**RESUME SCORING USING NLP**

**PROJECT REPORT**

****

**SUBMITTED BY**

**Meghna Lohani**

**16BCE1395**

**Aman Sharma**

**16BCE1224**

**(April 2019)**

**ABSTRACT**

The objective of the project is to create a Resume Scoring algorithm using Natural Language Processing. The algorithm will parse resumes one by one and will create a Candidate Profile based on the skills mentioned in the resume.

A corpus is created using Sketch Engine, Wikipedia pages for various required skills (example : Machine Learning, Data Science, Software developer, Programming) are given as input,

Sketch Engine creates a corpus from the useful data available on the given links.

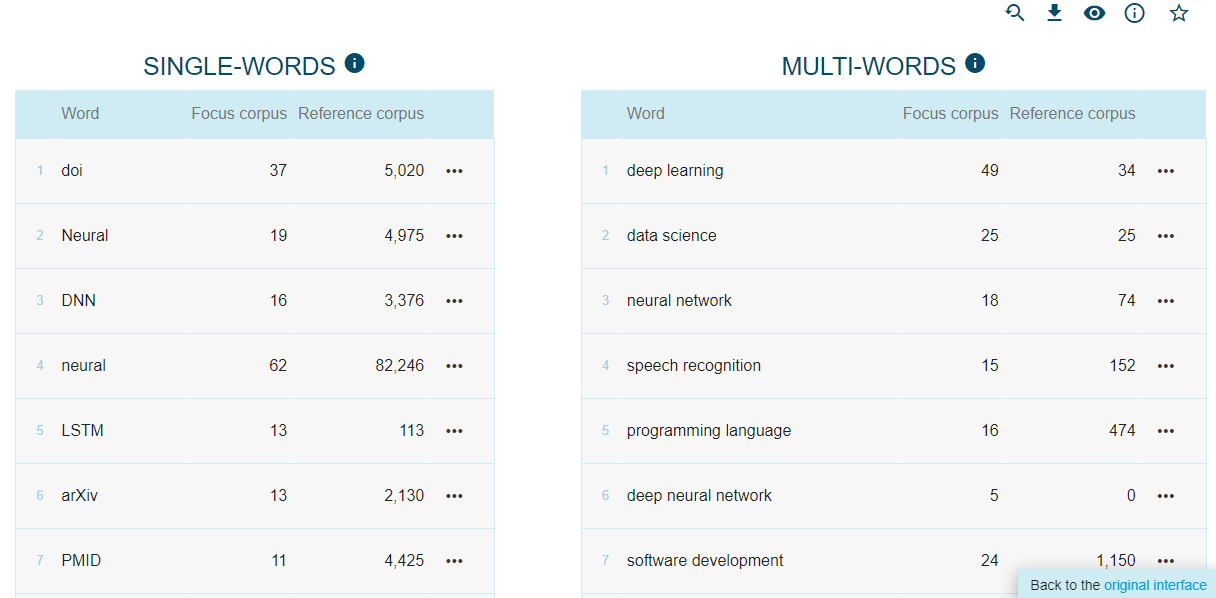
Word Embeddings are created from the corpus and these are used to match the skills in the candidate resume with the required skills.

Finally, the candidate profile is built and plotted as a bar graph for better visualization.

