

NOTES

Naive Bayes is the classification algorithm. It is based on the probability.

Conditional Probability:

It is the probability of A given B. Where A and B are the two random variables.

Example:

The outcome of one die is 6 possibilities and the outcome of other die is 6 possibilities. When we sum the outcomes of the two dies then there is a sample space of 36 outcomes.

What is the probability that $D_1 = 2$?

Table 1 shows the sample space of 36 outcomes.

Clearly, $D_1 = 2$ in exactly 6 of the 36 outcomes; thus $P(D_1=2) = \frac{6}{36} = \frac{1}{6}$.

Table 1

		D2					
		1	2	3	4	5	6
D1	1	2	3	4	5	6	7
	2	3	4	5	6	7	8
	3	4	5	6	7	8	9
	4	5	6	7	8	9	10
	5	6	7	8	9	10	11
	6	7	8	9	10	11	12

What is the probability that $D_1 + D_2 \leq 5$?

Table 2 shows that $D_1 + D_2 \leq 5$ for exactly 10 of the same 36 outcomes, thus $P(D_1 + D_2 \leq 5) = \frac{10}{36}$.

Table 2

		D2					
		1	2	3	4	5	6
D1	5	6	7	8	9	10	11
	6	7	8	9	10	11	12

Each cell the sum of outcomes d1 and d2 (dice).

Now, How many possibilities have the sum is less than 5 int he table.

What is the probability that $D_1 + D_2 \leq 5$?

Table 2 shows that $D_1 + D_2 \leq 5$ for exactly 10 of the same 36 outcomes, thus $P(D_1 + D_2 \leq 5) = \frac{10}{36}$.

Table 2

		D2					
		1	2	3	4	5	6
D1	1	2	3	4	5	6	7
	2	3	4	5	6	7	8
	3	4	5	6	7	8	9
	4	5	6	7	8	9	10
	5	6	7	8	9	10	11
	6	7	8	9	10	11	12

What is the probability that $D_1 = 2$ given that $D_1 + D_2 \leq 5$?

Table 3 shows that for 3 of these 10 outcomes, $D_1 = 2$.

Thus, the conditional probability $P(D_1=2 | D_1 + D_2 \leq 5) = \frac{3}{10} = 0.3$.

$$P(D_1 + D_2 \leq 5) = \frac{10}{36}$$

Q. Then what is the probability that $D_1 = 2$ given the condition $D_1 + D_2 \leq 5$.

A. The answer is nothing but the probability of $D_1=2$ and $D_1 + D_2$ is less than 5.

This is called conditional probability as the probability of some thing, when some thing else is already happened.(or given already).

conditional-prob

given

conditioned-on

$$P(D_1=2 | D_1+D_2 \leq 5) = \frac{3}{10}$$

= $\frac{3}{10}$

Computing conditional probability:

(3/36)

(3/10)

$P(A|B) = \frac{P(A \cap B)}{P(B)}$ if $P(B) \neq 0$

$D_1 = 2 : A$

$D_1 + D_2 \leq 5 : B$

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

def

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

if $P(B) \neq 0$

Here in the earlier notation for the definition of conditional probability, the conditioning event B is that $D_1+D_2 \leq 5$, and the event A is $D_1 = 2$. We have $P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{3/36}{10/36} = \frac{3}{10}$, as seen in the table.

Use in inference [edit]

Defining conditional probability:

Conditional prob:

$$P(A|B) = P_{\text{fix}}(A=a \mid B=b)$$

\downarrow
value
 \uparrow
value

$$\begin{cases} A: \gamma \cdot V \\ B: \gamma \cdot V \end{cases}$$

def: $P(A|B) = \frac{P(A \cap B)}{P(B)}$; $P(B) \neq 0$

Independent events and mutually exclusive events:

The events are said to be independent, if the occurrence of one event does not depend on the occurrence of other event.

Independent Events & Mutually Exclusive Events

A, B are said be "independent"

Def: $\begin{cases} P(A|B) = P(A) \\ P(B|A) = P(B) \end{cases}$

$\begin{array}{c} \text{⑥ } \boxed{5} \quad \text{⑦ } \boxed{3} \\ \text{1 } \quad \text{2 } \end{array}$

$\begin{aligned} A: & \text{ getting value of 6} \\ & \text{in die 1's throw} \\ & (D_1=6) \end{aligned}$

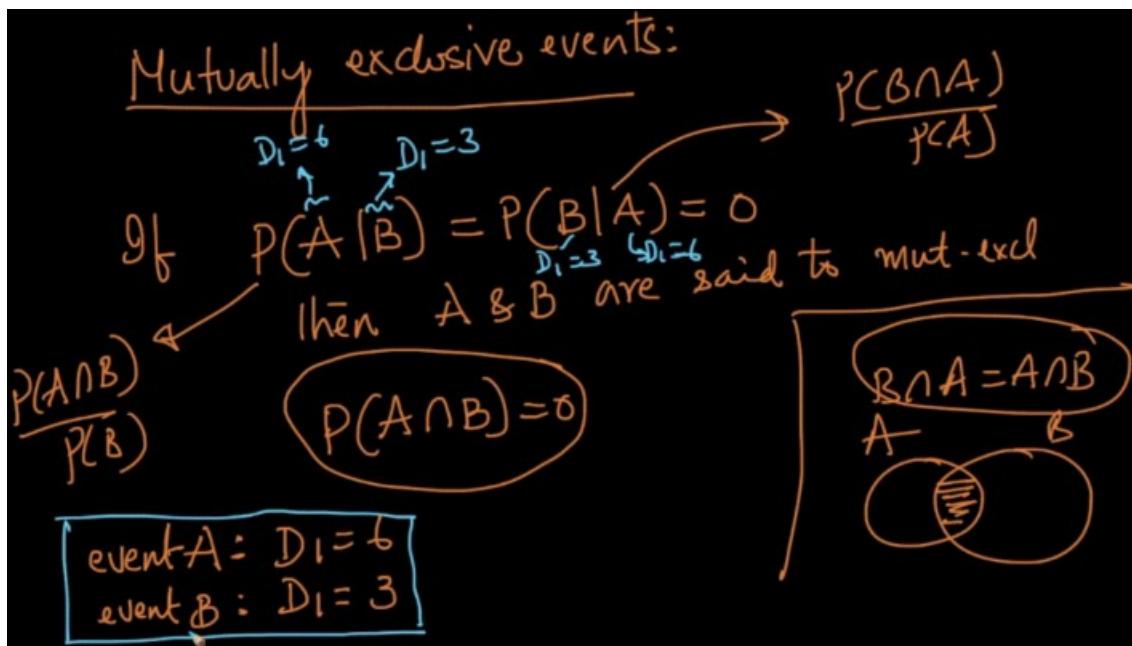
$\begin{aligned} B: & \text{ getting a value of 3} \\ & \text{in die 2's throw} \\ & (D_2=3) \end{aligned}$

$$\begin{cases} P(D_1=6 \mid D_2=3) \\ = P(D_1=6) \\ P(D_2=3 \mid D_1=6) = P(D_1=3) \end{cases}$$

Here is the mathematical derivation of independent events.

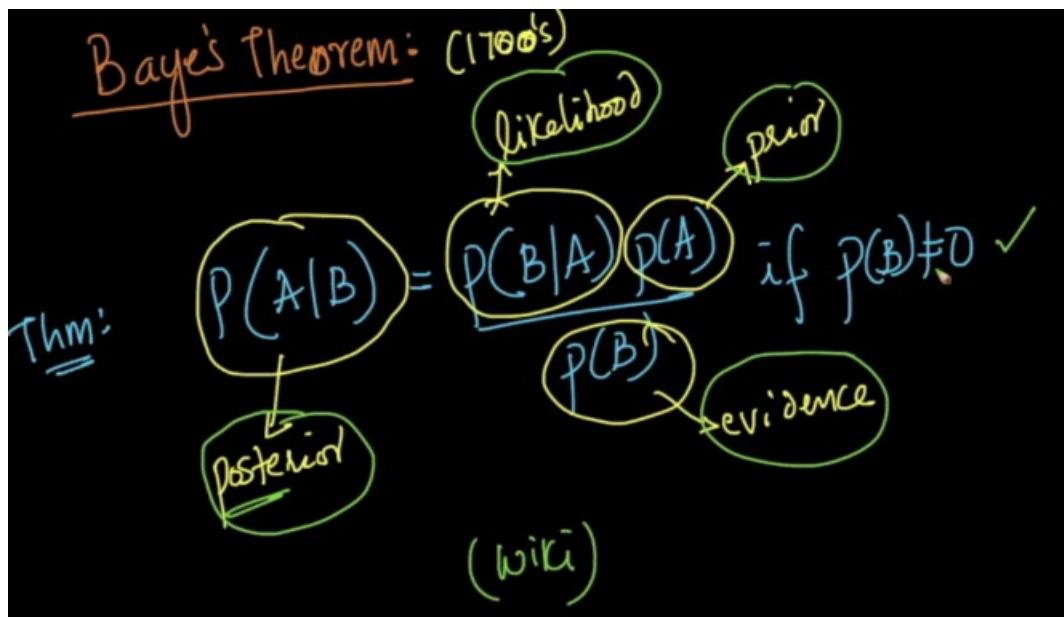
Mutually exclusive events:

We can understand this concept as the events that are occurring at the same time of one die only. Which is zero, because the probability of occurrence of 3 is nothing to do with the occurrence of the 6 of the same die.



Bayes theorem:

Defining bayes theorem:



Proof:

$$\text{Proof: } P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A, B)}{P(B)} \xrightarrow{\text{defn}} A \& B$$

$A \cap B = B \cap A \rightarrow \text{set theory}$

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

$P(A) \quad P(B|A) = \frac{P(B \cap A)}{P(A)} \rightarrow \text{defn.}$

Interchanging the equations of conditional probability, we get the bayes theorem.

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)} \rightarrow \text{Bayes Thm.}$$

Example:

A more complicated example [edit]

The entire output of a factory is produced on three machines. The three machines account for different amounts of the factory output, namely 20%, 30%, and 50%. The fraction of defective items produced is this: for the first machine, 5%; for the second machine, 3%; for the third machine, 1%. If an item is chosen at random from the total output and is found to be defective, what is the probability that it was produced by the third machine?

Once again, the answer can be reached without recourse to the formula by applying the conditions to any hypothetical number of cases. For example, if 100,000 items are produced by the factory, 20,000 will be produced by Machine A, 30,000 by Machine B, and 50,000 by Machine C. Machine A will produce 1000 defective items, Machine B 900, and Machine C 500. Of the total 2400 defective items, only 500, or 5/24 were produced by Machine C.

A solution is as follows. Let A_i denote the event that a randomly chosen item was made by the i th machine (for $i = 1, 2, 3$). Let B denote the event that a randomly chosen item is defective. Then, we are given the following information:

$$P(A_1) = 0.2, \quad P(A_2) = 0.3, \quad P(A_3) = 0.5. \quad \checkmark$$

If the item was made by the first machine, then the probability that it is defective is 0.05; that is, $P(B|A_1) = 0.05$. Overall, we have

$$P(B|A_1) = 0.05, \quad P(B|A_2) = 0.03, \quad P(B|A_3) = 0.01. \quad \checkmark$$

To answer the original question, we first find $P(B)$. That can be done in the following way:

$$P(B) = \sum_i P(B|A_i) P(A_i) = (0.05)(0.2) + (0.03)(0.3) + (0.01)(0.5) = 0.024.$$

Hence 2.4% of the total output of the factory is defective.

We are given that B has occurred, and we want to calculate the conditional probability of A_3 . By Bayes' theorem

Calculating the probability of defective of the total factory output..

Once again, the answer can be reached without recourse to the formula by applying the conditions to any hypothetical number of cases. For example, if 100,000 items are produced by the factory, 20,000 will be produced by Machine A, 30,000 by Machine B, and 50,000 by Machine C. Machine A will produce 1000 defective items, Machine B 900, and Machine C 500. Of the total 2400 defective items, only 500, or 5/24 were produced by Machine C.

