Resume Scoring and Filtering Web Application

Khushi Shivde, Mehul Agrawal, Monish Ostwal

Department of Information Technology
SGSITS, Indore

Abstract- Companies often receive thousands of resumes for each job posting and employ dedicated screeners to shortlist qualified applicants. In this paper, we present a decision support tool to help these screeners shortlist resumes efficiently. HR recruiter mines resumes to extract salient aspects of candidate profiles like skills, experience in each skill, education details and past experience. Extracted information is presented in the form of facets to aid recruiters in the task of screening. We also employ Information Retrieval techniques to rank all applicants for a given job opening.

I. Introduction

In the present system the candidate has to fill each and every information regarding their resume in a manual form which takes a large amount of time and then also the candidates are not satisfied by the job which the present system prefers according to their skills. Let me tell you a ratio of 5:1 means, If 5 people are getting a job out of that 5, only a single guy will be satisfied by his/her job. Let me tell you an example : If I am a good python developer and a particular company hired me and they are making me work on Java, my python skills are pretty useless. And on the other hand if there is a vacant place in a company so according to the owner of the company he/she will prefer a best possible candidate for that vacancy. So our system will act as a handshake between these two entities. The company who prefers the best possible candidate and the candidate who prefers the best possible job according to his or her skills and ability.

To give the context for our work, we now give an overview of a typical recruitment process. This is illustrated in Figure 1. The process starts when a business unit decides to hire employees to meet its business objectives. The business unit creates a job role that specifies the role, job category, essential skills, location of the opening and a brief job description detailing the nature of work. It might also specify the total work experience that the prospective employee should possess, along with the desired experience level for each skill. The job openings are advertised through multiple channels like online job portals, newspaper advertisements, etc. Candidates who are interested to apply for the job opening upload their profile through a designated website. The website typically provides an on{line form where the candidate enters details about her application like personal information, education and experience details, skills, etc. We call this Candidate Metadata. The candidates can also upload their resumes through the website. The objective of allowing the candidate to enter meta-data in an online form is to capture the information in a more structured format to facilitate automated analysis. However, real life experience suggests that most candidates do not specify a lot of information in the on{line forms and hence Candidate
Meta-data is often incomplete.

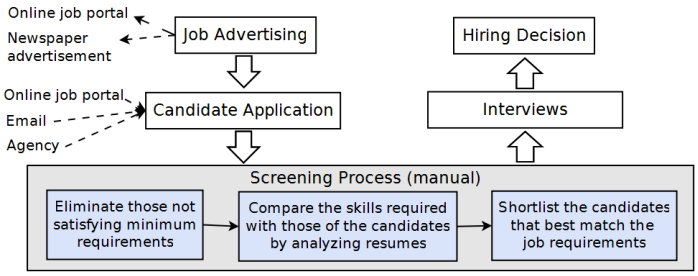

Figure 1: Recruitment process with manual screening

II. Description of System

The aim of this work is to find the right candidates resume from the pool of resumes. To achieve this objective,we have developed a machine learning based solution, The complete framework for the proposed model is shown in Figure 2. The proposed model worked mainly in two steps: i) Prepare and ii) Deploy and Inference.

2.1 Preprocessing

In this process, the CVs being provided as input would be cleansed to remove special or any junk characters that are there in the CVs. In cleaning, all special characters, the numbers, and the single letter words are removed. We got the clean dataset after these steps having no special characters, numbers or single letter word. The dataset is split into the tokens using the NLTK tokenizers [12]. Further, the preprocessing steps are applied on tokenized dataset such as stop word removal, stemming, and lemmatization. The raw CV file was imported and the data in the resume field was cleansed to remove the numbers and the extra spaces in the date. Data Masking was done as:

- Mask string fragments like \x
- Mask fragments for escape sequences like \a, \b, \t, \n
- Mask all numbers
- Replace all the single letter words with an empty string
- Mask email addresses.
- Stop words were masked from the dataset
- Lemmatization

Stop words removal: The stop words such as and, the, was, etc. are frequently appeared in the text and not helpful for the prediction process, hence it is removed. Steps to filter the Stop Words:

1. We have tokenize the input words into individual tokens and stored it in an array

2. Now, each word matches with the list of Stop Words present in NLTK library.

3. If the words present in the list of StopWords[], filtered from the main sentence array.

4. The same process repeated until the last element of the tokenized array is not matched.

5. Resultant array does not have any stop words.

Stemming: Stemming is the method of decreasing word inflection to its root forms such as mapping a group of words to the same *stem* even though the stem itself is not a valid term in the language. Stem (root) is the part of the word to which you add inflectional (changing/deriving) affixes such as (-ed, -ize, -s, -de, -ing, mis). For example the words like: Playing, Plays, Played are mapped to their root word Play, the words like: python, pythoner, pythoning, pythoned mapped to their root word python.

Lemmatization: Unlike Stemming, lemmatization decreases the inflected phrases to ensure that the root word belongs to the language correctly. Lemmatization comprises the following routine steps1:

•Transform the corpus of text into a list of words.

•Create a concordance of the corpus, i.e., of all the items of the word list as they occur in the corpus.

