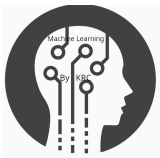
# Machine Learning
# (Basic Algorithms: Naive Bayes and KNN)

Prof K R Chowdhary

MBM University

September 27, 2024



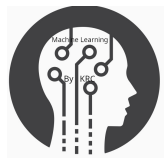
# Introduction

$\Rightarrow$ Four simple algorithms: *Naive Bayes*, *Nearest Neighbors*, the *Mean Classifier*, and the *Perceptron*. All these algorithms are readily usable and easily implemented from scratch in their most basic form.

$\Rightarrow$ Task: (Spam filtering). Given a set of $m$ e-mails $x_i$, denoted by $\mathbf{X} := \{x_1, ..., x_m\}$ and associated labels $y_i$, denoted by $\mathbf{Y} := \{y_1, ..., y_m\}$. Here the labels satisfy $y_i \in \{spam, nospam\}$.

$\Rightarrow$ Key assumption: pairs $(x_i, y_i)$ are drawn jointly from some probability distribution $p(x, y)$, which represents the e-mail generating process for a user.

$\Rightarrow$ Also, there is sufficiently strong dependence between $x$ and $y$ that we will be able to estimate $y$ given $x$ and a set of labeled instances $\mathbf{X}, \mathbf{Y}$.



# Example of spam e-mail

To undisclosed-recipients:; 13:03

**Exciting Freelance Opportunity for Academic Paper Review and Edit**

Dear Candidate,

I hope this email finds you well.

We at Paperpedia Private Limited are excited to announce a freelance opportunity for individuals passionate about reviewing and editing academic publication papers. If you have experience in this field and are eager to showcase your expertise, we invite you to join our team.

Role Description:

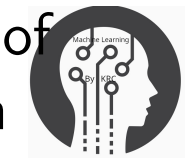
Reviewing and editing academic publication papers to ensure accuracy, clarity, and coherence of content.
Adhering to academic standards and guidelines, including those set by WOS (Web of Science) and SCOPUS.

Compensation for successfull WOS reviwing and editing is upto INR 150k for Q1 journals
All payments are excluding APC

$\Rightarrow$ The e-mails such above are text, and algorithms will require data in a *vectorial* form.

$\Rightarrow$ *Converting text into a vector*: *bag-of-words* representation. To list of all possible words occurring in $\mathbf{X}$ (*dictionary*), we assign a unique number (position in dictionary).

$\Rightarrow$ Next, simply count in each document $x_i$, the number of times a given word $j$ occurs. This is then used as the value of the $j^{th}$ coordinate of $x_i$, shown in table, next page.



# Vector space representation of Strings,

Table 1: $x_1$: BB Roy of Great Britain has very good dog $x_2$: The dog used to hunts dogs only

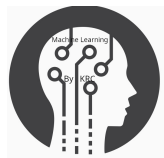
| | BB | Roy | of | Great | Britain | has | very | good | dog | the | used | to | hunts | dogs | only |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $x_2$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

$\Rightarrow$ The representation in Table 1 is such a case. Now, it is easy to compute distances, similarities, and other statistics directly from the *vectorial representation*.

$\Rightarrow$ **Naive Bayes**: Say, in a Cancer test, doctor uses outcomes of the test to infer whether the patient is diseased. In the context of spam filtering, actual text of the e-mail $x$ corresponds to the *test* and the label $y$ is like diagnosis. Bayes:

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

We may have a good estimate of $p(y)$, that is, the probability of receiving a *spam* or *nospam* mail.



# Naive Bayes Probability ...

$\Rightarrow$ Let us denote the number of *spam* and *nospam* e-mails in $\mathbf{X}$ by $m_{spam}$ and $m_{nospam}$, and $m_{spam} + m_{nospam} = m$. Thus,

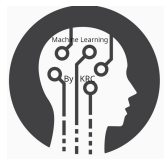
$$p(nospam) \approx \frac{m_{nospam}}{m} \qquad (2)$$

$$p(spam) \approx \frac{m_{spam}}{m} \qquad (3)$$

$\Rightarrow$ The key problem is: we do not know $p(x|y)$ or $p(x)$ in (1). We may compute the *likelihood ratio* ($L(x)$), and forgo with the need of $p(x)$:

$$L(x) = \frac{p(spam|x)}{p(nospam|x)} = \frac{p(x|spam)p(spam)}{p(x|nospam)p(nospam)}$$

$\Rightarrow$ Possibilities are: 1: $L(x)$ exceeds a threshold $c$, then decide that $x$ is spam, 2: $c$ is large, then our algorithm is conservative, and classifies an email as spam only if $p(spam|x) \gg p(nospam|x)$, 3: $c$ is small, then algorithm aggressively classifies emails as spam.