A solution is as follows. Let A_i denote the event that a randomly chosen item was made by the i th machine (for $i = 1, 2, 3$). Let B denote the event that a randomly chosen item is defective. Then, we are given the following information:

$$P(A_1) = 0.2, \quad P(A_2) = 0.3, \quad P(A_3) = 0.5.$$

If the item was made by the first machine, then the probability that it is defective is 0.05; that is, $P(B|A_1) = 0.05$. Overall, we have

$$P(B|A_1) = 0.05, \quad P(B|A_2) = 0.03, \quad P(B|A_3) = 0.01.$$

To answer the original question, we first find $P(B)$. That can be done in the following way:

$$P(B) = \sum_i P(B|A_i) P(A_i) = (0.05)(0.2) + (0.03)(0.3) + (0.01)(0.5) = 0.024.$$

Hence 2.4% of the total output of the factory is defective.

We are given that B has occurred, and we want to calculate the conditional probability of A_3 . By Bayes' theorem,

$$P(A_3|B) = P(B|A_3) P(A_3)/P(B) = (0.01)(0.5)/(0.024) = 5/24.$$

Given that the item is defective, the probability that it was made by the third machine is only 5/24. Although machine 3 produces half of the total output, it produces a much smaller fraction of the defective items. Hence the knowledge that the item selected was defective enables us to replace the prior probability $P(A_3) = 1/2$ by the smaller posterior probability $P(A_3|B) = 5/24$.

Next, the probability of the randomly chosen item belongs to the machine A3.

denote the event that a randomly chosen item is defective. Then, we are given the following information:

$$P(A_1) = 0.2, \quad P(A_2) = 0.3, \quad P(A_3) = 0.5.$$

If the item was made by the first machine, then the probability that it is defective is 0.05; that is, $P(B|A_1) = 0.05$. Overall, we have

$$P(B|A_1) = 0.05, \quad P(B|A_2) = 0.03, \quad P(B|A_3) = 0.01.$$

To answer the original question, we first find $P(B)$. That can be done in the following way:

$$P(B) = \sum_i P(B|A_i) P(A_i) = (0.05)(0.2) + (0.03)(0.3) + (0.01)(0.5) = 0.024.$$

Hence 2.4% of the total output of the factory is defective.

We are given that B has occurred, and we want to calculate the conditional probability of A_3 . By Bayes' theorem,

$$P(A_3|B) = P(B|A_3) P(A_3)/P(B) = (0.01)(0.5)/(0.024) = 5/24.$$

Given that the item is defective, the probability that it was made by the third machine is only 5/24. Although machine 3 produces half of the total output, it produces a much smaller fraction of the defective items. Hence the knowledge that the item selected was defective enables us to replace the prior probability $P(A_3) = 1/2$ by the smaller posterior probability $P(A_3|B) = 5/24$.

Interpretations [edit]

The interpretation of Bayes' theorem depends on the interpretation of probability ascribed to the terms. The two main interpretations are described below.

	Relative size	Case B	Case B	Total
Condition A	w	x	w+x	
Condition Ā	v	z	v+z	

Data:

	Predictors				Response	
	Outlook	Temperature	Humidity	Wind	Class	(C)
Day1	Sunny	Hot	High	Weak	No	Play=Yes
Day2	Sunny	Hot	High	Strong	No	Play=No
Day3	Overcast	Hot	High	Weak	Yes	
Day4	Rain	Mild	High	Weak	Yes	
Day5	Rain	Cool	Normal	Weak	Yes	
Day6	Rain	Cool	Normal	Strong	No	
Day7	Overcast	Cool	Normal	Strong	Yes	
Day8	Sunny	Mild	High	Weak	No	
Day9	Sunny	Cool	Normal	Weak	Yes	
Day10	Rain	Mild	Normal	Weak	Yes	
Day11	Sunny	Mild	Normal	Strong	Yes	
Day12	Overcast	Mild	High	Strong	Yes	
Day13	Overcast	Hot	Normal	Weak	Yes	
Day14	Rain	Mild	High	Strong	No	

We can take the proportionality from the naive bayes and derive the LHS of the naive bayes proportional to the numerator of Naive bayes.

Example:

Temperature=Cool, Humidity=High, Wind=Strong) that will have to be classified (i.e., are we going to play tennis under the conditions specified by x').

With the MAP rule, we compute the posterior probabilities. This is easily done by looking up the tables we built in the learning phase.

$$\begin{aligned} P(\text{Class}_{\text{Play}=\text{Yes}}|x') & \propto [P(\text{Sunny}|\text{Class}_{\text{Play}=\text{Yes}}) \times P(\text{Cool}|\text{Class}_{\text{Play}=\text{Yes}}) \times \\ & P(\text{High}|\text{Class}_{\text{Play}=\text{Yes}}) \times P(\text{Strong}|\text{Class}_{\text{Play}=\text{Yes}})] \times \\ & P(\text{Class}_{\text{Play}=\text{Yes}}) \end{aligned}$$

$$\textcolor{brown}{\cancel{2/9}} \times 3/9 \times 3/9 \times 9/14 = 0.0053$$

$$P(\text{Class}_{\text{Play}=\text{No}}|x') = [P(\text{Sunny}|\text{Class}_{\text{Play}=\text{No}}) \times P(\text{Cool}|\text{Class}_{\text{Play}=\text{No}}) \times$$

$$\textcolor{brown}{Q} \quad P(\text{High}|\text{Class}_{\text{Play}=\text{No}}) \times P(\text{Strong}|\text{Class}_{\text{Play}=\text{No}})] \times \\ P(\text{Class}_{\text{Play}=\text{No}})$$

$$= 3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.0205$$

Since $P(\text{Class}_{\text{Play}=\text{Yes}}|x')$ less than $P(\text{Class}_{\text{Play}=\text{No}}|x')$, we classify the new instance x'

Contingency tables:

The screenshot shows a Google Docs spreadsheet with three tables illustrating the frequency and probability of playing given different weather conditions.

P(Outlook=o Class Play=Yes No)		Frequency		Probability in Class	
Outlook =		Play=Yes	Play=No	Play=Yes	Play=No
Sunny		2	3	2/9	3/5
Overcast		4	0	4/9	0/5
Rain		3	2	3/9	2/5
		total= 9	total=5		

P(Temperature=t Class Play=Yes No)		Frequency		Probability in Class	
Temperature =		Play=Yes	Play=No	Play=Yes	Play=No
Hot		2	2	2/9	2/5
Mild		4	2	4/9	2/5
Cool		3	1	3/9	1/5
		total= 9	total=5		

P(Humidity=h Class Play=Yes No)		Frequency		Probability in Class	
Humidity =		Play=Yes	Play=No	Play=Yes	Play=No
High		3	4	3/9	4/5

The time complexity during the training time is $O(nd)$. Then we can store the probability values in the dictionary and retrieve during the test time.

In case of categorical features, implementing naive bayes is straight forward aka **counting** the unique values of the features.

Train and time complexity:

Classification Phase

Let's say, we get a new instance of the weather condition, $x' = (\text{Outlook}=\text{Sunny}, \text{Temperature}=\text{Cool}, \text{Humidity}=\text{High}, \text{Wind}=\text{Strong})$ that will have to be classified (i.e., are we going to play tennis under the conditions specified by x').

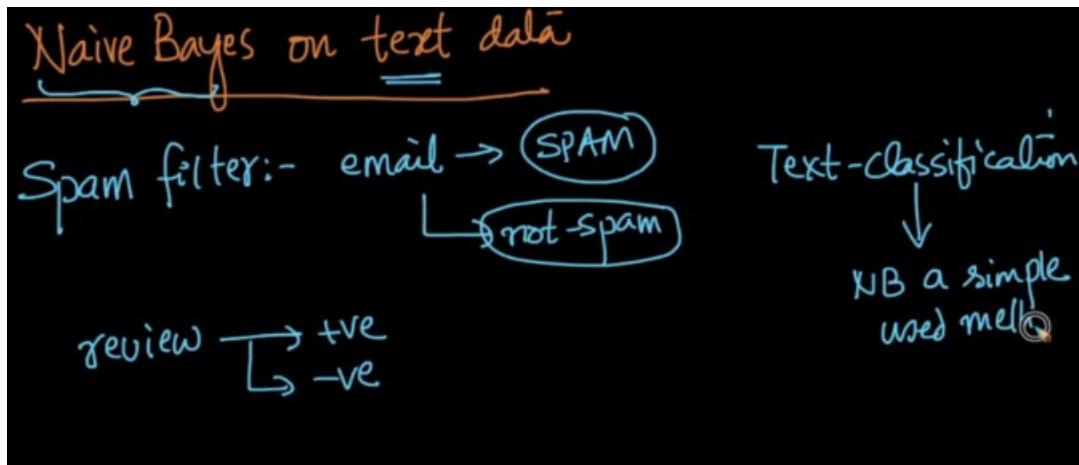
With the MAP rule, we compute the posterior probabilities. This is easily done by looking up the tables we built in the learning phase.

$$\begin{aligned} P(\text{ClassPlay}=\text{Yes}|x') &= [P(\text{Sunny}|\text{ClassPlay}=\text{Yes}) \times P(\text{Cool}|\text{ClassPlay}=\text{Yes}) \times \\ &\quad P(\text{High}|\text{ClassPlay}=\text{Yes}) \times P(\text{Strong}|\text{ClassPlay}=\text{Yes})] \times \\ &\quad P(\text{ClassPlay}=\text{Yes}) \\ &= 2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.0053 \end{aligned}$$

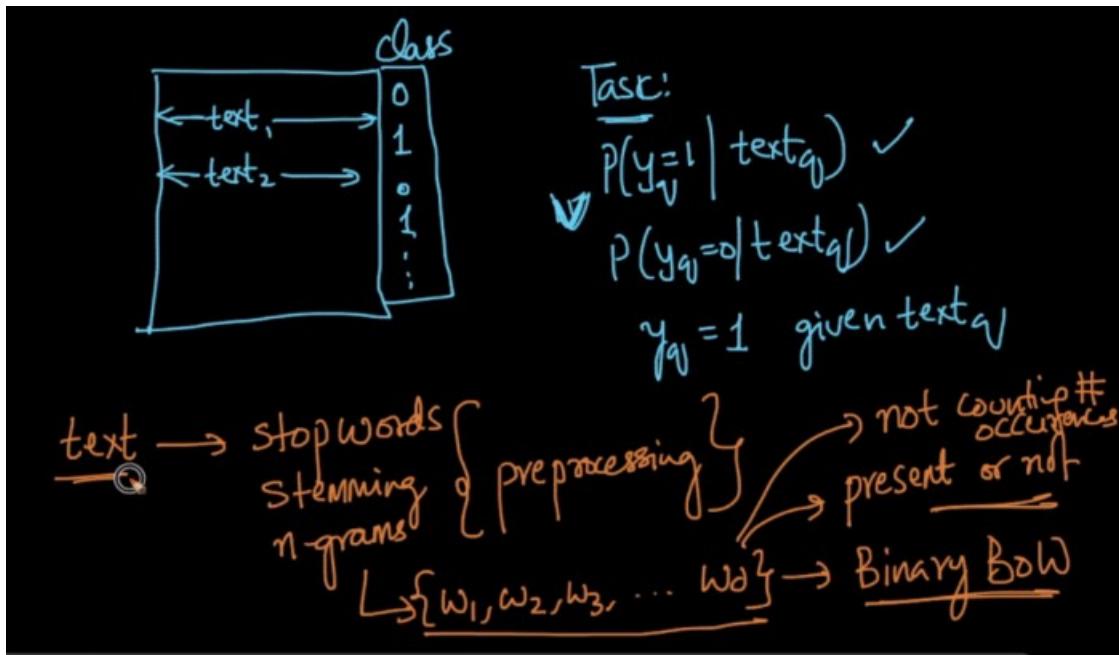
$$\begin{aligned} P(\text{ClassPlay}=\text{No}|x') &= [P(\text{Sunny}|\text{ClassPlay}=\text{No}) \times P(\text{Cool}|\text{ClassPlay}=\text{No}) \times \\ &\quad P(\text{High}|\text{ClassPlay}=\text{No}) \times P(\text{Strong}|\text{ClassPlay}=\text{No})] \times \\ &\quad P(\text{ClassPlay}=\text{No}) \end{aligned}$$

Naive Bayes on text data:

NB is often used for text classification, spam filtering.



