“RESUME SCREENING USING ML”

1. Anand Kumar (University Roll No. CSE/124/17)

2. Pooja Kumari (University Roll No. CSE/092/17)

3. Shreyash Pandey (University Roll No. CSE/116/17)

4. Nirmal Kumar Dubey (University Roll No. CSE/111/17)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

R.V.S. COLLEGE OF ENGINEERING AND TECHNOLOGY

KOLHAN UNIVERSITY

2017-2021

A
Project Report
On

(RESUME SCREENING USING ML)

Submitted in partial fulfillment of the requirements
for

Bachelor of Technology in
Computer Science and Engineering

by
1. Anand Kumar University Roll No- CSE/124/17
2. Pooja Kumari University Roll No- CSE/092/17
3. Shreyash Pandey University Roll no- CSE/116/17
4. Nirmal Kumar Pandey University Roll no- CSE/111/17

Under the guidance of
(Prof. Mukesh Raj)
Asst. Professor (Dept. of Computer Science and Engineering)

Department Of Computer Science and Engineering
R.V.S. College of Engineering and Technology
Jamshedpur – 831012

2017 - 2021

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project work entitled "Resume Screening using ML" is done by Anand Kumar, Pooja Kumari, Shreyash Pandey and Nirmal Kumar Pandey, ***Regd. No: - KU1723486, KU1723531, KU1723554 and KU1723525*** *in partial fulfilment of the requirements for the 8th Semester Sessional Examination of Bachelor of Technology in* ***Computer Science & Engineering*** *during the academic year 2017-21. This work is submitted to the department as a part of evaluation of 8th Semester Project.*

Asst. Professor Mukesh Raj — **Project Guide**

(Prof. Jeevan Kumar) — **H.O.D, CSE**

Signature of External

Date:
Place: RVSCET, Jamshedpur

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1. Anand Kumar

2. Pooja Kumari

3. Shreyash Pandey

4. Nirmal Kumar Dubey

ACKNOWLEDGEMENT

We avail this opportunity to express our profound sense of gratitude to Asst. Prof. Mukesh Raj for her constant supervision, guidance and encouragement right from the beginning till the completion of the project.

We wish to thank all the faculty members and supporting staffs of Dept. of Computer Science and Engineering for their valuable suggestions, timely advice and encouragement.

1. Anand Kumar

2. Pooja Kumari

3. Shreyash Pandey

4. Nirmal Kumar Dubey

Abstract

In the current smart world, everything should be done faster, smarter, and accurate way. The various organization's recruitment processes will be done face to face in an arranged venue. But, during some pandemics like Covid-19 face to face recruitment process will be very difficult. In the proposed system, a smarter way of performing the recruitment processes anywhere around the world based on the company requirements is performed. The aim of this project deals with making the process of candidate recruitment easier for companies. The amount of manual work that goes into recruiting processes is reduced and the initial scanning process of candidates was performed. By eliminating the redundant candidates helps in retaining only the applicable ones. Achieve this through the help of resume scanning, initial aptitude testing of candidates, and an interview session where the candidate answers questions asked by the interviewer. With this model, all the time and manual labour that is wasted in eliminating the redundant candidates is accomplished. It chooses the one who is best applicable to a job by comparing it with the job description based on the resumes received. Our model is working accurately for some of the predefined parameters of the company in a recruitment process by providing more security and reliability.

Contents

List of Figures

Chapter 1

Introduction

1.1 Purpose

A typical job posting on the Internet receives a massive number of applications within a short window of time. Manually filtering out the resumes is not practically possible as it takes a lot of time and incurs huge costs that the hiring companies cannot afford to bear. In addition, this process of screening resumes is not fair as many suitable profiles don't get enough consideration which they deserve. This may result in missing out on the right candidates or selection of unsuitable applicants for the job. In this paper, we describe a solution that aims to solve these issues by automatically suggesting the most appropriate candidates according to the given job description. Our system uses Natural Language Processing to extract relevant information like skills, education, experience, etc. from the unstructured resumes and hence creates a summarised form of each application. With all the irrelevant information removed, the task of screening is simplified and recruiters are able to better analyse each resume in less time. After this text mining process is completed, the proposed solution employs a vectorisation model and uses cosine similarity to match each resume with the job description. The calculated ranking scores can then be utilized to determine best-fitting candidates for that particular job opening.

1.2 Project Scope

- Separating the right candidates from the pack
- Making sense of candidate CVs
- Filtering the candidates based on their performance.

1.3 Product Features

- You do not need copy PDF text infor
- mation from hundreds PDF files again. Using it, you can batch process PDF Data one time.
- Create one single Excel, CSV or XML file from all PDF files.
- It is a single standalone program which is free.
- Make any change to text or images in a PDF without losing formatting.
- User-friendly and simple steps which is easily operatable.

Chapter 2

System Analysis

After the extensive analysis of the problems in the system, we are familiarized with the requirement that the current system needs. The requirement that the system needs is categorized into the Software requirements and Hardware requirements. These requirements are listed below:

2.1 Software Requirements

Operating System: Windows 8/8.1/10

IDE: The Jupyter Notebook

- The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results.
- Notebook documents contains the inputs and outputs of a interactive session as well as additional text that accompanies the code but is not meant for execution. In this way, notebook files can serve as a complete computational record of a session, interleaving executable code with explanatory text, mathematics, and rich representations of resulting objects.
- JupyterLab also offers a unified model for viewing and handling data formats. JupyterLab understands many file formats (images, CSV, JSON, Markdown, PDF, Vega, Vega-Lite, etc.) and can also display rich kernel output in these formats. See File and Output Formats for more information.

- JupyterLab extensions can customize or enhance any part of JupyterLab, including new themes, file editors, and custom components.
- JupyterLab is served from the same server and uses the same notebook document format as the classic Jupyter Notebook.

Language Used: Python 3.6

- Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.
- Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Libraries Used: Pandas, Numpy, Matplotlib, Sklearn, Seaborn, RE, Nltk, Wordcloud

2.2 Hardware Requirements

Processor: Intel i3 or Above
Clock Speed: 2GHZ
System Bus: 64bit
Ram: 4GB or more
Monitor: LCD
Keyboard: QWERTY (101keys)

Chapter 3

Feasibility Analysis

A feasibility study is a preliminary study which investigates the information of prospective users and determines the resources requirements, costs, benefits and feasibility of proposed system. A feasibility study takes into account various constraints within which the system should be implemented and operated. In this stage, the resource needed for the implementation such as computing equipment, manpower and costs are estimated. The estimated are compared with available resources and a cost benefit analysis of the system is made. The feasibility analysis activity involves the analysis of the problem and collection of all relevant information relating to the project. The main objectives of the feasibility study are to determine whether the project would be feasible in terms of economic feasibility, technical feasibility and operational feasibility and schedule feasibility or not. It is to make sure that the input data which are required for the project are available. Thus, we evaluated the feasibility of the system in terms of the following categories:

- Technical feasibility
- Operational feasibility
- Economic feasibility
- Schedule feasibility

3.1 Technical feasibility

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at the point in time there is no any detailed designed of the system, making it difficult to access issues like performance, costs (on account of the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis; understand the different technologies involved in the proposed system. Before commencing the project, we have to be very clear about what are the technologies that are to be required for the development of the new system. Is the required technology available? Our system "Resume Screening using ML" is technically feasible

since all the required tools are easily available. Python based Web App with HTML/CSS can be easily handled. Although all tools seem to be easily available there are challenges too.

3.2 Operational feasibility

Proposed project is beneficial only if it can be turned into information systems that will meet the operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to Implementation? The proposed was to make a simplified web application. It is simpler to operate and can be used in any webpages. It is free and not costly to operate.

3.3 Economic feasibility

Economic feasibility attempts to weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system. A simple economic analysis which gives the actual comparison of costs and benefits are much more meaningful in this case. In addition, this proves
to be useful point of reference to compare actual costs as the project progresses. There could be various types of intangible benefits on account of automation. These could increase improvement in product quality, better decision making, and timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information. This is a web-based application. Creation of application is not costly.

3.4 Schedule feasibility

A project will fail if it takes too long to be completed before it is useful. Typically, this means estimating how long the system will take to develop, and if it can be completed in a given period of time using some methods like payback period. Schedule feasibility is a measure how reasonable the project timetable is. Given our technical expertise, are the project deadlines reasonable? Some

project is initiated with specific deadlines. It is necessary to determine whether the deadlines are mandatory or desirable.

A minor deviation can be encountered in the original schedule decided at the beginning of the project. The application development is feasible in terms of schedule.

Chapter 4

Language Used - Python

What is Python?

- Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming.
- Python’s elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.
- The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.
- Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development.
- Python supports multiple programming pattern, including object-oriented, imperative, and functional or procedural programming styles.
- Python is not intended to work in a particular area, such as web programming. That is why it is known as multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc.

History of Python:

- Python was invented by Guido van Rossum in 1991 at CWI in Netherland. The idea of Python programming language has taken from the ABC programming language or we can say that ABC is a predecessor of Python language.

- There is also a fact behind the choosing name Python. Guido van Rossum was a fan of the popular BBC comedy show of that time, "Monty Python's Flying Circus". So, he decided to pick the name Python for his newly created programming language.

