MCSE 666:Pattern and Speech Recognition

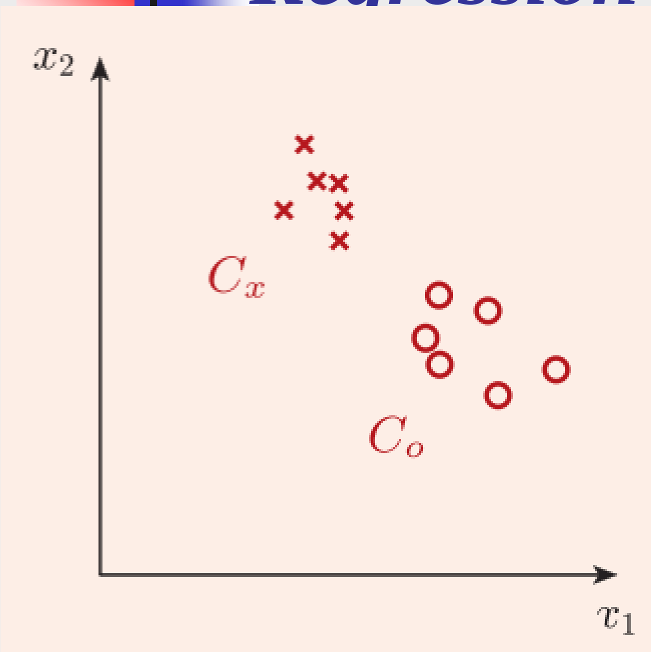
Classifier Learning

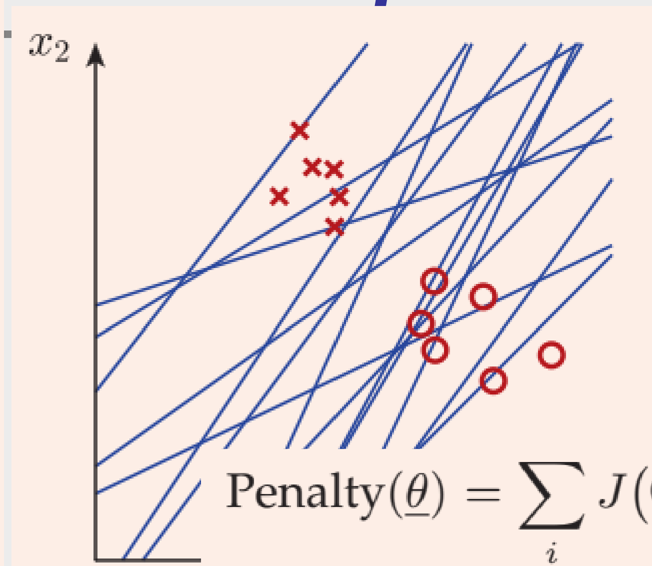
(K-Nearest Neighbors and Naive Bayes)

