**Introduction:**

My solution uses multiclass classification technique to predict the tags based on the job description.

After exploring training dataset, I came to know that maximum 5 tags out of 12 are possible because few tags are mutually exclusive.

Type 1:  "part-time-job"  /   "full-time-job"

Type 2:  "hourly-wage"  /  "salary"

Type 3:  "associate-needed"  /  "bs-degree-needed"  /  "ms-or-phd-needed" / "licence-needed"

Type 4:   "1-year-experience-needed"   /   "2-4-years-experience-needed" / "5-plus-years-experience-needed"

Type 5:  "supervising-job"

**About datasets:**

***train.tsv*** data set having tags & description, for few descriptions tags are not present so it was the challenge to handle these records.

The datasets ***train.tsv & test.tsv*** description having many special words like â€¢ . So it was very difficult to handle. Few algorithms are giving error because of these special characters.

**Preprocessing steps:**

* Read training dataset by *readLine* method because of another method like *read*.*table* or *read*.*delm* will not work properly because few descriptions don't have any tags.
* Create a data frame with 13 column. 1 column for description and other 12 column for tags. I am initializing the data frame with 0 and NA because I am using for loop to set the tag value so it will take very less time in function because space will not be initialized during a loop.

data.frame': 4375 obs. of 13 variables:

$ desc : chr "THE COMPANY Employer is a midstream service provider to the onshore Oil and Gas markets. It is a a fast growing filtration "| \_\_truncated\_\_ "ICR Staffing is now accepting resumes for Industrial Maintenance Mechanics. This is an experienced position, requiring prior ha"| \_\_truncated\_\_ "This is a great position for the right person. HEALTHCAREseeker.com (www.example.com) is an established company that places Re"| \_\_truncated\_\_ "A large multi-specialty health center is expanding it's Adult Medicine department and is seeking to hire a F/T or P/T Internist"| \_\_truncated\_\_ ...

$ tag1 : num 0 0 1 0 0 0 0 0 0 0 ...

$ tag2 : num 0 1 0 0 1 0 1 0 0 1 ...

$ tag3 : num 0 0 0 0 0 0 0 0 0 1 ...

$ tag4 : num 0 1 0 0 0 0 0 0 0 0 ...

$ tag5 : num 0 0 0 0 0 1 0 0 0 0 ...

$ tag6 : num 0 0 0 0 1 0 0 1 1 0 ...

$ tag7 : num 0 0 0 0 0 0 1 0 0 0 ...

$ tag8 : num 1 0 0 1 0 0 0 0 0 0 ...

$ tag9 : num 0 0 0 0 0 0 0 0 0 0 ...

$ tag10: num 0 1 0 0 0 0 0 0 1 0 ...

$ tag11: num 1 0 0 0 1 1 1 1 0 0 ...

$ tag12: num 1 0 0 0 1 0 0 0 0 0 ...

* Create 5 vector for 5 types of tag.
* Read the test dataset in the data frame.
* Merge training and test data set for processing only description column.
* Create *DocumentTermMatrix* from a description by below steps.
* Convert merged dataset into corpus data.
* Converting descriptions in lower character.
* Delete punctuation.
* Delete English words like a, an, the, this, that, etc.
* Uses stem function to stem the words.
* Create *DocumentTermMatrix* from corpus dataset.
* Create frequency matrix from *DocumentTermMatrix.*
* Remove Sparse Terms from a frequency *DocumentTermMatrix.*
* splitting the data set back to training and test dataset.

**Model justification:**

My solution uses multiclass classification technique to predict the tags based on the job description. So here, I am using ***XGboost*** algorithm because it can do parallel computation on windows. It is faster than classical ***glm.***

One of the special features of is ***xgb.train*** the capacity to follow the progress of the learning after each round, because of the way boosting works.

**Package required:**

install.packages("xgboost")