- Python 2.0 was released on 16 October 2000, with many major new features, including a cycle-detecting garbage collector and support for Unicode.

- Python 3.0 was released on 3 December 2008. It was a major revision of the language that is not completely backward-compatible. Many of its major features were backported to Python 2.6.x and 2.7.x version series. Releases of Python 3 include the 2 to 3 utilities, which automates (at least partially) the translation of Python 2 code to Python 3.

The Python Programming Language:

- Python is an example of a high-level language, other high-level languages you might have heard of are C, C++, Perl and Java.

- There are also low-level languages which are sometimes referred as "machine languages" or "assembly languages".

- Computers can only understand low level languages, so programs written in high level language have to be processed before they can run and it take some time for processing which is a small disadvantage of high-level language.

- The advantages are enormous. First, it is much easier to program in a high-level language. Programs written in a high-level language take less time to write, they are shorter and easier to read, and they are more likely to be correct.

- Second, high-level languages are portable, meaning that they can run on different kinds of computers with few or no modifications. Low-level programs can run on only one kind of computer and have to be rewritten to run on another.

- Due to these advantages, almost all programs are written in high-level languages. Low-level languages are used only for a few specialized applications.

- Two kinds of programs process high-level languages into low-level languages: Interpreters and Compilers.

- An interpreter reads a high-level program and executes it, meaning that it does what the program says. It processes the program a little at a time, alternately reading lines and performing computations.

- A compiler reads the program and translates it completely before the program starts running. In this context, the high-level program is called the source code, and the translated program is called the object code or the executable. Once a program is compiled, you can execute it repeatedly without further translation.

- Python is considered an interpreted language because Python programs are executed by an interpreter.

- Working in interactive mode is convenient for testing small pieces of code because you can type and execute them immediately. But for anything more than a few lines, you should save your code as a script so you can modify and execute it in the future.

Chapter 5

Libraries used

1. NumPy:

- NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

- NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.

- NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.

2. Pandas:

- Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

- Pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive.

- It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way toward this goal.

3. Matplotlib:

- Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

- It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

- Pyplot is a Matplotlib module which provides a MATLAB-like interface.[10] Matplotlib is designed to be as usable as MATLAB, with the ability to use Python, and the advantage of being free and open-source.

4. Sklearn:

- Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.

- Scikit-learn is an open-source machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data pre-processing, model selection and evaluation, and many other utilities.

- This library is built upon the SciPy (Scientific Python) library that you need to install before you can use scikit-learn. It is licensed under a permissive simplified

BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

5. Seaborn:

- Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures.

- Seaborn helps you explore and understand your data. Its plotting functions operate on data frames and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

- A dataset-oriented API for examining relationships between multiple variables. Specialized support for using categorical variables to show observations or aggregate statistics. Options for visualizing univariate or bivariate distributions and for comparing them between subsets of data.

6: RE:

- Regular expressions (called REs, or regexes, or regex patterns) are essentially a tiny, highly specialized programming language embedded inside Python and made available through the re module.

- The regular expression language is relatively small and restricted, so not all possible string processing tasks can be done using regular expressions.

- A regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. Regular expressions are widely used in UNIX world.

7: NLTK:

- Natural Language Processing with Python provides a practical introduction to programming for language processing.
- The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language.
- NLTK is intended to support research and teaching in NLP or closely related areas, including empirical linguistics, cognitive science, artificial intelligence, information retrieval, and machine learning

8: Word Cloud:

- Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance.
- Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analysing data from social network websites.

Chapter 6

Literature Review

6.1 Case Study on talent acquisition:

6.1.1 First Generation Hiring System

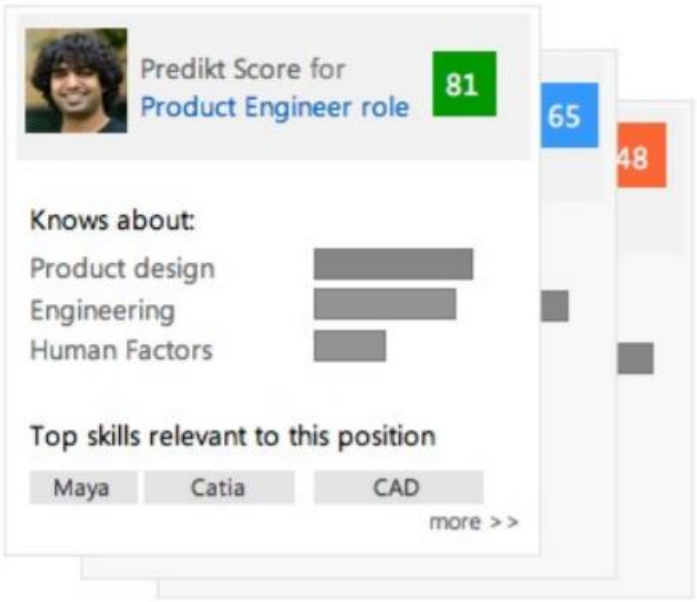
In this System the Hiring team would publish their vacancies and invite applicants. Methods of publishing were newspaper, television and mouth. The interested candidates would then apply by sending their resumes. These resumes were then received and sorted by the hiring team and shortlisted candidates were called for further rounds of interviews. The whole process would take lot of time and human efforts to find right candidate suitable for their job roles.

6.1.2 Second Generation Hiring System

As the industries have grown, there hiring needs has rapidly grown. To serve this hiring needs certain consultancy units have come into existence. They offered a solution in which the candidate has to upload their information in a particular format and submit it to the agency. Then these agencies would search the candidates based on certain keywords. These agencies were middle level organizations between the candidate and company. These systems were not flexible as the candidate has to upload there resume in a particular format, and these formats changed from system to system.

6.1.3 Third Generation Hiring System

This is our proposed system, which allow the candidates to upload their resumes in flexible format. These resumes are then analysed by our system, indexed and stored in a specific format. This makes our search process easy. The analysing system works on the algorithm that uses Natural Language Processing, sub domain of Artificial Intelligence. It reads the resumes and understands the natural language/format created by the candidate and transforms it into a specific format. This acquired knowledge is stored in the knowledge base. The system acquires more information about candidate from his social profiles like LinkedIn and GitHub and updates the knowledge base.



An example of Resume

6.2 Intelligent searching

Put simply, Artificial Intelligence or "AI" is an add-on to system, complementing to provide the online recruitment solution. As the name suggests, AI enables a combination of an applicant-tracking system and an artificial intelligence resume parsing, searching and matching engine. The result is a supercharged tool giving incredibly accurate candidate matching to jobs, and 'talent pool' searching that makes other systems look like they're from the stone-age.

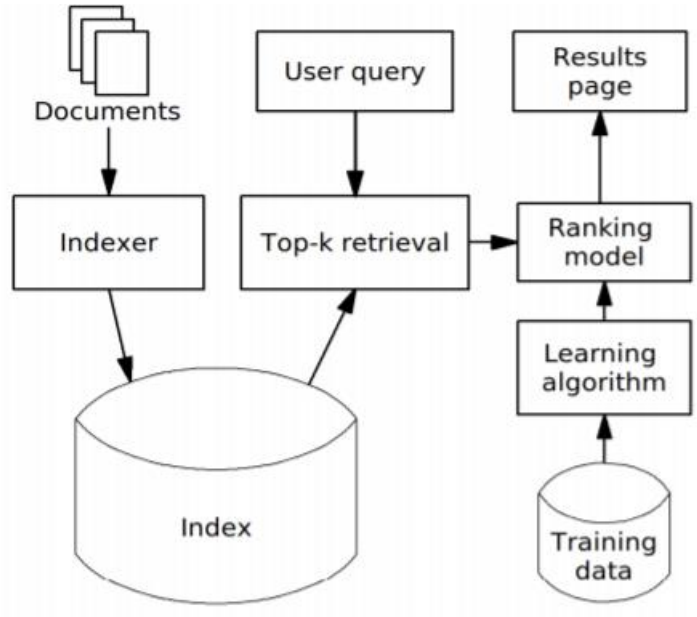
6.2.1 Identifying "best" applicants in recruiting using data envelopment analysis

Selecting the most promising candidates to fill an open position can be a difficult task when there are many applicants. Each applicant achieves certain performance levels in various categories and the resulting information can be overwhelming. We demonstrate how data envelopment analysis (DEA) can be used as a fair screening and sorting tool to support the candidate selection and decision-making process. Each applicant is viewed as an entity with multiple achievements. Without any a priori preference or information on the multiple achievements, DEA identifies the

non-dominated solutions, which, in our case, represent the ‘‘best’’ candidates. A DEA-aided recruiting process was developed that (1) determines the performance levels of the ‘‘best’’ candidates relative to other applicants; (2) evaluates the degree of excellence of ‘‘best’’ candidates’ performance; (3) forms consistent trade off information on multiple recruiting criteria among search committee members, and, then, (4) clusters the applicants.

6.3 A short introduction to learning to rank

Learning to rank refers to machine learning techniques for training the model in a ranking task. Learning to rank is useful for many applications in Information Retrieval, Natural Language Processing, and Data Mining. Intensive studies have been conducted on the problem and significant progress has been made. This short paper gives an introduction to learning to rank, and it specifically explains the fundamental problems, existing approaches, and future work of learning to rank.