Dr. Md. Aminul Haque Akhand

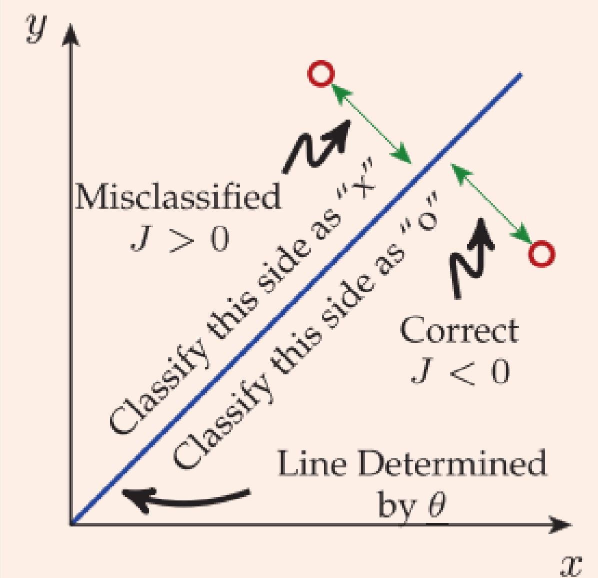
Dept. of CSE, SUB

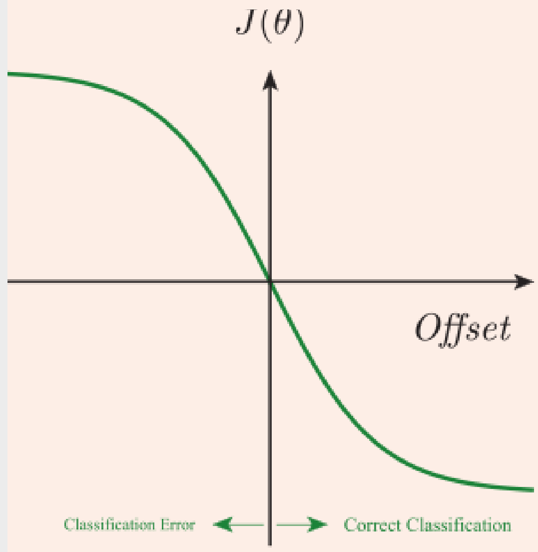
Regression and Classification

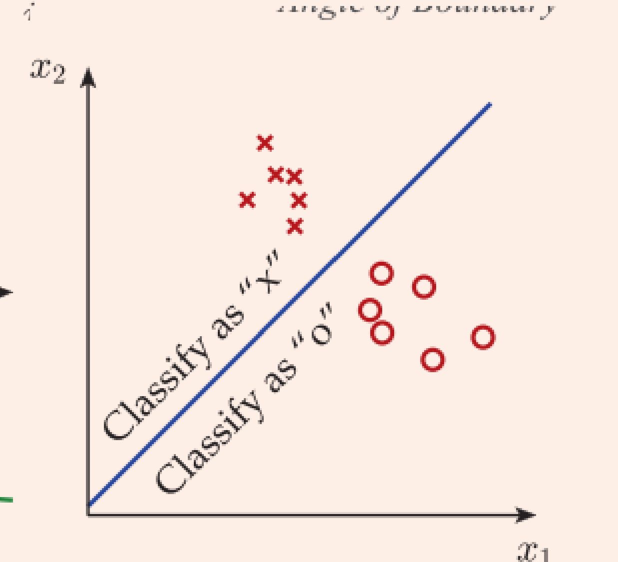




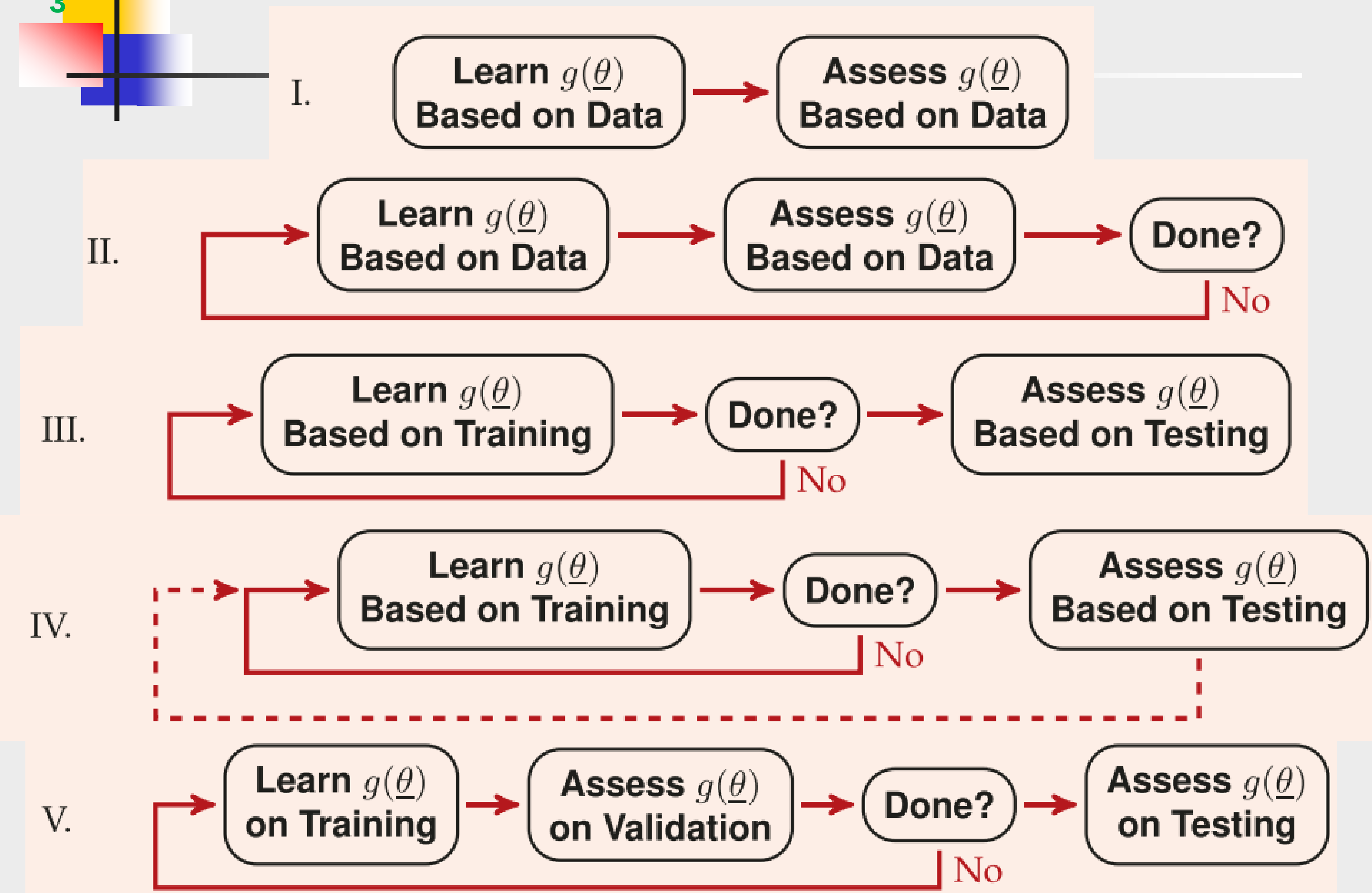
$$\text{Penalty}(\underline{\theta}) = \sum_i J\big(\text{Offset from line } \underline{\theta} \text{ to } \underline{x}_i \in C_x\big) + \sum_j J\big(\text{Offset from line } \underline{\theta} \text{ to } \underline{x}_j \in C_o\big)$$







