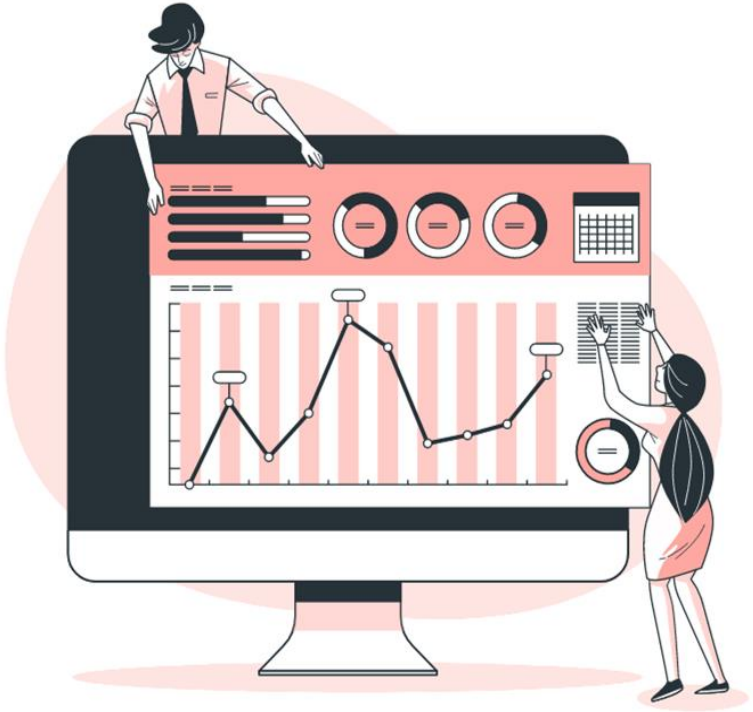
Applicant Shortlisting using Python and NLP

Solution document for the proposed problem in Round-2.

Jaynil Pandya.
jaynil@ieee.org



Problem Statement

For the role of a data scientist in a company, you have to hire two applicants. You have received an overwhelming number of responses on your website with over 200 applicants.
You now have a day to shortlist the candidate and onboard him as soon as possible. What will you do?

Table of contents

01

Assumptions and Dataset used to generate results.

Assumptions

- The resumes supplied by candidates are in simple format with popular fonts.
- They are in PDF format and not password protected.
- While extracting the text, we eliminate the numbers and only focus on the keywords.
- Spelling errors in resumes and / or alternate spellings are not accounted for.
- The program only scores the candidates on the basis of keywords and hence it doesn't include the personal details. The final results are given on the basis of the *filename* of the resume, hence the names should be uniform.

Dataset

- For the dataset, we have 3 simple resumes for an ideal data scientists' role.
- These include the keywords and the dataset is varied in terms of what kind of field a candidate is into.

02

Approach and technologies used

Approach to solve the problem

- To shortlist and select 1-2 candidates out of 200, we would need to accurately analyse their resumes and use Natural Language Processing to determine the final scores of candidates, and based upon that we could shortlist for Data Scientist Role.

- First, we create a csv file which has various top skills required by data scientists. These skills are clustered on the basis of seven fields.

statistics	python	machine learning	deep learning	r programming	nlp	data engineering
statistical modelling	numpy	linear regression	neural networks	shiny	sentiment analysis	aws
probability	pandas	logistic regression	keras	ggplot	chat bot	ec2
normal distribution	scikit learn	k means	cnn	cran	word cloud	instances
hypothesis testing	sklearn	random forest	convolutional neura	tidyr	word to vector	azure
bayesian inference	matplotlib	svm	object detection	knitr		sql
factor analysis		naïve bayes	yolo			nosql
monte carlo		decision trees	gans			kubernetes
			open cv			hadoop
						spark
						tableu
						power bi

- The resumes of all the candidates are stored in a particular folder in the database. To read the resumes one by one, we have used an NLP Algorithm which coverts the PDF based resume into a simple text form and performs matching function in accordance to the keywords mentioned in the dictionary.

```
# Function to read resumes from the folder in a sequence
path = '/content/Data/Resumes'
can_files = [os.path.join(path, f) for f in os.listdir(path) if os.path.isfile(os.path.join(path, f))]
```

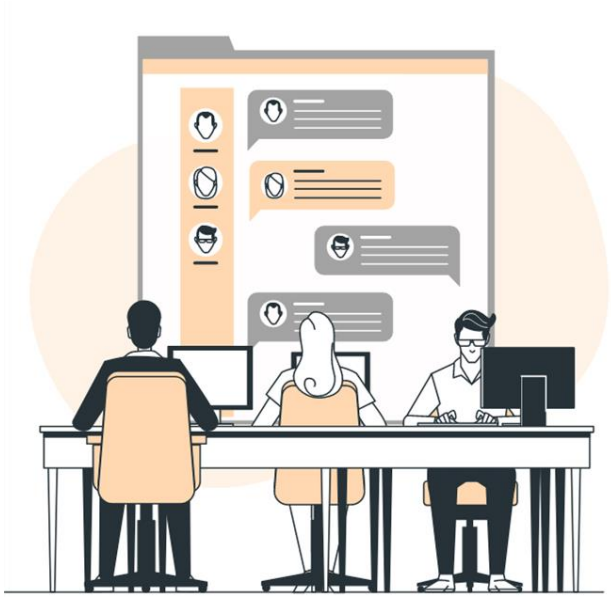
- Then, we have declared a function create_profile( ) which performs text cleaning, i.e. removes spaces, converts the text into lower case. This ensures that the text remains uniform.
- Matcher function from the spacy library is then used to match the occurrences of each word in a category.
- A scoring algorithm then counts these occurrences and computes the scores of candidates.
- We finally create a candidate database which orders the candidates based on their scores.
- Finally, we plot a graph which represents the candidates on y-axis and skill-based scores on x-axis. Such a plot would be useful in determining whether a candidate has core focus on a specialization or has a mix-bag of skills.