A model for processing

6.4 Weaknesses

- Prior systems needed lot of human efforts and time.
- Cost of hiring is high.
- Potential candidate may lose the opportunity because of ambiguous keyword matching.
- Resumes needed to be in specific format.

6.5 How to overcome

- Use of NLP to read resumes allow candidates the freedom to choose any format that's available to them.
- Machine learning is used to rank candidates in accordance to requirements
- Which reduces the efforts of sorting thousands of resumes?
- Use of NLP can be used to get mean out of ambiguous data.
- Five benefits of A.I.
    - Goes Beyond Key Words
    - Fast and Accurate
    - Perfect For the New World of Social Recruiting
    - Customizes to your Needs
    - Gets Smarter

Chapter 7

Implementation Details

7.1 Assumptions and Dependencies

7.1.1 Assumptions

The following Assumption was taken into consideration:
• The Parser parses the resumes and convert them into the text file and then that text file is read to get the attributes of the candidate and store them in structured form in the json file.
• This json file contains the attributes of the student in ranked format and then it is read to show the output to the student/employer.

7.1.2 Dependencies

The dependencies are as follows:
• For User interface Django web framework is used.
• Python programming language, NLTK, Apache Tika is used to parse the document and store the information in structured form.
• To get user information form LinkedIn, GitHub there api is used which provides data in structured format.

7.2 Implementation Methodologies

7.2.1 Modular description of project

Different modules or components created are domain establishment, data collection, parsing, ranking and database component. Parsing and Ranking is the heart of our system which is created using python, nltk, tika libraries. This component does the morphological analysis, syntactic analysis, semantic analysis and generates the parsed and ranked data of the candidate according to his/her

skills. Then this information is stored in the database and retrieved and shown to the users whenever required.

7.3 Detailed Analysis and Description of Project

Domain Establishment: This module is responsible for creating user accounts and database creation as the proposed system is domain independent and would be used by multiple users.

Registration or Login Module: If the new user wants to interact with our system, he needs to simply register into our system by completely filling details i.e., validation. If the user is already existing, he needs to login.

Parsing & Ranking: Parsing module is responsible for parsing the document and storing it in json format which will later be used by the ranking module. Ranking module will then use the json file and rank the candidate's information according to his/her skills and the information will
be stored in the database.

Morphological Analysis: Morphology in linguistics is the study and description of how words are formed in natural language. In this phase the sentence is broken down into tokens- smallest unit of words, and determine the basic structure of the word.

Syntactic Analysis: The objective of the syntactic analysis is to find the syntactic structure of the sentence. It is also called Hierarchical analysis/Parsing, used to recognize a sentence, to allocate token groups into grammatical phrases and to assign a syntactic structure to it.

Semantic Analysis: Semantic Analysis is related to create the representations presentations for meaning of linguistics inputs. It deals with how to determine the meaning of the sentence from the meaning of its parts.

7.3.1 Use-Case Report

Title	**Resume Screening using ML**
Description	The current recruitment process is more tedious and time consuming which forces the candidates to fill all their skill and information manually. And HR team requires more man power to scrutinize the resumes of the candidates. So that motivated to build a solution that is more flexible and automated which will ease the burden on the employer for searching potential candidate and the burden of the candidate to find job suitable to his/her interests.
Primary actor	Candidate in search of good job and Employer in search of potential Candidate.
Pre-condition	There is no special requirement in submitting the resumes as our system is accepting different formats of resumes.
Post-condition	Candidate will see himself/herself ranked in his/her mentioned skills and the employer will get list of all potential candidate according to his/her need.
Main success scenario:	• Candidate submits resumes and social profile (LinkedIn, stack overflow, GitHub) links. • The parser and ranker will parse and rank the candidate skills.
Frequency of use	User can interact with the system at any time and get information.
System requirement	Normal, no specific requirement.

Chapter 8

Workflow

8.1 Importing the libraries:

- First, we have to import all the libraries which are required for manipulation or analysis of the data file.
- Like, NumPy and pandas for statistical calculation and analysis of the csv file.
- Matplotlib for plotting the graph like bar chart and pie chart to show the data in the graphical form.
- We require all these in order to manage all the operations from importing till filtering all the candidates as per the given criteria.
- It is essential filter the candidates as per the criteria, if we will not do so than the use of manipulating all the data is not beneficial.

8.2 Importing the dataset:

- After importing the libraries, we are supposed to give the dataset, which contains all the details of the candidates who have applied for different post.
- So, we are having a dataset which is in csv (comma separated values) format.

8.3 Displaying the categories:

- It will display all the categories which we are having in the dataset.
- There are many categories like ‘Data Science’, ‘HR’, ‘Business Analyst’ ‘Blockchain’, ‘Testing’, etc.

- As there are lots of applicants so all are from various backgrounds or disciplines that's why there are lots of categories.

8.4 Distinct categories:

- It shows that how many candidates are present in the various profiles.
- It will help the firm in analysing the total number of applicants in various distinct as per the available vacancies.

8.5 Visualize the dataset:

- It will show you the dataset in the graphical form and a pie chart form.
- It will help you in visualizing the dataset in an efficient manner.

8.6 Clean Dataset:

- A helper function to remove the URLs, hashtags, mentions, special letters and punctuations.
- It will clean the dataset, and present the dataset in more efficient way.
- After cleaning the dataset, a work cloud module is imported to make it appear as a word cloud.

8.7 Train Machine Learning model for Resume Screening:

- Now the next step in the process is to train a model for the task of resume screening.
- We will use the one vs the rest classifier, KNeighborsClassifier.
- First, we will split the data into training and test sets.
- Now, let's train the model and print the classification report.
- So, this way we can train machine learning model for the task of resume screening.

Chapter 9

Machine Learning approach for resume recommendation

9.1 Introduction

Talent acquisition is an important, complex, and time-consuming function within Human Resources (HR). The sheer scale of India's market is overwhelming. Not only is there a staggering one million people coming into the job market every month, but there is also huge turnover. As per LinkedIn, India has the highest percentage of the workforce that is "actively seeking a new job". Clearly, this is an extremely liquid, massive market but one that also has many frustrating inefficiencies. The most challenging part is the lack of a standard structure and format for resume which makes short listing of desired profiles for required roles very tedious and time-consuming. Effective screening of resumes requires domain knowledge, to be able to understand the relevance and applicability of a profile for the job role. With a huge number of different job roles existing today along with the typically large number of applications received, short-listing poses a challenge for the human resource department. Which is only further worsened by the lack of diverse skill and domain knowledge within the HR department, required for effective screening. Being able to weed out non-relevant profiles as early as possible in the pipeline results in cost savings, both in terms of time as well as money.

Today the industry face three major challenges:

• Separating right candidates from the pack - India being a huge job market and with millions seeking jobs; it is humanly impossible to screen the CVs and find the right match. This makes the whole hiring process slow and inefficient costing resources to the companies.

• Making sense of candidate CVs - Second challenges are posed by the fact that the CVs in the market are not standard practically every resume in the market has different structure and format. HR has to manually go through the CVs to find the right match to the job description. This is resource intensive and prone to error whereby a right candidate for the job might get missed in the process.

• Knowing that candidates can do the job before you hire them -The third and the major challenge is mapping the CV to the job description to understand if the candidate would be able to do the job for which she is being hired.

To overcome the mentioned issues in the resume short-listing process, in this paper we present an automated Machine Learning based model. The model takes the features extracted from

the candidate's resume as input and finds their categories, further based on the required job description the categorised resume mapped and recommend the most suitable candidate's profile to HR. Our main contributions are listed below:

1. We developed an automated resume recommendation system.

2. Machine learning based classification techniques with similarity functions are used to find most relevant resume.

3. Linear SVM classifier performed best for our case compared to another ML classifiers.

9.2 Problem Statements

Today the major problem being faced across the industry is how to acquire the right talent, using minimal resources over the internet and in minimal time. As described in section 1, there are three major challenges that are required to be overcome, to bring efficiencies to the complete process.

- Separating the right candidates from the pack
- Making sense of candidate CVs
- Knowing that candidates can do the job before you hire them

Our group intends to provide a solution to the above-mentioned challenges by automating the process. The solution would help to find the right CV from the large dumps of CVs; would be agnostic to the format in which CV has been created and would give with the list of CVs which are the best match to the job description provided by the recruiter. The proposed solution involves supervised learning to classify the resumes into various categories corresponding to the various domains of expertise of the candidates. A multi-pronged approach to classification is proposed as follows:

- Perform NER, NLP, and Text classification using n-grams.
- Use distance-metric based classification.
- The solution shall provide a feedback loop closure to adjust/improve the accuracy by incorporating the feedback corresponding to the incorrectly screened profiles.

9.3 Methodology

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. The proposed model worked in mainly in two steps: i) Prepare and ii) Deploy and Inference.

Dataset Description: The data was downloaded from the online portal(s) and from Kaggle. The data is in Excel format, with three column ID, Category, and Resume.

ID - The sequence number of the resume,

Category – Industry sector to which the resume belongs to, and

Resume - The complete CV of the candidate. The number of instances for the different domain.