Use of Data in Learning



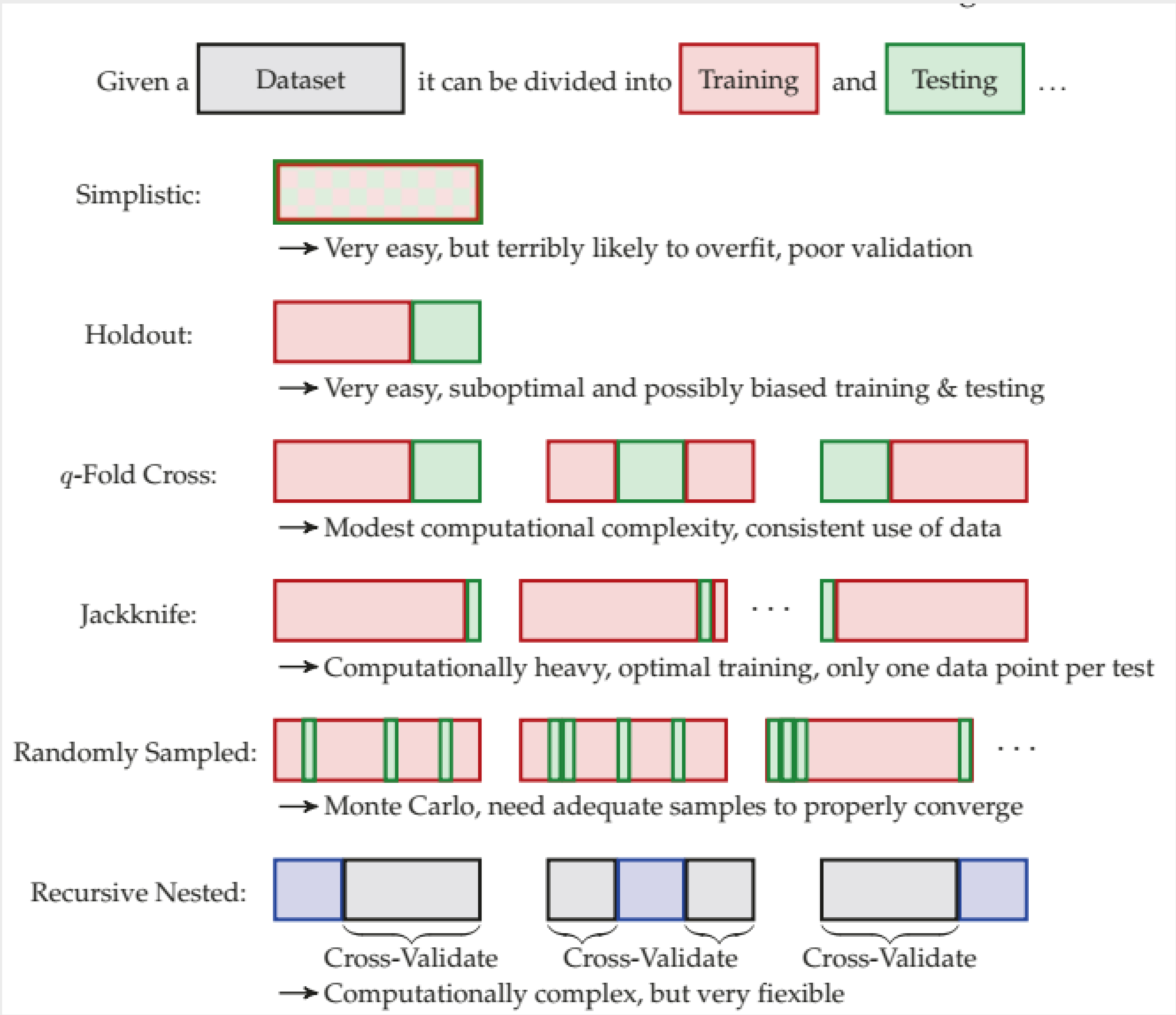
Classifier Evaluation / Performance Measure

1. Test Set Accuracy
2. Test Set Error Rate
3. Confusion Matrix

Given a classifier g, the corresponding confusion matrix from (3.19),

$$
\begin{array}{cc}
 & \text{Is classified as} \ldots \\
 & \begin{array}{cccc} C_1 & C_2 & \cdots & C_K \end{array} \\
\text{True Class} \quad \begin{array}{c} C_1 \\ \vdots \\ C_K \end{array} & \left[\begin{array}{cccc}
S_g(C_1|C_1) & S_g(C_2|C_1) & \cdots & S_g(C_K|C_1) \\
\vdots & \vdots & & \vdots \\
S_g(C_1|C_K) & S_g(C_2|C_K) & \cdots & S_g(C_K|C_K)
\end{array}\right]
\end{array}
$$