The binary bag of words of text data is obtained by most common text preprocessing techniques.



The probability of text is spam can be written as follows:

$$\text{text} \xrightarrow{\text{preproc}} \{w_1, w_2, \dots, w_d\}$$

$$P(y=1 | \text{text}) = P(y=1 | \underbrace{w_1, w_2, \dots, w_d}_{\text{features}})$$

class prior α $P(y=1) * P(w_1 | y=1) + P(w_2 | y=1)$
 \dots
 $P(w_d | y=1)$

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

likelihood

The words are features in this case.

$$P(y=0 | \text{text}) \propto P(y=0) \cdot \prod_{i=1}^d p(w_i | y=0)$$

$$P(y=1) = \frac{\# \text{ Train pts with } y=1}{\text{Total # Train pts}}$$

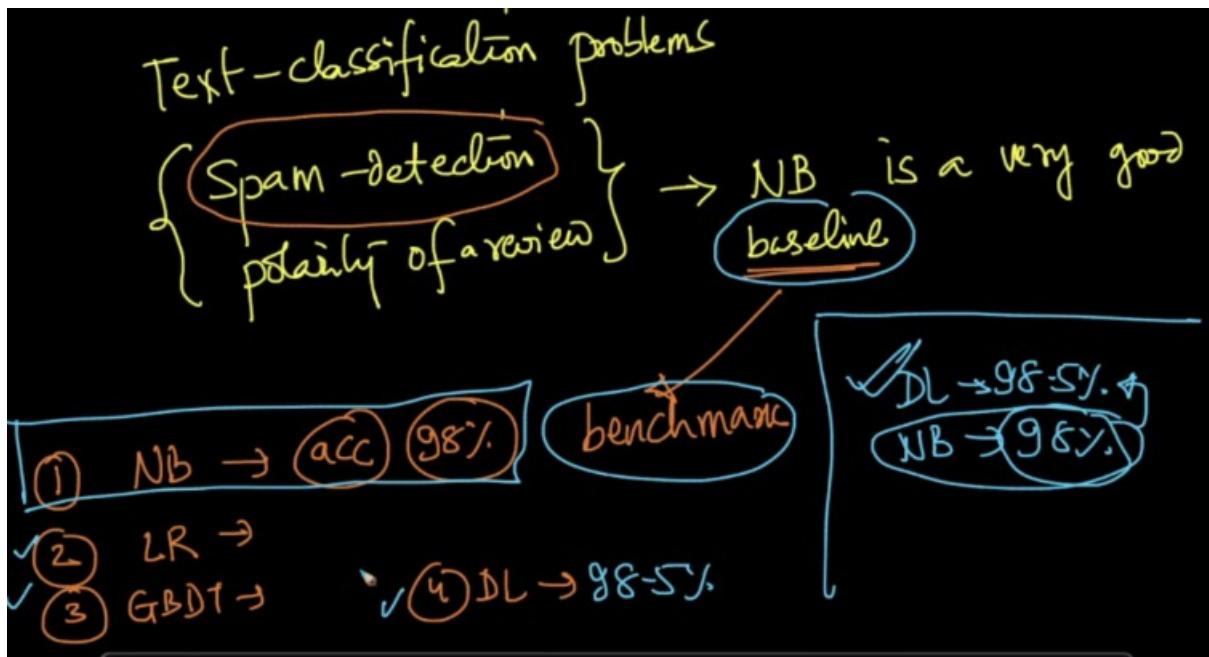
$$P(y=0) = \frac{\# \text{ Train pts with } y=0}{\text{Total # Train pts}}$$

The below part of the snap is the calculation of **prior** probabilities.

Calculation of likelihood:

$$P(w_i | y=1) = \frac{\# \text{ Train data pts with contain } w_i \text{ & } y=1}{\# \text{ Train data pts with } y=1}$$

Naive bayes is often used as a baseline model for improving the accuracy.



Laplace and additive smoothing:
Summary of training the models:

Laplace Smoothing:-

Training:-

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

During the text time if the words does not occur in the traning data.

Test: $\text{text}_q = (w_1, w_2, w_3, w')$

w' is not present in $\{w_1, w_2, w_3, \dots, w_n\}$

$\{w'\}$ is not present in $\{\text{set of words in training data}\}$

$\{w'\}$ is not present in $\{\text{English has tens of thousands of words}\}$

$\{w'\}$ is not present in $\{n\text{-grams}\}$

Questioning the model, with test data point.

$$\begin{aligned}
 P(y=1 | \text{text}_q) &= P(y=1 | w_1, w_2, w_3, w') \\
 &= p(y=1) + p(w_1 | y=1) + p(w_2 | y=1) \\
 &\quad + p(w_3 | y=1) + p(w' | y=1)
 \end{aligned}$$

$p(w' | y=1)$; $p(w' | y=0)$ (ignoring it or dropping it)

$p(w' | y=0) = 1 \times p(w' | y=1) = 1$

Ignoring the word that is not there in the training data is nothing but making the word probability equal to 1. This is logically incorrect.

Making the probability of occurrence of the word to zero will make the whole probability reduce to zero.

$$\begin{aligned}
 P(y=1 | \text{text}_W) &= P(y=1 | w_1, w_2, w_3, w') \\
 &= P(y=1) + P(w_1 | y=1) + P(w_2 | y=1) \\
 &\quad + P(w_3 | y=1) + P(w' | y=1) \\
 P(y=0 | \text{text}_W) &= P(w' | y=0); P(w_1 | y=0) \quad (\text{ignoring it}) \\
 &\quad ; P(w_2 | y=0) \quad (\text{ignoring it}) \\
 &\quad ; P(w_3 | y=0) \quad (\text{ignoring it}) \\
 &\quad \times P(w' | y=0) = 1 \quad (\text{dropping it})
 \end{aligned}$$

Here is the intuition of the missing word problem, the probability reduces to zero.

$$\left\{ \underbrace{P(w' | y=1)}_{0} = \frac{P(w', y=1)}{P(y=1)} \right. \\
 = \frac{\# \text{train pts s.t. } w' \text{ occurs in } y=1}{\# \text{train pts where } y=1} \\
 = \frac{0}{n_1} = \boxed{0}$$

To overcome this problem, we use something called laplace smoothing / additive smoothing.

Laplace Smoothing (not Laplacian smoothing)
Additive Smoothing

$$P(\underline{\omega}^i | y=1) = \frac{\alpha + n_i}{n_1 + \alpha K}$$

$\alpha = 1$ Typically

$\begin{cases} 1 \rightarrow \text{present} \\ 0 \rightarrow \text{not present} \end{cases}$

$K = \# \text{ distinct values}$
 $\underline{\omega}^i \text{ can take}$

Here K is the distinct values the **feature** can take, like when $k=2$ it can take the word in spam or not spam.



Typically, alpha is taken as 1.

Example:

$$P(\underline{\omega}^i | y=1) = \frac{\alpha + n_i}{100 + 2\alpha}$$

Let $n_1 = 100$

$\alpha = 2$ because $\underline{\omega}^i = 0 \text{ or } 1$

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

case 2 :- $0 \neq P(y=1 | \text{text}_y) \leftarrow P(\underline{\omega}^i | y=1) \neq 0$

Why don't we just replace with some small value instead of laplace smoothing ?

By placing laplace smoothing, we can interpret that the occurrence of words that does not occur can be given equi probable, whatever the value of alpha is.

Case 2 :- $\alpha = 10000$

$$P(w_i | y=1) = \frac{0 + 10000}{100 + 20000} = \frac{10000}{21000} \approx \frac{1}{2}$$

$n_i = 10$

$\checkmark w_i = 0 \rightarrow$ two possibilities

$\checkmark w_i = 1 \rightarrow$

$$P(w_i | y=1) = P(w_i | y=0) = \frac{1}{2}$$

If we apply laplace smoothing every word gets applied by the laplace term. (or) additive – smoothing.

Laplace Smoothing

for all words

$$P(w_i | y=1) = \frac{(\# \text{ data pts with } w_i \text{ by } y=1) + \alpha}{(\# \text{ data pts } y=1) + \alpha k}$$

w_i present in my training data

$$P(w_1 | y=1) = \frac{2 + \alpha}{50 + \alpha} = \frac{2}{50}$$

$\alpha = 1 \Rightarrow \frac{2+1}{50+1} = \left(\frac{3}{54}\right)$

$\alpha = 10 \Rightarrow \frac{2+10}{50+20} = \left(\frac{12}{70}\right)$

$\alpha = 100 \Rightarrow \frac{2+100}{50+200} = \left(\frac{102}{250}\right)$

$\alpha = 1000 \Rightarrow \frac{2+1000}{50+2000} = \left(\frac{1002}{2050}\right)$

If the alpha increases, the likelihood probabilities are moved to uniform distribution.

If the ratio value is very less, we can apply laplace smoothing to make the probabilities to be equal to half, which is very promising.

This is called Smoothing because, we are eventually moving the values of probabilities to uniform distribution by applying laplace smoothing.

Generally, we apply alpha equal to 1.

Log – probabilities and numerical stability:

For value of d equal to 100. calculation of posterior probabilities.

Log-probabilities vs numerical stability

$$P(y=1 | w_1, w_2, \dots, w_d) = P(y=1) * P(w_1 | y=1) * P(w_2 | y=1) * \dots * P(w_d | y=1)$$

d is large (let d=100)

When the values of the probabilities are so small then there is the problem of numerical stability and numerical underflow issues.

Most of the languages will tend to round the values, which will lead to rounding errors.

$$\left\{ \begin{array}{l} 0.2 \times 0.1 \times 0.2 \times 0.1 \\ = 0.0004 \end{array} \right. \quad \left| \begin{array}{l} \log(0.0004) = -2.23979 \\ \text{all of which are } 0 \text{ to } 1 \end{array} \right. \\ \xrightarrow{\text{100 numbers}} \begin{array}{l} \downarrow \\ \text{numerical stability} \end{array} \quad \begin{array}{l} \downarrow \\ \text{numerical underflow} \end{array} \quad \left\{ \begin{array}{l} \text{python} \\ \text{double precision} \\ \text{16 significant digits} \\ (\text{rounding}) \end{array} \right.$$

Instead of using these small values, we will use log of each probabilities. It makes more efficient.

