2nd

Naive Bayes :→ classification algorithm
↳ probability - based

KNN → neighborhood based classification.

Conditional Probability = $(P(A/B)) = Pr(A = a / B = b)$

↓ value that A takes

↓ value that B takes.

Always read equations in english.

$$P(A/B) = \frac{P(A \cap B)}{P(B)}$$

* Independent Events & mutually Exclusive events :→

A, B are said to be independent.

$P(A/B) = P(A)$

$P(B/A) = P(B)$

for ex :→
A: getting value of 6 in die 1 throw ($D_1 = 6$)
B: getting a value of 3 in dice2's throw ($D_2 = 3$)

A, B are said to be mutually exclusive if.

$P(A/B) = P(B/A) = 0$

$\frac{P(A \cap B)}{P(B)}$ $\quad$ $\frac{P(B \cap A)}{P(A)}$

, so, $P(A \cap B)$ should be 0

as $A \cap B = B \cap B$.

For ex :→ probability of getting 3 in die1 is 0 if die 2 is getting 6 in it.

Scanned by TapScanner

(1700's)

imp Bayes Theorem :→

$$P(A/B) = \frac{\overset{\nearrow \text{likelihood}}{P(B/A)} \cdot P(A) \xrightarrow{\text{prior}}}{\underbrace{P(B)}_{\text{evidence}}} \quad \text{if } P(B) \neq 0.$$

posterior probability

Proof of Bayes theorem :→

$$P(A/B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A,B)}{P(B)}$$

↗ also means ∩

In set theory $A \cap B = B \cap A$ ←

$$P(A/B) = \frac{P(B \cap A)}{P(B)} = \frac{P(B,A)}{P(B)} \quad - (1)$$

$$P(B/A) = \frac{P(B \cap A)}{P(A)}$$

$$P(B \cap A) = P(B/A) \cdot P(A) \quad (2)$$

Put $P(B \cap A)$ in (1) eq

$$P(A/B) = \frac{P(B/A) \cdot P(A)}{P(B)}$$

Scanned by TapScanner

* Naive Bayes Algorithm

↓

unsophistic

↓

simplistic

$$\underbrace{P(C/f_1, f_2, f_3, f_4)}_{x_q} = P(f_1/c) * P(f_2/c) * P(f_3/c) * P(f_4/c) * P(c)$$

Naive bayes is much better in terms of space complexity at runtime

$O(d*c) \rightarrow$ naive bayes

$O(n*d) \rightarrow$ KNN

memory efficient at run-time

* Naive Bayes on text data :→

↓

first technique used in case of spam filters → email → spam / not spam

review → +ve / -ve

Text classification → Naive Bayes is a simple used method.

← text1 →	0
← text2 →	1
	0
	1
	:

Task

① ← $P(y_q = 1 / text_q)$ → compute these

② ← $P(y_q = 0 / text_q)$

So if ① > ②

then we can say that $y_q = 1$ for given $text_q$.

Preprocessing :→ text ⟶ stopwords

Stemming

n-grams

$\hookrightarrow \{w, w_2, w_3, \ldots w_d\}$

↓

Binary BoW

text $\xrightarrow{\text{pre-pro}} \underbrace{\{w_1, w_2, \ldots w_d\}}_{\text{set of words.}}$

$$P(y_1 = 1 / \text{text}) = P(y_1 = 1 / w_1, \underbrace{w_2, w_3, \ldots, w_d}_{\text{features}})$$

class prior

likelihoods

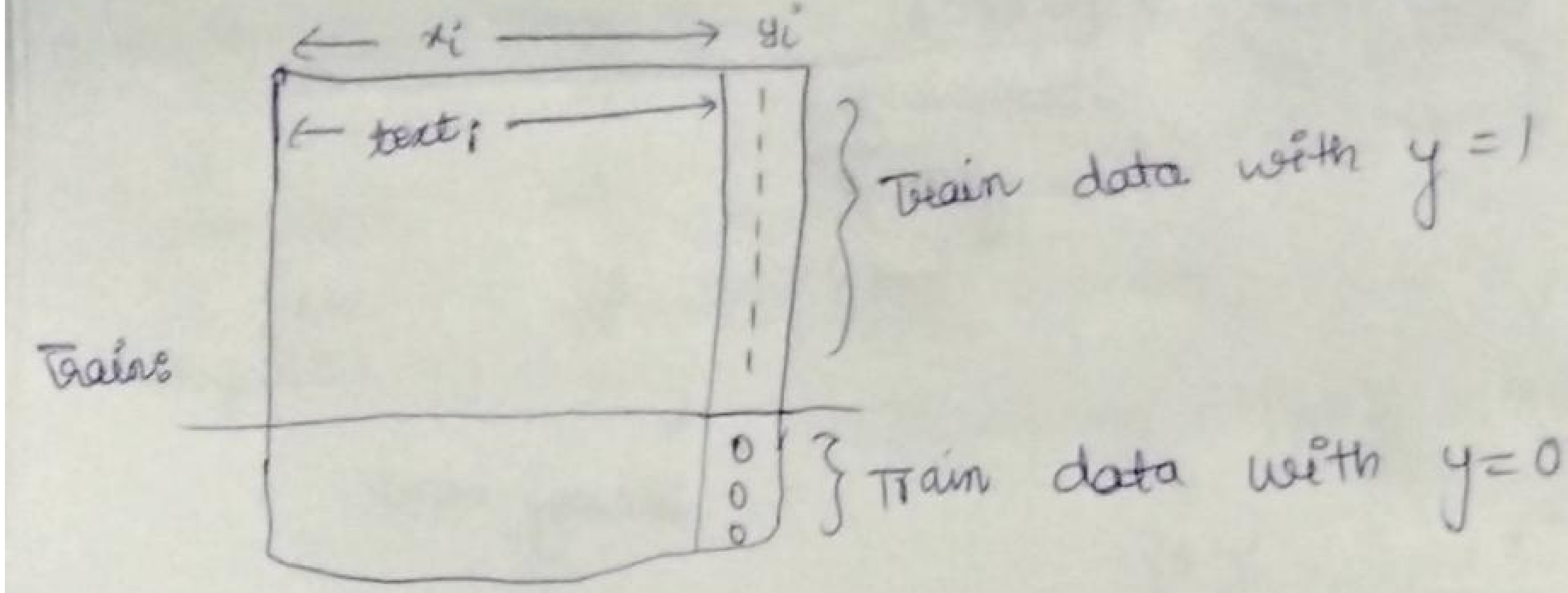
$$= P(y=1) * P(w_1/y=1) * P(w_2/y=1) \ldots\ldots P(w_d/y=1)$$