Classifier Validation



K-Nearest Neighbor (KNN) Classifier

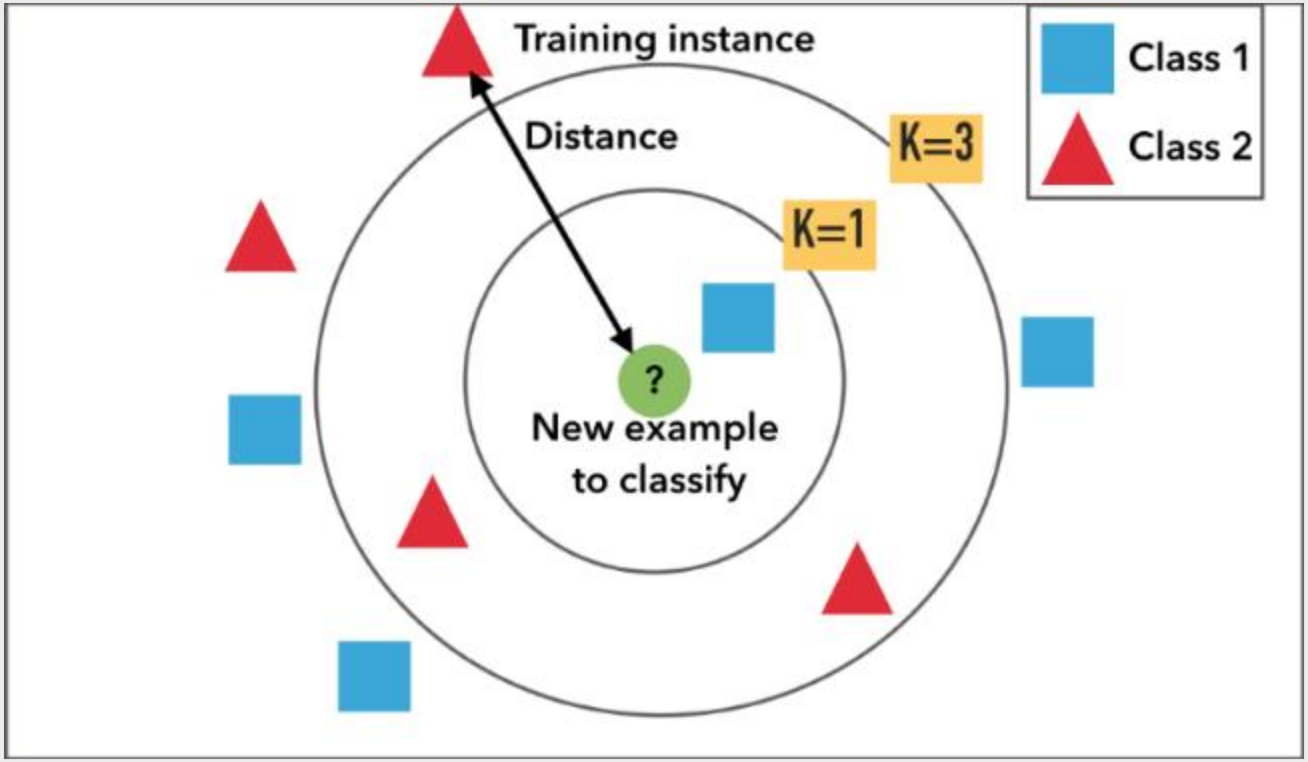
KNN Overview

- *KNN algorithm is one of the simplest classification algorithm*
- *Non-parametric*

  It does not make any assumptions on the underlying data distribution
- *Lazy learning algorithm.*
  - *there is no explicit training phase or it is very minimal.*
  - *also means that the training phase is pretty fast .*
  - *Lack of generalization means that KNN keeps all the training data.*
- *Its purpose is to use a database in which the data points are separated into several classes to predict the classification of a new sample point.*

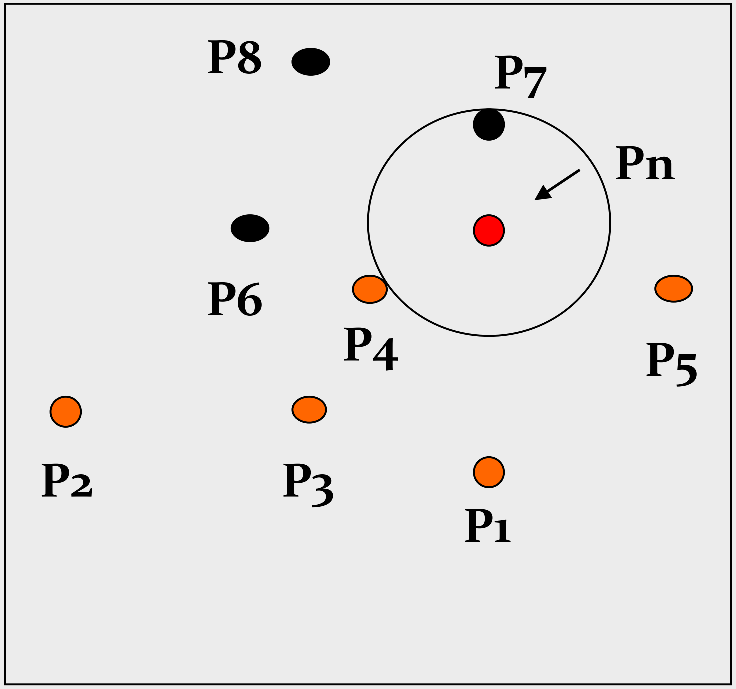
KNN

- *KNN Algorithm is based on feature similarity*
- *How closely out-of-sample features resemble our training set determines how we classify a given data point*

