. justification of the error functions in ANNs
  - e.g.justification of the inductive bias of decision trees

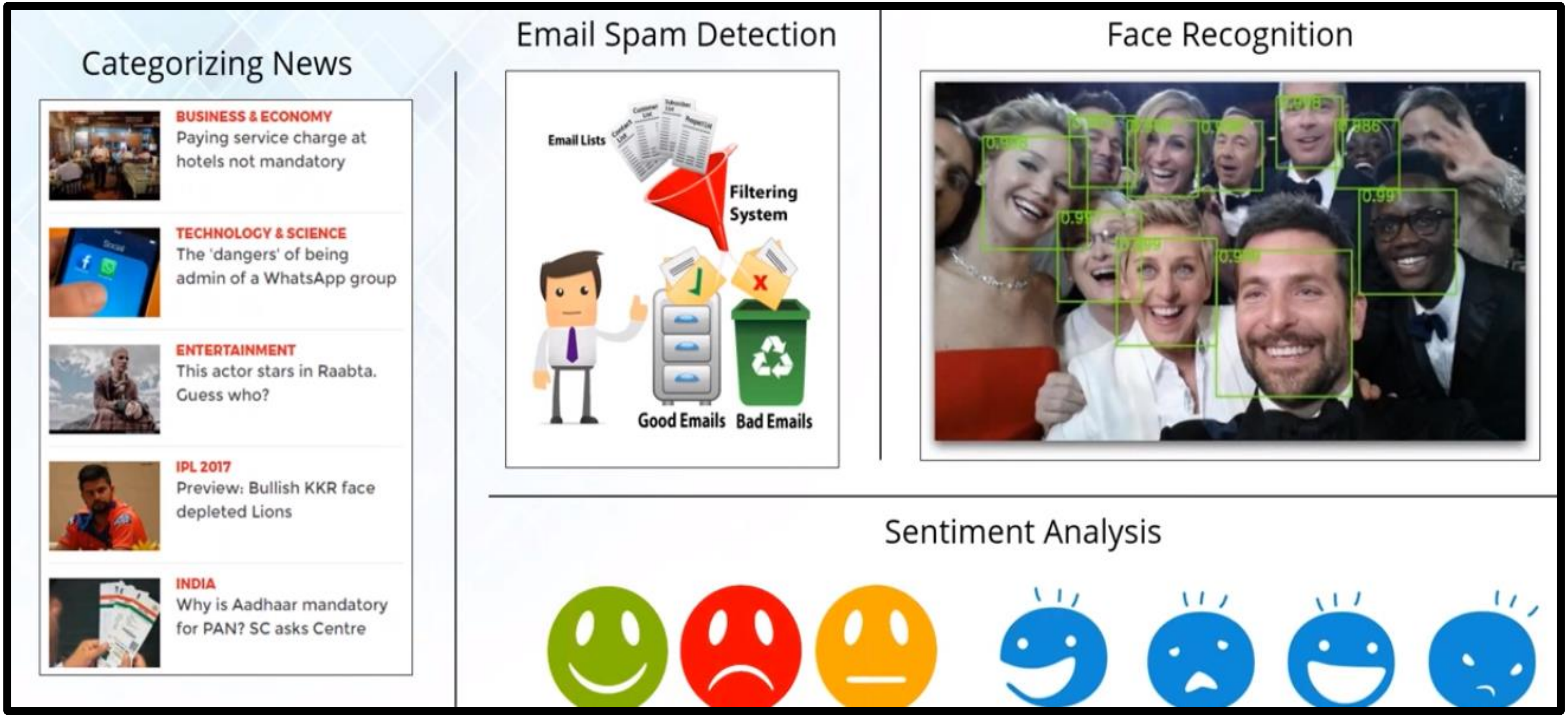
Features of Bayesian Learning Methods

- Each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct.
- Prior knowledge can be combined with observed data to determine the final probability of a hypothesis.
- Bayesian methods can accommodated hypothesis that make probabilities predictions.
- New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities
- Provides a standard of optimal decision making against which other practical methods can be measured.

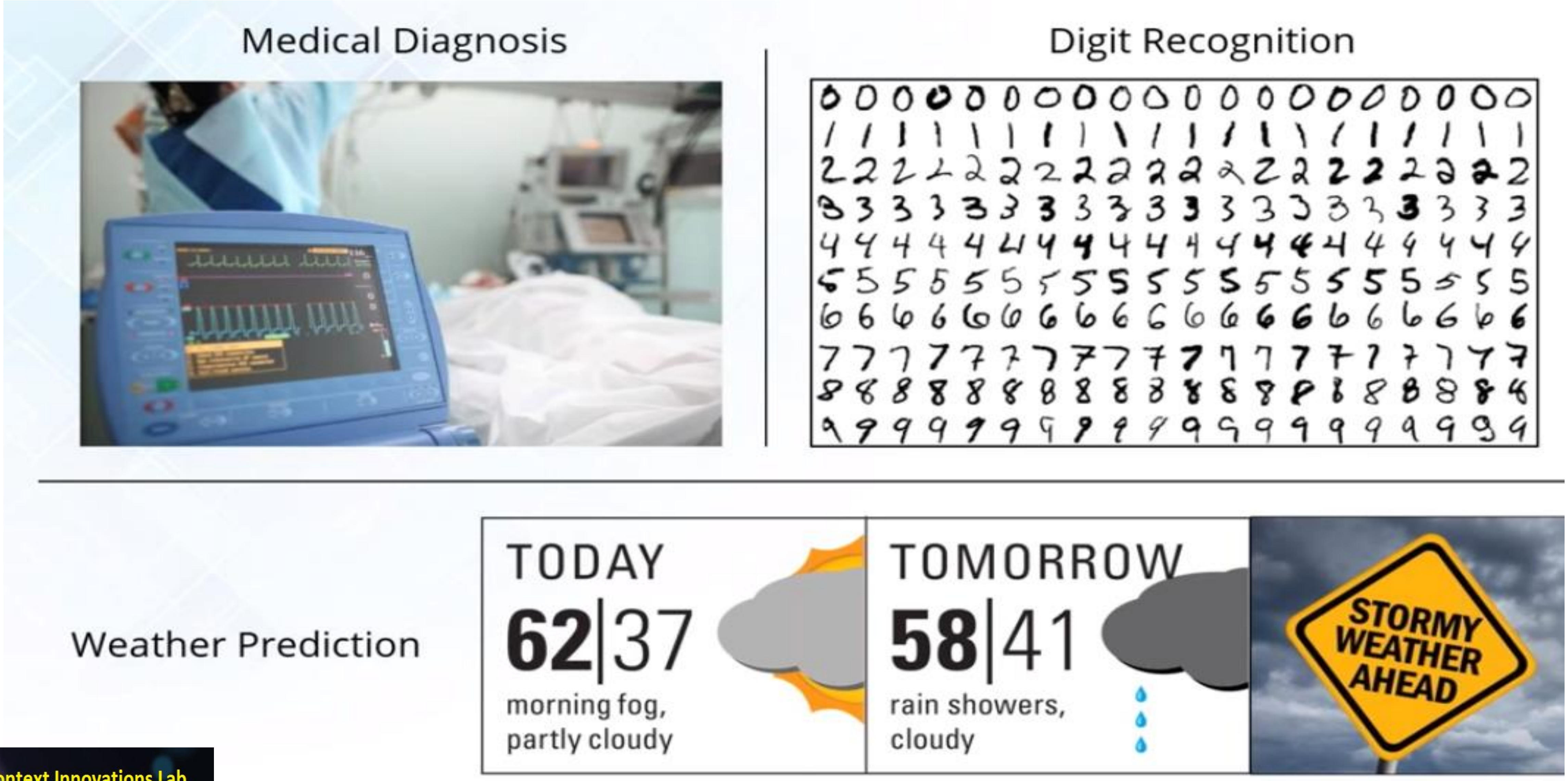
Practical Difficulties

- Initial knowledge of many probabilities is required
- Significant computational costs required

Use cases


Categorizing News
BUSINESS & ECONOMY
Paying service charge at hotels not mandatory
TECHNOLOGY & SCIENCE
The 'dangers' of being admin of a WhatsApp group
ENTERTAINMENT
This actor stars in Raabta. Guess who?
IPL 2017
Preview: Bullish KKR face depleted Lions
INDIA
Why is Aadhaar mandatory for PAN? SC asks Centre
Email Spam Detection
Email Lists
Filtering System
Good Emails
Bad Emails
Face Recognition
Sentiment Analysis

Use cases



Context Innovations Lab

Basic Formulas for Probabilities

- ***Product Rule***: probability $P(A \wedge B)$ of a conjunction of two events A and B:

$$P(A \wedge B) = P(A \mid B)\, P(B) = P(B \mid A)\, P(A)$$

- ***Sum Rule***: probability of a disjunction of two events A and B:

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

- ***Theorem of total probability***: if events $A_1, \ldots, A_n$ are mutually exclusive with $\Sigma_{i=1}^{n} P(A_i) = 1$

, then $P(B) = \sum_{i=1}^{n} P(B|A_i)P(A_i)$

Conditional Probability

Definition. The **conditional probability** of an event *A* given that an event *B* has occurred is written: **P(A|B)** and is calculated using:

$P(A|B) = P(A \cap B) / P(B)$ as long as $\boldsymbol{P(B) > 0}$.

Example :

P(A) = 4/52

P(B) = 4/51

P(A and B) = 4/52*4/51= 0.006

$$P\left(\frac{B}{A}\right) = \frac{P(\text{A and B})}{P(A)} = \frac{0.006}{0.077} = 0.078$$

4.2 Bayes Theorem

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$$

- $P(h)$ = prior (initial) probability that hypothesis h holds , before we observed any training data.
- $P(D)$ = prior probability of training data D
- $P(h|D)$ = posterior probability of h given D *(it holds after we have seen the training data D)*
- $P(D|h)$ = probability of observing data D given some world in which hypothesis h holds.

4.2.1 Maximum a posterior (MAP) hypothesis

- In many learning scenarios , the learner considers some set of candidate hypotheses H and is interested in finding the most probable hypotheses hЄH given the observed data D .
- Any such maximally probable hypothesis is called a maximum posteriori (MAP) hypothesis h_{MAP}:

$$\begin{aligned} h_{MAP} &= \arg\max_{h\in H} P(h|D) \\ &= \arg\max_{h\in H} \frac{P(D|h)P(h)}{P(D)} \\ &= \arg\max_{h\in H} P(D|h)P(h) \end{aligned}$$

4.2.2Maximum Likelihood

- In some cases we will assume that every hypothesis in H is equally probable a priori (***P*(*h*$_i$)** **= *P*(*h*$_j$) for all *h*$_i$** in H) hen can further simplify and need to consider the term **P(D|h)** is often called the likelihood of the data D given h and hypothesis that maximizes **P(D|h)** is called a ***Maximum likelihood*** (ML) hypothesis $\boldsymbol{h_{ML}}$

$$h_{ML} = \arg\max_{h_i \in H} P(D|h_i)$$

An Example : Cancer Patient Diagnosis

- To illustrate Bayes Rule , Consider a medical diagnosis problem in which there are two alternative hypotheses :

1. That the patient has a particular form of cancer and
2. That the patient does not.

The available data is from a particular laboratory test with two possible outcomes :

+ : positive

- : negative

Example : Medical Cancer Test Details of Patient

A patient takes a lab test and the result comes back positive. The test returns a correct positive result in only 98% of the cases in which the disease is actually present, and a correct negative result in only 97% of the cases in which the disease is not present. Furthermore, .008 of the entire population have this cancer.

$P(cancer) =$ 0.008

$P(\neg cancer) =$ 0.992

$P(+|cancer) =$ 0.98

$P(-|cancer) =$ 0.02

$P(+|\neg cancer) =$ 0.03

$P(-|\neg cancer) =$ 0.97

Example : Does patient have cancer or not?

The Maximum a posterior hypothesis for Patient having cancer/no cancer :

$\mathbf{cancer}_{\boldsymbol{MAP}}$ = P(+|cancer) P(cancer) = (0.98)(0.008) = **0.0078**

$\neg\boldsymbol{cancer}_{\boldsymbol{MAP}}$ = P(+| $\neg$ cancer) P($\neg$ cancer) =(0.03)(0.992) = **0.0298**

4.3 Relation to Concept Learning

- Consider our usual concept learning task
  - instance space ***X***, hypothesis space ***H***, training examples *D*
  - consider the **FindS** learning algorithm (outputs most specific hypothesis from the version space $VS_{H,D}$)
- What would Bayes rule produce as the MAP hypothesis?
- Does *FindS* output a MAP hypothesis??

Brute Force Bayes Concept Learning

- Assume that the learner considers some finite hypothesis space H defined over the instance space X , in which the task is to learn some target concept **c: X-> {0,1}**
- Assume fixed set of instances **<x_1,…, x_m>**
- Assume *D* is the set of classifications: ***D* = <*c*(x_1),…,*c*(x_m)>**
- Assume that the learner has given some sequence of training examples **<<x_1,d_1><x_2,d_2>,…………………….<x_m, d_m>>** where x_i is some instance from X and where **d_i** is the target value of **x_i** **(i.e d_i= c(x_i)).**

Brute Force MAP Learning Algorithm

1. For each hypothesis *h* in *H*, calculate the posterior probability

$$P(h\,|\,D) = \frac{P(D\,|\,h)P(h)}{P(D)}$$

2. Output the hypothesis h_{MAP} with the highest posterior probability

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}}\, P(h|D)$$

Assumptions

The probability distribution P(h) and P(D|h) is chosen to be consistent with the following assumptions :

1. The training data D is noise free(i.e. $d_i = c(x_i)$)
2. The target concept c is contained in the hypothesis space H
3. We have no *a priori reason* to believe that any hypothesis is more probable than any other.