9.3.1 Pre-processing

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 tokenizes. Further, the pre-processing 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 string 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

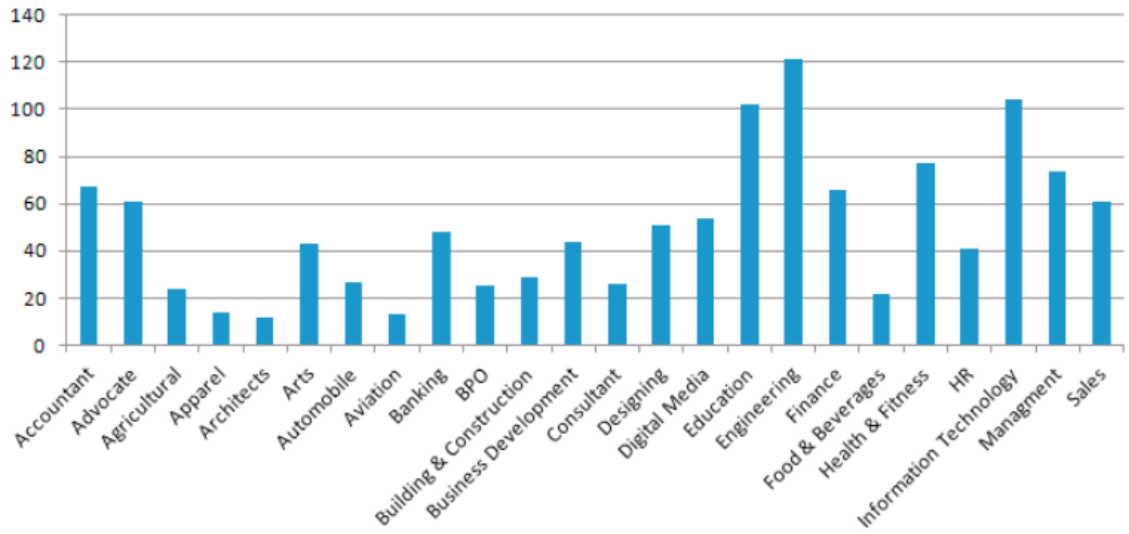


Fig. 9.1 Data distribution over the different domains.

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

1. We have tokenized 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:

   (a) from nltk. corpus import stops words /*Imported Stop Word module from NLTK corpus*/

   (b) StopWords [] = set (stopwords. words('english')) /* Get set of English Stop Words*/

   (c) It returns total of 179 stop words, that can be verified using (len (StopWords)) and can be viewed by print (StopWords) function.

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:

$$\begin{pmatrix} Playing \\ Plays \\ Played \end{pmatrix} ------> \begin{pmatrix} Play \\ (root\ word) \end{pmatrix}$$

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

The next step is feature extraction. On pre-processed 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 pre-processing 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 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 frequency, 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:

- to use a logarithmic form for frequency, sub-linear df is set to True
- min df is the minimum numbers of documents a word must be present in to be kept
- In order to ensure that the all-feature vector have a Euclidean norm of 1, norm is set to l2.
- gram range is set to (n1, n2), where n1=1, and n2=2. It indicates that both uni-grams and bigrams considered.
- to reduce the non-informative features, stop words is set to "english" to remove all common pronouns ("a", "an", "the", "there", ...)

9.3.2 Deployment and Inference

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

9.4 Models

Two 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 and ii) Recommendation -The model would create a summary of the resume and job description provided by the recruiter and give the list of most relevant resume based on the similarity between resume and jobs description.

9.4.1 Classification

The classification was done using four different models and their accuracy score was recorded.

1. **Random Forest (RF)**: RF is an ensemble learning method for classification that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) of the individual trees.

2. **Multinomial Naive Bayes (NB)**: NB classifiers are a family of simple "probabilistic classifiers" based on Bayes' theorem with strong independence assumptions between the features.

3. **Logistic Regression (LR)**: LR uses a logistic function to model a binary dependent variable.

4. **Linear Support Vector Classifier (Linear SVM)**: A SVM is a supervised machine learning classifier which defined by a separating hyperplane. In two-dimensional space, a hyperplane is a line which separates a plane into two separate planes, where each plane belongs to a Class. For example, if the training sample contained two class data such as male (Class 0) and female (Class 1), then the output of the SVM is a line which separates the complete data into two classes using a line. Each plane represents a class of the data either male (Class 0) or female (Class 1).

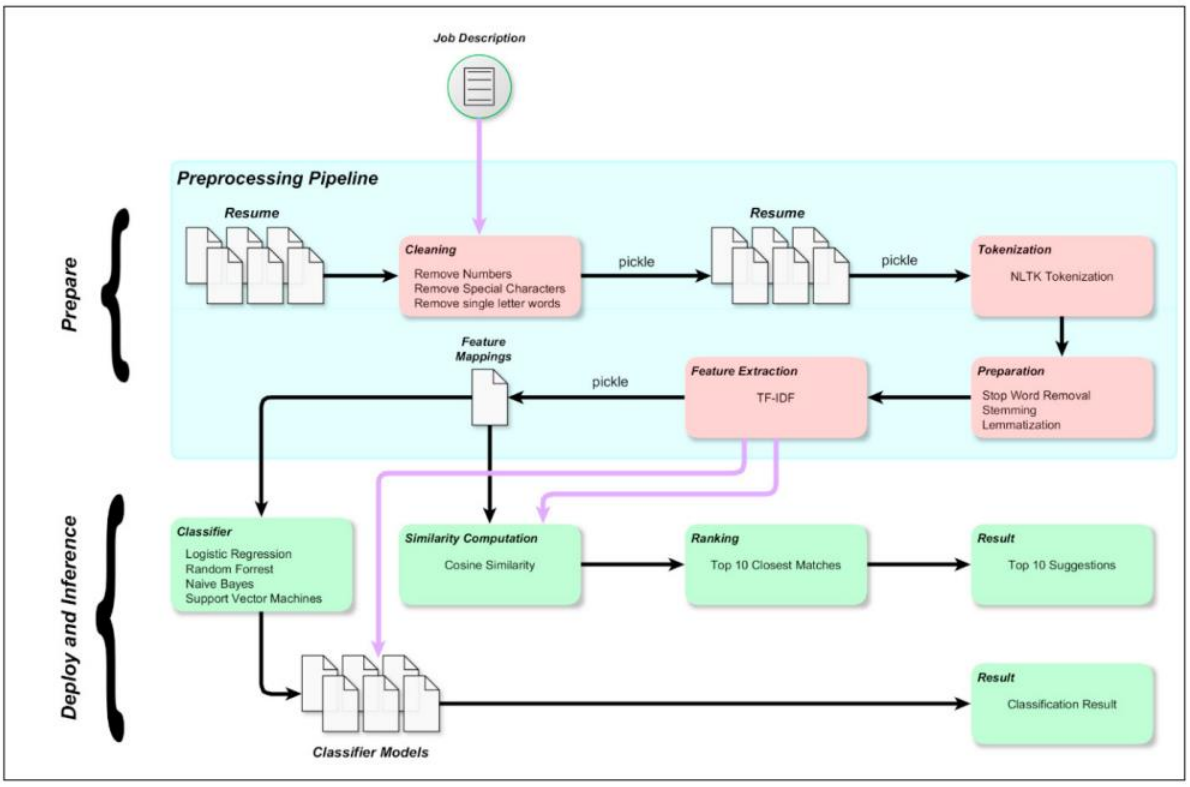


Fig. 9.2 A complete framework of the proposed model

We have started our experiment with RF classifier, the extracted tf-idf features set is fed into the RF classifier to pre-dict the resume category. The RF classifier yielded an accuracy of 38.99% on 10-fold cross-validation. The obtained results were not satisfactory and hence we used another popular classifier named "NB" for this task. NB classifier predicted the categories of resume with an accuracy of 44.39%, which was improved than the earlier classifier's accuracy (RF). However, 44.39% accuracy of NB classifier indicated that more than 50% of the resume was misclassified.

We have used another classifier “Linear SVM” on the same data and achieved 78.53% accuracy. In order to improve the model accuracy, LR classifier was used and obtained 62.40% of accuracy which was lower than that of accuracy of “Linear SVM” classifier. The accuracy of all the models was calculated using 10-fold cross-validation, the average accuracy obtained from the classifiers.

The accuracy score of Linear Support Vector Classifier higher compared to other models have we found this model to reliable and best fit for our objective. Continue with our best model (LinearSVC), we are going to look at the confusion matrix as shown in Figure 4, and show the discrepancies between predicted and actual labels. This is for the classification scope of our problem.