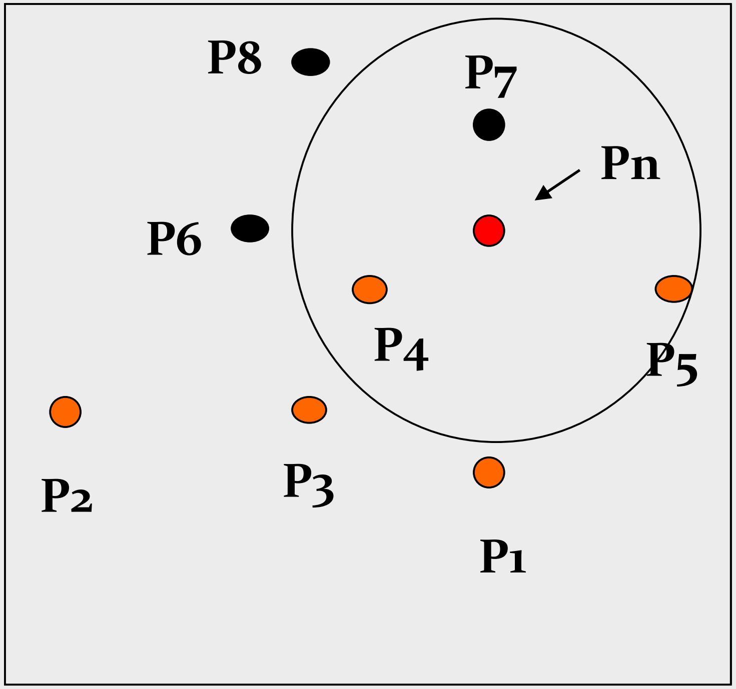


KNN for K = 1 or K = 3

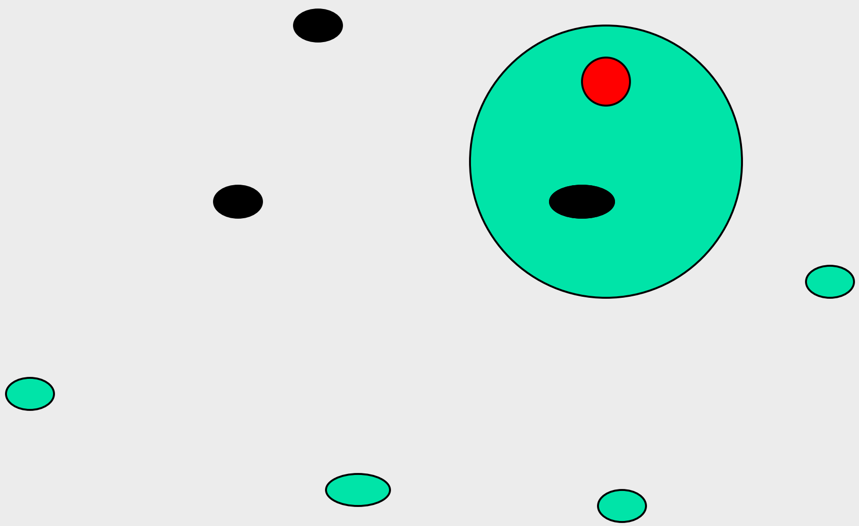
k = 1



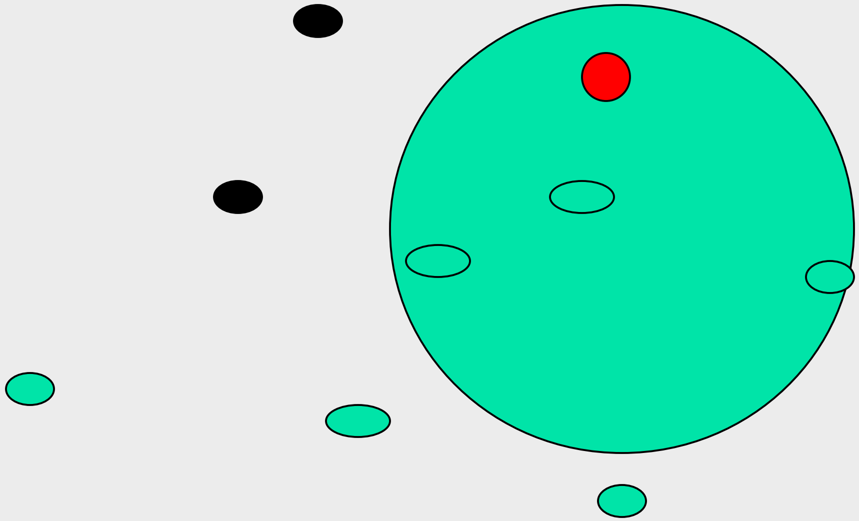
k = 3



1-Nearest Neighbor



3-Nearest Neighbor



Classification Steps

- **Training phase:** a model is constructed from the training instances.
  - classification algorithm finds relationships between predictors and targets
  - relationships are summarised in a model

- **Testing phase:** test the model on a test sample whose class labels are known but not used for training the model

- **Usage phase:** use the model for classification on new data whose class labels are unknown

https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_knn_algorithm_finding_nearest_neighbors.htm

KNN Features

- All instances correspond to points in an n-dimensional Euclidean space
- Classification is delayed till a new instance arrives
- Classification done by comparing feature vectors of the different points
- Target function may be discrete or real-valued

➢ It uses the local neighborhood to obtain a prediction

➢ The K memorized examples more similar to the one that is being classified are retrieved

KNN Features

- A distance function is needed to compare the examples similarity

Euclidean :