The Values of P(h) and P(D|h)

- Choose *P*(*h*) to be *uniform* distribution
  - ***P*(*h*) = 1/|*H*| for all *h* in *H***
- Choose *P*(*D*|*h*):

$$P(D|h) = \begin{cases} 1 \text{ if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \; (h \text{ consistent with } D) \\ 0 \text{ otherwise} \end{cases}$$

Two cases

- By Applying Bayes theorem $P(h|D) = \frac{P(D|h)P(h)}{P(D)}$

- **Case1 :** When h is inconsistent with training data D:

  $$P(h|D) = 0.P(h)/P(D) = 0$$

- **Case 2:** When h is consistent with D , we have

  $$P(h|D) = (1*1/|H|)/(|V_{SH,D}|/|H|)$$
  $$= 1/|V_{SH,D}|$$

To Summarize

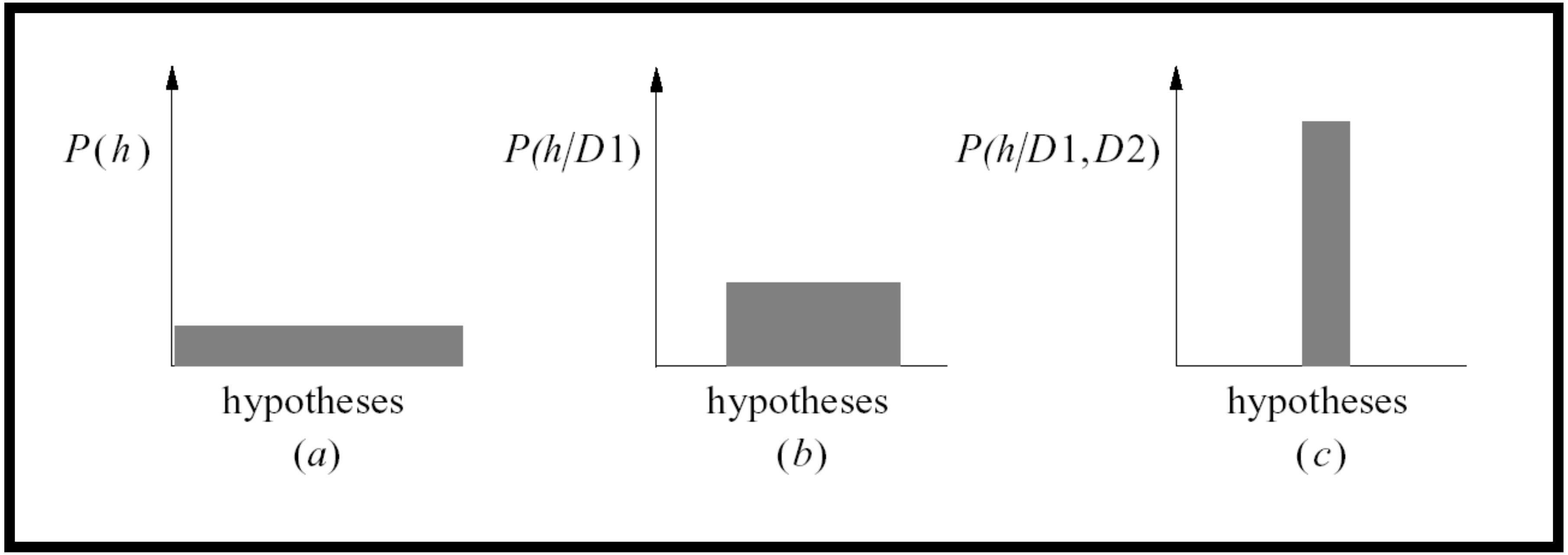
- To summarize , Bayes theorem implies that the posterior probability P(h|D) under our assumed P(h) and P(D|h) is

$$P(h|D) = \begin{cases} \frac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$$

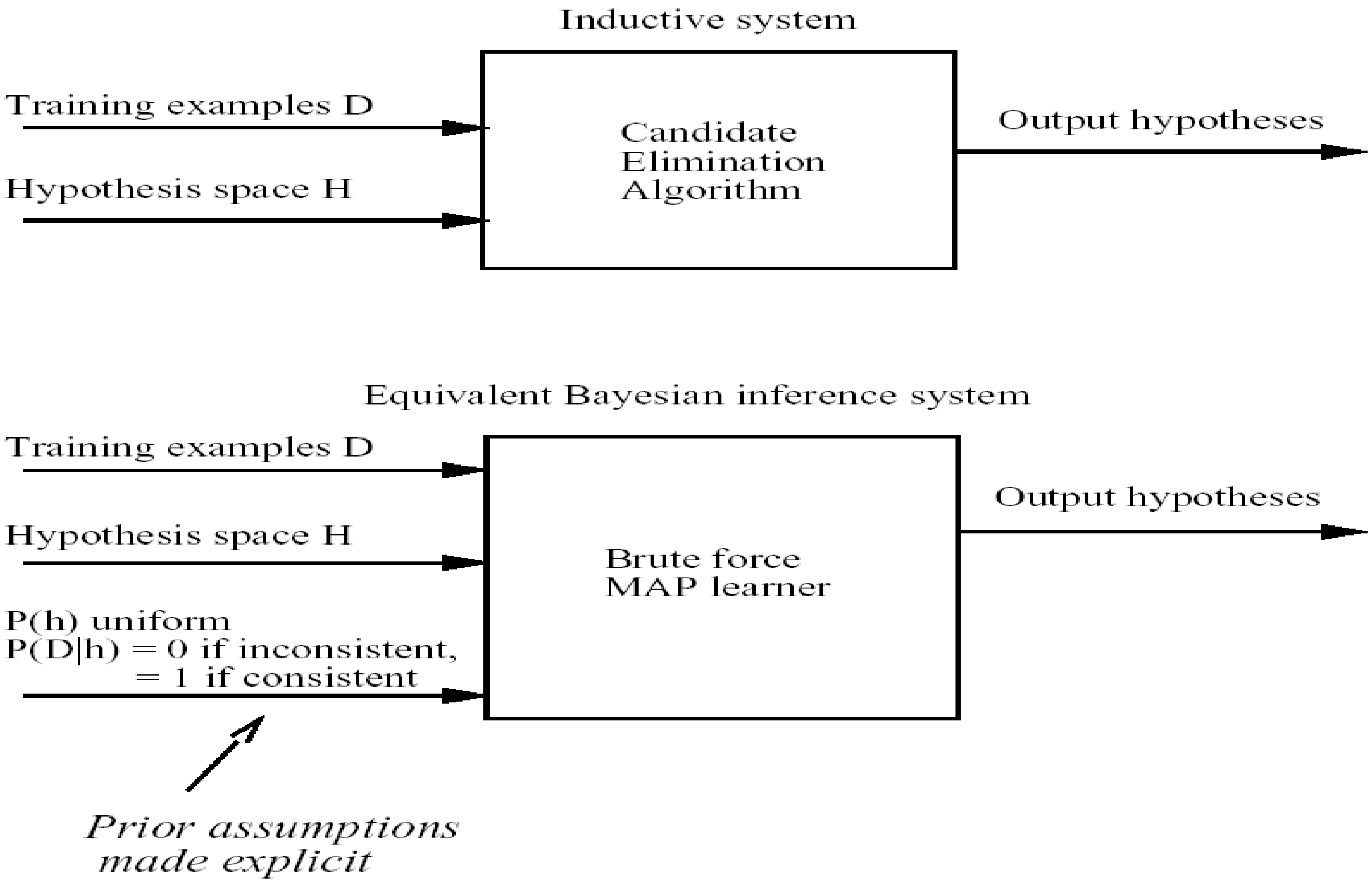
Refer

- Refer the text book “ Machine Learning “ Tom M Mitchell : Page No 159 to 16

4.4 MAP hypothesis and Consistent Learners



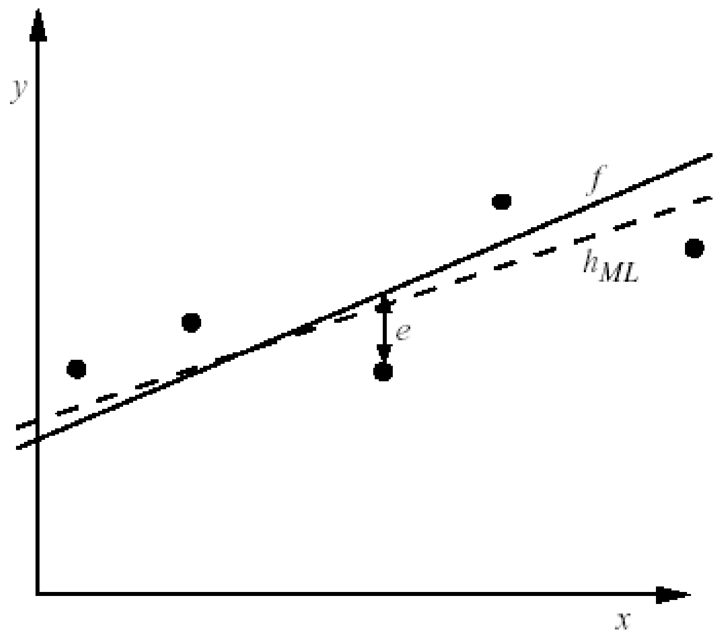
Characterizing Learning Algorithms by Equivalent MAP Learners



4.5 Maximum likelihood and Least Squared Error hypothesis

- A straightforward Bayesian analysis will show that under certain assumptions any learning algorithm that minimizes the squared error between the output hypothesis predictions and the training data will output a maximum likelihood hypothesis.

Learning A Real Valued Function



- Consider any real-valued target function *f*

  Training examples <x_i, d_i>, where d_i is noisy training value

  - $d_i = f(x_i) + e_i$
  - e_i is random variable (noise) drawn independently for each x_i according to some Gaussian distribution with mean=0

- Then the maximum likelihood hypothesis h_{ML} is the one that minimizes the sum of squared errors:

$$h_{ML} = \arg\min_{h \in H} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

Learning A Real Valued Function

$$
\begin{aligned}
h_{ML} &= \underset{h \in H}{\operatorname{argmax}}\, p(D|h) \\
&= \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} p(d_i|h) \\
&= \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2}
\end{aligned}
$$

- Maximize natural log of this instead...

$$
\begin{aligned}
h_{ML} &= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2 \\
&= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} -\frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2 \\
&= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} -\left(d_i - h(x_i)\right)^2 \\
&= \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} \left(d_i - h(x_i)\right)^2
\end{aligned}
$$

Reference

- Refer the text book “ Machine Learning “ Tom M Mitchell : Page No 164 to 167

4.6 Maximum Likelihood Hypothesis Learning to Predict Probabilities

- Consider predicting survival probability from patient data
- Training examples $<x_i, d_i>$, where d_i is 1 or 0
- Want to train neural network to output a *probability* given x_i (not a 0 or 1)
- In this case can show

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} d_i \ln h(x_i) + (1 - d_i) \ln(1 - h(x_i))$$

- Weight update rule for a sigmoid unit:

$$w_{jk} \leftarrow w_{jk} + \Delta w_{jk}$$

where

$$\Delta w_{jk} = \eta \sum_{i=1}^{m} (d_i - h(x_i)) \; x_{ijk}$$

Reference

- For Complete Derivation Refer the text book “ Machine Learning “ Tom M Mitchell : **Page No 168 to 171**

4.6 Naive Bayes classifier /Bayes Rule

- Highly Bayesian learning method is the naïve Bayes learner often called the naïve Bayes Classifier .
- Bayesian Classifier assumes that all the variables are ***conditionally independent*** given the value of the target variable.
- The naïve Bayes Classifier applies to learning tasks where each ***instance x is described by a conjunction of attribute values*** and where ***the target function f(x) can take on any value from some finite set V.***
- A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values **<a_1, a_2 , a_3 ,------- a_n >.** The learner is asked to predict the target value, or classification, for this new instance.

- The Bayesian approach to classifying the new instance is to assign the most probable target value, V_{MAP}, given the attribute values <a_1, a_2 , a_3 ,------- a_n >.that describe the instance.

$$v_{MAP} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j | a_1, a_2 \ldots a_n)$$

We can use Bayes theorem to rewrite this expression as

$$v_{MAP} = \underset{v_j \in V}{\operatorname{argmax}} \frac{P(a_1, a_2 \ldots a_n | v_j) P(v_j)}{P(a_1, a_2 \ldots a_n)}$$

$$= \underset{v_j \in V}{\operatorname{argmax}} P(a_1, a_2 \ldots a_n | v_j) P(v_j)$$

Naive Bayes classifier:

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_i P(a_i | v_j)$$

Illustrative Example

- Example: Play Tennis

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Learning Phase

P(Outlook | Play)

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Learning Phase

P(Temperature|Play)

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Learning Phase

P(Humidity | Play)

Humidity	Play=*Yes*	Play=*No*
High	3/9	4/5
Normal	6/9	1/5

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Learning Phase

P(Wind | Play)

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Learning Phase

P(Outlook | Play)

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

P(Temperature | Play)

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

P(Humidity | Play)

Humidity	Play=*Yes*	Play=No
High	3/9	4/5
Normal	6/9	1/5

P(Wind | Play)

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

$P(\text{Play}=Yes) = 9/14$

$P(\text{Play}=No) = 5/14$

Example

- Test Phase
  - Given a new instance,

    $\mathbf{x}'$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)
  - Look up tables

P(Outlook=*Sunny*|Play=*Yes*) = 2/9

P(Temperature=*Cool*|Play=*Yes*) = 3/9

P(Huminity=*High*|Play=*Yes*) = 3/9

P(Wind=*Strong*|Play=*Yes*) = 3/9

P(Play=*Yes*) = 9/14

P(Outlook=*Sunny*|Play=*No*) = 3/5

P(Temperature=*Cool*|Play==*No*) = 1/5

P(Huminity=*High*|Play=*No*) = 4/5

P(Wind=*Strong*|Play=*No*) = 3/5

P(Play=*No*) = 5/14

  - MAP rule

P(*Yes*|$\mathbf{x}'$): [P(*Sunny*|Yes)P(*Cool*|*Yes*)P(*High*|Yes)P(*Strong*|*Yes*)]P(Play=*Yes*) = 0.0053

P(*No*|$\mathbf{x}'$): [P(*Sunny*|No) P(*Cool*|No)P(*High*|*No*)P(*Strong*|*No*)]P(Play=*No*) = 0.0206

Given the fact P(*Yes*|$\mathbf{x}'$) < P(*No*|$\mathbf{x}'$), we label $\mathbf{x}'$ to be "*No*".

