Almas Jaipuri  | June 7, 2019

PROFILE SCANNER

An automated recruitment tool – pROCESS DOCUMENT



# BACKGROUND

This tool has been designed for scanning the candidate profiles received by recruitment agencies or internal recruiters giving the following benefits –

1. It will improve the efficiency of identifying the optimum candidates without any human bias.
2. One recent study suggested that human recruiters spend only 6 seconds reviewing an individual resume and 80% of this time is spent on only six data points: name, current title/company, current position start and end date, previous title/company, previous position start and end date, and education, these details will be identified by the tool and presented at one place for the evaluation.
3. The first round of elimination can very well be completed by this tool rendering more time and better info points for the further evaluation.

# ASSUMPTIONS

The below assumptions are to be taken into consideration while designing the solution and should be heeded for further use of the tool-

1. The services of Google Natural Language API has been rendered for the text mining and identification of NERs and tokens, as the APIs cannot call their services from behind a corporate proxy till it is bypassed hence this tool cannot be used over Tech Mahindra network without bypassing the proxy.
2. Numerous packages have been used in the integrated R environment for the tool development hence it should be ensured that the mentioned packages have been installed by the user to avoid dealing with undesired errors.
3. The ranking for the candidates has been established on the basis of mathematical model of probability which uses Jaccard similarity scores to determine the profile suitability to the JD. In cases, it might produce results which would be different from a manual identification which can be perfectly explained by the differences in human emotions and the mechanisms of the algorithm.

# TECHNICAL SPECIFICATIONS

## software used –

R Studio

## packages to be installed –

|  |  |  |
| --- | --- | --- |
| dplyr | textreadr | tm |
| googleLanguageR | knitr | ggplot2 |
| quanteda | stringr | udpipe |
| lattice | scales | readtext |
| qdap | tm | wordcloud |
| plotrix | dendextend | ggplot2 |
| ggthemes | RWeka | quanteda |
| tidytext | tidyr | igraph |
| ggraph | widyr | sqldf |
| stringr | text2vec | lsa |
| tcR | vegan | proxy |
| Rlist | shinydashboard | shiny |
| dplyr | ggplot2 | xlsx |
| DT | readxl | fuzzyjoin |

## data containers –

A folder is to be created with all the profiles of the candidates to be screened along with a file named “Skills” with the job description in it.

The files kept can be either in “doc”, ”docx”, ”txt” or “pdf” format.

## function descriptions –

### Model\_code.R

### Functions.R

### App.R

It contains following functions for creating Rshiny dashboard

Ui function

Server function

### Profile\_Scanner.R

File to be executed mainly for the dashboard. It calls the other files and functions.

Following amendments need to be done before each execution –

1. Location for code files to be sourced.
2. Json file for API authentication to be kept at a location and the location address should be updated in gl\_auth() function.
3. Location for data i.e. candidate profiles should be updated.
4. Location for app file to be updated for creating the dashboard.

Almas Jaipuri

June 7, 2019