\* In aptitude test questions are comes from these topics: -permutations and combinations, profit & loss, average, HCF and LCM. Refer ‘indiasbig’ website to study about these topics. Study minimum 4 questions of each topic from this website.

\* For textual data use multinomial naïve bayes and for tabular data use gaussian naïve bayes.

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Part of Speech Tagging (POS)

\* POS means verb, adverb, noun, pronoun, punctuation mark, etc.

\* POS tagging works is to label each word with their respective POS. It is required because one same word has different meanings in different sentences. For e.g.: - Same word is noun and same word is verb in different sentences.

\* When we using same word as noun it’s meaning is different and using same word as verb so their meaning is different. Word is same but their meaning becomes different according to their POS.

E.g.: - **‘I like google’ and ‘I want to work in google’.**

\* In first sentence google is verb (kriya) and in second google is noun.

\* As human we, understand the meaning of word in sentence, but model is not capable of it. If we directly pass ‘google’ word into model then it gets confuse to recognise the meaning of google in sentence. It treats ‘google’ word same in both sentences.

\* That’s why we are studying POS that it will tag each word with their respective POS.

E.g.: -

**\* D1 = ‘I left the room’**

**\* D2 = ‘Left of the room’**

\* In D2, ‘left’ refers to direction and in D1 ‘left’ means leave. In both sentences ‘left’ word has different meaning.

E.g.: -

**\* D1 = ‘I ran towards the shop’.**

**\* D2 = ‘I ran out of milk’.**

\* In both sentences ‘ran’ word has different meaning.

Why POS is needed?

\* One word has different meaning in different sentence. At some place same word is noun, in another sentence same word is verb, in another sentence same word is adjective, etc.

\* To deal with this problem we have POS.

\* When discussion happens in interview regarding POS, then talk about why POS is needed.

\* POS tag each word, such as which word is verb, which is noun, etc.

E.g.: -

**\* D1 = Why not tell someone?**

\* Why = adverb, not = adverb, tell = verb, someone = noun, ? = punctuation.

\* It will tag each word of sentence like this. This is the work of POS to tag or label each word.

\* We make a model which will tag each word like this.

How it’s works?

**\* D1 = ‘Nitesh loves campus.’**

**\* D2 = ‘Can Nitish google campus.’**

**\* D3 = Will Arihant google campus?’**

**\* D4 = ‘Arihant loves Will.’**

**\* D5 = ‘Will loves google.’**

\* We take small sentence as example, some words we take which have multiple meanings according to the nature of sentence. Same word has different POS according to the nature of sentence.

\* In these documents some words have 2 meanings. In D2 ‘google’ is verb and in D5 it is noun.

\* In POS we first assign POS to words manually. Like in D1 ‘Nitesh’ is noun, ‘loves’ is verb, ‘campus’ is noun. In D2 ‘Can’ is model verb, ‘Nitesh’ is noun, ‘google’ is verb, ‘campus’ is noun. In D3 ‘will’ is model verb, ‘Arihant’ is noun, ‘google’ is verb and ‘campus’ is noun. In D4 ‘Arihant’ is noun, ‘loves’ is verb and ‘Will’ is noun. In D5 ‘Will’ is noun, ‘loves’ is verb and ‘google’ is noun.

\* First, we do this thing, tag each word with their respective POS manually like above.

\* Then we have calculated 2 things: -

(I) Emission probability

\* Emission means each word repeats how many times in corpus.

\* To find emission probability, we look at all unique words in vocabulary / dictionary. In above example vocabulary is (Nitesh, loves, campus, google, will, arahant, can).

\* We calculate both these by using given data.

|  |  |  |  |
| --- | --- | --- | --- |
|  | noun | model verb | verb |
| nitesh | 2 | 0 | 0 |
| loves | 0 | 0 | 3 |
| campus | 3 | 0 | 0 |
| google | 1 | 0 | 2 |
| will | 2 | 1 | 0 |
| arihant | 2 | 0 | 0 |
| can | 0 | 1 | 0 |

\* We take only 3 POS (noun, model verb and verb) here, but in real time POS will be more, such as pronoun, adjective, etc.

\* First task is to create matrix like this.

\* Then see difference part of speech in vocabulary and count which part of speech is used for each word in corpus.

\* From above matrix we have to find emission probability like this: -

|  |  |  |  |
| --- | --- | --- | --- |
|  | noun | model verb | verb |
| nitesh | 2/10 | 0/2 | 0/5 |
| loves | 0/10 | 0/2 | 3/5 |
| campus | 3/10 | 0/2 | 0/5 |
| google | 1/10 | 0/2 | 2/5 |
| will | 2/10 | 1/2 | 0/5 |
| arihant | 2/10 | 0/2 | 0/5 |
| can | 0/10 | 1/2 | 0/5 |
| **TOTAL** | 10 | 2 | 5 |
|  |  |  |  |

\* These are known as the emission probabilities. These are use in further calculations.