$$d(x,y) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2}$$

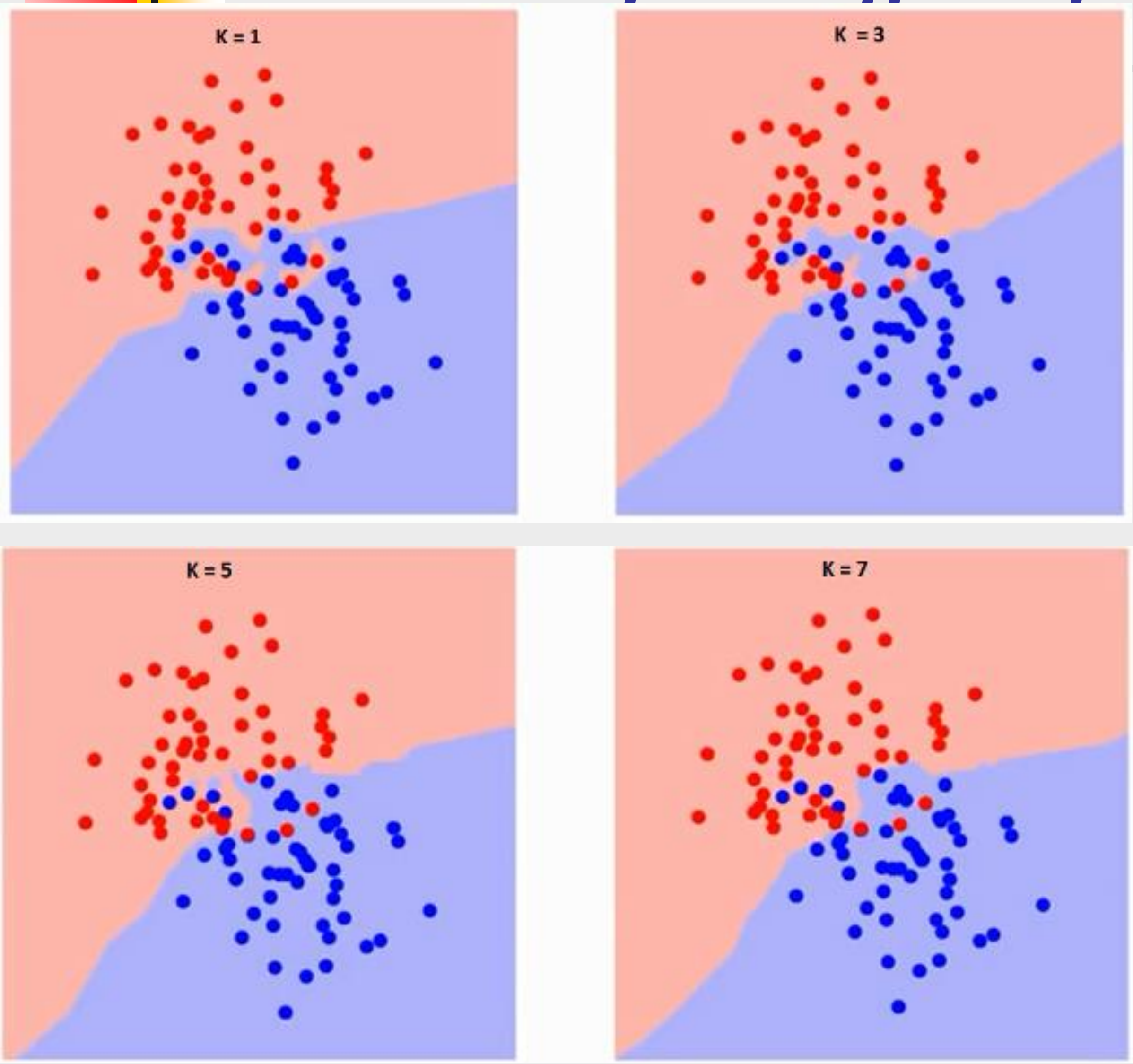
Manhattan / city - block :

$$d(x,y) = \sum_{i=1}^{m}\left|x_i - y_i\right|$$

- This means that if we change the distance function, we change how examples are classified

Learning is very simple but Classification is time consuming

KNN Classifier : Effect of K Value



The boundary becomes smoother with increasing value of K.

With K increasing to infinity it finally becomes all blue or all red depending on the total majority.

https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/#h-what-is-knn-k-nearest-neighbor-algorithm

KNN Classifier: Conclusions

Pros of KNN

•Very simple algorithm to understand and interpret.

•Very useful for nonlinear data because there is no assumption about data in KNN.

•Versatile algorithm as we can use it for classification as well as regression.

•High accuracy but there are much better supervised learning models than KNN.

Cons of KNN

•Computationally a bit expensive algorithm because it stores all the training data.

•High memory storage required as compared to other supervised learning algorithms.

•Prediction is slow in case of big N.

•Very sensitive to the scale of data as well as irrelevant features.

Applications of KNN

Banking System: to predict weather an individual is fit for loan approval? Does that individual have the characteristics similar to the defaulters one?

Calculating Credit Ratings: can be used to find an individual's credit rating by comparing with the persons having similar traits.

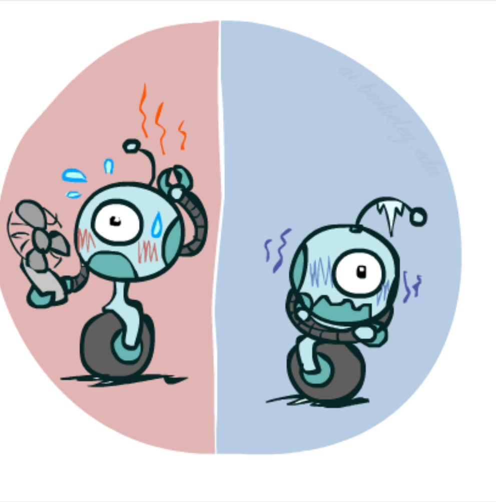
Other areas: can be used are Speech Recognition, Handwriting Detection, Image Recognition and Video Recognition.

Naïve Bayes (NB) Classifier

Probability Distributions

Associate a probability with each value

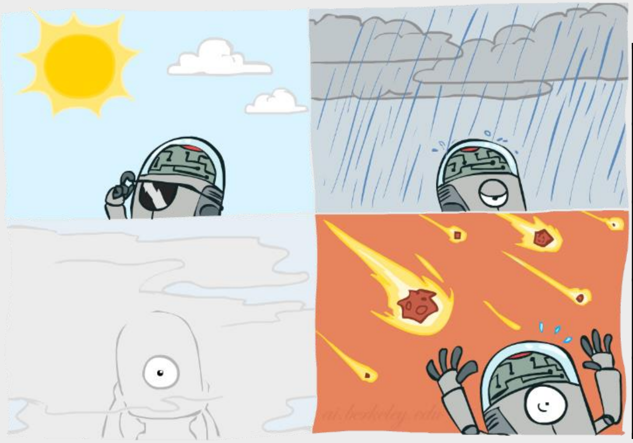
Temperature:



$P(T)$

T	P
hot	0.5
cold	0.5

Weather:



$P(W)$

W	P
sun	0.6
rain	0.1
fog	0.3
meteor	0.0

Probabilistic Models

- A probabilistic model is a joint distribution over a set of random variables

- Probabilistic models:
  - (Random) variables with domains
  - Assignments are called *outcomes*
  - Joint distributions: say whether assignments (outcomes) are likely
  - *Normalized:* sum to 1.0
  - Ideally: only certain variables directly interact

Distribution over T, W

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

Marginal Distributions

- Marginal distributions are sub-tables which eliminate variables
- Marginalization (summing out): Combine collapsed rows by adding

$P(T,W)$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

$$P(t) = \sum_s P(t,s)$$

$P(T)$

T	P
hot	0.5
cold	0.5

$$P(s) = \sum_t P(t,s)$$

$P(W)$

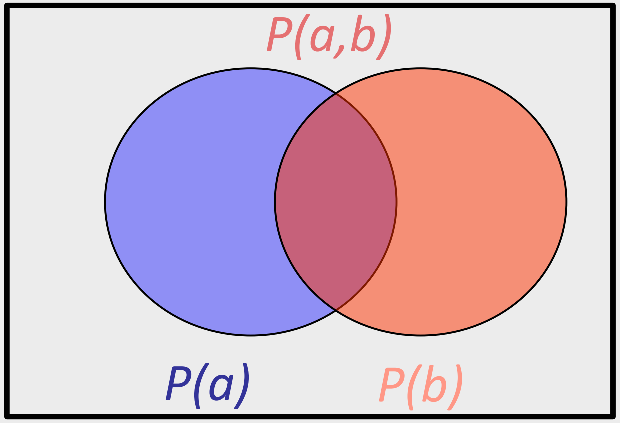
W	P
sun	0.6
rain	0.4

$$P(X_1 = x_1) = \sum_{x_2} P(X_1 = x_1, X_2 = x_2)$$

Conditional Probabilities

- A simple relation between joint and conditional probabilities
  - In fact, this is taken as the *definition* of a conditional probability

$$P(a|b) = \frac{P(a,b)}{P(b)}$$



$P(T,W)$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

$$P(W=s|T=c) = \frac{P(W=s,T=c)}{P(T=c)} = \frac{0.2}{0.5}$$

$$= 0.4$$

$$= P(W=s,T=c) + P(W=r,T=c)$$

$$= 0.2 + 0.3 \quad = 0.5$$

Normalization Trick

$$P(a|b) = \frac{P(a,b)}{P(b)}$$

$$P(W=s|T=c) = \frac{P(W=s,T=c)}{P(T=c)}$$
$$= \frac{P(W=s,T=c)}{P(W=s,T=c)+P(W=r,T=c)}$$
$$= \frac{0.2}{0.2+0.3} = 0.4$$

$P(T, W)$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

$\longrightarrow$

$P(W|T=c)$

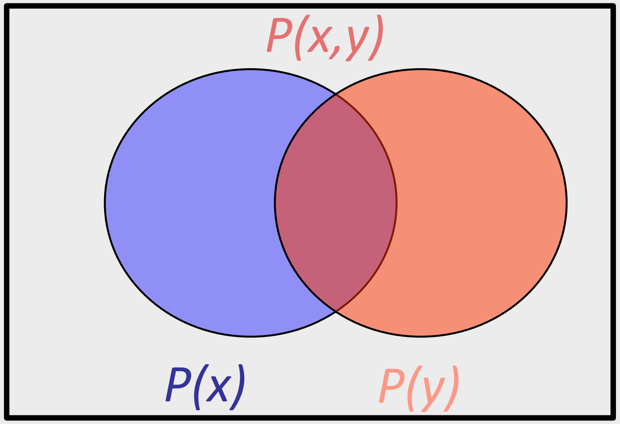
W	P
sun	0.4
rain	0.6

$$P(W=r|T=c) = \frac{P(W=r,T=c)}{P(T=c)}$$
$$= \frac{P(W=r,T=c)}{P(W=s,T=c)+P(W=r,T=c)}$$
$$= \frac{0.3}{0.2+0.3} = 0.6$$

Bayes' Rule

$$P(a|b) = \frac{P(a,b)}{P(b)}$$

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



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

$$P(x,y) = P(x|y)P(y) = P(y|x)P(x)$$

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

Bayes' Rule

Prior, conditional and joint probability

- Prior probability: $P(X)$
- Conditional probability: $P(X_1 | X_2), P(X_2 | X_1)$
- Joint probability: $\mathbf{X} = (X_1, X_2), P(\mathbf{X}) = P(X_1, X_2)$
- Relationship: $P(X_1, X_2) = P(X_2 | X_1)P(X_1) = P(X_1 | X_2)P(X_2)$
- Independence: $P(X_2 | X_1) = P(X_2), P(X_1 | X_2) = P(X_1), P(X_1, X_2) = P(X_1)P(X_2)$

Bayesian Rule

$$P(C | \mathbf{X}) = \frac{P(\mathbf{X} | C)P(C)}{P(\mathbf{X})} \qquad Posterior = \frac{Likelihood \times Prior}{Evidence}$$

Naïve Bayes (NB) Classifier

- Establishing a probabilistic model for classification
  - Discriminative model $P(C|\mathbf{X}) \quad C = c_1, \cdots, c_L, \mathbf{X} = (X_1, \cdots, X_n)$

- MAP classification rule
  - MAP: Maximum A Posterior
  - Assign x to c^* if

$$P(C = c^* | \mathbf{X} = \mathbf{x}) > P(C = c | \mathbf{X} = \mathbf{x}) \quad c \neq c^*, \; c = c_1, \cdots, c_L$$

Naïve Bayes (NB) Classifier

Naïve Bayes Algorithm (for discrete input attributes)