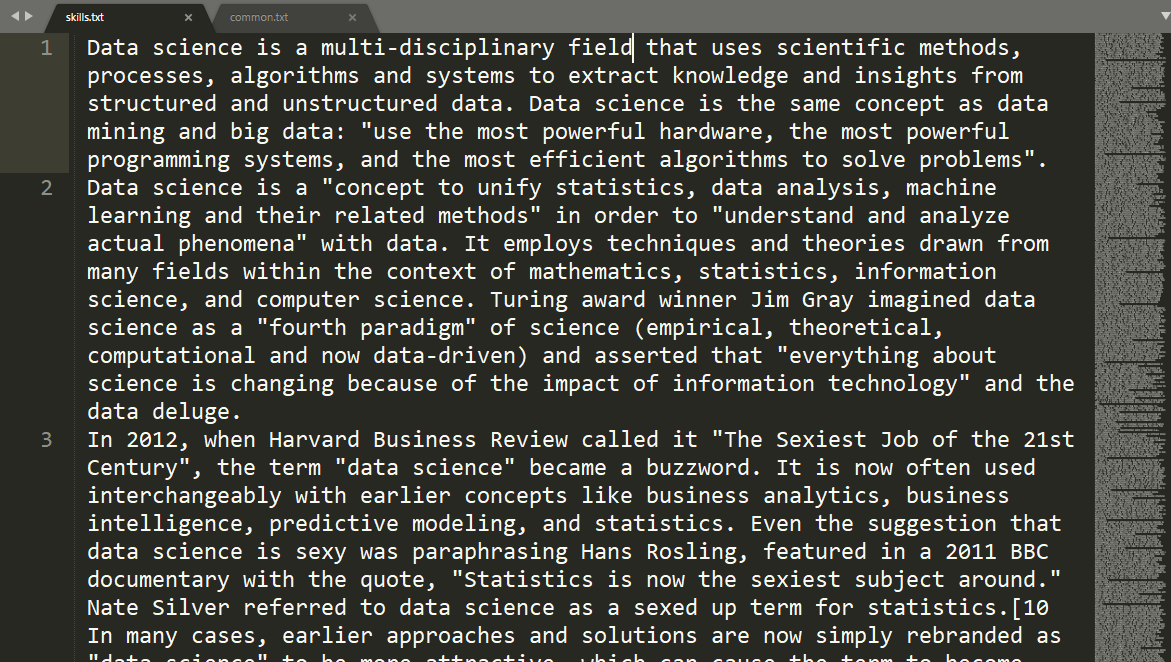
**Workflow**

1. Creating corpus using Sketch Engine
2. Preprocessing
3. Removing common words that are not required
4. Creating phrases of words that frequently occur togther (bigrams)
5. Creating word embedding using Genism , Word2Vec
6. Testing the model
7. Extracting resumes (using PyPDF) and converting to string
8. Building candidate profile using model.most\_similar(skills), where skills is an array of required skills and Creating a matcher using Spacy to match the wods in resume to most\_similar(skills)
9. Counting words under each category to build a candidate profile.
10. Printing and visualizing the candidate profile
11. **Creating corpus using Sketch Engine**

Sketch Engine is an online tool to analyse how language works. It algorithms can analyse authentic texts of billion words to identify what is typical in a language, or rare or usual or emerging. It is used to create the corpus for our application since the available corpus did not contain the required words. Wikipedia pages are given as input and a text corpus is created from the useful information available on the given pages.

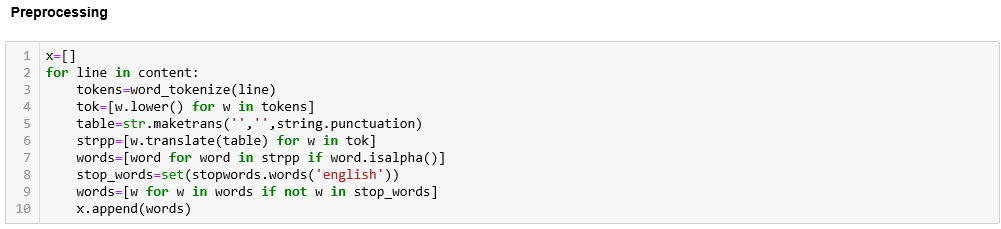


Corpus :



1. **Preprocessing**

Skills.txt is preprocessed before creating word emeddings, it involves removal of white space, stop words, digits etc and <p> tags.



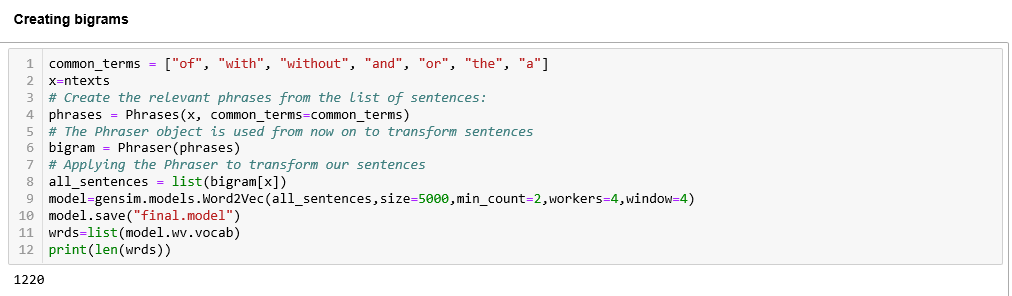
1. **Removing common words that are not required**

A text file is created named commons.txt which contains all the words that occur frequently and are not useful for technical skill matching (source : Wikipedia)



1. **Creating phrases of words that frequently occur togther (bigrams)**

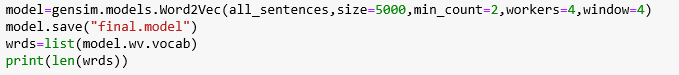
Bigrams are those words which occur together almost all the time. Example : New York, Machine Learning. Bigrams are created using phrases.



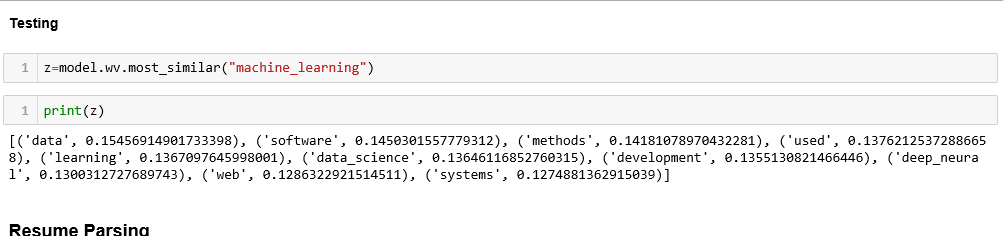
1. **Creating word embedding using Genism , Word2Vec and building a word2vec model.**

The Gensim library provides a [simple API](https://radimrehurek.com/gensim/models/word2vec.html) to the [Google word2ve](https://code.google.com/archive/p/word2vec/)c algorithm which is used to create word embeddings.