•Assign the word-forms to their lemmas based on the concordance.

2.2 Feature Extraction

The next step is feature extraction. On a preprocessed dataset, we have extracted the features using the Tf-Idf. The cleansed data was imported and feature extraction was carried out using Tf-Idf. The machine learning based classification model or learning algorithms need a fixed size numerical vector as input to process it. ML based classifiers did not process the raw text having variable size in length. Therefore, the texts are converted to a required equal length of vector form during the preprocessing steps. There are many approaches used to extract the features such as BoW(Bag of Words), tf-idf (Term Frequency, Inverse Document Frequency) etc. In the BoW model, for each document, a complaint, the narrative in our case, the presence (and often the frequency) of words is taken into consideration, but the order in which they occur is ignored. Specifically, we have calculated tf-idf (term fre- quency, and inverse document frequency) for each term present in our dataset using the *scikit learn* library function: sklearn.feature extraction.text.TfidfVectorizer to calculate a tf-idf vector.

2.3 Model Selection

In this process the tokenized CV data and the job descriptions (JD) would be compared and the model would provide CVs relevant to the job description as an output.

Model 1: Supervised Learning Model

Model 2: Model based on Cosine Similarity

Model 3: Unsupervised Learning Model

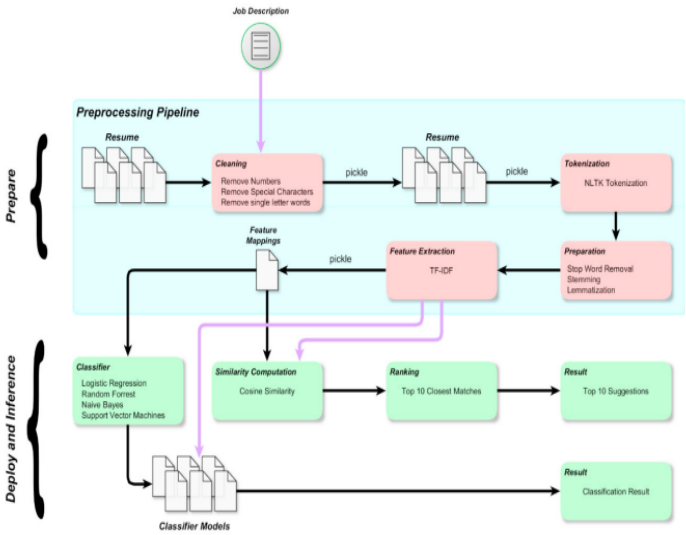

Figure 2.3: Architecture of the application

2.4 Results

Three models have been built on the cleansed data: i) Classification - Based on the resume and category the model has been designed to categories the resume in the right category, ii) Resume filtering based on job description by cosine similarity and iii) Unsupervised Model-In which the training data consists of a set of input vectors x without any corresponding target values. The goal in such problems may be to discover groups of similar examples within the data, where it is called clustering, or to determine the distribution of data within the input space, known as density estimation, or to project the data from a high-dimensional space down to two or three dimensions for the purpose of visualization.