4.7 Event Models

- The assumptions on distributions of features are called the *event model* of the Naive Bayes classifier.
- **For discrete features** like the ones encountered in document classification (include spam filtering), multinomial and Bernoulli distributions are popular.
- **For Continuous feature , Gaussian naive Bayes distributions is popular.**

1. Gaussian naive Bayes

- When dealing with continuous data, a typical assumption is that the ***continuous values associated with each class are distributed according to a Gaussian distribution.***
- Then, the probability *distribution* of v given a class C_k, $p(x = v \mid C_k)$ can be computed by plugging V into the equation for a Normal distribution parameterized by μ_k and σ_k^2.

$$p(x = v \mid C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}}\, e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}}$$

2. Multinomial naive Bayes

Its is used when we have **discrete data** (*e.g. movie ratings ranging 1 and 5 as each rating will have certain* ***frequency*** *to represent*). In text learning we have the count of each word to predict the class or label.
The Multinomial Naive Bayes's conditional distribution is:

The term frequencies can then be used to compute the maximum-likelihood estimate based on the training data to estimate the class-conditional probabilities in the multinomial model:

$$\hat{P}(x_i \mid \omega_j) = \frac{\sum tf(x_i, d \in \omega_j) + \alpha}{\sum N_{d \in \omega j} + \alpha \cdot V}$$

where

- x_i: A word from the feature vector $\mathbf{x}$ of a particular sample.
- $\sum tf(x_i, d \in \omega_j)$: The sum of raw term frequencies of word x_i from all documents in the training sample that belong to class ω_j.
- $\sum N_{d \in \omega j}$: The sum of all term frequencies in the training dataset for class ω_j.
- α: An additive smoothing parameter ($\alpha = 1$ for Laplace smoothing).
- V: The size of the vocabulary (number of different words in the training set).

The class-conditional probability of encountering the text $\mathbf{x}$ can be calculated as the product from the likelihoods of the individual words (under the *naive* assumption of conditional independence).

$$P(\mathbf{x} \mid \omega_j) = P(x_1 \mid \omega_j) \cdot P(x_2 \mid \omega_j) \cdot \ldots \cdot P(x_n \mid \omega_j) = \prod^{m} P(x_i \mid \omega_j)$$

3. Bernoulli naive Bayes

It assumes that all our features are binary such that they take only two values. Means **0s** can represent “word does not occur in the document” and **1s** as "word occurs in the document" .

$$p(\mathbf{x} \mid C_k) = \prod_{i=1}^{n} p_{ki}^{x_i} (1 - p_{ki})^{(1-x_i)}$$

3. Bernoulli naive Bayes

It assumes that all our features are binary such that they take only two values. Means **0s** can represent “word does not occur in the document” and **1s** as "word occurs in the document" .

$$p(\mathbf{x} \mid C_k) = \prod_{i=1}^{n} p_{ki}^{x_i} (1 - p_{ki})^{(1-x_i)}$$

Lab Program

- **Assuming a set of documents that need to be classified, use the naïve Bayesian Classifier model to perform this task. Built-in Libraries can be used to write the program. Calculate the accuracy, precision, and recall for your data set.**

4.8 Learning to Classify Text – Algorithm

S1: LEARN_NAIVE_BAYES_TEXT (*Examples*, V)

S2: CLASSIFY_NAIVE_BAYES_TEXT (*Doc*)

- *Examples is a set of text documents along with their target values.*
- *V is the set of all possible target values.*
- *This function (S1) learns the probability terms* $\boldsymbol{P(w_k \mid v_j)}$*, describing the probability that a randomly drawn word from a document in* ***class*** $\boldsymbol{v_j}$ *will be the English word* ***wk****. It also learns the class prior probabilities* $\boldsymbol{P(v_j)}$*.*

S1: LEARN_NAIVE_BAYES_TEXT (*Examples*, V)

[V: Class , W: Word, doc : Documents]

1. *collect all words and other tokens that occur in Examples*

- ***Vocabulary*** ← all distinct words and other tokens in *Examples*

2. *calculate the required* $\boldsymbol{P(v_j)}$ ***and*** $\boldsymbol{P(w_k \mid v_j)}$ *probability terms*

- For each target value v_j in *V* do

$$P(v_j) \leftarrow \frac{|docs_j|}{|Examples|}$$

  - **$docs_j$ ← subset of *Examples* for which the target value is v_j**
  - **$Text_j$ ← a single document created by concatenating all members of $docs_j$**
  - **n ← total number of words in $Text_j$ (counting duplicate words multiple times)**
  - **for each word w_k in *Vocabulary***

$$P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$$

n_k ← number of times word w_k occurs in $Text_j$ ಡಾ॥ ತ್ಯಾಗರಾಜು ಜಿ.ಎಸ್

S2:CLASSIFY_NAIVE_BAYES_TEXT (*Doc*)

- *positions* $\leftarrow$ all word positions in *Doc* that contain tokens found in *Vocabulary*
- Return v_{NB} where

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_{i \in positions} P(a_i | v_j)$$

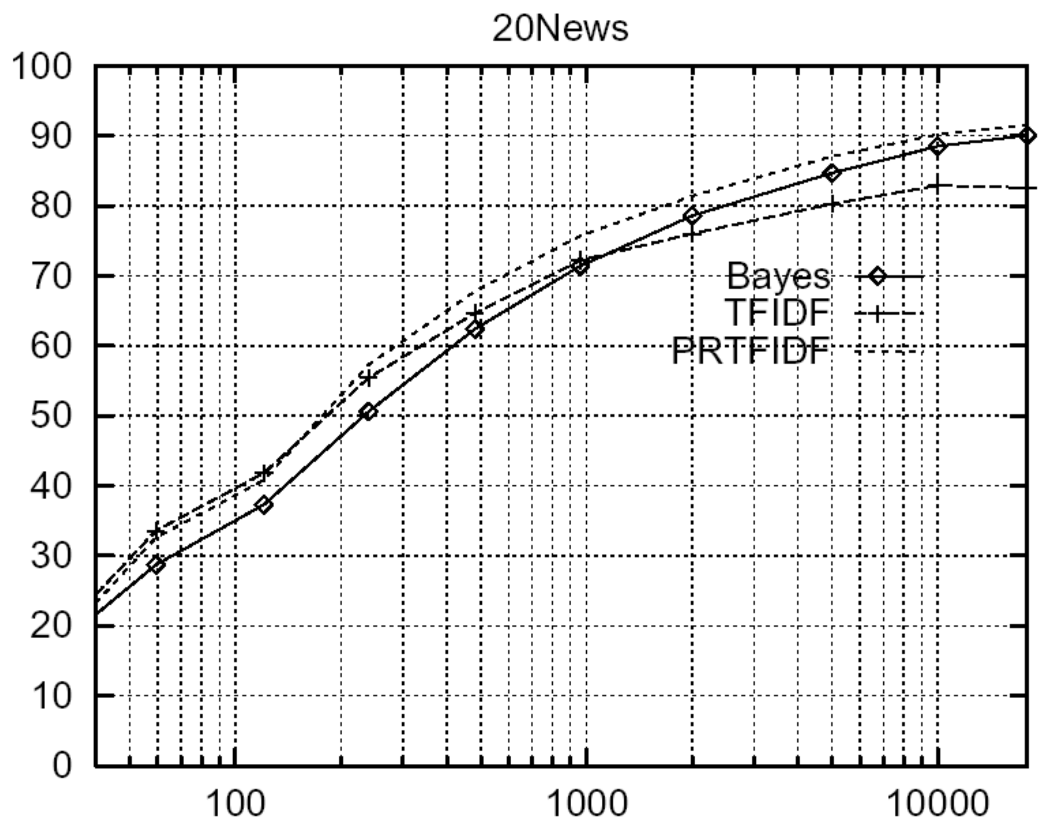
Twenty NewsGroups

- Given 1000 training documents from each group Learn to classify new documents according to which newsgroup it came from

comp.graphics	**misc.forsale**	**alt.atheism**	**sci.space**
comp.os.ms-windows.misc	**rec.autos**	**soc.religion.christian**	**sci.crypt**
comp.sys.ibm.pc.hardware	**rec.motorcycles**	**talk.religion.misc**	**sci.electronics**
comp.sys.mac.hardware	**rec.sport.baseball**	**talk.politics.mideast**	**sci.med**
comp.windows.x	**rec.sport.hockey**	**talk.politics.misc**	
		talk.politics.guns	

- Naive Bayes: 89% classification accuracy

Learning Curve for 20 Newsgroups



- Accuracy vs. Training set size (1/3 withheld for test)

Example :

- In the example, we are given a sentence " ***A very close game***", a training set of five sentences (as shown below), and their corresponding category (Sports or Not Sports).
- The goal is to build a Naive Bayes classifier that will tell us which category the sentence " ***A very close game***" belongs to.
- Applying a Naive Bayes classifier, thus the strategy would be calculating the probability of both "A very close game **is Sports**", as well as it's ***Not Sports.*** The one with the higher probability will be the result.

Text	Category
"A great game"	Sports
"The election was over"	Not sports
"Very clean match"	Sports
"A clean but forgettable game"	Sports
"It was a close election"	Not sports

Step 1: Feature Engineering

- **word frequencies**, i.e., counting the occurrence of every word in the document.
- ***P(a very close game)*** = P(a) X P(very) X P(close) X P(game)
- ***P(a very close game | Sports)*** = P(a|Sports) X P(Very|Sports) X P(close|Sports) X P(game|Sports)
- **P(*a very close game* | *Not Sports*)** = P(*a* | *Not Sports*) x P(*very* | *Not Sports*) x P(*close* | *Not Sports*) x P(*game* | *Not Sports*)

Step 2: Calculating the probabilities

- Here , the word "close" does not exist in the category Sports, thus **P(*close* |*Sports*)** = 0, leading to **P(*a very close game* | *Sports*)**=0.
- The probabilities are calculated using multinomial probability distribution function

$$P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$$

Word	P(word | Sports)	P(word | Not Sports)
a	$\frac{2+1}{11+14}$	$\frac{1+1}{9+14}$
very	$\frac{1+1}{11+14}$	$\frac{0+1}{9+14}$
close	$\frac{0+1}{11+14}$	$\frac{1+1}{9+14}$
game	$\frac{2+1}{11+14}$	$\frac{0+1}{9+14}$

$$P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$$

$$P(a|Sports) \times P(very|Sports) \times P(close|Sports) \times P(game|Sports) \times P(Sports)$$
$$= 4.61 \times 10^{-5}$$
$$= 0.0000461$$

$$\text{P(a — Not Sports)} \times P(very|Not\ Sports) \times P(close|Not\ Sports) \times P(game|Not\ Sports) \times P(Not\ Sports)$$
$$= 1.43 \times 10^{-5}$$
$$= 0.0000143$$

As seen from the results shown below, P(a very close game | Sports) gives a higher probability, suggesting that the sentence belongs to the Sports category.