Technologies used

Language Python

Visualization Matplotlib

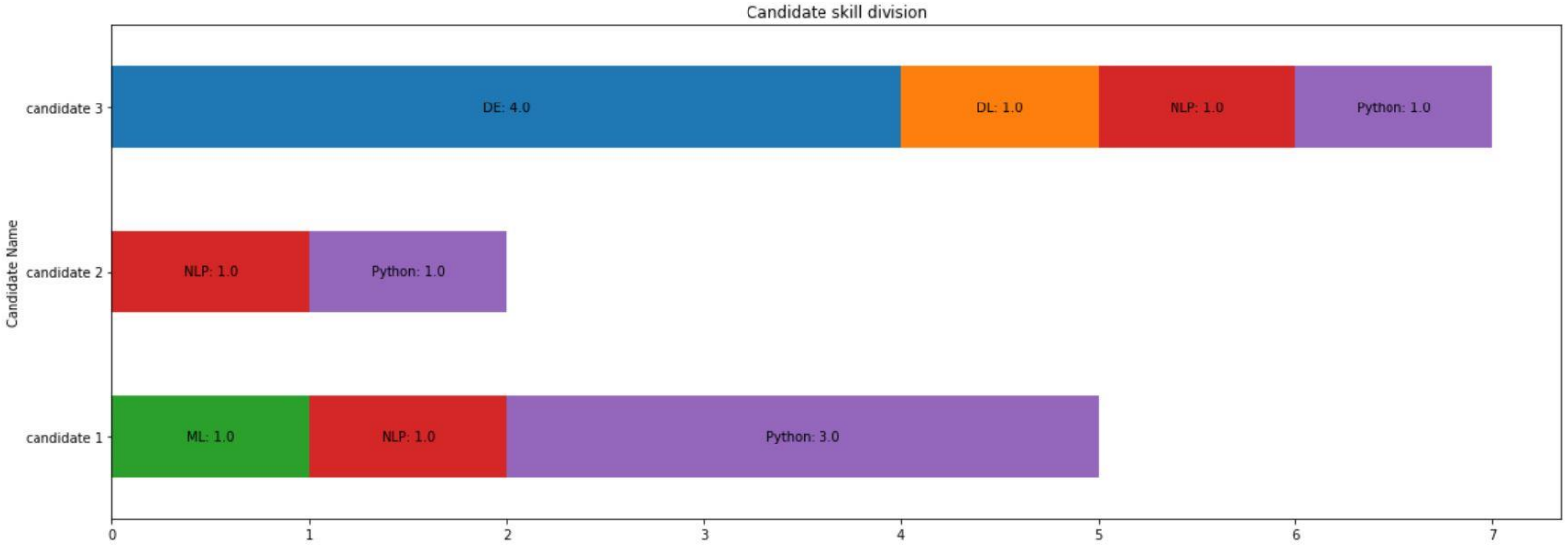
Dependencies Spacy, Pandas, PyPDF2



03

Results, advantages, and further possibilities

Results



- The above plot shows scores of different candidates divided on the basis of their skillsets. As we can see, candidate 3 has overall highest score and has greatest competency in DE (which is Data Engineering), in which the score is 4.

- Made using Prettytable library, the following tables show the computed scores of the candidates.
- These scores are weighted scores that are more useful considering the overall selection.
- Here, we can see the top two shortlisted candidates and the selected candidate simply represents the candidate with the highest score.

```
Top two shortlisted candidates:
+-------------+-------+
|     Name    | Score |
+-------------+-------+
| candidate 3 |   41  |
| candidate 1 |   34  |
+-------------+-------+

Selected Candidate:
+-------------+-------+
|     Name    | Score |
+-------------+-------+
| candidate 3 |   41  |
+-------------+-------+
```

Advantages and limitations

- The model presented to solve the problem statement can be very well curated into a full-fledged app that can solve a plethora of issues faced by hiring managers and recruiters.
- Automatic reading of the resumes saves time as the recruiter doesn't need to manually open each and every resume.
- The keyword file can be customized to fit the skillset needed by any given job. The model would remain pretty much the same except for a few variable changes.
- Use of Natural Language Processing helps to a great extent in filtering out the candidates who have the desired skills for the job.

Limitations:

- Since the project is keyword based, certain numerical data from the resume is not accounted for. This could very well hamper the chances of a candidate who has stated his/her achievement in numbers.
- Details like college CGPA, extracurricular activities, data on internships and projects is also missed out and only limited to consideration of keywords.
- Thus, this model is useful for a first stage filtering of candidates, with the selection being a subject to personal interviews.