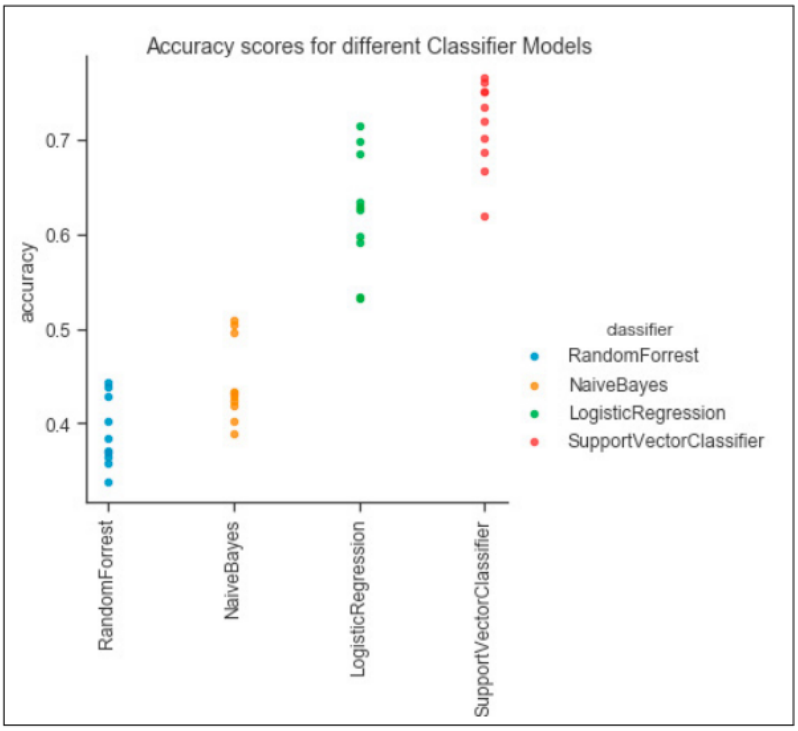


Fig. 9.3 A complete framework of the proposed model

9.4.2 CV Recommendation Model

The recommendation model is designed to take job description and CVs as input and provide the list of CVs which are closest to the provided job description. This is done using two approaches

i) Content Based Recommendation using Cosine Similarity

ii) k-Nearest Neighbours

i) Content Based Recommendation

Considering this is the case of document similarity identification, we have gone with the Content-based recommender where Job Description provided by the employer is matched with the content of resumes in the space and the top n (n being configurable) matching resumes are recommended to the recruiter. The model takes the cleansed resume data and job description and combines the two into a single data set, and then computes the cosine similarity between the job description and CVs.

ii) k-Nearest Neighbours

In this model, k-NN is used to identify the CVs that are nearest to the provided job description, in other words, the CVs that are close match to the provided job description. First, to get the JD and CVs to a similar scale, we have used an open-source library called "genism", this library generates the summary of the provided text in the provided word limit. So, to get the JD and CVs to similar word scale this library was used to generate a summary of JD and CVs and then k-NN was applied to find the CVs which are closely matching the provided JD.

9.5 Implications

The model designed is best suited for the first level of screening of the resumes by the recruiter. This would help the recruiter to classify the resumes as per the requirements and easily identify the CVs that are the best match to the job description. The model would assist the recruiter in hastening the profile shortlisting, at the same time ensuring credibility of the shortlisting process, as they would be able to screen thousands of resumes very quickly, and with the right fit, which would not have been possible for a human to do in near real time. This would aid in making the recruitment process efficient and very effective in identifying the right talent. Also, this would help the recruiter to reduce the resources spent in identifying the right talent making the process cost-effective. On the second level, the model provides the ranking to the CVs as per their fit vis-a-vis the job description, making it easier for the recruiter by giving the resume list in order of their relevance to the job. The recommendation made by the model are currently for the varied industry but the model and be further enhanced to target specific industry which would make it more effective, and give better recommendations.

Chapter 10

Testing

10.1 Unit Testing:

Unit testing is performed for testing modules against detailed design. Inputs to the process are usually compiled modules from the coding process. Each module is assembled into a larger unit during the unit testing process. Testing has been performed on each phase of project design and coding. We carry out the testing of module interface to ensure the proper flow of information into and out of the program unit while testing. We make sure that the temporarily stored data maintains its integrity throughout the algorithm's execution by examining the local data structure. Finally, all error-handling paths are also tested.

10.2 Integration Testing:

Integration testing is used to verify the combining of the software modules. Integration testing addresses the issues associated with the dual problems of verification and program construction. System testing is used to verify, whether the developed system meets the requirements.

- Big Bang is an approach to Integration Testing where all or most of the units are combined together and tested at one go. This approach is taken when the testing team receives the entire software in a bundle.

- Top Down is an approach to Integration Testing where top-level units are tested first and lower level units are tested step by step after that. This approach is taken when top-down development approach is followed. Test Stubs are needed to simulate lower-level units which may not be available during the initial phases.

- Bottom Up is an approach to Integration Testing where bottom level units are tested first and upper-level units' step by step after that. This approach is taken when bottom-up development approach is followed. Test Drivers are needed to simulate higher level units which may not be available during the initial phases.

10.3 Performance Testing:

It is done to test the run-time performance of the software within the context of integrated system. These tests are carried out throughout the testing process. For example, the performance of individual module is accessed during white box testing under unit testing.

10.4 Verification and Validation:

The testing process is a part of broader subject referring to verification and validation. We have to acknowledge the system specifications and try to meet the customer’s requirements and for this sole purpose, we have to verify and validate the product to make sure everything is in place. Verification and validation are two different things. One is performed to ensure that the software correctly implements a specific functionality and other is done to ensure if the customer requirements are properly met or not by the end product. Verification is more like 'are we building the product, right?' and validation is more like 'are we building the right product?'

Chapter 11

Activity Diagram

11.1 Candidate Side:

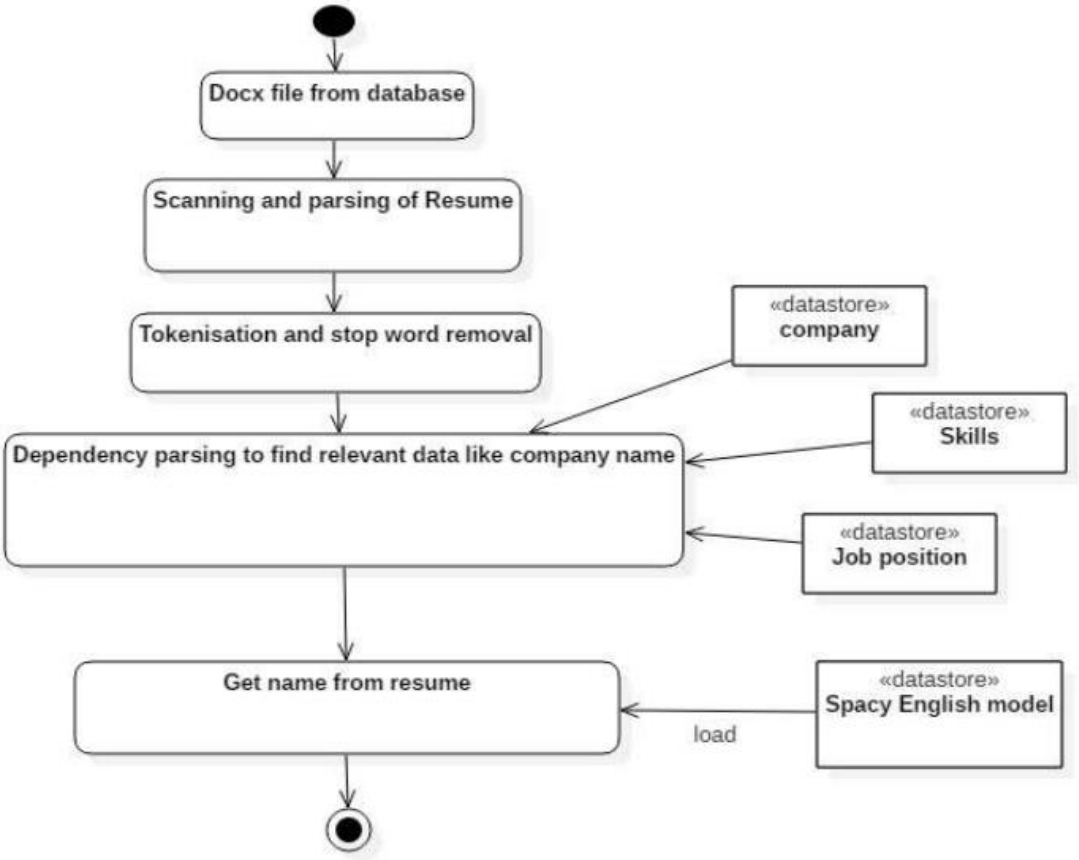


Fig. 11.1 Activity diagram of resume acceptance into the system

- At the candidate's side, the candidate fulfils the initial step of feeding the resume input for the given job description. In this primitive model, there will only be one job description fed into the system as per the recruiter's requirements. Feeding the resume input is a very easy step for the job applicant. All they have to make sure into create their account on this web application and upload their resume on the homepage after they log into the system. The job applicant's details including their resume will directly be saved in the database of the system. As soon as the job applicant clicks on submit, the data extraction using Natural

Language Processing starts on the resume they submitted. This resume is raw and unstructured data and it is a challenge to extract relevant important data.

- To keep up with the simplicity of the working of this web application, resumes in only .docx format are accepted for further data pre-processing. Initially, for the ease of scanning and parsing through the resume, the resume in .docx format is converted into .txt format.