4.9 Bayesian Network (BAYESIAN BELIEF NETWORKS)

- Bayesian Belief networks ***describe conditional independence*** among *subsets* of variables

Conditional Independence

- **Definition:** X is *conditionally independent* of Y given Z if the probability distribution governing X is independent of the value of Y given the value of Z; that is, if

$$(\forall x_i, y_j, z_k)\ P(X = x_i | Y = y_j, Z = z_k) = P(X = x_i | Z = z_k)$$

more compactly, we write

$$P(X|Y, Z) = P(X|Z)$$

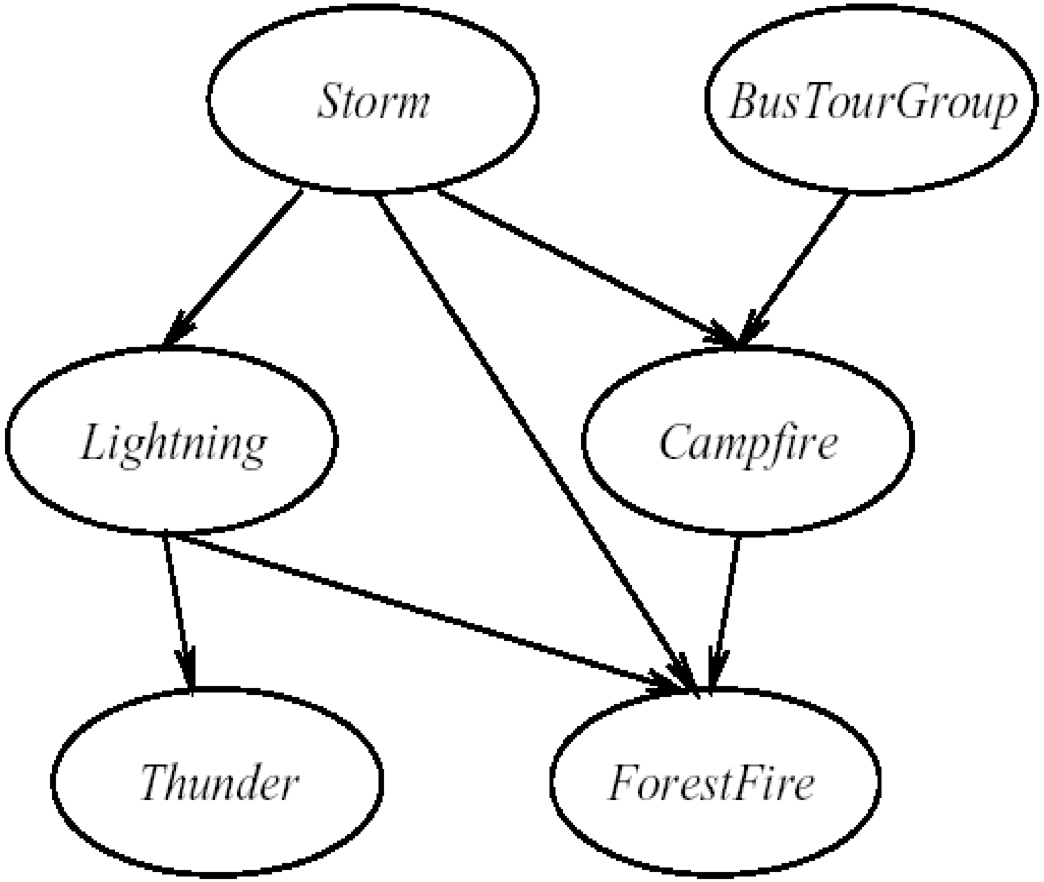
- **Example:** *Thunder* is conditionally independent of *Rain*, given *Lightning*

$$P(Thunder|Rain, Lightning) = P(Thunder|Lightning)$$

- Naive Bayes uses cond. indep. to justify

$$P(X, Y|Z) = P(X|Y, Z)\ P(Y|Z) = P(X|Z)\ P(Y|Z)$$

Bayesian Belief Network (1/2)



	S,B	$S,\neg B$	$\neg S,B$	$\neg S,\neg B$
C	0.4	0.1	0.8	0.2
$\neg C$	0.6	0.9	0.2	0.8

- Network represents a set of conditional independence assertions:
  - Each node is asserted to be conditionally independent of its non descendants, given its immediate predecessors.
  - Directed acyclic graph

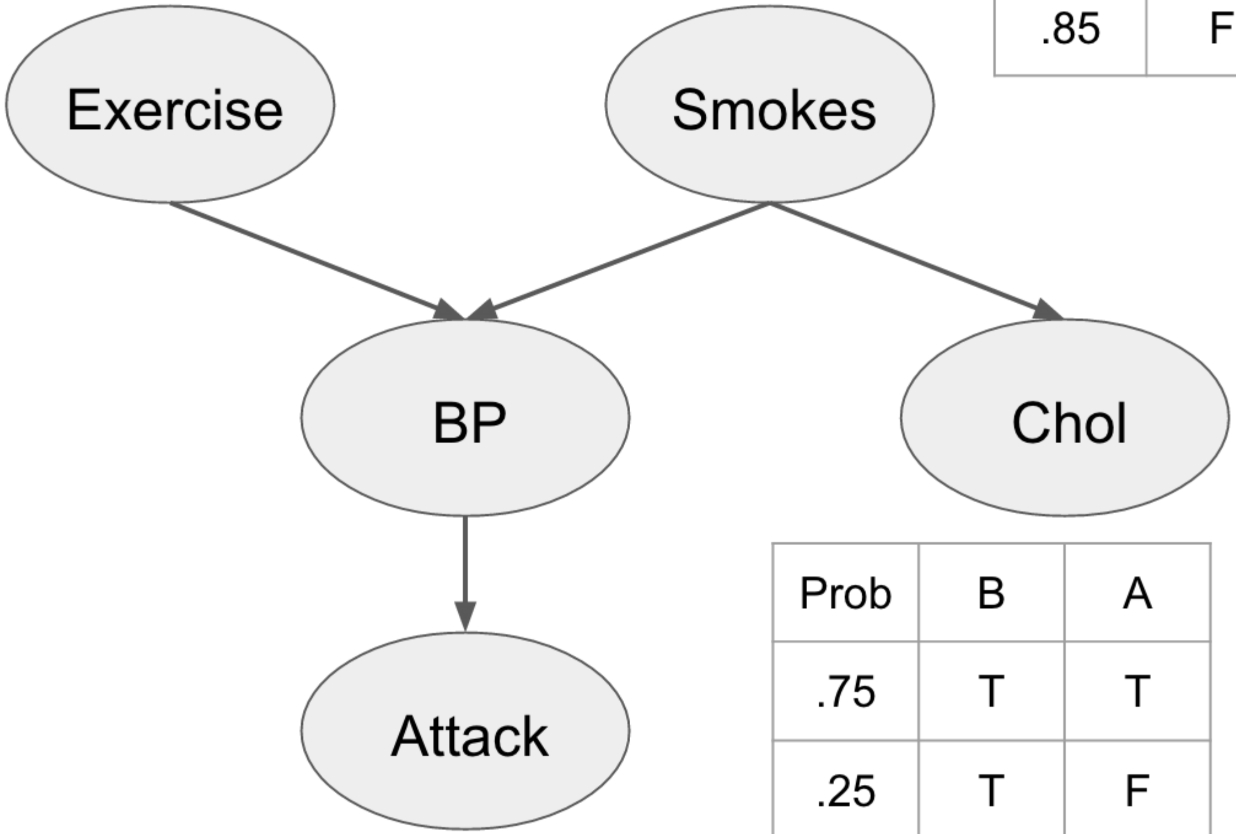
Bayesian Belief Network (2/2)

- Represents joint probability distribution over all variables
  - e.g., *P*(*Storm*, *BusTourGroup*, . . . , *ForestFire*)
  - in general, $$P(y_1, \ldots, y_n) = \prod_{i=1}^{n} P(y_i | Parents(Y_i))$$

    where *Parents*(Y_i) denotes immediate predecessors of Y_i in graph
  - so, joint distribution is fully defined by graph, plus the P(y_i|*Parents*(Y_i))

Prob	E
.4	T
.6	F

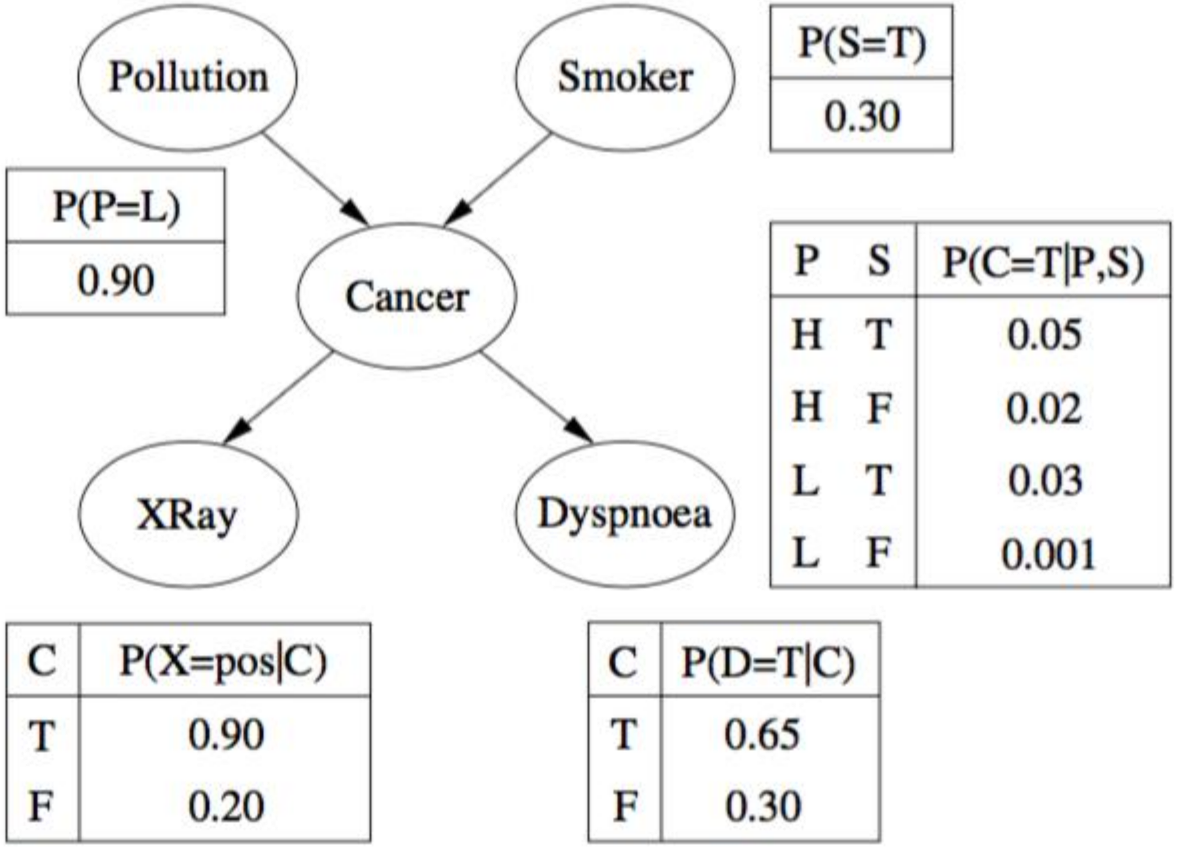
Prob	S
.15	T
.85	F



Prob	E	S	B
.45	T	T	T
.55	T	T	F
.05	T	F	T
.95	T	F	F
.95	F	T	T
.05	F	T	F
.55	F	F	T
.45	F	F	F

Prob	S	C
.8	T	T
.2	T	F
.4	F	T
.6	F	F

Prob	B	A
.75	T	T
.25	T	F
.05	F	T
.95	F	F



P(P=L)
0.90

P(S=T)
0.30

P	S	P(C=T|P,S)
H	T	0.05
H	F	0.02
L	T	0.03
L	F	0.001

C	P(X=pos|C)
T	0.90
F	0.20

C	P(D=T|C)
T	0.65
F	0.30

FIGURE 2.1

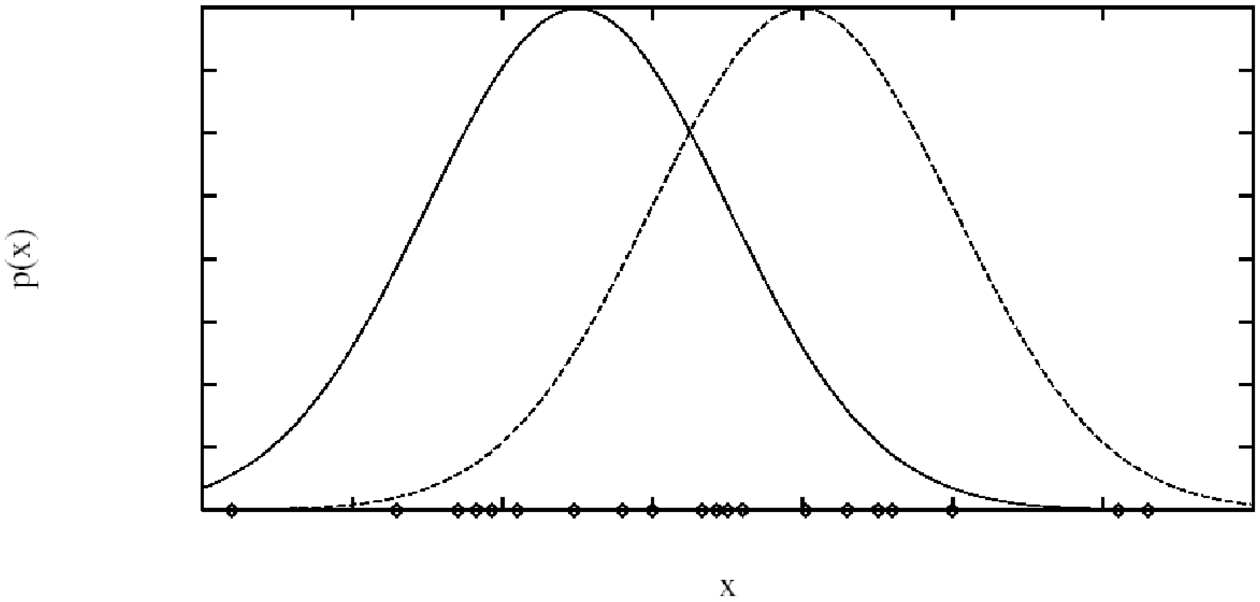
A BN for the lung cancer problem.