We use log – probabilities:

Log-probabilities:

$$\log \left(\frac{P(y=1 | w_1, w_2, \dots, w_d)}{P(y=0 | w_1, w_2, \dots, w_d)} \right) = P(y=1) + \prod_{i=1}^d P(w_i | y=1) - P(y=0) - \prod_{i=1}^d P(w_i | y=0)$$

$\boxed{x \uparrow ; \log x \uparrow \text{monotonic fn}}$

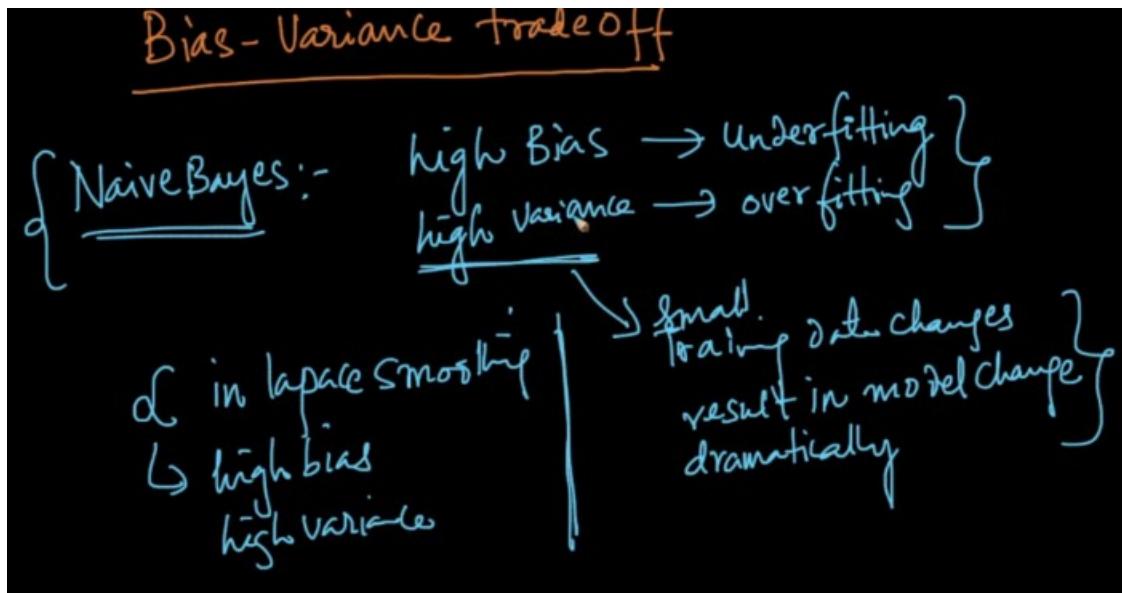
The product becomes summation for log probabilities (basic property of the log).

$$\log \left(\frac{P(y=1 | w_1, w_2, \dots, w_d)}{P(y=0 | w_1, w_2, \dots, w_d)} \right) = \log(P(y=1)) + \sum_{i=1}^d \log(P(w_i | y=1))$$

$$\log(a+b) = \underbrace{\log a}_{\text{mult}} + \underbrace{\log b}_{\text{addn}}$$

Bias – variance tradeoff:

In naive bayes the value of alpha judges the bias – variance tradeoff.



Case 1: when alpha = 0

When alpha = 0, then we will overfit the data because for the words that occur very less frequent also, we calculate the probability value.

Case 1:- $\alpha = 0$

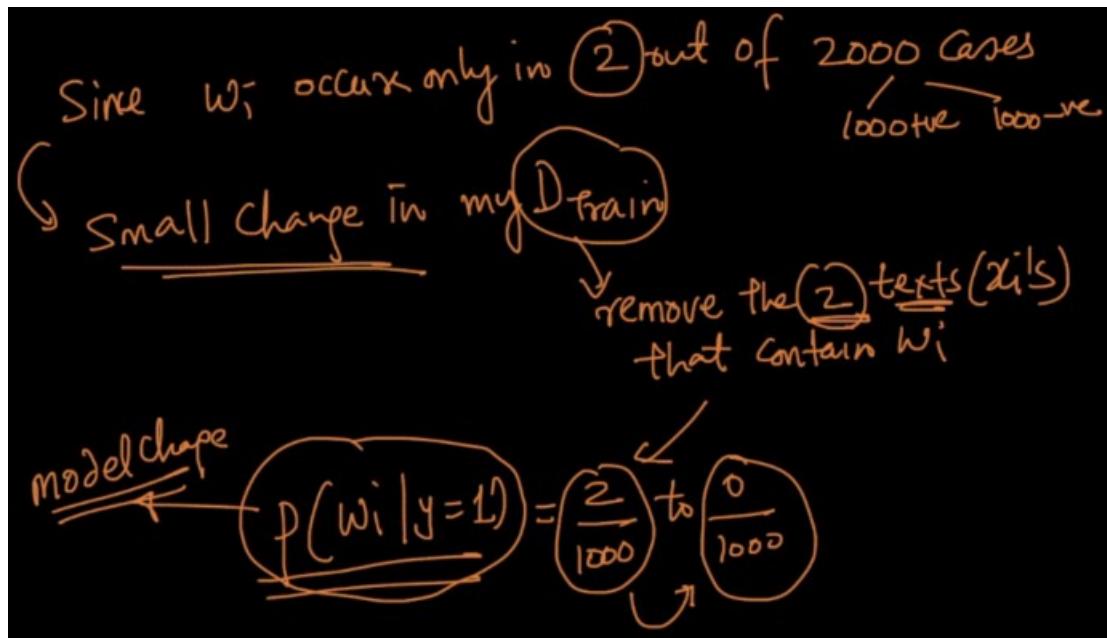
$$p(w_i | y=1) = \frac{\# \text{Train data pts } w_i \text{ occurs & } y=1}{\# \text{Train pts with } y=1}$$

$(n = 2000 \text{ pts} \rightarrow 1000 \text{ tve})$

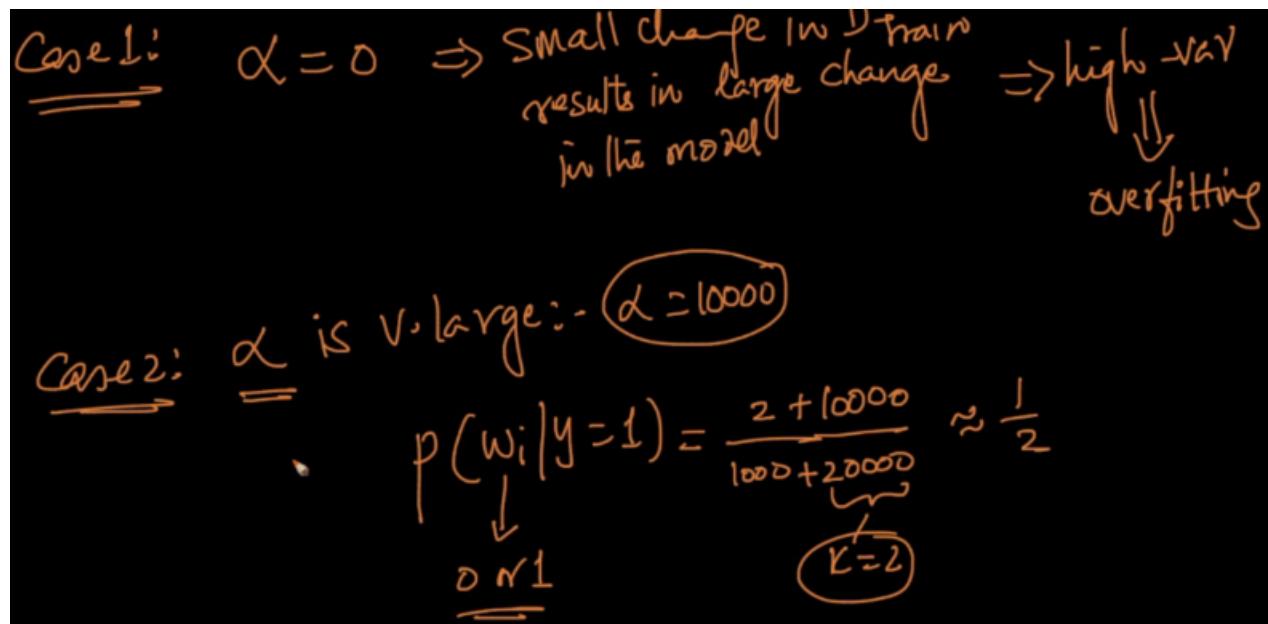
$$= \frac{② \rightarrow \text{only } 2 \text{ times}}{1000 \rightarrow \text{tve data pts}} \Rightarrow \underline{\underline{\text{rare}}}$$

{ Words that are rare } $\cup \{ p(w_i | y=1) \} \rightarrow \text{overfitting}$

By removing the two texts out of all the words, the probability of occurrence will become zero.



Case - 2 : alpha is very large.



If alpha is very large then every word in the text will move towards the uniform distribution, i.e., 1/2.

Given the model, that cannot distinguish between the two classes for large values of alpha.

This is the case of under fitting.

$$\begin{aligned}
 & \text{for } \alpha \approx 0 \\
 & p(w_i | y=1) \approx p(w_i | y=0) \approx \frac{1}{2} \\
 & \Downarrow \\
 & p(y=1 | x_w) \approx p(y=0 | x_w) \approx \frac{1}{2} \\
 & (\text{Underfitting}) \quad \left\{ \begin{array}{l} k-\text{NN} \\ k=n \\ x_q \rightarrow \text{Same class label} \end{array} \right.
 \end{aligned}$$

Comparision of NB with KNN:

$$\begin{aligned}
 & \alpha = \text{V-large:} \\
 & p(y=1 | w_1, w_2, \dots, w_d) = p(y=1) \cdot \prod_{i=1}^d p(w_i | y=1) \approx \frac{1}{2} \\
 & p(y=0 | w_1, w_2, \dots, w_d) = p(y=0) \cdot \prod_{i=1}^d p(w_i | y=0) \approx \frac{1}{2}
 \end{aligned}$$

When alpha is very large then the probability of the majority class wins over the minor class. In addition if we have high alpha values then every probability values of the words will be equal to half.

This is same as KNN with K is the hyper parameter.

$\alpha = \text{v-large}$:

$$P(y=1 | w_1, w_2, \dots, w_d) = \frac{n_1/n}{1 + \prod_{i=1}^d p(w_i | y=1)}$$

$$P(y=0 | w_1, w_2, \dots, w_d) = \frac{n_0/n}{1 + \prod_{i=1}^d p(w_i | y=0)}$$

Compare

$\alpha \rightarrow y_{\text{all}} = 1$

$n \rightarrow n_1 \text{ +ve}$

$n \rightarrow n_2 \text{ -ve}$

$n_1 > n_2$

Summary:

Case 2: $\alpha \rightarrow \text{very large}$

Underfitting \rightarrow high bias

Case 1: $\alpha = 0$

Overfitting \rightarrow high var

How to find the right alpha ?

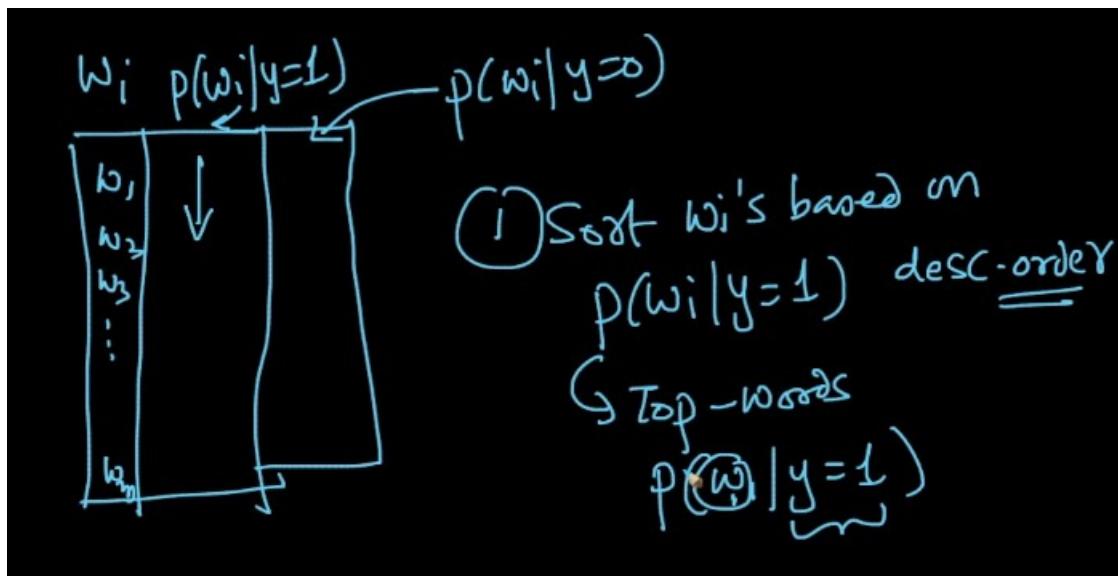
(Q) How to find the right α

\rightarrow KNN ; right $\alpha \rightarrow$ using simple CV
or logistic CV

\rightarrow right $\alpha: \rightarrow$ simple CV
or logistic CV

Feature importance and interpretability:

For every word w_1 and probability of likelihood for the both classes is calculated.



