Naive Bayes on Text Data

- Performs super good on textual data as per observed

Ex We have data about reviews and sentiment

review	sentiment

Step 1: Text Preprocessing

such as → Remove html tags
→ " special characters.
→ Converting every thing to lower case
→ Removing stopwords
→ Stemming

Step 2: Vectorizing (text to numbers)

- Different techniques to Vectorization
  - BOW
  - OHE
  - TFID
  - Embedding

Step 3: apply Naive Bayes

Bag of Words (Vectorization)

Ex	reviews	sentiment
	liked the movie	+ve
	hated the movie	-ve
	-	-
	-	

- BOW will include all the available words in text
- Let's say we have 10 lakh words
- You see in that 10 lakh, there were 35000 unique words
- and in among 35000, let's say you will choose 5000 most frequent words

Ex : hate, like, movie, actor, ... 5000 words

- What you'll do is you will visit every review and ask how many times a review contain hate, like, and all those 5000 words

Ex

Most frequent words	hate	like	movie	. . . . .
Review 1	0	1	1	. . . . .
Review 2	2	0	3	. . . . .
Review 3	0	3	4	. . . . .
⋮	-	-	-	
⋮	-	-	-	
Review 50000	-	-		

- Now we have the dataset, which contains 50000 rows and 5000 columns. (Vectorization)
- Now, let's say we have a new query (review), we have to predict its sentiment
- Review was: "Adipurush is great movie, I liked it very much"
- Now we will try to represent text in review same as in columns we have
- We had 5000 columns:

Review	hate	like	movie	actor .....
Adipurush:	0	2	3	5

- This tells how many times word hate, like, actor has come in review of adipurush
- Basically, we converted into vector of dimension $\Rightarrow (1, 5000)$
- Now we have to tell if the query is +ve or -ve
- We have to find 2 probabilities

↓

P(+ve | -ve)

P(+ve | hate=0, like=2.....) & P(-ve | hate=0, like=2.....)

P(+ve | hate=0, like=2) = P(hate=0|+ve) X P(like=2|+ve)..... XP(+ve)

- similarly we will calculate the -ve prob
- let's say we have +ve prob = 0.8, and -ve = 0.1, hence we will assign adipurush sunrise positive
- refer to code

Date - 20-May