Lab Program

- **Write a program to construct a Bayesian network considering medical data. Use this model to demonstrate the diagnosis of heart patients using standard Heart Disease Data Set. You can use Python ML library API.**

4.10 EM Algorithm

Generating Data from Mixture of *k* Gaussians



- Each instance x generated by
  1. Choosing one of the *k* Gaussians with uniform probability
  2. Generating an instance at random according to that Gaussian

Gaussian Distribution

- Univariate Gaussian Distribution

$$\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\sigma}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

mean　　variance

- Multi-Variate Gaussian Distribution

$$\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi|\Sigma|)^{1/2}} \exp\left\{ -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}-\boldsymbol{\mu}) \right\}$$

mean　　covariance

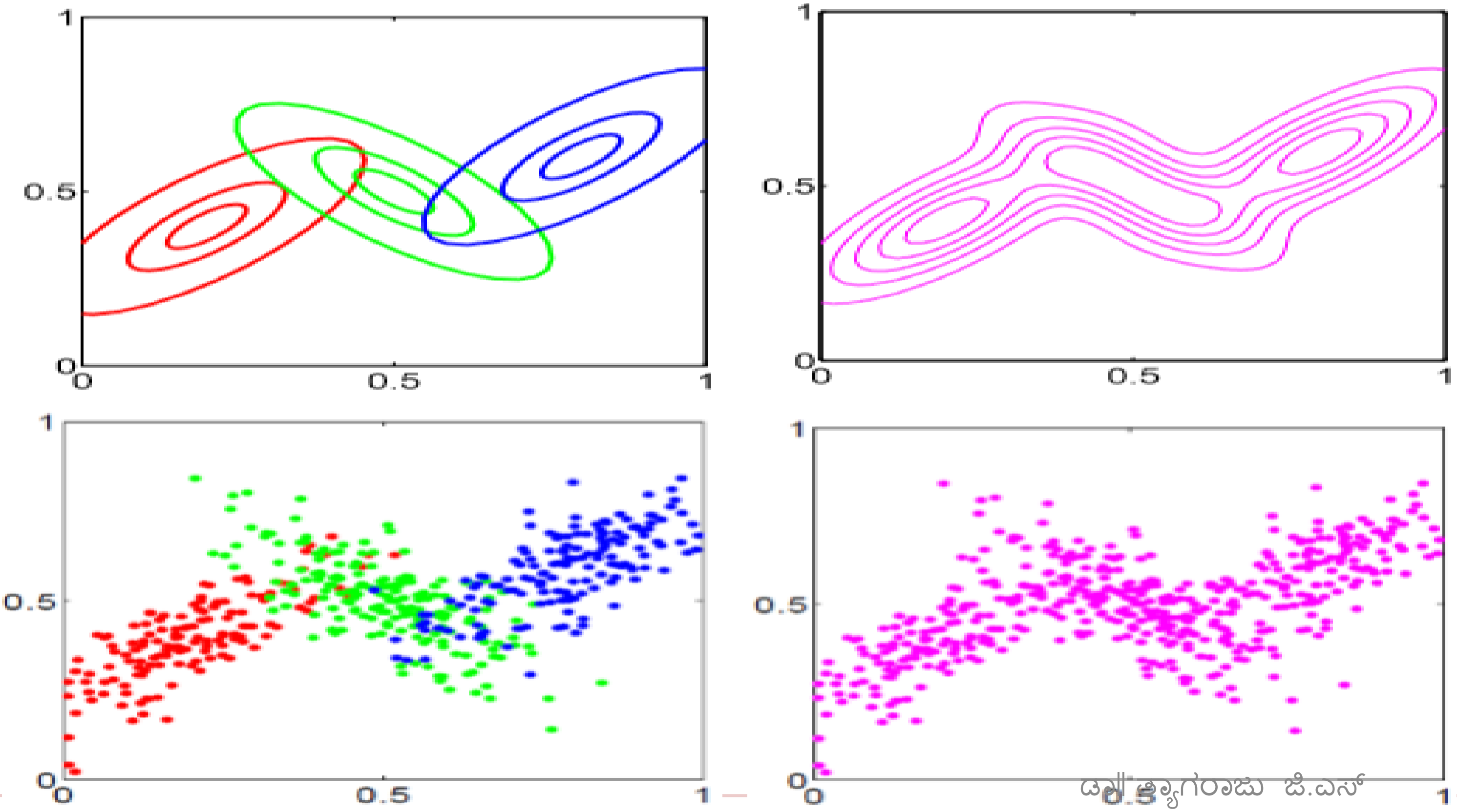
Gaussian Mixtures

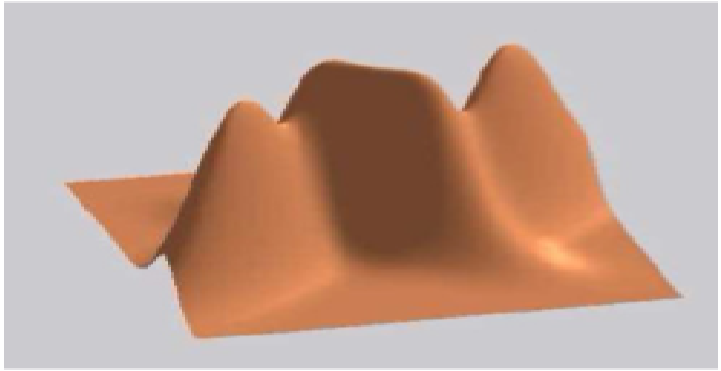
- Linear super-position of Gaussians

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \Sigma_k)$$

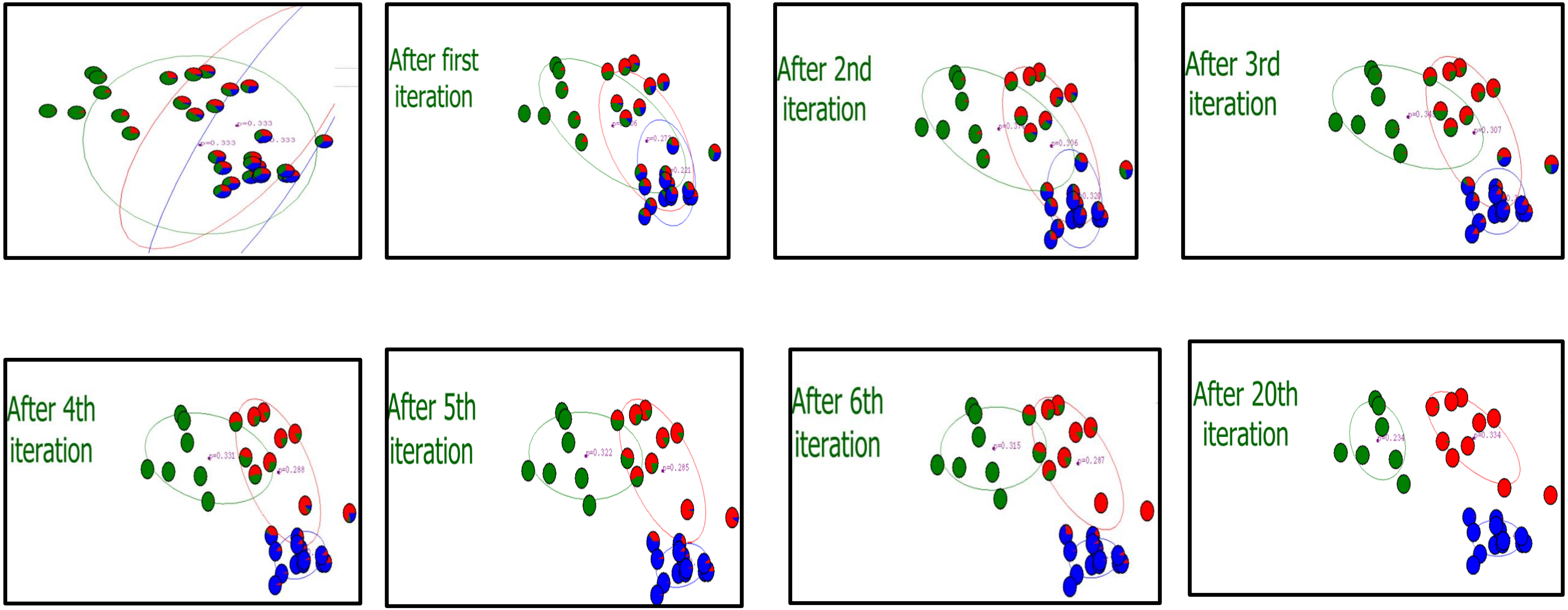
Number of Gaussians

Mixing coefficient: weightage for each Gaussian dist.





GMM : Example


After first iteration
After 2nd iteration
After 3rd iteration
After 4th iteration
After 5th iteration
After 6th iteration
After 20th iteration

Expectation Maximization (EM) Algorithm

- When to use:
  - Filling in missing data in samples
  - Discovering the value of latent variables
  - Estimating the parameters of HMMs
  - Estimating parameters of finite mixtures
  - Unsupervised learning of clusters
  - Semi-supervised classification and clustering

Expectation Maximization (EM) Algorithm

- EM is typically used to compute maximum likelihood estimates given incomplete samples.
- The EM algorithm estimates the parameters of a model iteratively.
  - Starting from some initial guess, each iteration consists of
    - an E step (Expectation step)
    - an M step (Maximization step)

EM Algorithm

- Given:
  - Instances from *X* generated by mixture of *k* Gaussian distributions
  - Unknown means $<\mu_1,\ldots,\mu_k>$ of the *k* Gaussians
  - Don't know which instance x_i was generated by which Gaussian
- Determine:
  - Maximum likelihood estimates of $<\mu_1,\ldots,\mu_k>$

- EM Algorithm:
- Pick random initial $h = <\mu_1, \mu_2>$ then iterate

  E step: Calculate the expected value $E[z_{ij}]$ of each ***hidden variable z**$_{ij}$*, assuming the current hypothesis $h = <\mu_1, \mu_2>$ holds.

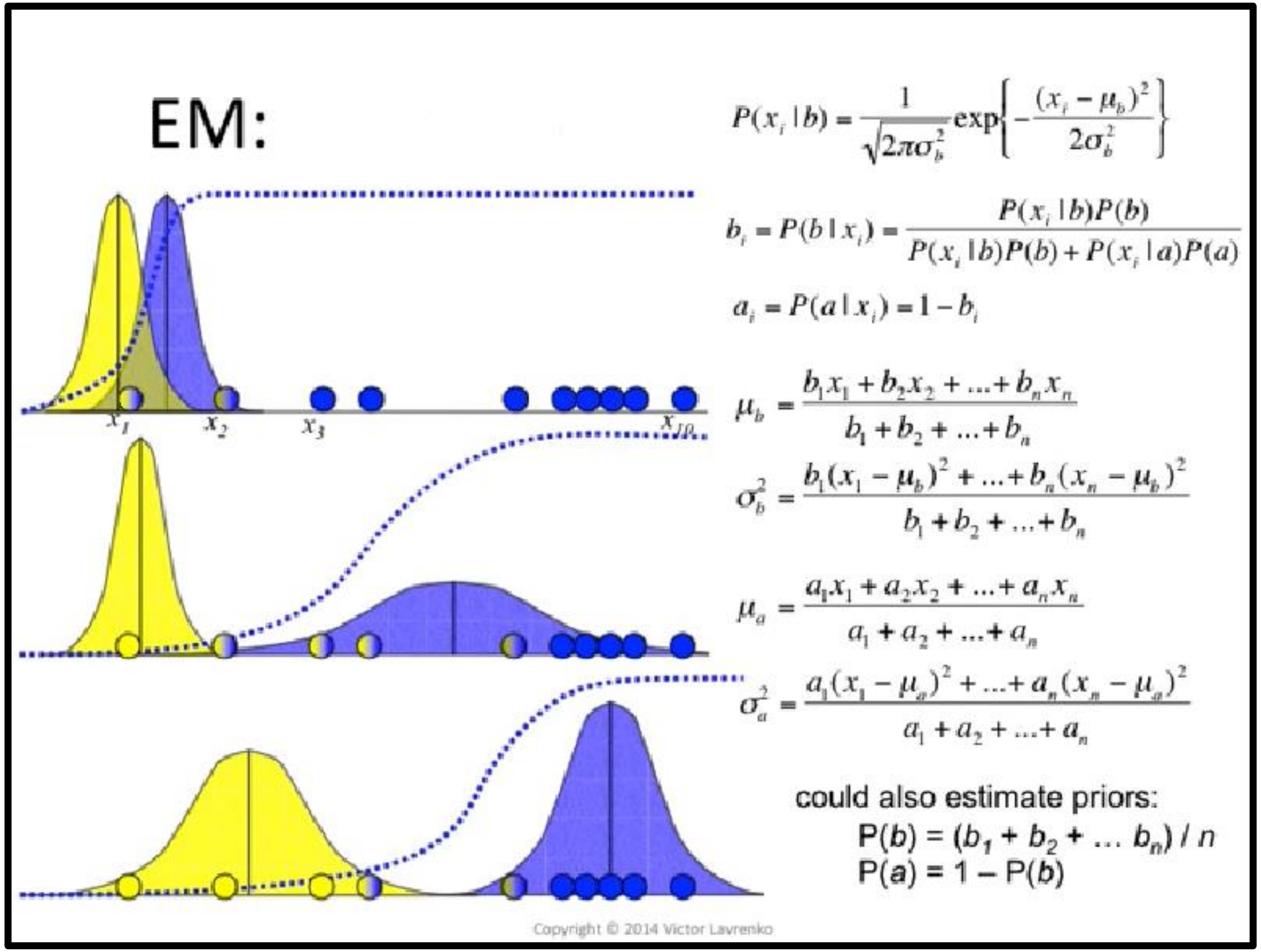
$$E[z_{ij}] = \frac{p(x = x_i|\mu = \mu_j)}{\Sigma_{n=1}^{2} p(x = x_i|\mu = \mu_n)} = \frac{e^{-\frac{1}{2\sigma^2}(x_i-\mu_j)^2}}{\Sigma_{n=1}^{2} e^{-\frac{1}{2\sigma^2}(x_i-\mu_n)^2}}$$

  M step: Calculate a new maximum likelihood hypothesis $h' = <\mu'_1, \mu'_2>$, assuming the value taken on by each hidden variable z_{ij} its expected value $E[z_{ij}]$ calculated above. Replace $h = <\mu_1, \mu_2>$ by $h' = <\mu'_1, \mu'_2>$.