The words which have the highest probability are very useful or important features or words in determining the data point belongs to that class.

Those words are very important word or feature that a point is more important.

w_i 's which have high values of $P(w_i | y=1)$ → important words/features in determining that point is +ve class

$P(w_i | y=1) \rightarrow$ high
 $\hookrightarrow w_i$ is important word/feature

Feature importance in naive bayes is trivial, for a class we can figure out the words with highest value of probability.

Feature importance can be obtained directly from the model.

feature importance in NB:

{ +ve class:- find words (w_i) with highest value of $P(w_i | y=1)$ ✓ }

✓ { -ve class : " $P(w_i | y=0)$ ✓ }

→ determined/obtained directly from the model

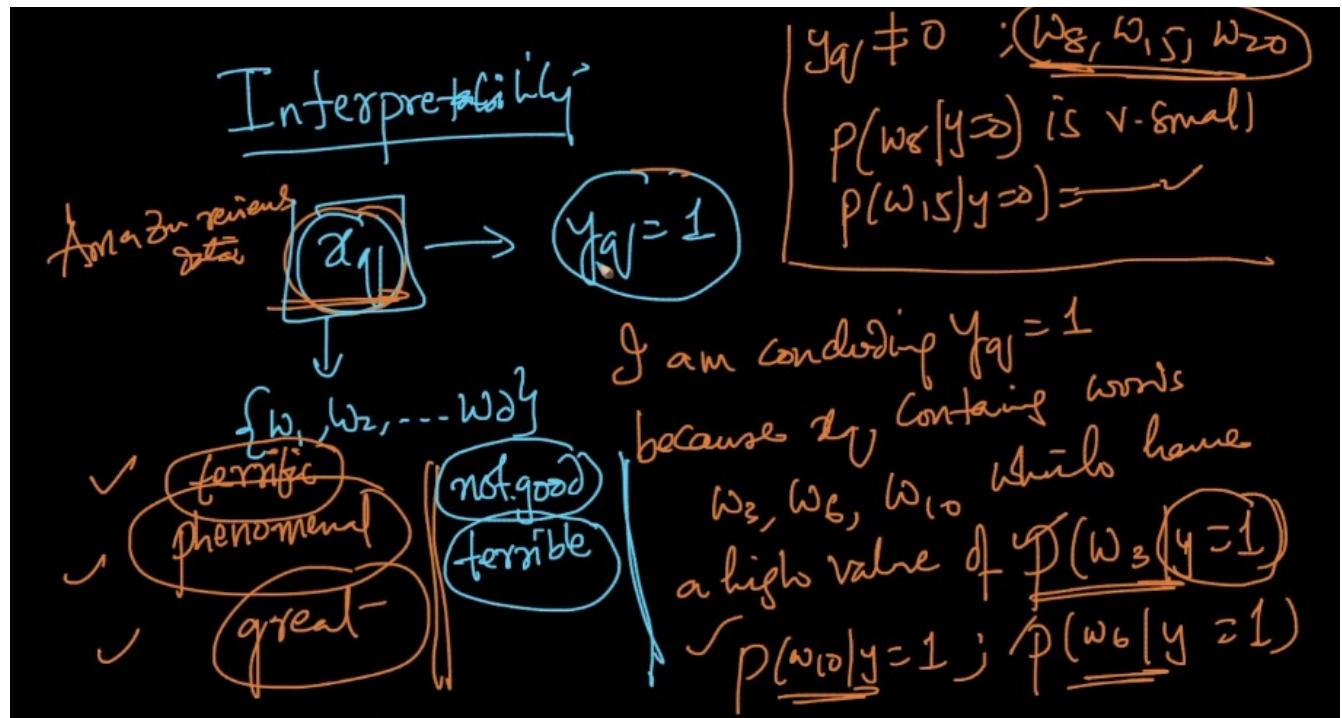
In KNN it is not possible to get feature importance, we used forward feature selection as the criteria.

Interpretability:

If the query point has the highest probable features, then we can conclude the query point belongs to the class, because the query point has the high probable words.

We can declare the query point as a positive or negative, if it contains the words that has the most probable value words of the classes. like (for pos – great, good, terrific, phenomenal, neg – not good, terrible).

This is how we interpret the queried review with the probability values of the training data.



The conditional probabilities are nothing but the likelihood values.

Imbalanced data:

In case of imbalanced data, the majority or dominating class has an advantage. Though the conditional probabilities are same (or) equal.

$$p(y=1 | w_1, w_2, \dots, w_d) = p(y=1) \prod_{i=1}^d p(w_i | y=1) \quad \checkmark$$

|| (let)

~~Compare~~

$$p(y=0 | w_1, w_2, \dots, w_d) = p(y=0) \prod_{i=1}^d p(w_i | y=0)$$

|| (0=1)

imbalanced data

① class priors → *majority | dominating class
has an advantage*

We can use the standard techniques of upsampling and down sampling. (or)
we can drop the prior probabilities.