# Naive Bayes Probability ...

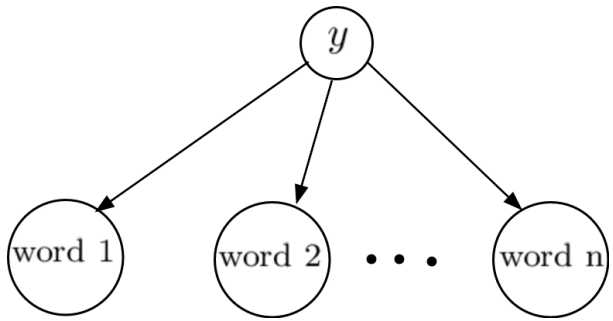
$\Rightarrow$ Main obstacle in computing of (1) is, "No access to $p(x|y)$." So, we make approximation. To model the distribution of the test outcomes, say, $T_1$ and $T_2$, we assume that they are conditionally independent of each other, given the diagnosis.

$\Rightarrow$ Similarly, occurrence of each word in a email is taken as a separate test, and outcomes are combined in a Naive Bayes,

$$p(x|y) = \prod_{j=1}^{n} p(w^j|y), \qquad (4)$$

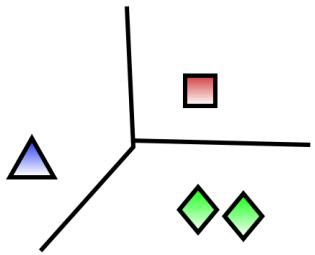
$\Rightarrow$ $n$ is word count in email $x$, $w^j$ is $j^{th}$ word in $x$, and $y$ is label (spam/nospam).

$\Rightarrow$ Equation (4) assumes that probability of occurrence of a word in email is independent of all other words given the label of the email. This is not always true. e.g., "Delhi" is more likely to be preceded with "New". But, it suffices for our purposes (see fig.) of Naive Bayes.



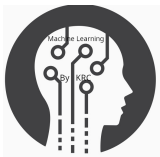
# Nearest Neighbor Classifier (NNC)

$\Rightarrow$ It is simpler estimator than *Naive Bayes*. In most basic form, it assigns the label of its nearest neighbor to an observation $x$ depending on whether the query point $x$ is closest to the $\square$, $\Diamond$ or $\triangle$.



$\Rightarrow$ This requires a distance measure $d(x, x')$ between pairs of observations, which need not even be symmetric. E.g., we could use string edit distances or information theory based measures to compare two documents.

$\Rightarrow$ Problem with NNC is: Estimates can be very noisy when data itself is noisy. E.g., if a spam email is erroneously labeled as *nonspam*, then all emails which are similar to this, will share the same fate. See Fig. 1, (next pg.) where it is beneficial to pool together number of neighbors (the $k$-nearest neighbors of $x$) and use a majority vote to decide the class membership of $x$.



# $k$-Nearest neighbor classifiers using Euclidean distances

$\Rightarrow$ NNC algorithms can yield excellent performance when distance measure is good.

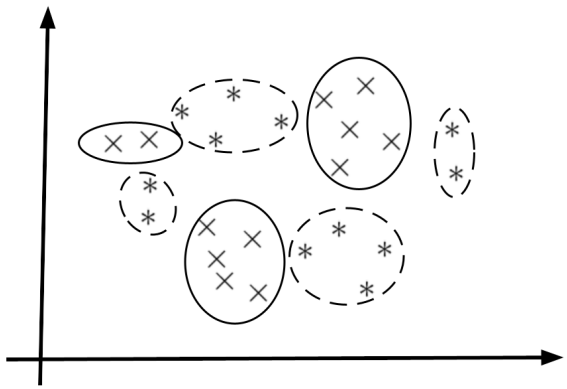


Figure 1: KNN Classifier

$\Rightarrow$ It is trivial to extend this algorithm to regression. Change is: *return average of values* $y_i$ instead of their majority vote.
$\Rightarrow$ $d(x_i, x)$ calc. will be costly when observations are large or $x_i$ in a high dimensional space.

**Algorithm 1** $k$-NNC

1: Classify($\mathbf{X}, \mathbf{Y}, x$) {reads documents $\mathbf{X}$, labels $\mathbf{Y}$ and query $x$}
2: **for** $i = 1$ to $N$ **do**
3: Compute distance $d(x_i, x)$
4: **end for**
5: Compute set $I$ containing indices for the $k$ smallest distances $d(x_i, x)$
6: **return** Label of majority: $\{y_i$, here $i \in I\}$