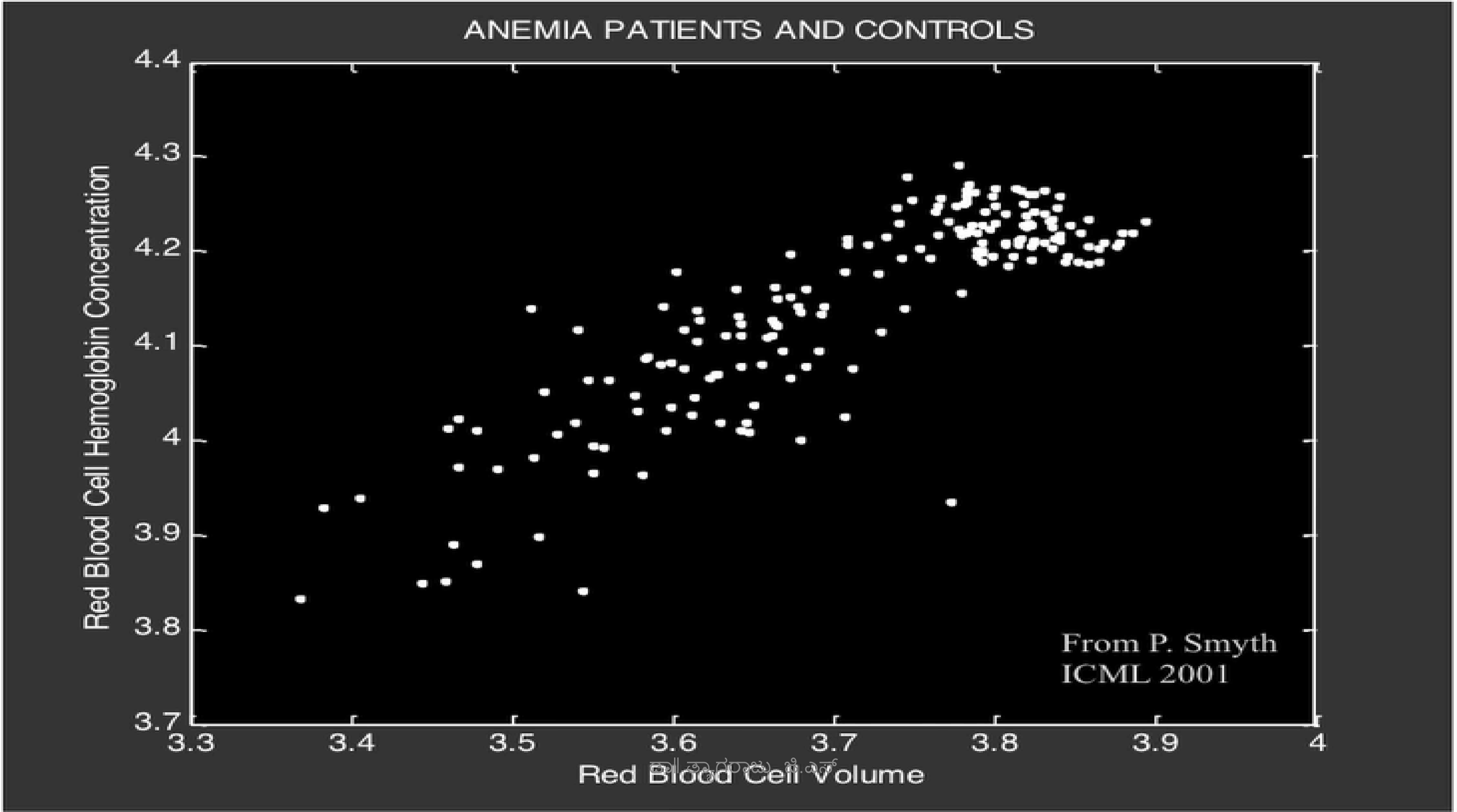
$$\mu_j \leftarrow \frac{\Sigma_{i=1}^{m} E[z_{ij}] \; x_i}{\Sigma_{i=1}^{m} E[z_{ij}]}$$

ಡಾ॥ ತ್ಯಾಗರಾಜು ಜಿ.ಎಸ್

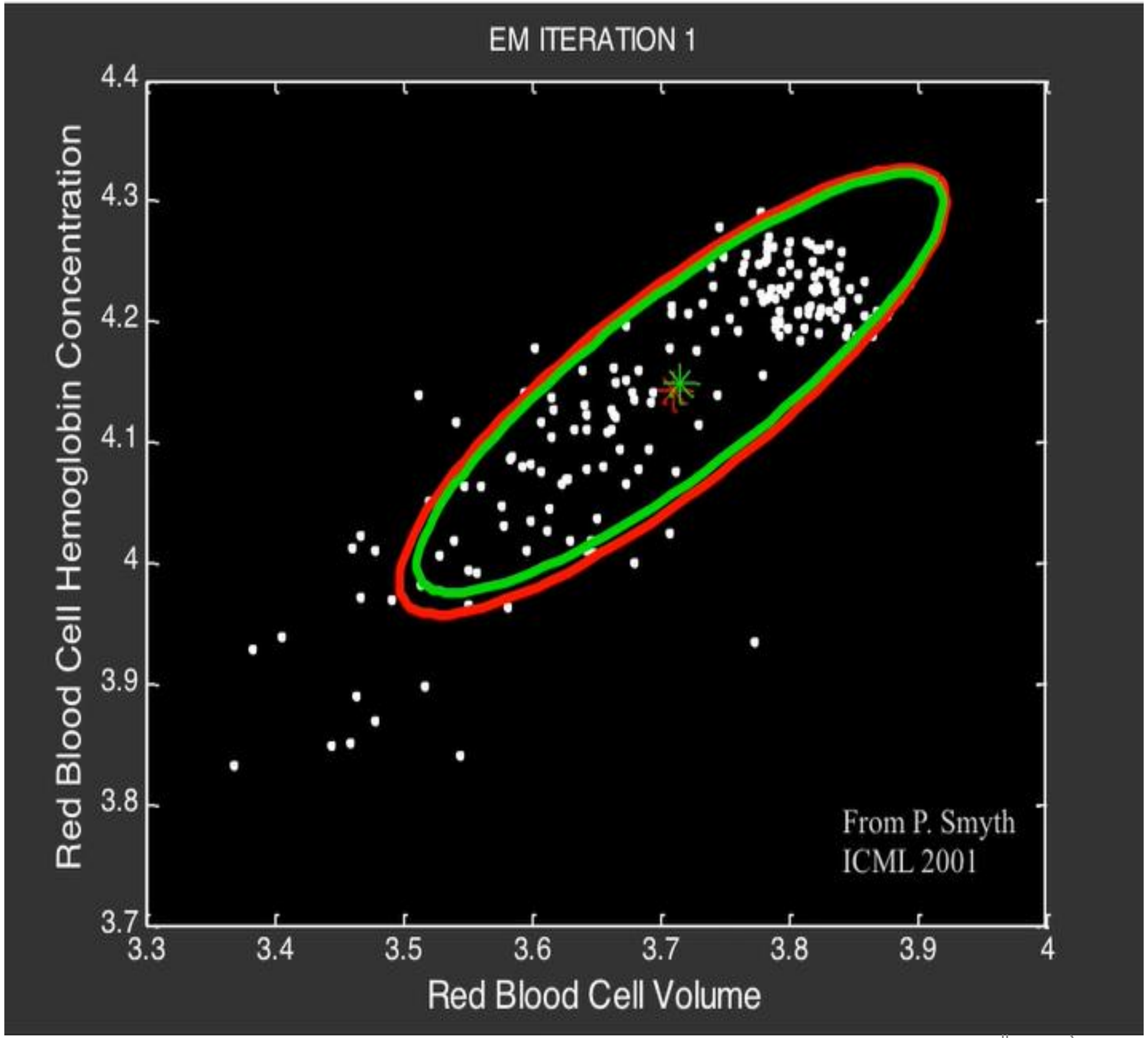
GMM :

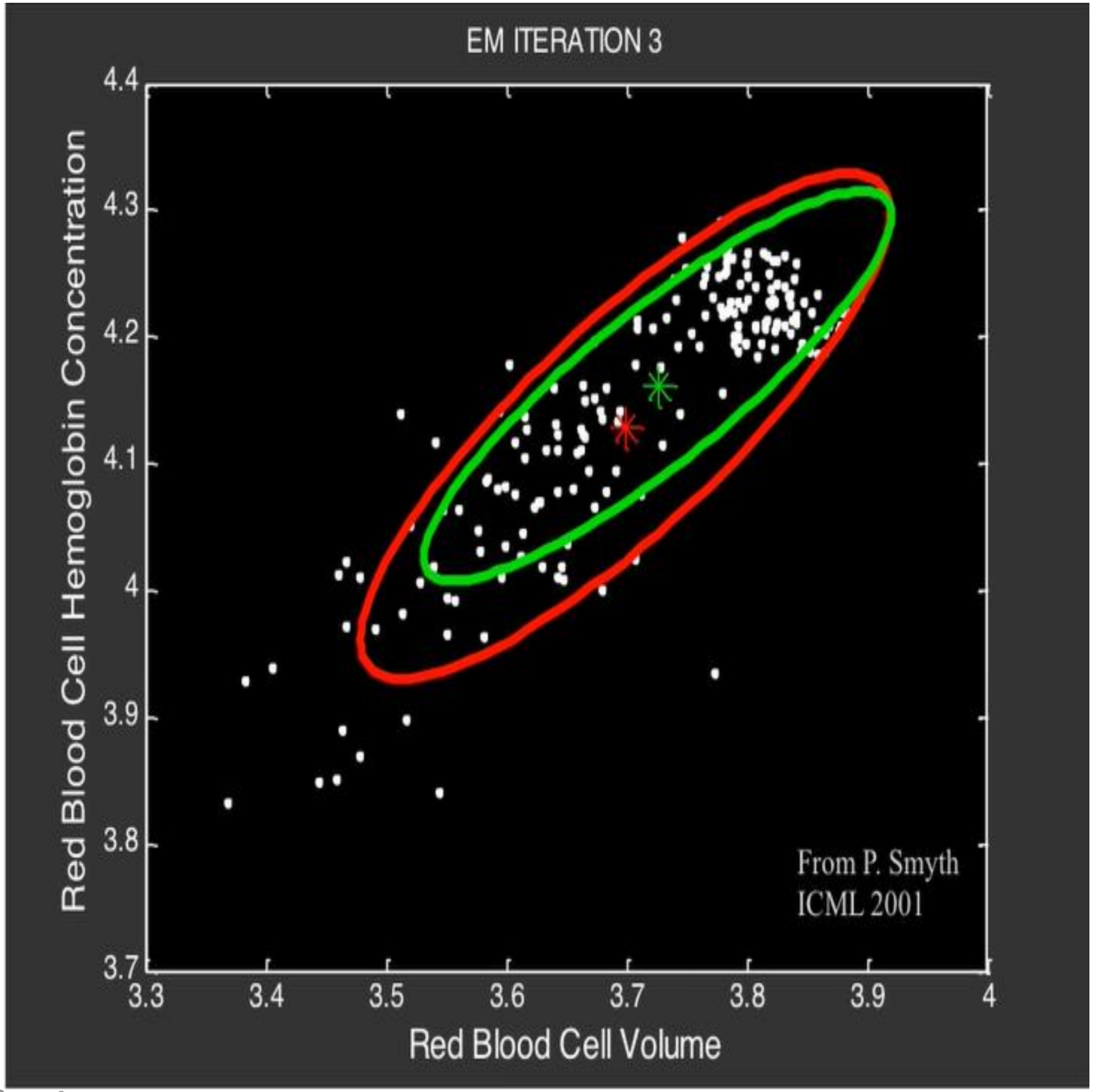

EM:
x_1 x_2 x_3 x_{10}
$P(x_i \mid b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left\{-\frac{(x_i - \mu_b)^2}{2\sigma_b^2}\right\}$
$b_i = P(b \mid x_i) = \frac{P(x_i \mid b)P(b)}{P(x_i \mid b)P(b) + P(x_i \mid a)P(a)}$
$a_i = P(a \mid x_i) = 1 - b_i$
$\mu_b = \frac{b_1 x_1 + b_2 x_2 + \ldots + b_n x_n}{b_1 + b_2 + \ldots + b_n}$
$\sigma_b^2 = \frac{b_1 (x_1 - \mu_b)^2 + \ldots + b_n (x_n - \mu_b)^2}{b_1 + b_2 + \ldots + b_n}$
$\mu_a = \frac{a_1 x_1 + a_2 x_2 + \ldots + a_n x_n}{a_1 + a_2 + \ldots + a_n}$
$\sigma_a^2 = \frac{a_1 (x_1 - \mu_a)^2 + \ldots + a_n (x_n - \mu_a)^2}{a_1 + a_2 + \ldots + a_n}$
could also estimate priors:
$P(b) = (b_1 + b_2 + \ldots b_n) / n$
$P(a) = 1 - P(b)$
Copyright © 2014 Victor Lavrenko