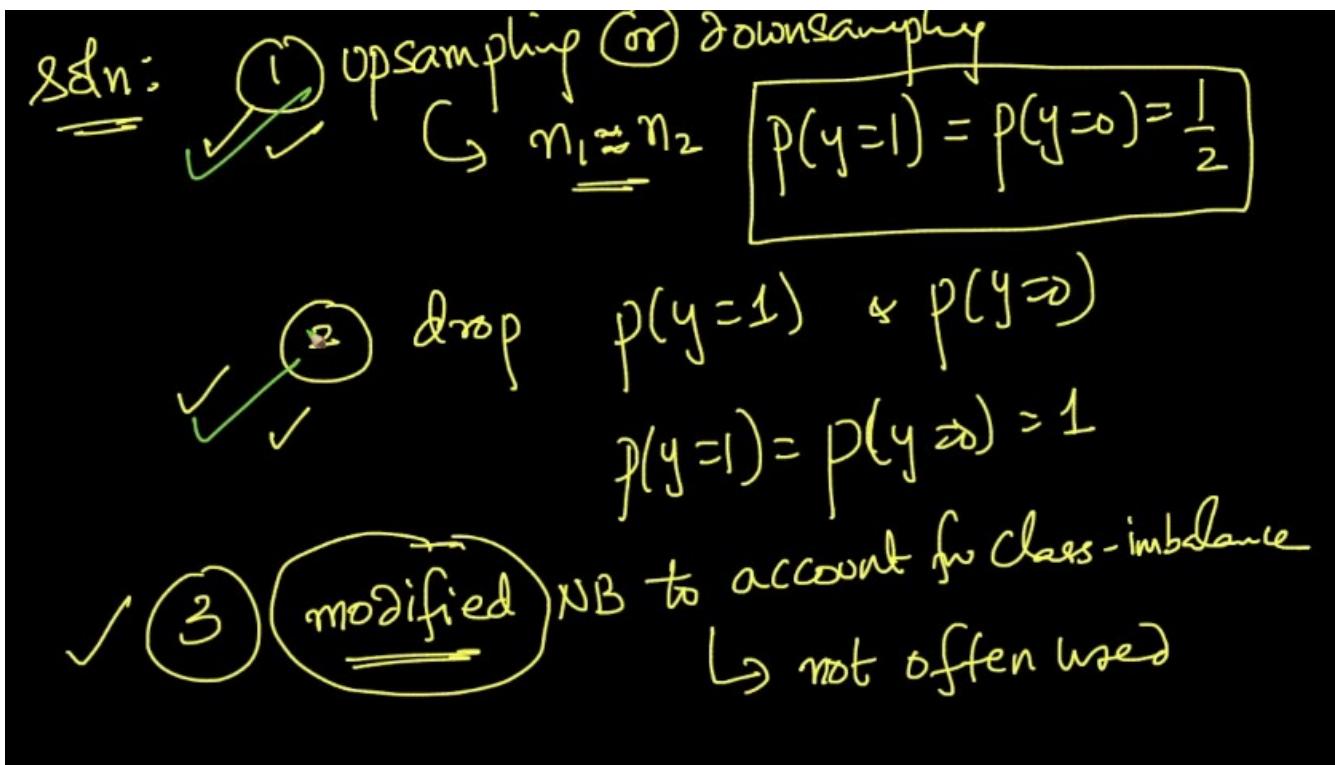
solution: ① upsampling or down sampling

$\hookrightarrow n_1 \approx n_2$

$p(y=1) = p(y=0) = \frac{1}{2}$

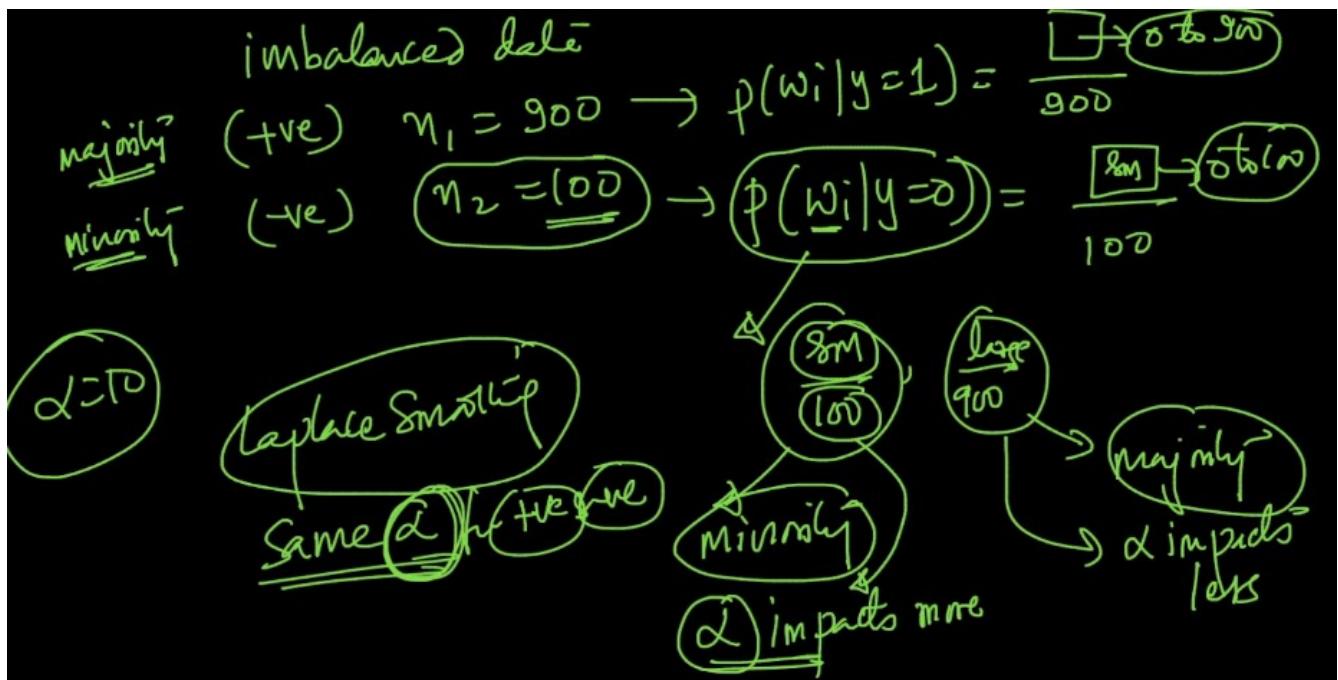
② drop $p(y=1)$ & $p(y=0)$

$p(y=1) = p(y=0) = 1$

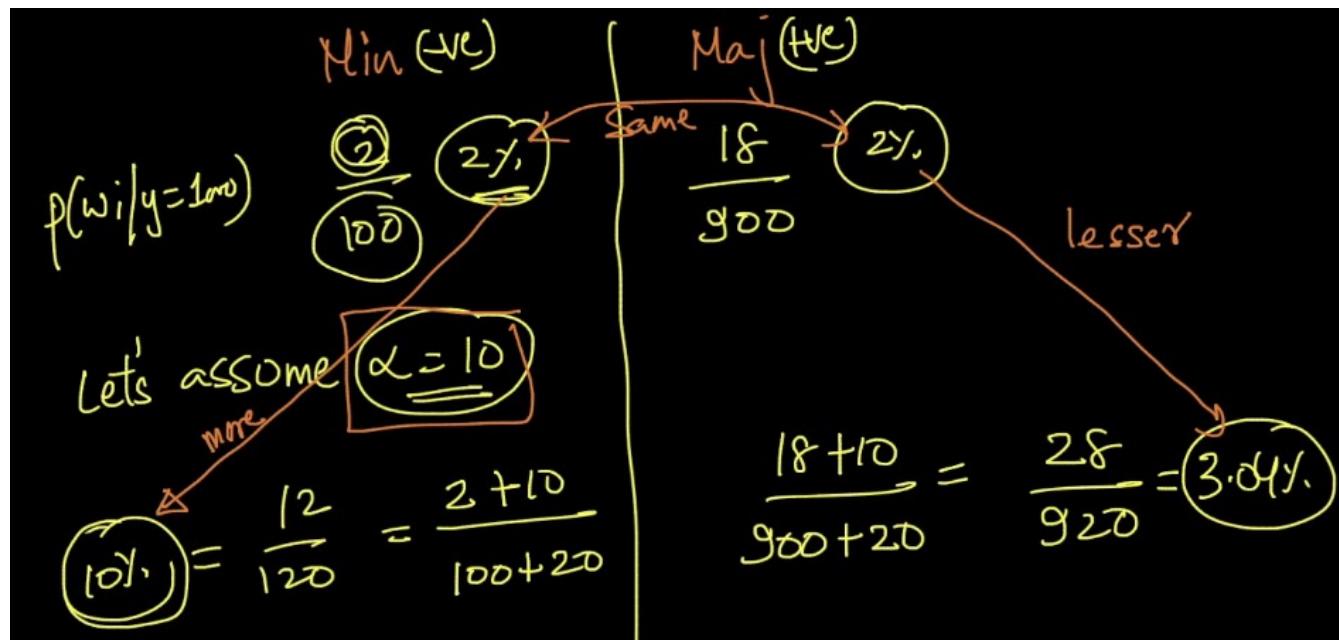


The majority class has the more range of values than the minority class range of values.

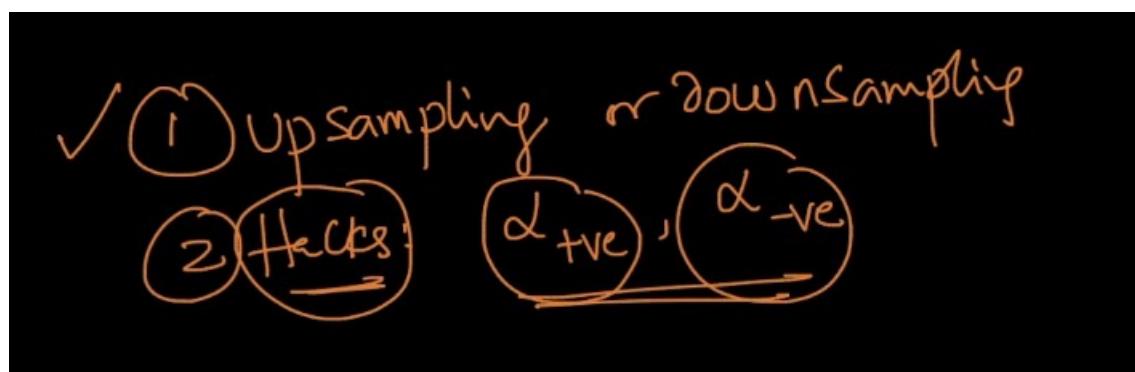
Impact of the laplace smoothing:



With the laplace smoothing, the minority changes a lot than the majority class. For the same alpha value.



Upsampling and down sampling can be used to make the affect of alpha less. We can use different alpha values for positive and negative classes.(this is a hack).



Naive bayes imacted by imbalanced data, it affects the prior probabilities and also impacts the likelihood probabilities for the same values of alpha.

Outliers:

During test time:

If we get the outlier during the test time, laplace smoothing takes care of the outlier point.

Outliers :-

NB

Text-classfn:-

outlier at test-time $\rightarrow \underline{x_1 = w_1, w_2, w_3, w'}$

$w' \notin \{w_1, w_2, \dots, w_m\} \leftarrow$ Set of words that I have seen in D_{train}

Laplace Smoothing

$$P(w/y=1) = \frac{O+\alpha}{n_1+2\alpha}$$

During outliers in training data:

If the words in the positive and negative classes occur very few times then this can be taught of an outlier.

Outlier can be taught of there are not many points around the point. This can be taught of an outlier.

Outliers in Training Data:

$\{w_1, w_2, w_3, \dots, w_m\} \leftarrow$ Set of words in D_{train}

$w_8 \rightarrow$ occurs $\overbrace{\text{very few times}}$ in pos & neg classes

outlier



First solution:

If a word occurs fewer than 10 times, then just ignore that word. Droping that one word can be the choive of doing.

Even if occurs in the traning data.

hack son:

if a word (w_j) occurs fewer than 10 times, then just ~~ignore~~ that word

$$w_j \notin \{w_1, w_2, \dots, w_m\}$$

Second solution:

The solution is using the laplace smoothing.

Outliers :-

NB

Text-classfmr:-

outlier at test time $\rightarrow \frac{\lambda_{q1} = w_1 w_2 w_3 w'}{w' \notin \{w_1, w_2, \dots, w_m\}}$ Set of words that I have seen in Train

Laplace Smoothing

$$P(w/y=1) = \frac{0+\alpha}{n_1+2\alpha}$$

Missing values:

In the case of text data, then there is no case of missing data.

In case of categorical data, we can use the new feature called NaN and make that as the new feature.

Missing-Values :-

① Text-Data :- Text : - $\{w_1, w_2, \dots, w_d\}$
↳ no case of missing data.

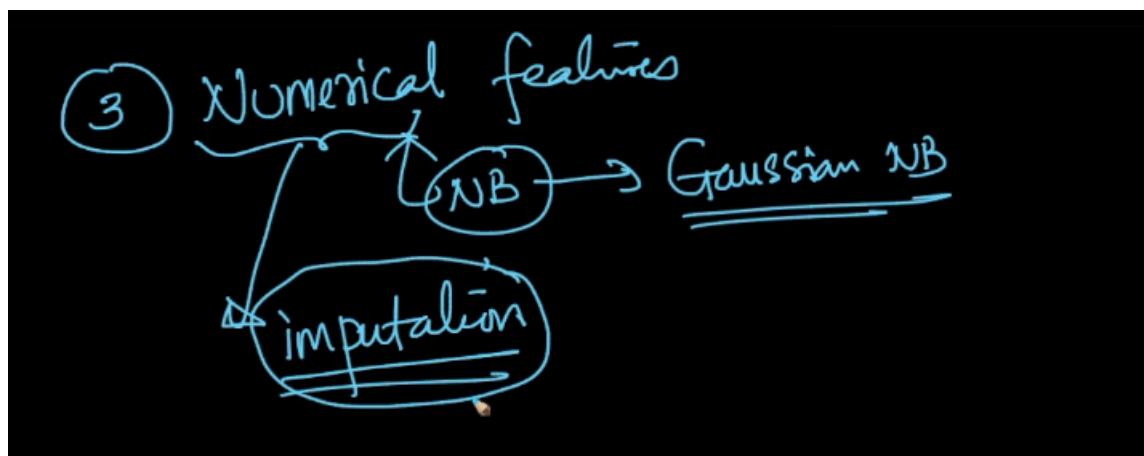
✓ ② Categorical features :-
• $f_i \in \{a_1, a_2, a_3\}$
→ consider NaN as a category
 $f_i \in \{a_1, a_2, a_3, \text{NaN}\}$

d_j

For numerical features:

We use the standard imputations techniques for the missing values.

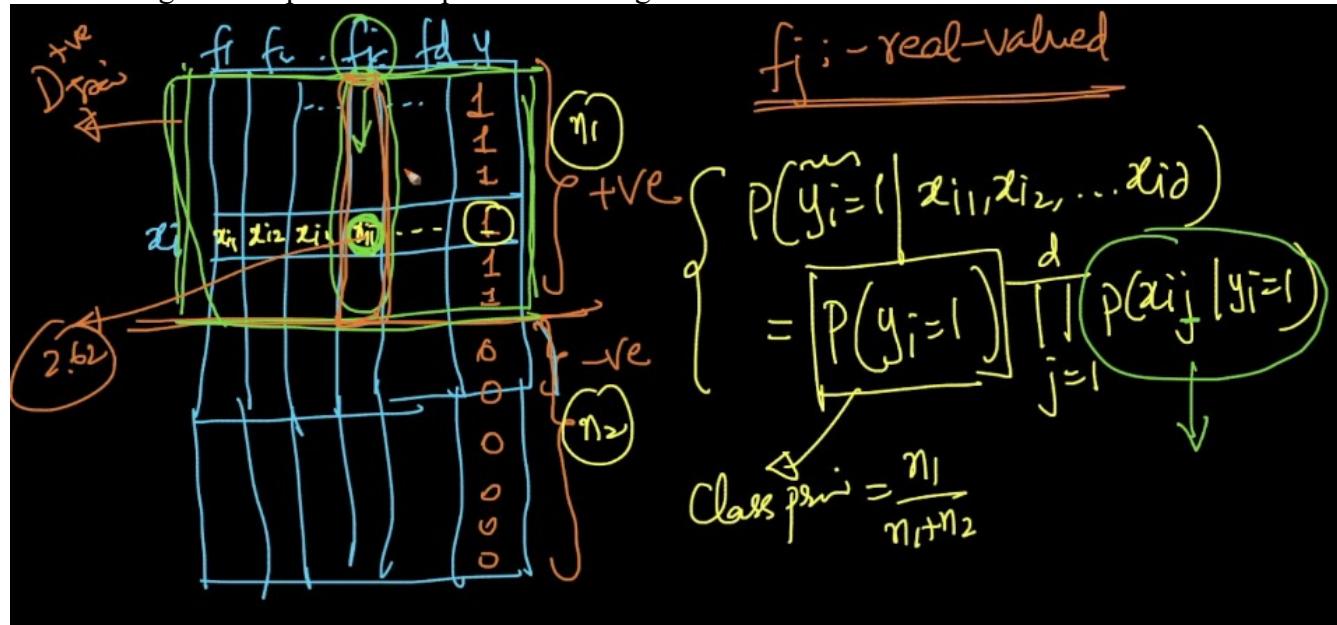
We use GaussianNB for numerical features.