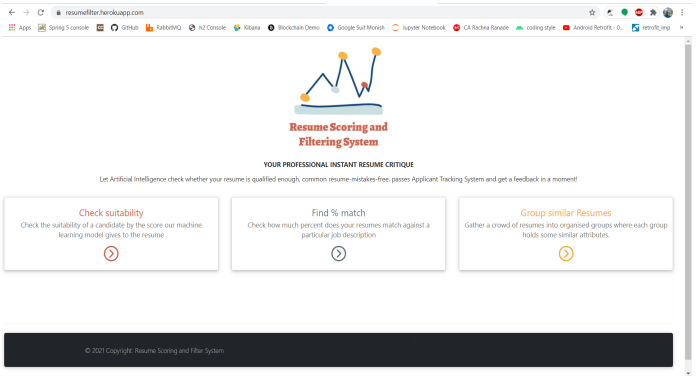

Figure 2.4.1: Homepage of the application

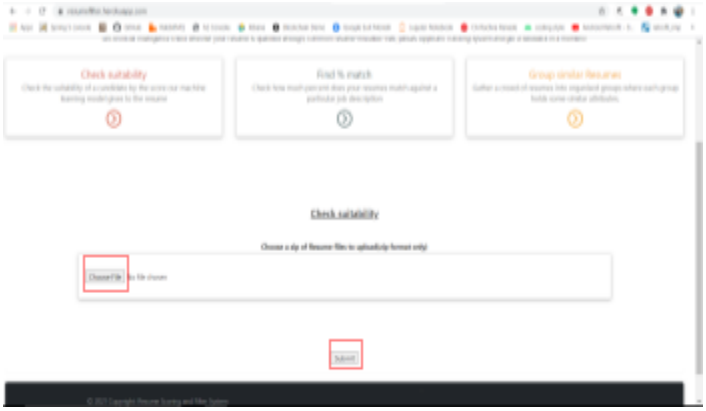

Figure 2.4.2: Selection of model and submitting of pdf/zip file

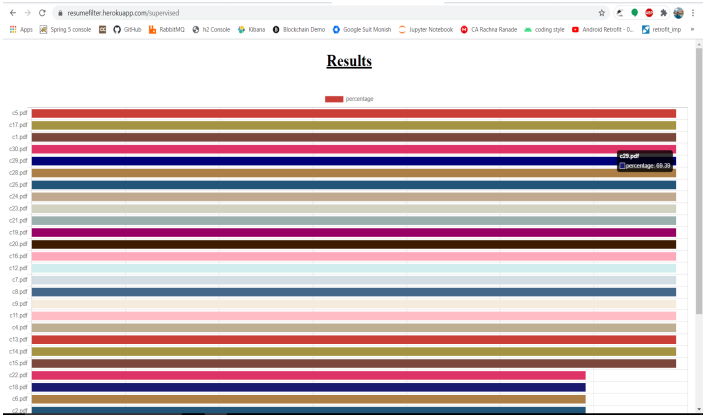

Figure 2.4.3: Results shown using bar chart

Figure 2.4.4: Results shown using pie chart

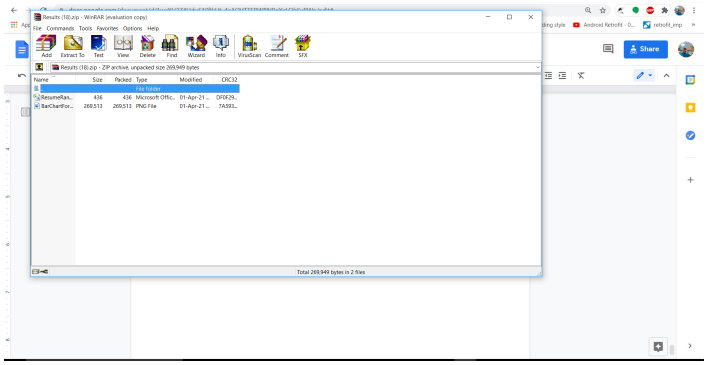

Figure 2.4.5:Extraction of downloaded zip file

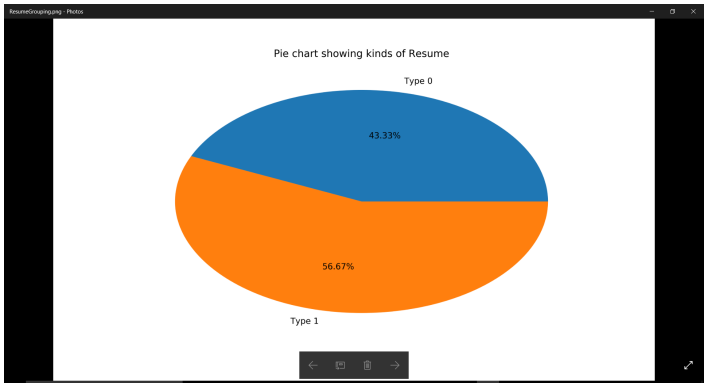

Figure 2.4.6: Downloaded graph of clustering results

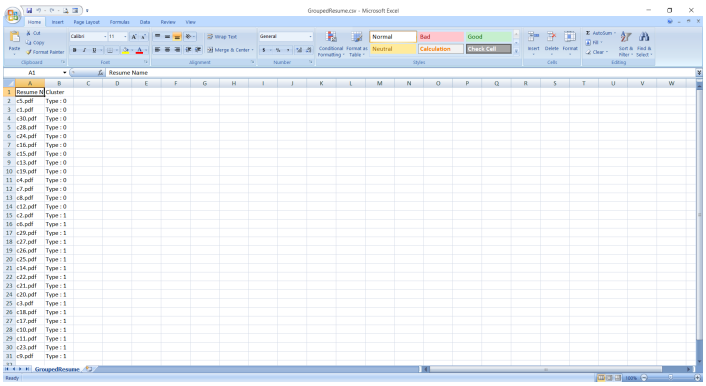

Figure 2.4.7: Result shown in excel format

III. Conclusion

Huge number of applications received by the organization for every job post. Finding the relevant candidate's application from the pool of resumes is a tedious task for any organization nowadays. The process of classifying the candidate's resume is manual, time consuming, and waste of resources. To overcome this issue, we have proposed an automated machine learning based model which recommends suitable candidate's resumes to the HR based on given job description. The proposed model worked in two phases: first, classify the resume into different categories. Second,recommends a resume based on the similarity index with the given job description..If an Industry provides a large number of resumes,then an Industry specific model can be developed by utilizing the proposed approach. By involving the domain experts like HR professionals would help to build a more accurate model, feedback of the HR professional helps to improve the model iteratively.

IV. References

[1]https://reader.elsevier.com/reader/sd/pii/S187705092030750X?token=C10A721DC71F5B4AE0B05D21CF4D3920CDC599267F5030272C4C773EE0ED290BB40DE236E8A34C6B831B34CE6795CD8B

[2] https://www.researchgate.net/publication/221614548

[3]https://vinayakjoglekar.wordpress.com/2014/10/03/resume-ranking-using-machine-learning-implementation/