(II) Transition probability

\* Transition Probability means probability of converting from one state to another.

\* Next step is to calculate transition probability. Transition probability means what is probability of convert one POS to another, such as noun to verb probability, verb to adverb probability, adverb to model verb probability, like this.

\* To calculate transition probability, we initialize starting and ending point like this.

**\* D1 = Starting point - ‘Nitesh loves campus.’ – ending point**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | noun | model verb | verb | ending point |
| starting point | 3 | 2 | 0 | 0 |
| noun | 0 | 0 | 5 | 5 |
| model verb | 2 | 0 | 0 | 0 |
| verb | 5 | 0 | 0 | 0 |
|  |  |  |  |  |

\* Next, we have to see that from starting and ending point how many times, each POS is present, like noun is present how many times at starting point.

\* In second line (noun, noun) means how many sentences starts from (noun, noun), means first 2 words of sentence are noun.

\* We have to find transition probability by using this matrix like this: -

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | noun | model verb | verb | ending point | **TOTAL** |
| starting point | 3/5 | 2/5 | 0/5 | 0/5 | 5 |
| noun | 0/10 | 0/10 | 5/10 | 5/10 | 10 |
| model verb | 2/2 | 0/2 | 0/2 | 0/2 | 2 |
| verb | 5/5 | 0/5 | 0/5 | 0/5 | 5 |
|  |  |  |  |  |  |

\* These are the **transition probabilities**.

\* By using these probabilities, it will be predicting or tagging each word with their respective POS.

E.g.: - Sentence = ‘Will will google campus.’

\* First will is ‘model verb’, second ‘will’ is noun, ‘google’ is verb and ‘campus’ is noun.

\* Actually, we have to predict that which word has which part of speech.

\* To do this, what will do?

\* Suppose we take all words as noun, like:

Emission

Probability 1/5 1/5 1/10 3/10

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Starting point======🡺 | will (noun)======🡺 | will (noun)======🡺 | google (noun)======🡺 | campus (noun)======🡺 | ending point |

Transition

Probability 3/5 0/0 0/0 0/0 1/2

\* Now it will calculate that what is the probability of ‘will’ as noun, ‘google’ s noun and ‘campus’ as noun.

\* Consider emission and transition probabilities of ‘will’ as a noun, ‘google’ as noun and ‘campus’ is a noun.

\* Upper side we have emission probabilities and lower side we have transition probabilities.

\* After calculating emission and transition probabilities, multiply all probabilities, results in a single value. In case that value comes out to be ‘0’ because multiple any number with ‘0’ result as ‘0’.

\* In laymen language, there is 0 probability that all words of this sentence are noun.

\* Next, we see another possible probability like: -

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| starting point=============🡺 | will (verb) =============🡺 | will (verb) =============🡺 | google (verb) =============🡺 | campus (verb) =============🡺 | ending point |
|  |  |  |  |  |  |

\*It calculates value for all permutations and combination like this. For this sentence 81 permutations and combinations are made and every permutation and combination have probability.

\* For which permutation and combination probability is highest, it will select that as output and convert POS of document according to that permutation and combination, which is called as Final Combination.

\* Maximum of the permutation and combinations have probabilities as 0. If emission or transition probability for any word or POS is 0 then output probability comes out to be 0.

Calculate output by using original POS of sentence: -

1/2 1/5 2/5 3/10

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Starting point======🡺 | will (model verb)======🡺 | will (noun)======🡺 | google (verb)======🡺 | campus (noun)======🡺 | ending point |

2/5 1 1/2 1 1/2

\* After calculating emission and transition probabilities of this we get some value, suppose that value is 0.6. So, this is the highest probability of permutation and combination among all permutation and combination.

\* So, in this same way it will predicting that ‘will’ is model verb, second ‘will’ is noun, ‘google’ is verb and ‘campus’ is noun.

**\* This is how POS tagging works internally.**

\* POS tagging takes more time in code execution so; it is recommended that use google Collab when working on POS.

\* POS process is very time consuming because it is doing many irrelevant calculations.

\* To optimize the process / calculations of POS, there is a technique called ‘Viterbi Algorithm’.

E.g., of Viterbi Algorithm: -

Sentence = ‘will will google campus’

\* Suppose all 4 words have probability that they can be noun, model verb or verb.

\* What is the probability that ‘will’ is a noun?

Working process of Viterbi Algorithm: -