Handling Numerical features(Gaussian NB):

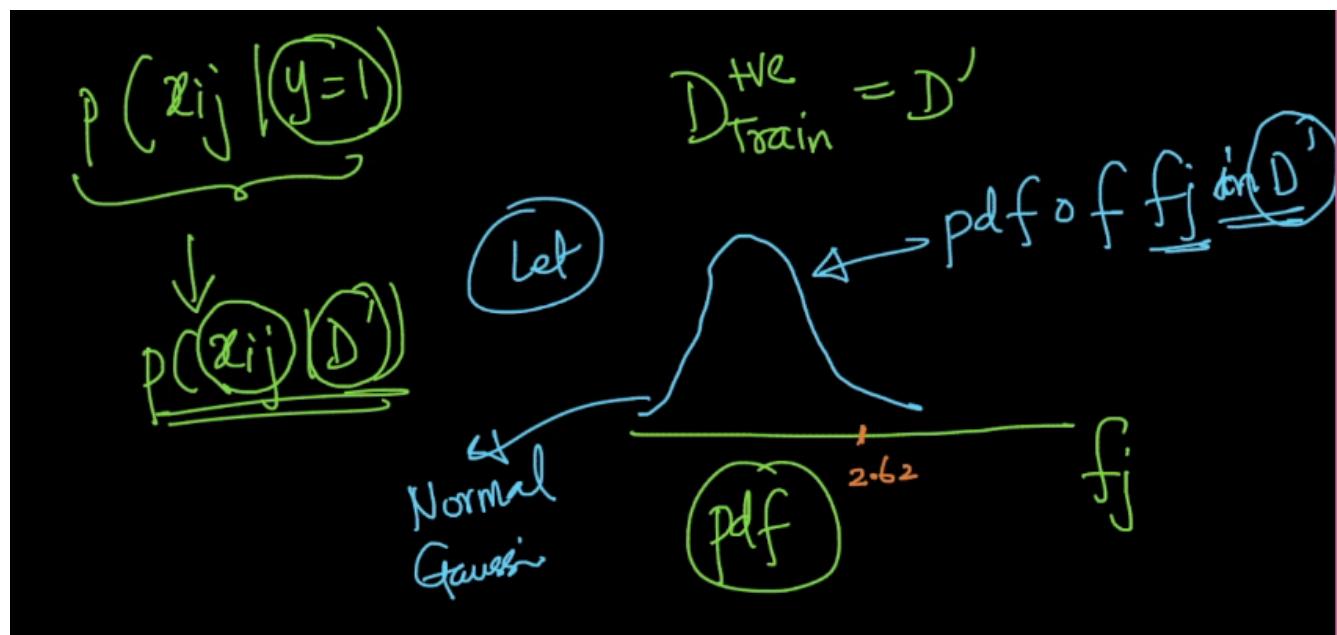
Step 1:

The training data is split into the positive and negative classes.

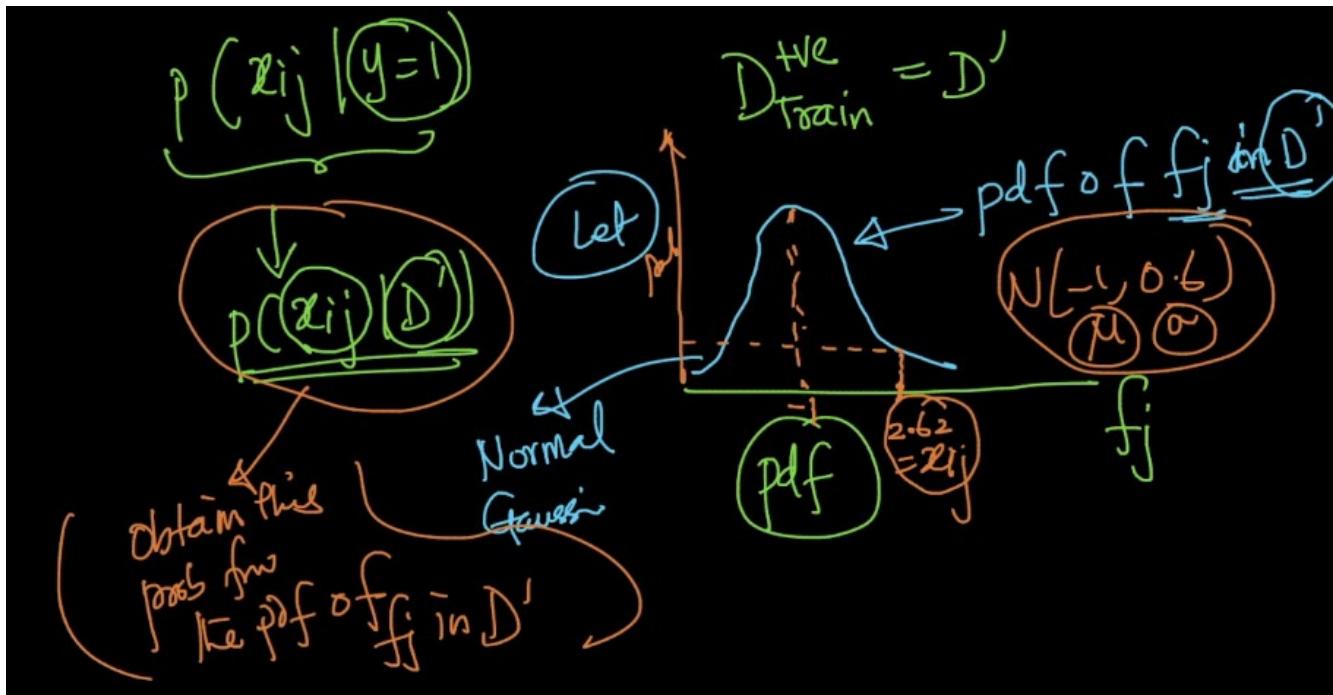


Calculating the probability for the two classes. The pdf is only plotted for only the positive labelled dataset (or class representative)

The pdf of the feature is plotted.

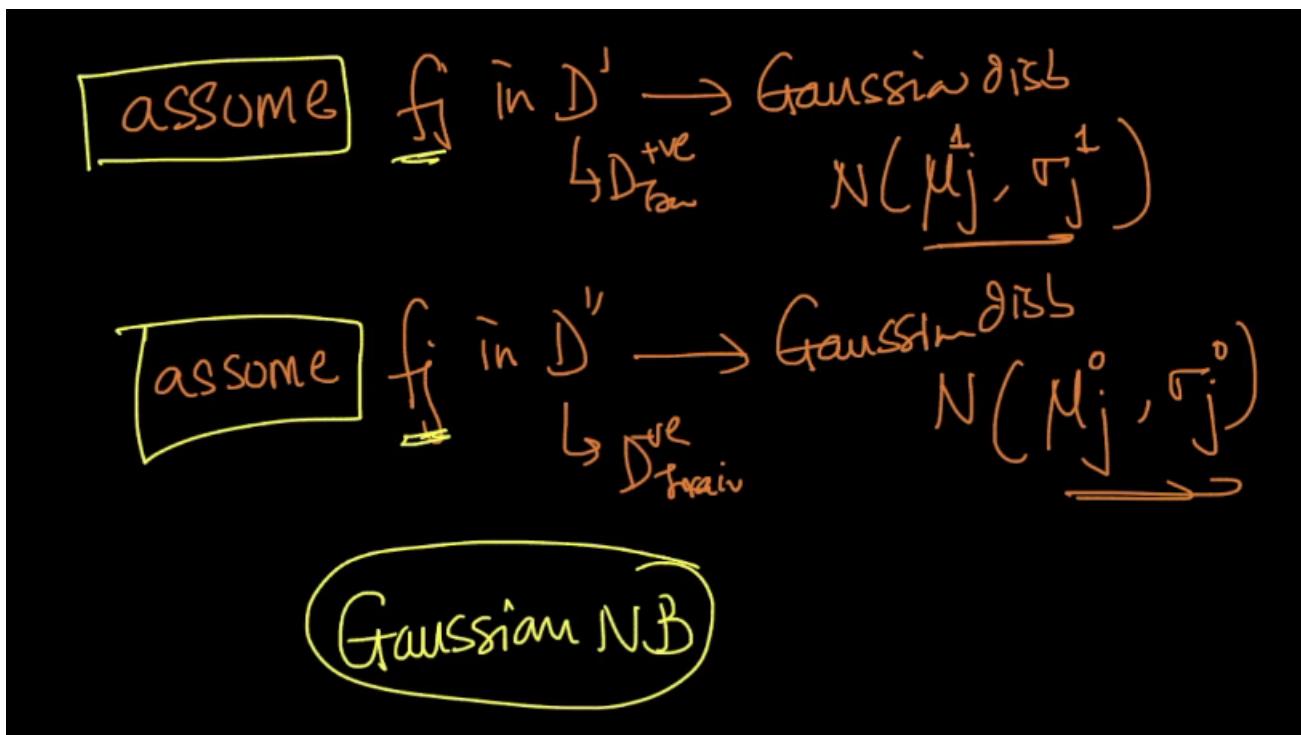


We can obtain the probability of single value from the feature by plotting the pdf of the feature.



Similarly we can get for the other class also.

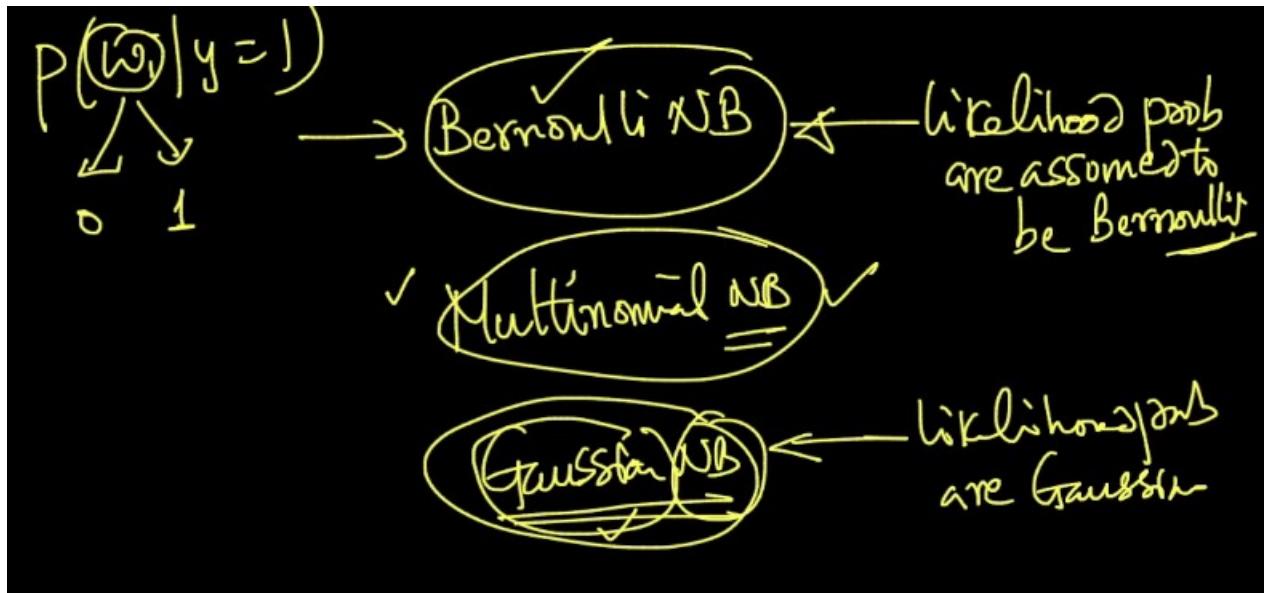
It is often assumed that the distribution of the feature is always a gaussian.



Such a model is called the gaussian naive bayes.

There are different types of NB such as:

- BernoulliNB (it is a binary random variable)
- MultinomialNB.
- GaussianNB.



Each of the feature is conditionally independent in case of Naive Bayes.

Multiclass classification:

We can compute priors and likelihood probabilities for any number of classes in the dataset. Then compare all of them.