– Learning Phase: Given a training set **S**,

For each target value of $c_i \ (c_i = c_1, \cdots, c_L)$

$\hat{P}(C = c_i) \leftarrow$ estimate $P(C = c_i)$ with examples in $\mathbf{S}$;

For every attribute value a_{jk} of each attribute $x_j \ (j = 1, \cdots, n; \ k = 1, \cdots, N_j)$

$\hat{P}(X_j = a_{jk} \mid C = c_i) \leftarrow$ estimate $P(X_j = a_{jk} \mid C = c_i)$ with examples in $\mathbf{S}$;

Output: conditional probability tables; for $x_j,\ N_j \times L$ elements

– Test Phase: Given an unknown instance $\mathbf{X}' = (a'_1, \cdots, a'_n)$,

Look up tables to assign the label c^* to $\mathbf{X}'$ if

$$[\hat{P}(a'_1 \mid c^*) \cdots \hat{P}(a'_n \mid c^*)]\hat{P}(c^*) > [\hat{P}(a'_1 \mid c) \cdots \hat{P}(a'_n \mid c)]\hat{P}(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

Play Tennis: Classification with NB

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

NB: Learning Phase

P(Outlook=o|Play=b)

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

P(Temperature=t|Play=b)

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

P(Humidity=h|Play=b)

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

P(Wind=w|Play=b)

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

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

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

NB: 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*".

NB for Continuous-values Input Attributes

- Numberless values for an attribute
- Conditional probability modeled with the normal distribution

$$\hat{P}(X_j \mid C = c_i) = \frac{1}{\sqrt{2\pi}\sigma_{ji}} \exp\left(-\frac{(X_j - \mu_{ji})^2}{2\sigma_{ji}^2} \right)$$

μ_{ji} : mean (avearage) of attribute values X_j of examples for which $C = c_i$

σ_{ji} : standard deviation of attribute values X_j of examples for which $C = c_i$

- Learning Phase: for $\mathbf{X} = (X_1, \cdots, X_n),\ C = c_1, \cdots, c_L$

  Output: $n \times L$ normal distributions and $P(C = c_i)\ \ i = 1, \cdots, L$

- Test Phase: for $\mathbf{X}' = (X'_1, \cdots, X'_n)$
  - Calculate conditional probabilities with all the normal distributions
  - Apply the MAP rule to make a decision

NB Remarks

Naïve Bayes based on the independence assumption

- Training is very easy and fast; just requiring considering each attribute in each class separately
- Test is straightforward; just looking up tables or calculating conditional probabilities with normal distributions

NB Classifier Online Resource

Naive Bayes Classifiers
https://www.geeksforgeeks.org/naive-bayes-classifiers/

Naive Bayes Classifier (Explained in Bangla) || Algorithm in Data Mining || Data Mining Tutorial

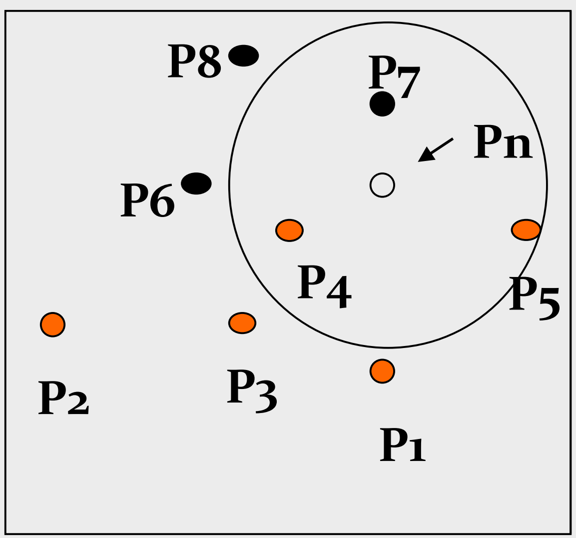
https://www.youtube.com/watch?v=cxWPanlW-L0

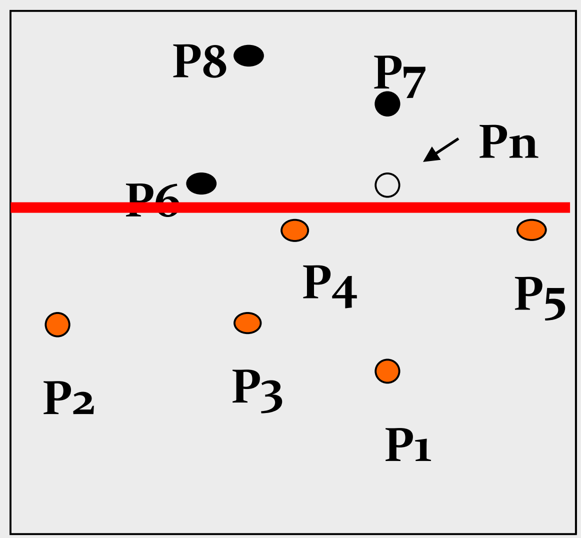
Solved Example Naive Bayes Classifier to classify New Instance PlayTennis Example Mahesh Huddar

https://www.youtube.com/watch?v=XzSlEA4ck2I&list=WL

Lazy & Eager Learning

- Two Types of Learning Methodologies
  - Lazy Learning
    - Instance-based learning. (k-NN)
  - Eager Learning
    - Decision-tree and Bayesian classification.
    - ANN & SVM





Lazy & Eager Learning : *Key Differences*

- Lazy Learning
  - Do not require model building
  - Less time training but more time predicting
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form its implicit global approximation to the target function
- Eager Learning
  - Require model building
  - More time training but less time predicting
  - must commit to a single hypothesis that covers the entire instance space

Thanks for your attention

Question and Answer