GMM : Example2

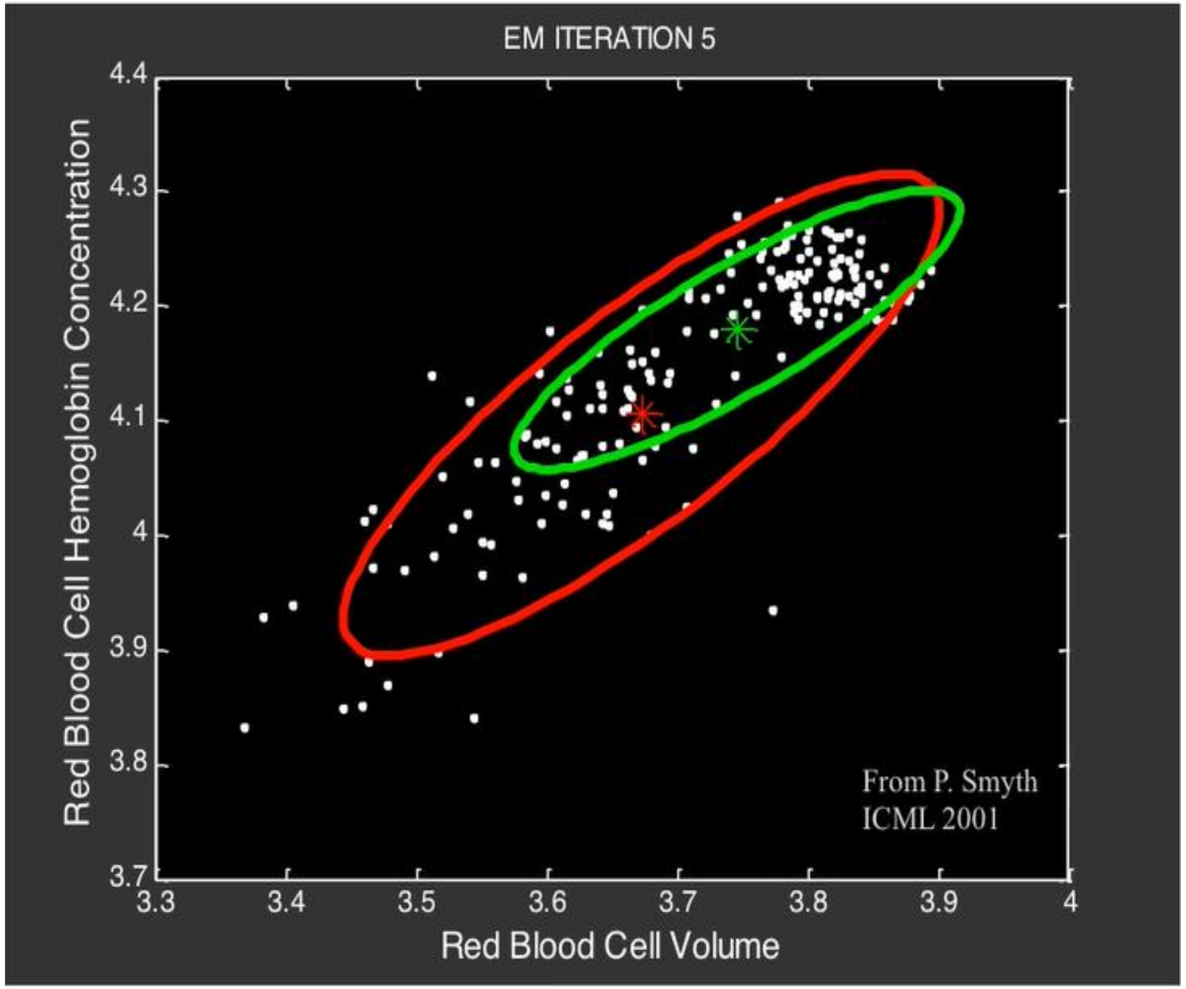


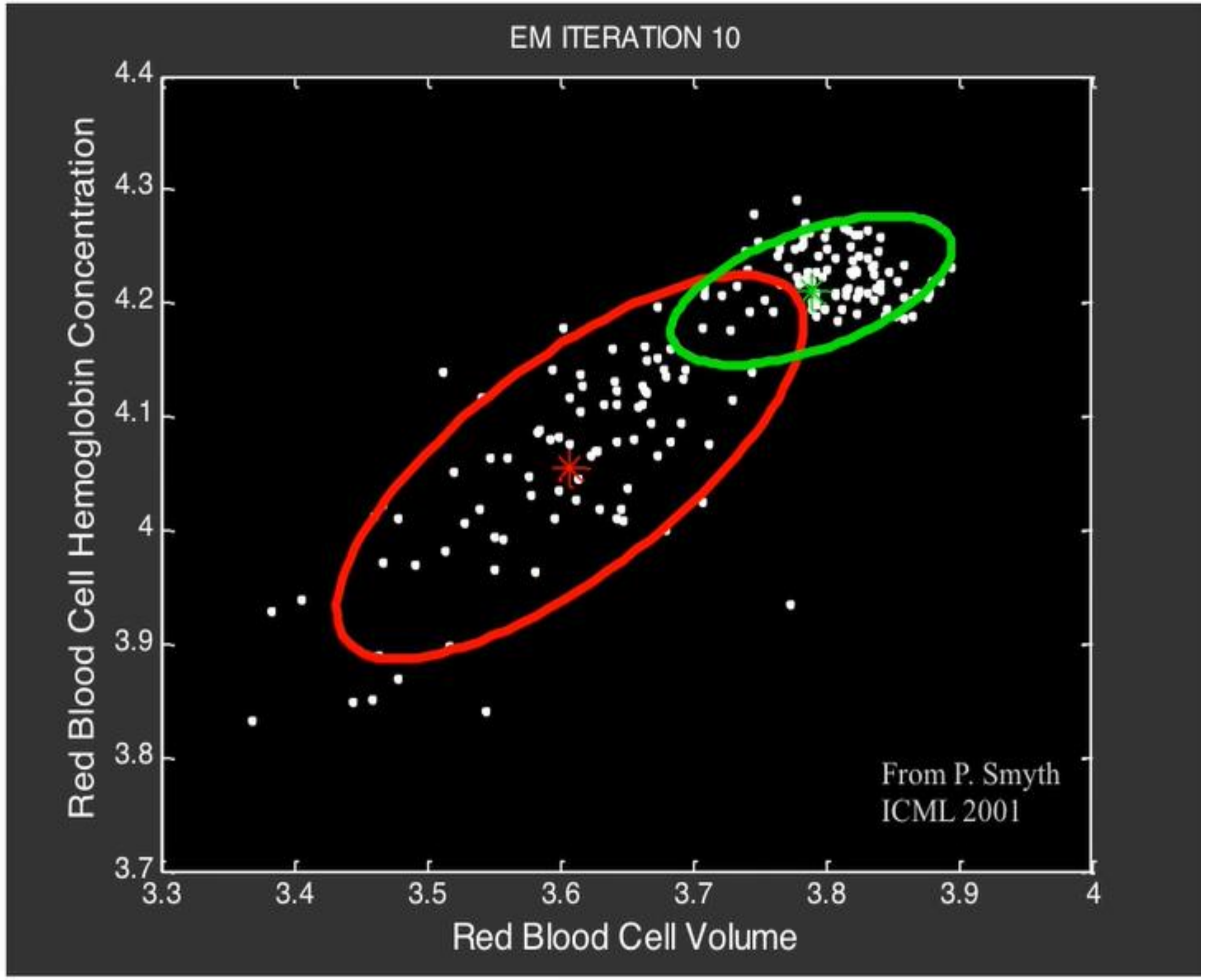
GMM : Example2



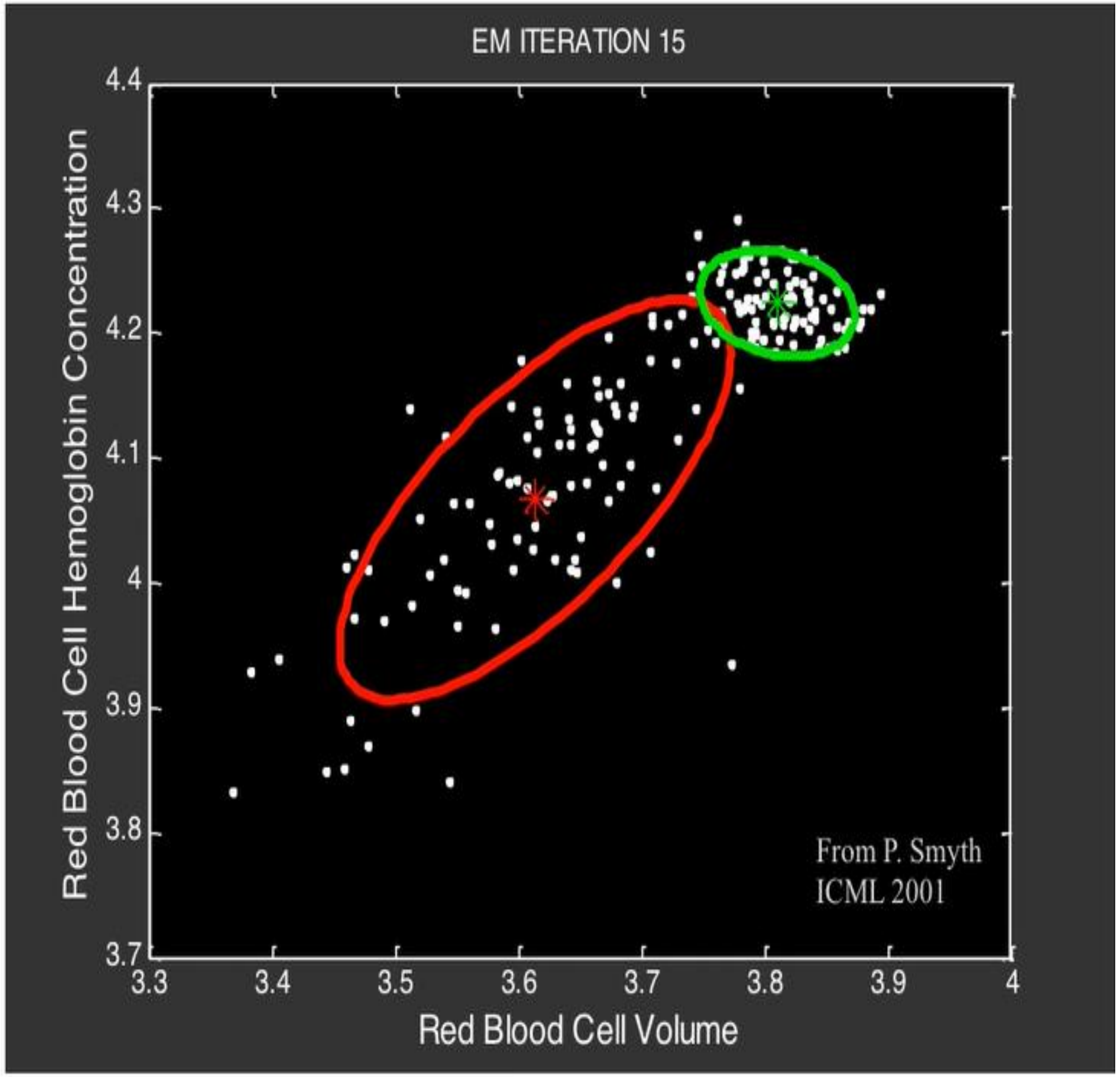


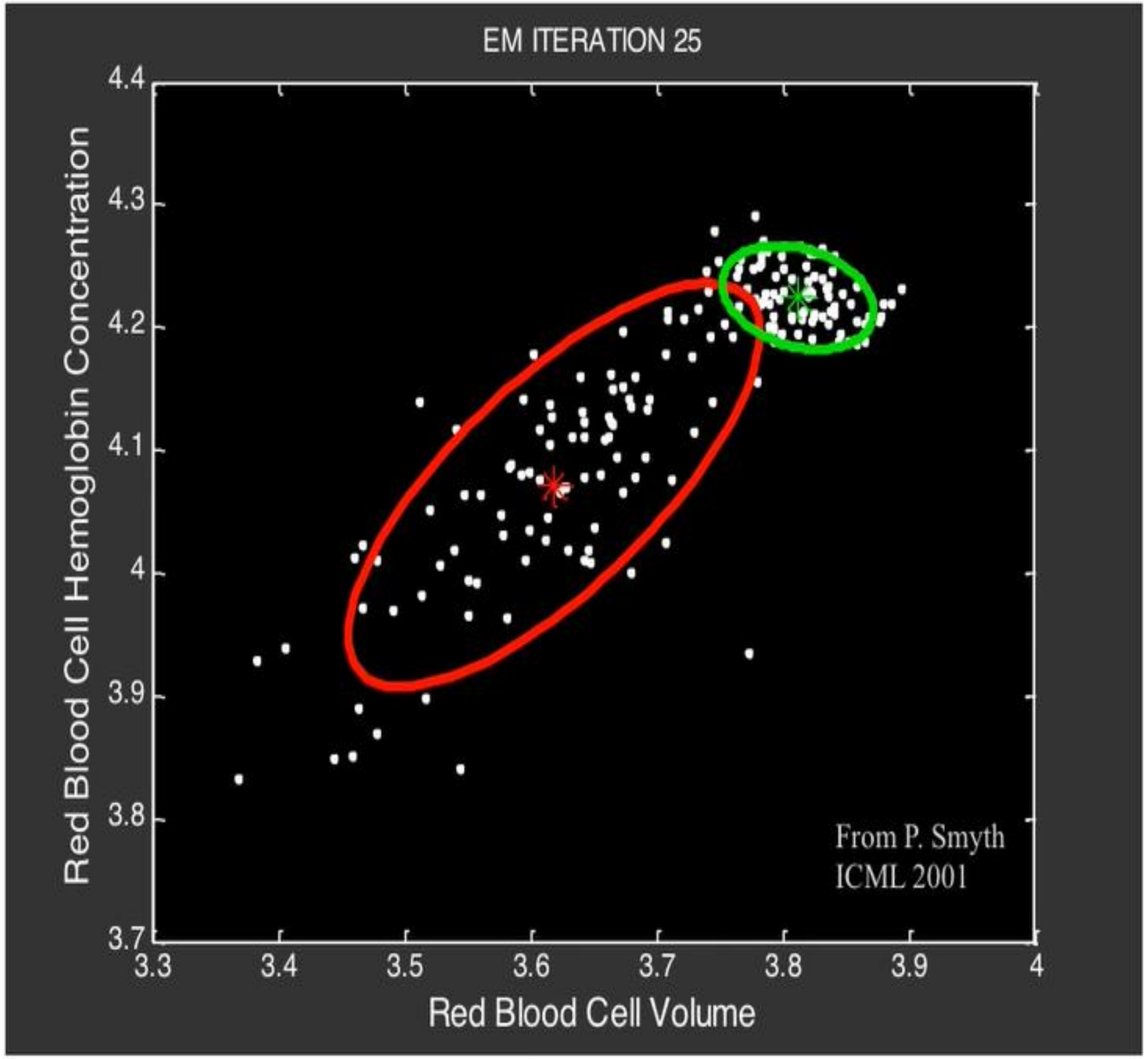
GMM : Example2





GMM : Example2





Reference

- Refer the text book “ Machine Learning “ Tom M Mitchell : Page No 191 to 194 for detailed explanation on EM Algorithm