Multi-class classfn:

(NB) → can do multi-class classfn

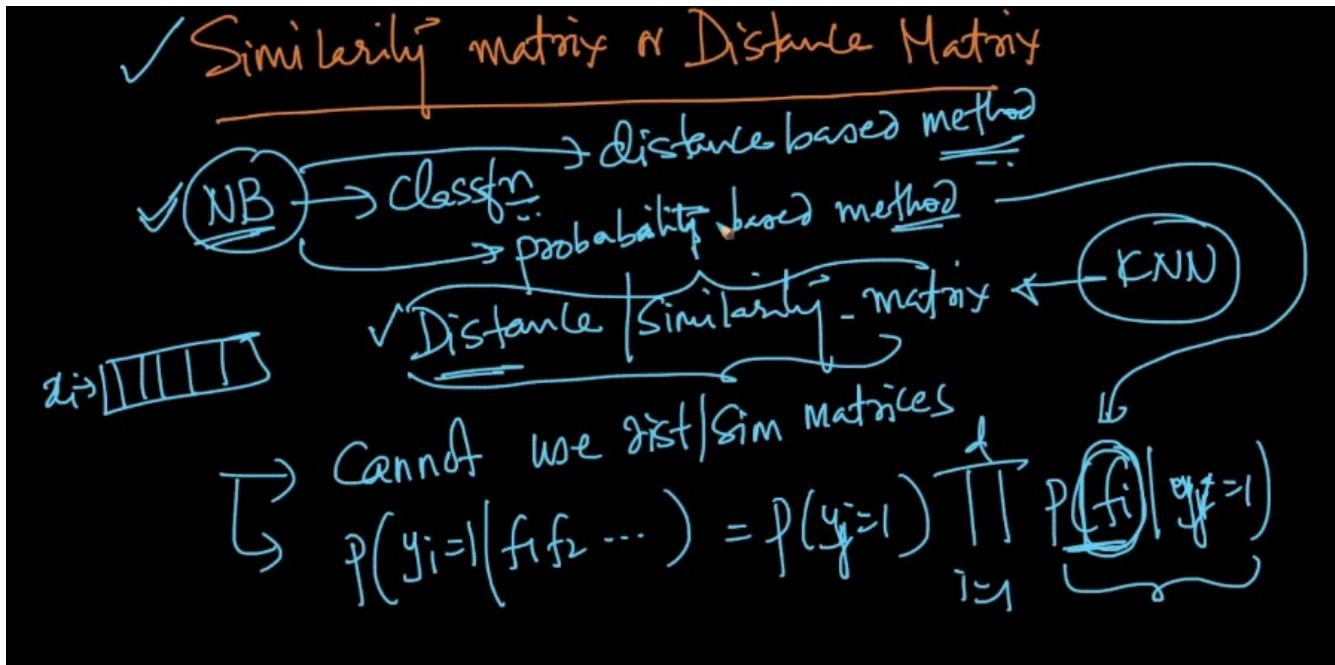
Compare

$$\left\{ \begin{array}{l} p(y_i=1 | w_1, w_2, \dots, w_d) \\ p(y_i=0 | w_1, w_2, \dots, w_d) \\ p(y_i=2 | w_1, w_2, \dots, w_d) \\ p(y_i=c-1 | w_1, w_2, \dots, w_d) \end{array} \right\} \rightarrow \text{is the largest } y_i = b \text{ given } \vec{x}_i = \{w_1, w_2, \dots, w_d\}$$

Similarity matrix or a distance matrix:

In general NB cannot use the distance or similarity matrix. We cannot compute the likelihood probabilities.

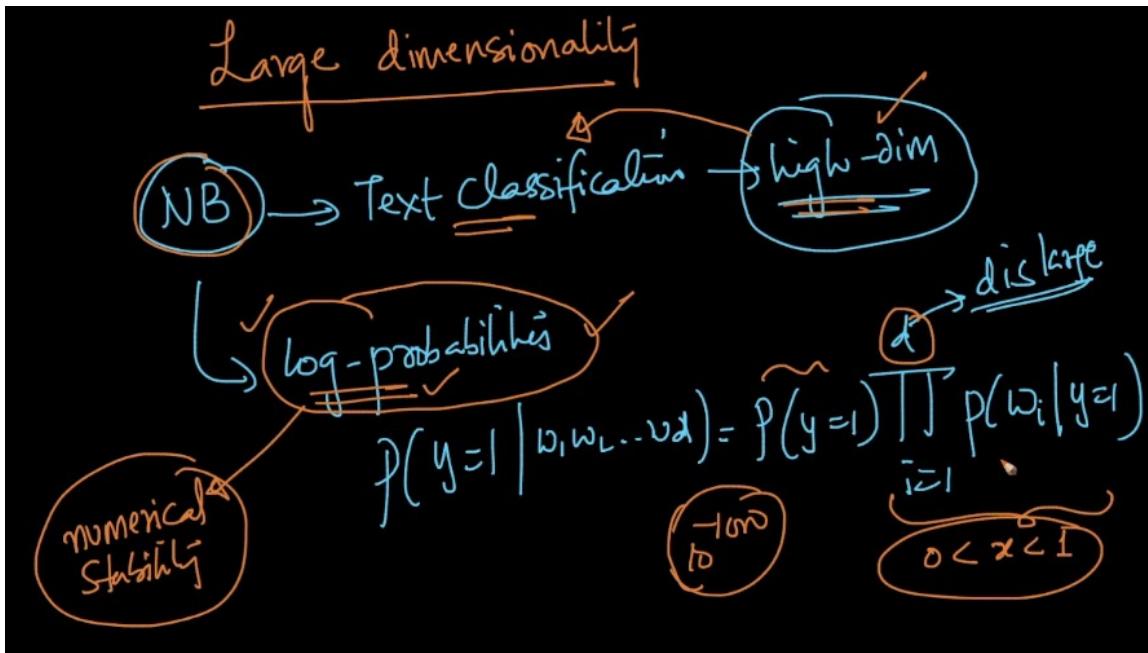
NB does not use the distance based algorithm. It is the probability based method.
We need actual feature values for calculations for probabilities.



Large dimensional values:

In case of large dimensions the calculation of likelihood probabilities multiplication can cause the

numerical stability issues. Use the **log probabilities** to overcome this problem in high dimensional data.

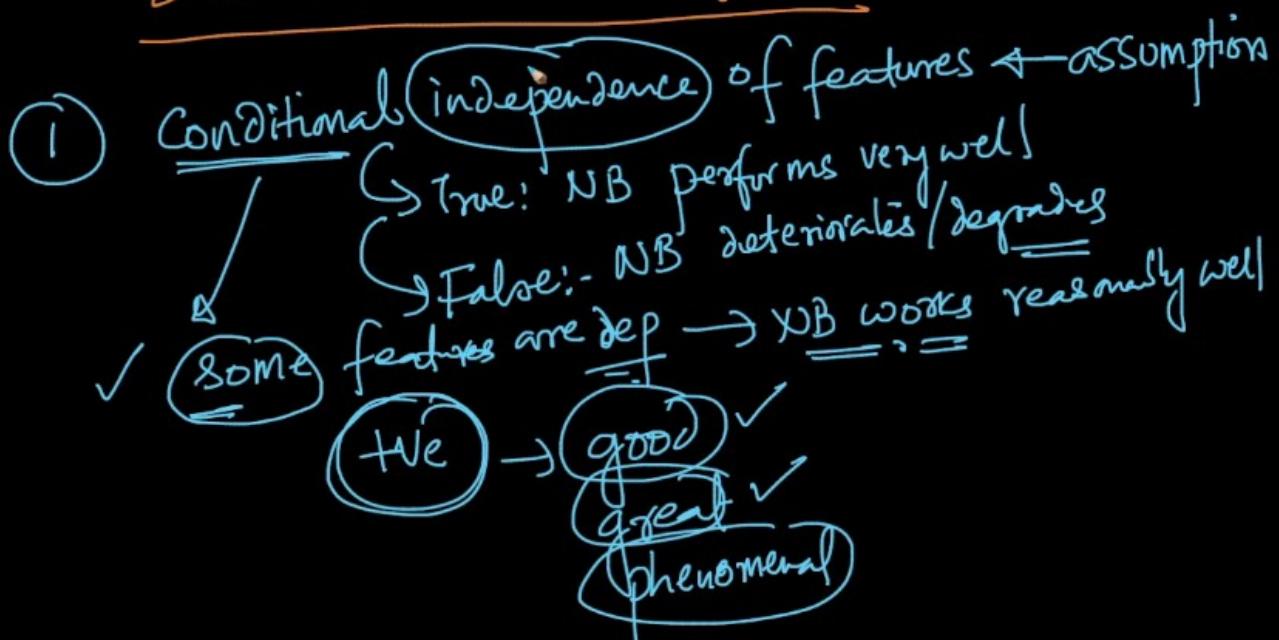


Best and worst cases of NB:

NB assumes conditional independence of the features assumption.

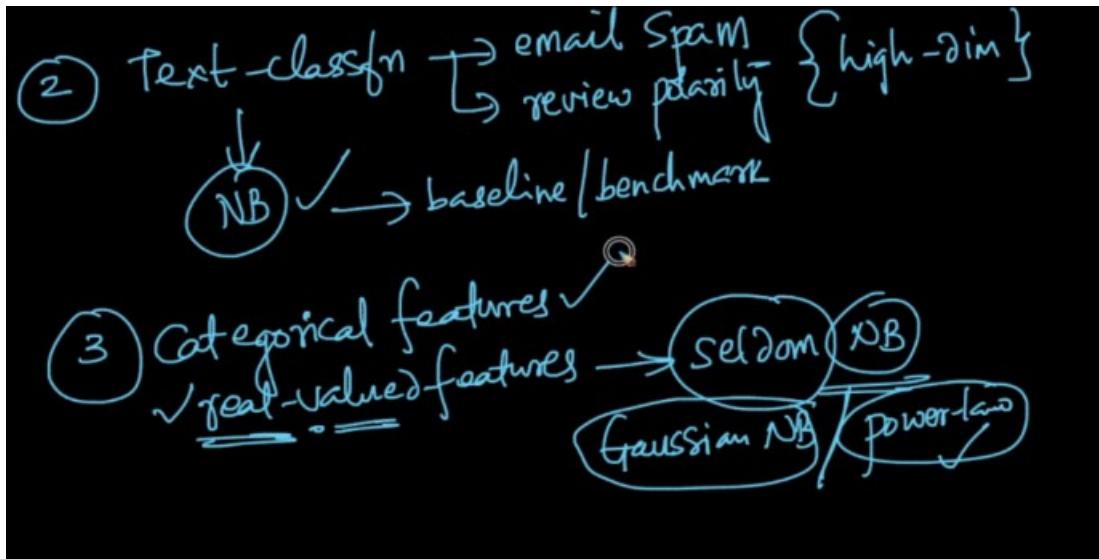
In some cases the words in text data does not mean too different, they mean one and the same like the words good, great and phenomenal.

Best & Worst Cases for NB



Even some features are dependent then naive bayes works reasonably well.

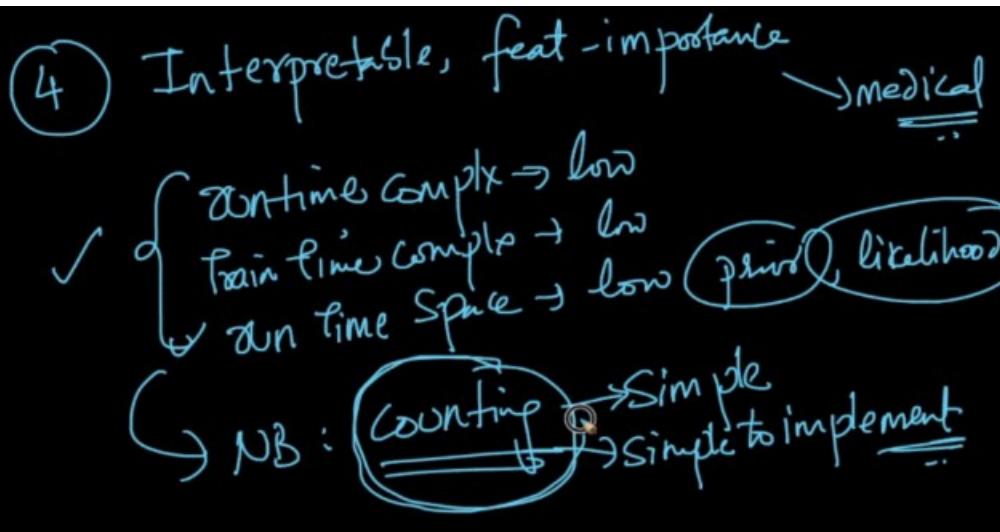
Theoretically, naive bayes works only when the features are independent.



Naive bayes is not most often used in case of real – valued features.

Naive bayes is super interpretable.

The run – time complexity is also low.



Naive bayes can easily be overfit the data. If there is not laplace smoothing.