Word embedding is one of the most popular representation of document vocabulary. It is capable of capturing context of a word in a document, semantic and syntactic similarity, relation with other words, etc.

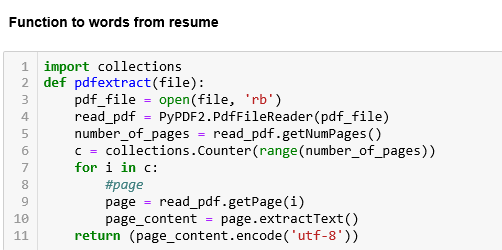


1. **Testing the model.**



1. **Extracting resumes (using PyPDF) and converting to string**

Resumes are stored in a folder and extracted one by one using PyPDF library and returned as a sequence of strings. The string is preprocessed and is further processed to create candidate profiles.

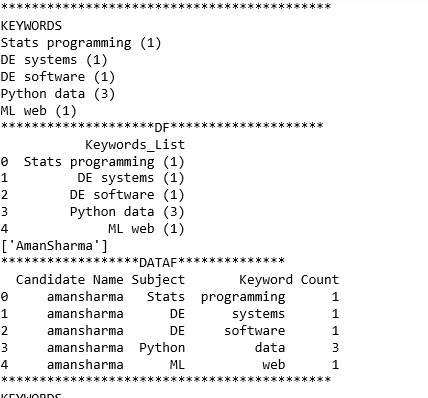


1. **Building candidate profile using model.most\_similar(skills), where skills is an array of required skills and creating a matcher using Spacy to match the wods in resume to most\_similar(skills)**

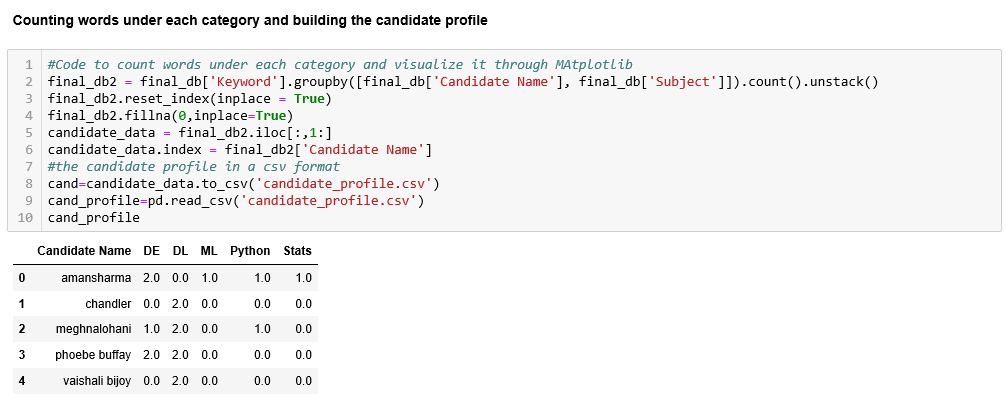
Spacy’s Phrase matcher is used to match the array(obtained from word2vec) with the extracted text. It assigns each skill to a particular keyword and scores accordingly.



**Output**

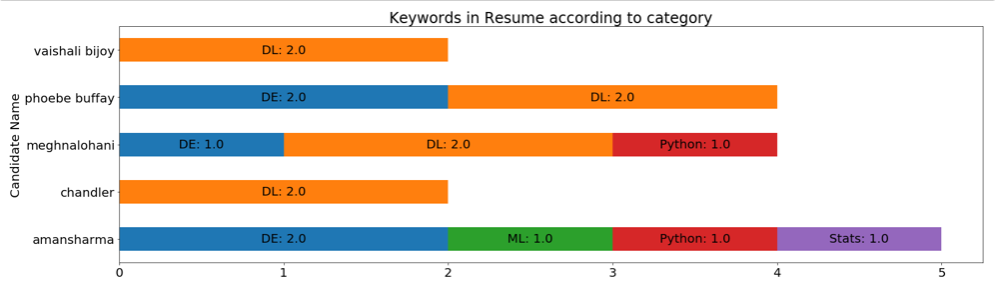


1. **Counting words under each category to build a candidate profile.**



1. **Printing and visualizing the candidate profile**

The candidate profile is visualized through a bar graph using scores of each skill. So, the recruiter can easily interpret the candidate profile.



**CONCLUSION**

The application is able to match each skill in the candidate resume to a particular keyword (Machine learning, Programming) based on semantic distance between the words, i.e. using Word Embeddings. Matching is done by Spacy’s matcher and a score is created for each keyword. The final scores are visualized.

**FUTURE SCOPE**

With a better refined corpus, the model can be made more accurate. Due to the unavailability of the required corpus, we created a small corpus using Sketch Engine, therefore the vocabulary is limited.

**REFERENCES**

1. https://machinelearningmastery.com/develop-word-embeddings-python-gensim/

2. https://www.shanelynn.ie/word-embeddings-in-python-with-spacy-and-gensim/

3. https://app.sketchengine.eu/