$$P(y_1 = 1 / \text{text}) \propto P(y=1) * \prod_{i=1}^{d} P(w_i / y=1)$$

$$P(y = 0 / \text{text}) \propto P(y=0) * \prod_{i=1}^{d} P(w_i / y=0)$$

$$P(y=1) = \frac{\text{no. of train points with } y=1}{\text{Total no. of train points.}}$$

$$P(y=0) = \frac{\text{no. of train points with } y=0}{\text{Total no. of train points.}}$$



$$P(w_i|y=1) = \frac{\text{no. of Train data points which contain } w_i \text{ \&\& } y=1}{\text{no. of Train data points with } y=1}$$

Text - classification problems.

- spam - detection
- Polarity of a review

} Naive Bayes is a very good Baseline → benchmark

* Laplace - Additive Smoothing :→ (not laplacian smoothing)

After training → all this data is already computed.

$$\left\{ \begin{array}{ll} P(y=1) ; P(y=0) \leftarrow \text{class priors} & \\ P(w_1|y=1) & P(w_1|y=0) \\ P(w_2|y=1) & P(w_2|y=0) \\ \vdots & \\ P(w_m|y=1) & P(w_m|y=0) \end{array} \right\} \rightarrow \text{likelih}$$

Test :→ $text_q = (w_1, w_2, w_3, w')$

training data.

very often ← ~~*~~ w' is not present in $\{w_1, w_2, w_3, \dots w_m\}$

$$P(1/text_q) = P(y=1/w_1, w_2, w_3, w')$$

$$= P(y=1) * P(w_1/y=1) * P(w_2/y=1) * P(w_3/y=1) * P(w'/y=1)$$

how do you get this probability as w' is not present in training data.

ignoring or dropping it will mean $P(w'/y=1) = 1$ which is not correct.

We have to get values of $P(w'/y=1)$ and $P(w'/y=0)$

$$P(w'/y=1) = \frac{P(w', y=1)}{P(y=1)}$$

$$= \frac{\text{no. of train points such that } w' \text{ occurs in } y=1}{\text{no. of train points where } y=1}$$

$$= \frac{0}{n_1} = 0$$ → This is also dangerous as it will make whole probability to be 0.

Laplace smoothing or additive smoothing :-

$$P(f_1 = a / y = 1) = \frac{0 + \alpha}{n_1 + \alpha k}$$

$\alpha = 1$ → typically (not always)

k = no. of distinct values which f_1 can take

f_1 ⇒ feature

$$P(w'/y=1) = \frac{0+\alpha}{100+2\alpha}$$

$K = 2$ → because w' is 0 or 1.
present or not.

Let $n_1 = 100$

Case 1 :- $\alpha = 1 = \frac{1}{102} \neq 0$

α small, we are getting rid of multiplying all the probabilities with 0.

$P(w'/y=1) \neq 0$

⇓ which implies

$P(y=1 | text_q) \neq 0$.

So, its not 0 anymore

Case 2 :- $\alpha = 10000$ → when α is large

$$P(w'/y=1) = \frac{0+10k}{100+2 \times 10k} = \frac{10k}{20100} \approx \frac{1}{2}$$

$$\boxed{P(w'/y=1) = P(w'/y=0) = \frac{1}{2}}$$

means equal probability of w' to be 0 or 1 because w' have only two possibilities (0 or 1)

Scanned by TapScanner

Laplace smoothing :→

find this for all words

$$P(w_i/y=1) = \frac{(\text{no. of data points with } w_i \,\&\, y=1) + \alpha}{(\text{no. of data points with } y=1) + \alpha k}$$

present in my training data

adding something to numerator & denominator, that's why it is called additive Smoothing

In this formula, as $\alpha \uparrow$,

$P(w_i/y=1)$ → likelihood probability is moving to a uniform distribution.

if n_1 is small

num or den is small → less confidence in ratio

often times, $\alpha = 1$ → add one smoothing

It is called smoothing because you are moving likelihood probability towards uniform distribution