After Careful Consideration...

Project: #5 - Resume Classification using NLP

Project sponsors: *Sunny Rathee | Prasant Das*
Mentors: *Monica Chen | Emma Wang | Rachel Mackintosh*
Team members: *Sophy Peng | Michael Chan | Mike Arinarkin*

Data Exploration, Preparation, and Insights:

- We received an archive of 26 resumes in doc and docx formats.
- After analysing the set we discovered two key insights.
    - First, some resumes were identical but uploaded with a different name. To save time, we did not take any special actions to highlight identical resumes. They are relatively easy to spot when they get exactly the same score. Moreover, top resumes are still being screened by a recruiter. However, with more time, we would create a check system to check for plagiarism of resumes within the given data set.
    - Second, resumes are of different length ranging from 1 to 7 pages. Currently, we are not accounting for this. However, if we had more time, we would write a function weighting resumes based on their lengths so that longer resumes wouldn't be unfairly biased.
- In addition, we downloaded a set of around 20 resumes found on the Internet. Some of them were in pdf format. Our model can read both doc/docx and pdf files.
- We also received a set of 4 job descriptions. We did not have to take any special actions preparing this data.

Model:

- We have a 2-part model.
- Part I: NLP using a dataset of skills. In short, the model compares the skills found in the resume against the skills found in the job description.
    - First, we took the set of all skills that could be added to the "skills" section on LinkedIn. We store a list of 51514 skills in *skills.txt* file
        - We did not have time to parse LinkedIn ourselves, so we took an existing list. However, we have to delete certain skills such as "A" (a programming language) and "one" to save time.
        - Moreover, we could not find the categories of skills (e.g. "Interpersonal Skills", "Tools & Technologies"). If we had them, we would be able to provide rating in each of the categories. Or if we had more time, we could have crawled for the data ourselves.
    - We use the set of skills we took from LinkedIn to search for them in the job description. We parse through the documents to get skills that are present in the JD. The more frequently skills are mentioned, the higher the weight assigned for the skill during rating. However, we did not have enough time to come up with a better weighting system e.g. one that is categorical based (for example, past job experience is more important than extracurriculars)

    - With more time and data, we want to create a machine learning algorithm that detects the keywords in the JD for you and determines the weight based on context.
    - Then we parse the resumes to look for matches against JD skills using the same process. We count every appearance of the skill.
- Part II: NLP using similarities.
    - We use Spacy Similarities with word vectors to find similarities in between resumes and job descriptions.
    - We did not have time to build our own corpus and train it. Moreover, we did not have enough data to train the model. That's the biggest trade-off we have made in this part.
- Scoring
    - We score all resumes for each position separately.
    - We take the top score in each of two methods and count it as 100%. All the scores below the top one are displayed a proportions of the top one.
        - For example, if the two top scores for Method I for the resumes are 50 and 40, we display 100% for the first resume and 80% for the second resume for Method I.
    - We combine the scores given by two methods into the final score we display on the portal. For now, we used simple averages. However, if we had more time, we would weigh them according to their importance.
- Tools we used:
    - Textract - extracting text from pdfs
    - Flask - to develop a fast web application
- NLP tools we used:
    - Textblob - extracting certain word types (to focus on nouns and verbs)
    - Ntlk - tokenize sentences to split them up more accurately
    - Spacy - for semantic similarity

Report and portal:

- Portal gives a visual report of top 10 resumes for a given job description. It allows you to see the scores from Method I and Method II as well as the combined score. Moreover, it allows you to download top resumes.
- As of now, you need to upload all of the resumes and all of the JDs before launching the portal. We are expanding the capability to allow to upload them while the portal is running (stored on some server), but we built the frontend to show how this would look like.