4.11 K Means Algorithm

- 1. The sample space is initially partitioned into K clusters and the observations are randomly assigned to the clusters.
- 2. For each sample:
  - Calculate the distance from the observation to the centroid of the cluster.
  - IF the sample is closest to its own cluster THEN leave it ELSE select another cluster.
- 3. Repeat steps 1 and 2 untill no observations are moved from one cluster to another

Basic Algorithm of K-means

Algorithm 1 Basic K-means Algorithm.

```
Select K points as the initial centroids.
repeat
    Form K clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change
```

Distance functions

Euclidean: $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan: $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski: $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

Details of K-means

1. Initial centroids are often chosen randomly.
   - *Clusters produced vary from one run to another*
2. The centroid is (typically) the mean of the points in the cluster.
3. 'Closeness' is measured by **Euclidean distance**, cosine similarity, correlation, etc.
4. K-means will converge for common similarity measures mentioned above.
5. Most of the convergence happens in the first few iterations.
   - *Often the stopping condition is changed to 'Until relatively few points change clusters'*

Euclidean Distance

$$d(i,j) = \sqrt{|x_{i_1} - x_{j_1}|^2 + |x_{i_2} - x_{j_2}|^2 + ... + |x_{i_p} - x_{j_p}|^2}$$

A simple example: Find the distance between two points, the original and the point (3,4)

$$d_E(O, A) = \sqrt{3^2 + 4^2} = 5$$

Update Centroid

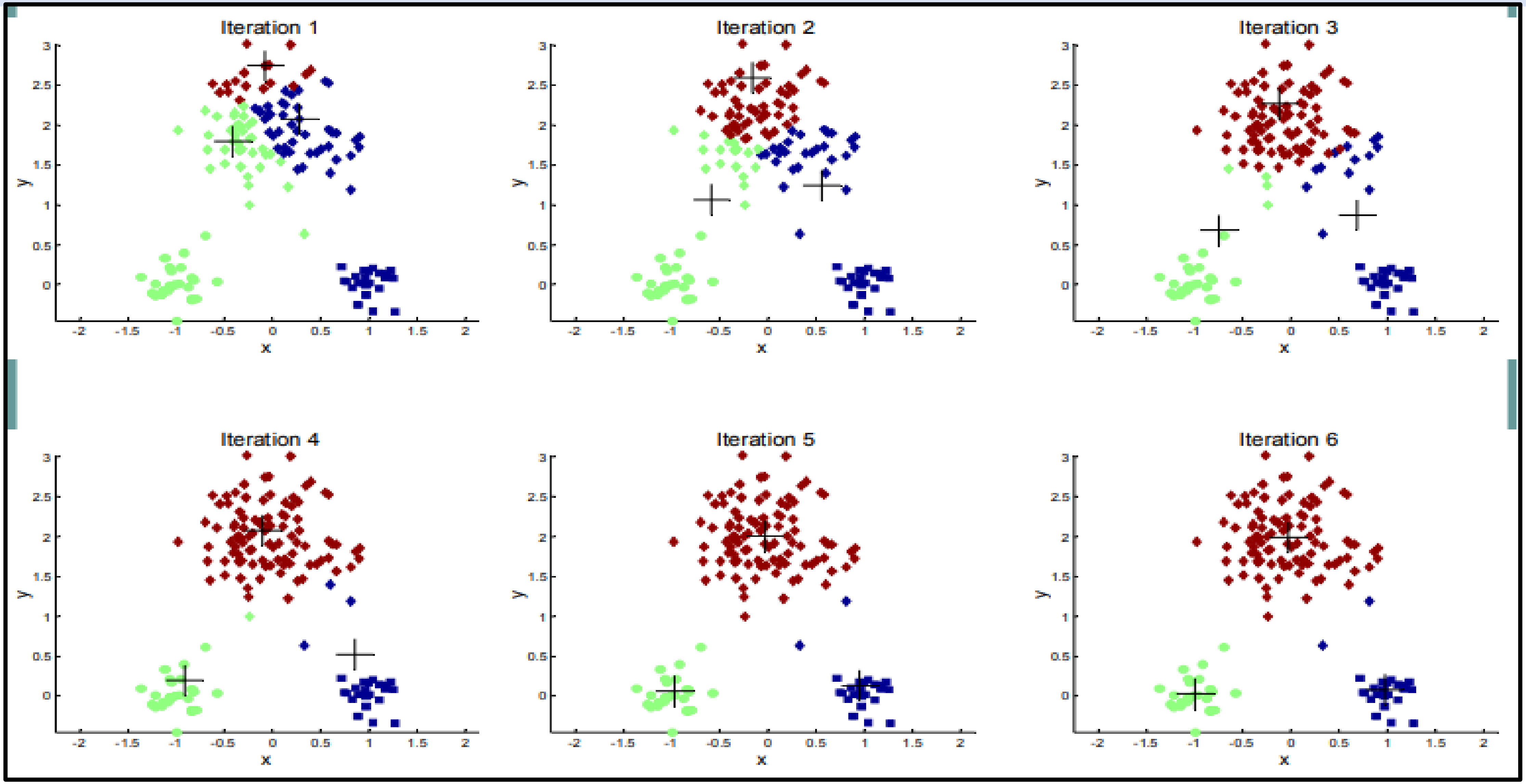
We use the following equation to calculate the n dimensional centroid point amid k n-dimensional points

$$CP(x_1, x_2, \ldots, x_k) = \left(\frac{\sum_{i=1}^{k} x1st_i}{k}, \frac{\sum_{i=1}^{k} x2nd_i}{k}, \ldots, \frac{\sum_{i=1}^{k} xnth_i}{k}\right)$$

Example: Find the centroid of 3 2D points, (2,4), (5,2) and (8,9)

$$CP = \left(\frac{2+5+8}{3}, \frac{4+2+9}{3}\right) = (5,5)$$

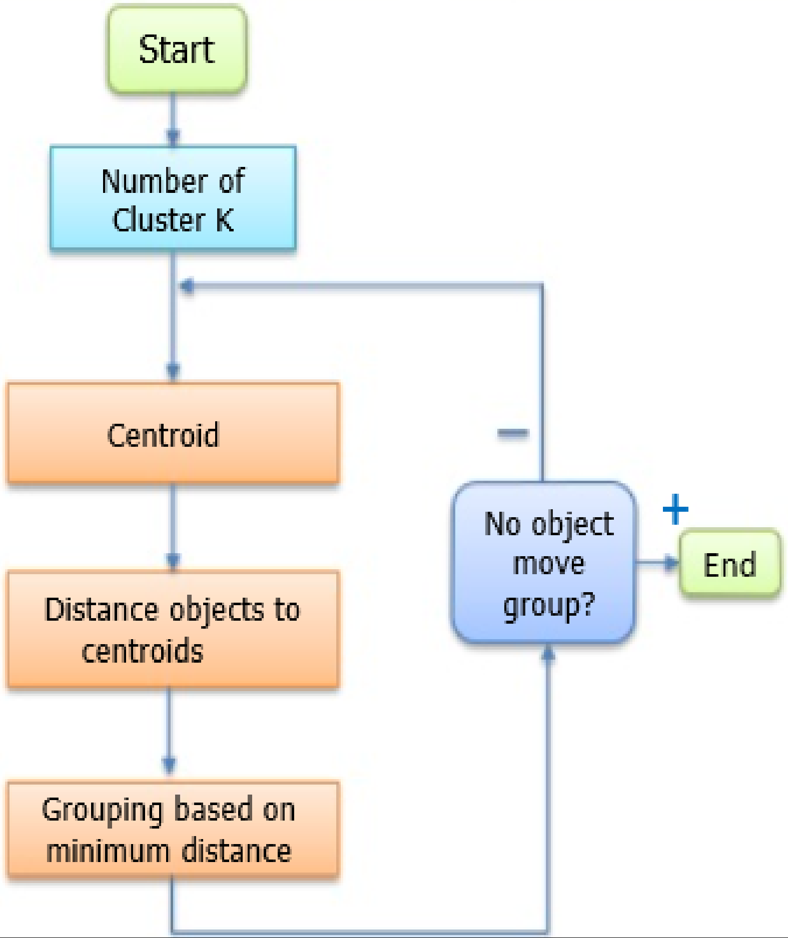
Examples of K Means



How the K-Mean Clustering algorithm works?

$$\left[\frac{x_1 + x_2 + x_3}{3} , \frac{y_1 + y_2 + y_3}{3} \right]$$

Process Flow of K-means



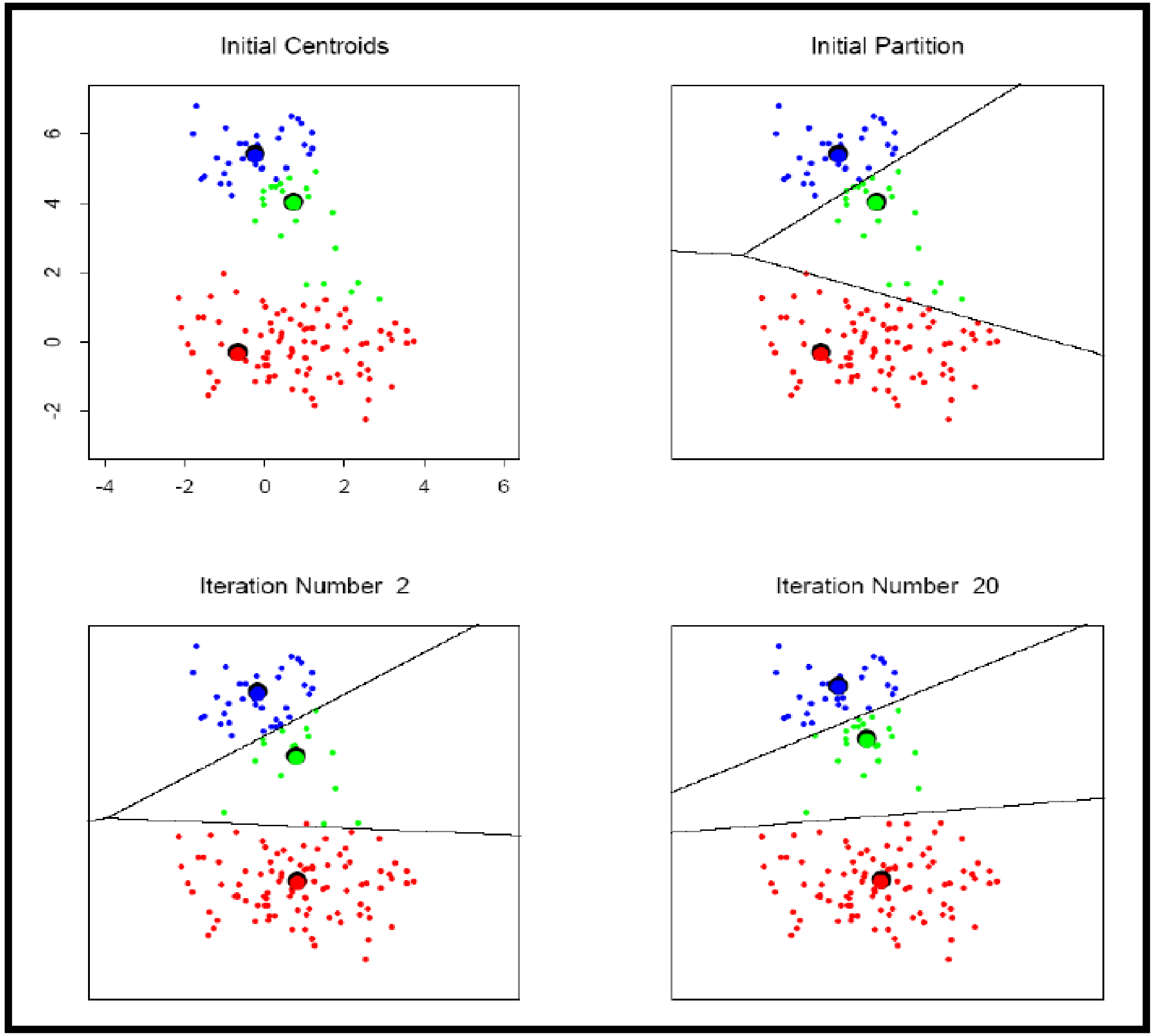
Iterate until *stable* (cluster centers converge):

1. Determine the centroid coordinate.

2. Determine the distance of each object to the centroids.

3. Group the object based on minimum distance (find the closest centroid)

K-means clustering example



Lab Program

- Apply EM algorithm to cluster a set of data stored in a .CSV file. Use the same data set for clustering using k-Means algorithm. Compare the results of these two algorithms and comment on the quality of clustering. You can add Python ML library classes/API in the program.

Derivation of the k –means Algorithm

- Refer the text book “ Machine Learning “ Tom M Mitchell : Page No 195 to 196.

End of the Module 4.