- Tokenization is then performed on this resume text file. After this, the resume in this text format undergoes data pre-processing. Hence, all the unnecessary stop words are truncated from the resume. For further processing, dependency parsing and Natural Language Processing using SpaCy are used to extract only that information which is relevant and vital for job matching. SpaCy uses the concept of NER (Named Entity Recognition) to identify categories such as name, company name etc. from raw and unstructured data. In order to achieve this, three datasets in .csv format are loaded into the system. These datasets have a catalogue of job positions, company names and skills in job position dataset, company dataset and skills dataset respectively.

- Since tokenization has already been performed on the resume, it is extremely essential to use dependency parsing for the relevant data extraction. In this concept, neighbouring words are also considered along with the main word for relevant data extraction with the help of the datasets.

- For example, to extract important data from the text like business technology analyst as a job position, when the parser lands on the word business, it will look for words before as well as after this word before comparing whether this group of words exists in the catalogue of numerous job positions.

- To extract the name from the resume, the ‘en’ model, i.e., the English model of SpaCy framework of NLP is used. The name is extracted accurately after this framework is loaded into the system. Finally, contact details of the candidate, such as phone number, email id etc. is extracted using Regex.

11.2 Recruiter Side:

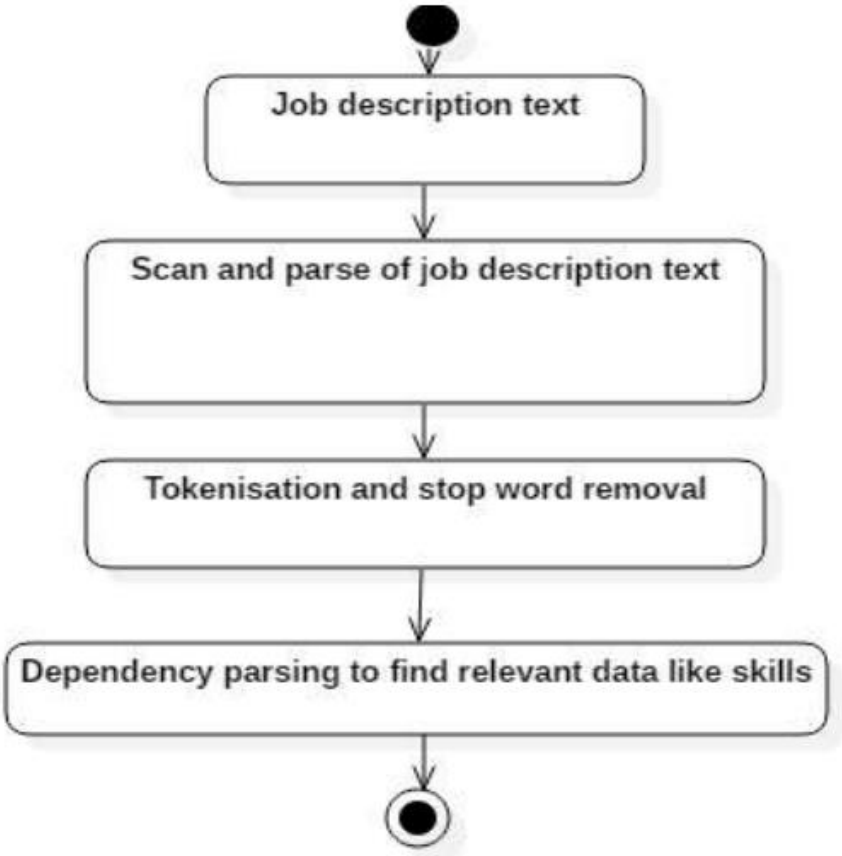


Fig. 11.2 Activity diagram of data extraction from job description

- It is imperative to understand what happens at the recruiter's side too. For keeping up the simplicity with this NLP model, there is only one description which is allowed for one job recruiter to be provided as input. The job description, just like any candidate's resume, has to go through scanning and parsing to only capture important data out of it. Here also tokenization, data pre-processing like stop word removal and dependency parsing is performed. Fig 4 depicts the extraction of relevant data from the job description.

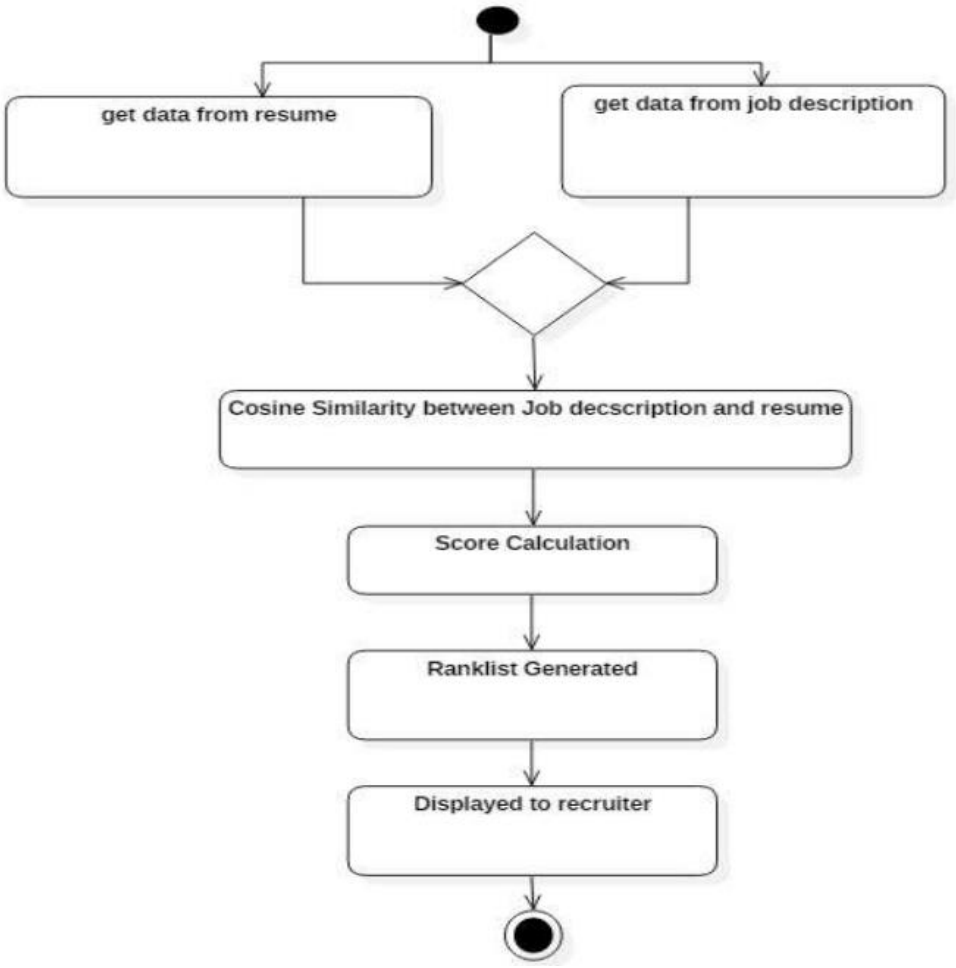


Fig. 11.3 Activity diagram depicting the output at the recruiter end

The prototype developed for this system is primitive and has a lot of scope for betterment in the future. Higher accuracy can be aimed for the NLP model for the future prototypes. A better database option can be used for storing humungous amounts of data of the job applicants as well as the job recruiters. Many job recruiters' accounts can be assimilated with the existing prototype for the flexibility of use for various types of job recruiters. This will fulfill the aim of developing this model for reducing the workload of the job recruiters in selecting the right employee for their respective company. The most important detail is that there were issues while calculating the number of years of relevant experience for any prospective job candidate. During the calculation of the number of years from the raw unstructured data of the candidate's resume, the number of years of education of the candidate is also getting added along with the number of years of work and internship experience. Hence, manual input had to be provided for the job applicant for the evaluation of the number of years of experience in the overall resume score.

Chapter 12

Snapshots

Coding Snapshot:

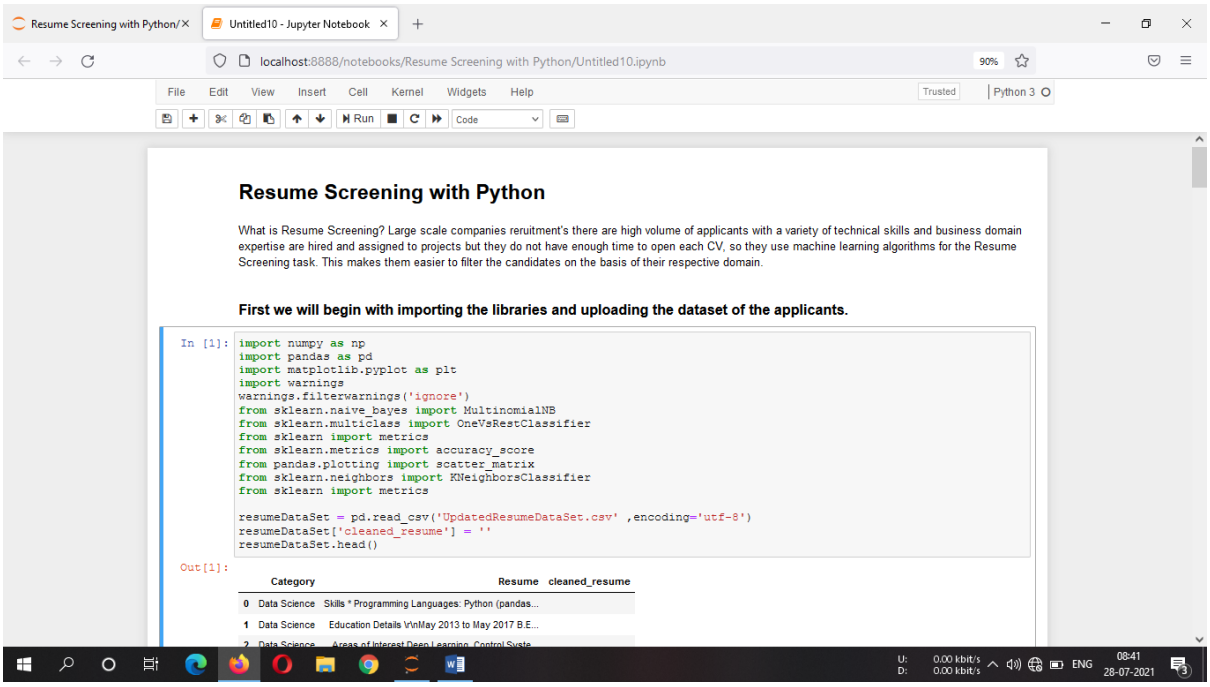


Figure 11.1

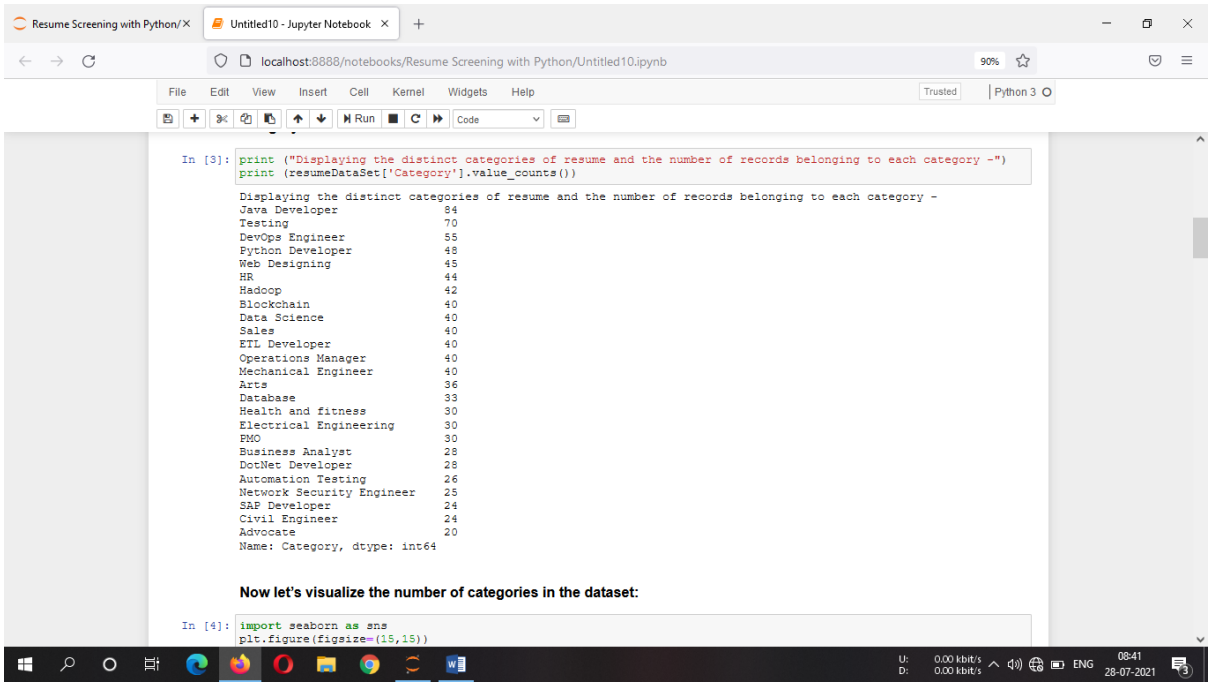


Figure 11.2

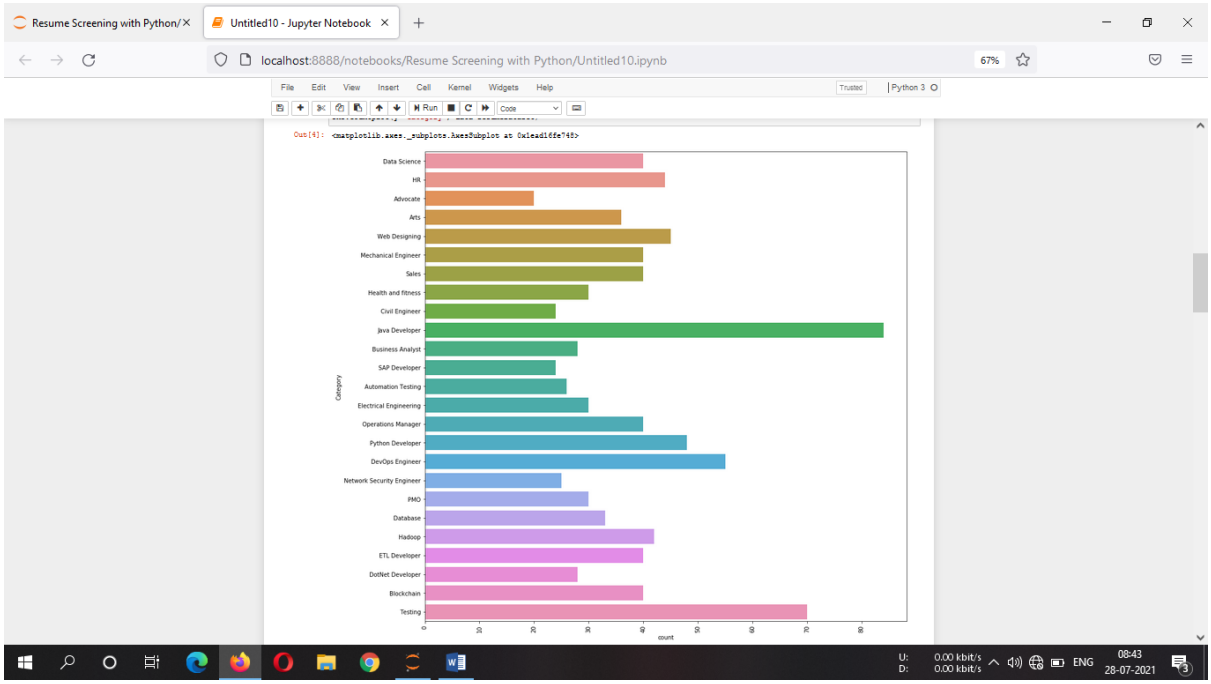


Figure 11.3

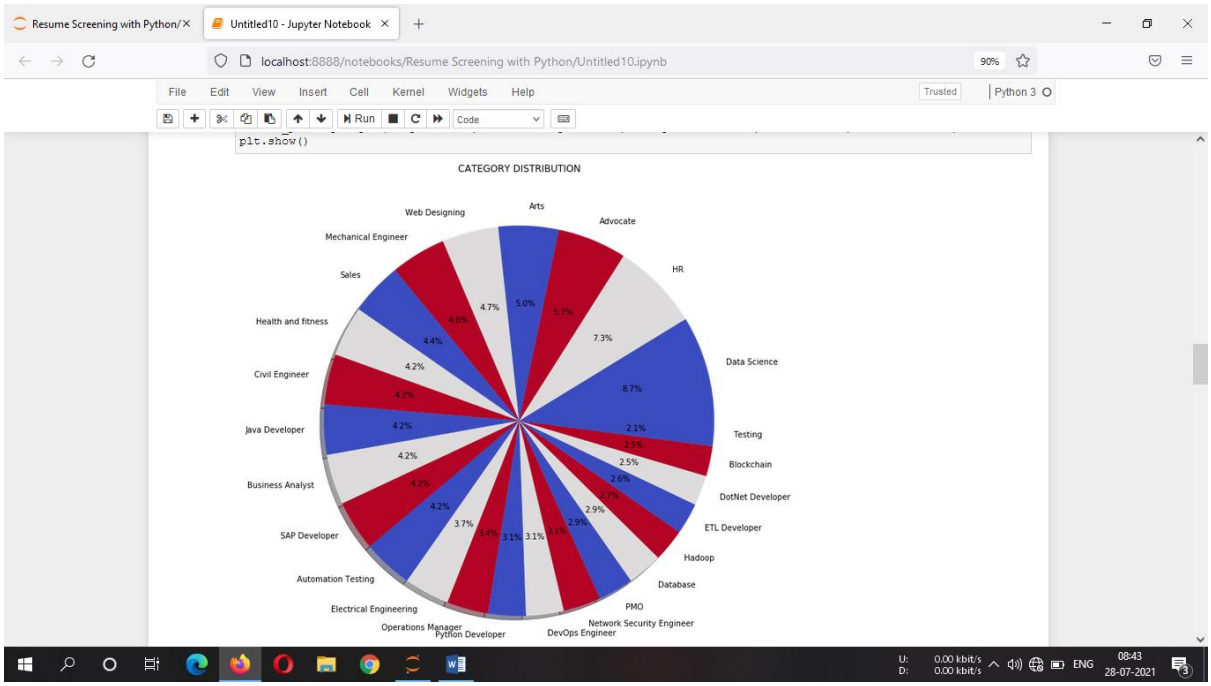


Figure 11.4

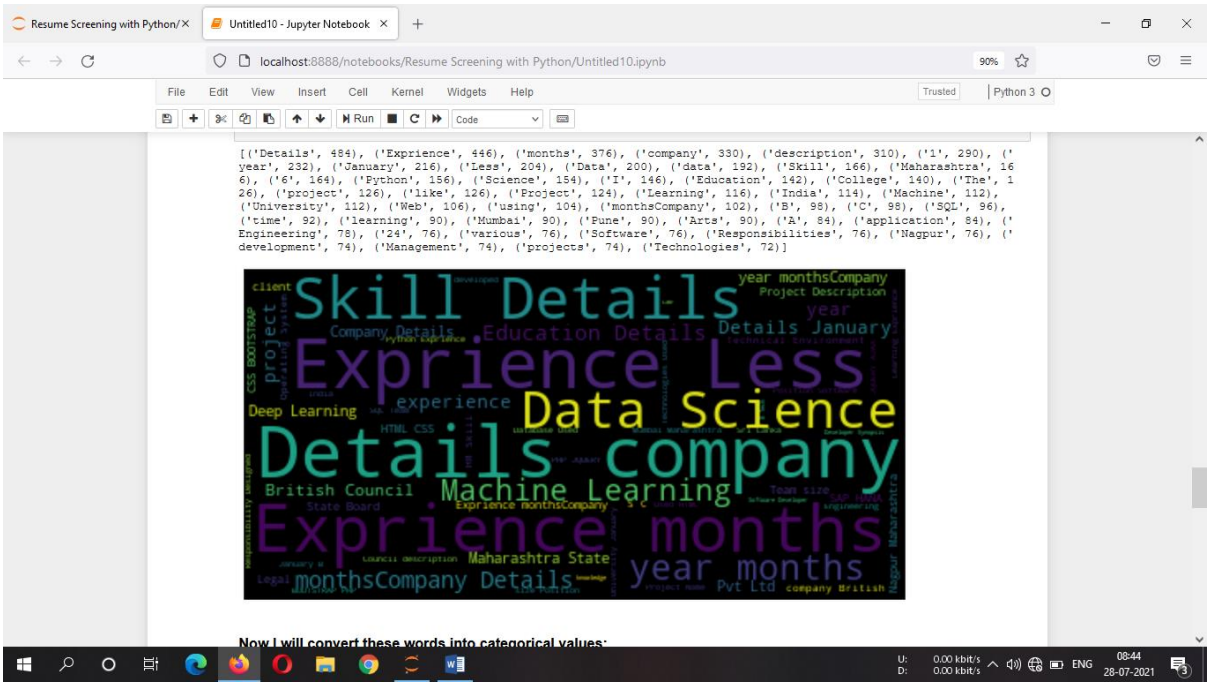


Figure 11.5

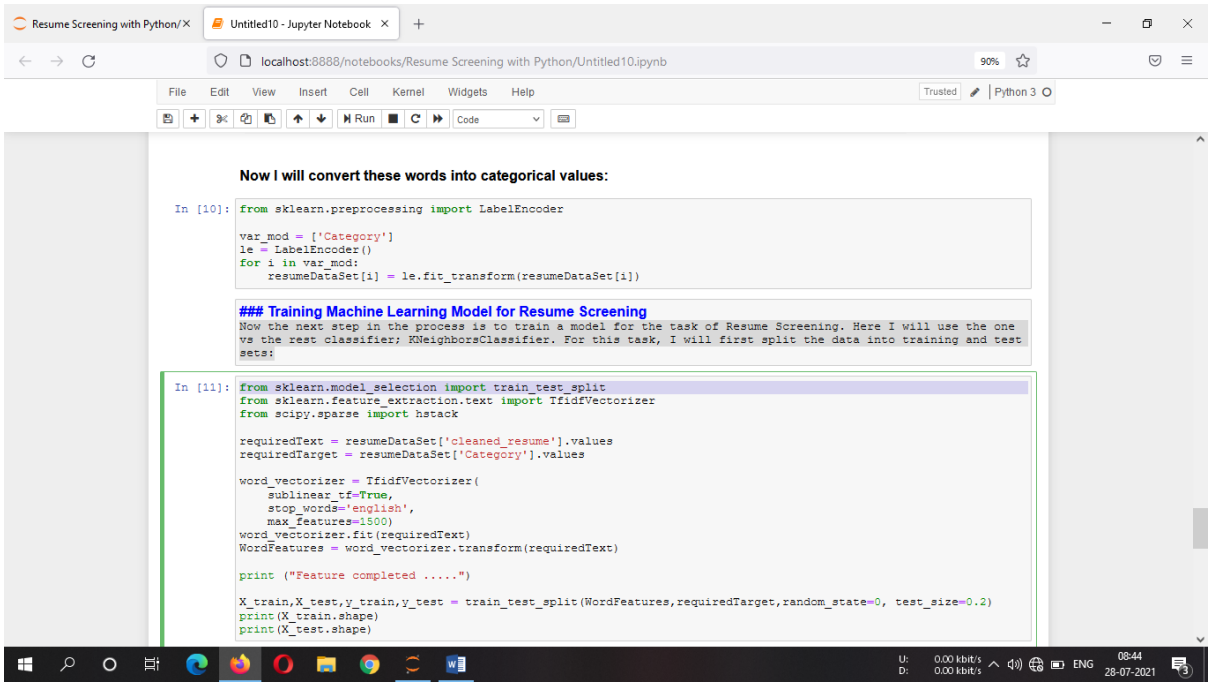


Figure 11.6

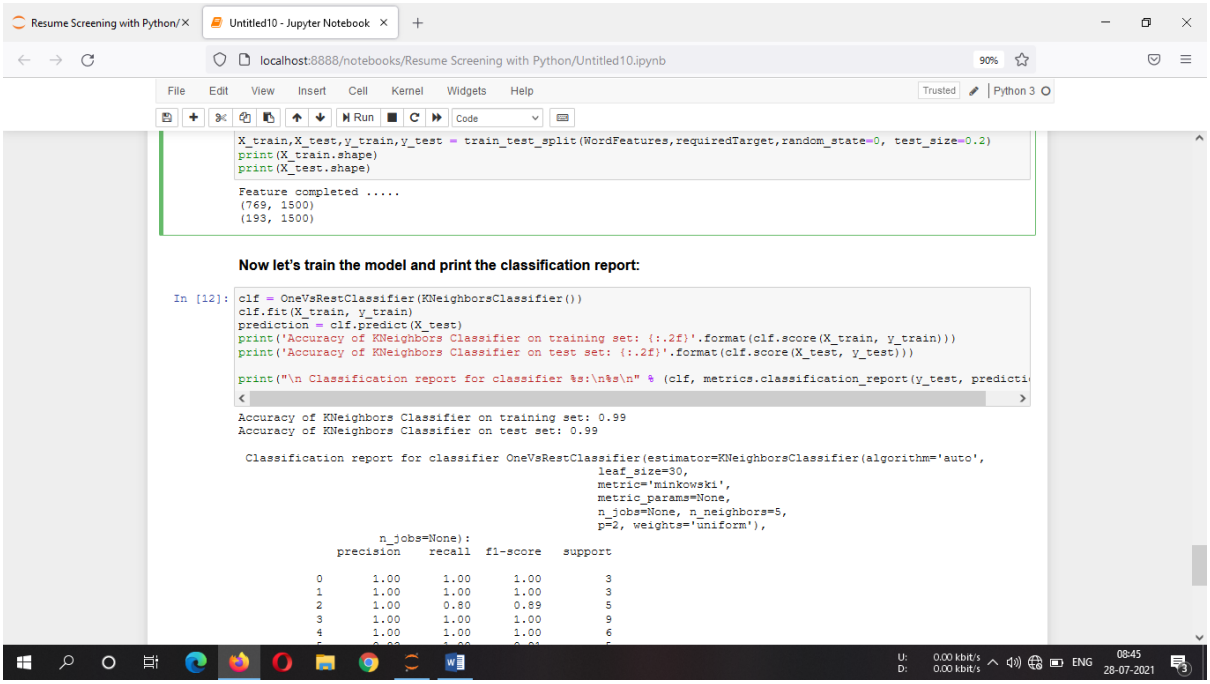


Figure 11.7

Chapter 12

Result and 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 resume to the HR based on given job description. The proposed model worked in two phases: first, classify the resume into different categories. Second, recommends resume based on the similarity index with the given job description. The proposed approach effectively captures the resume insights, their semantics and yielded an accuracy of 78.53% with Linear SVM classifier. The performance of the model may enhance by utilizing the deep learning models like: Convolutional Neural Network, Recurrent Neural Network, or Long-Short Term Memory and others. If an Industry provides a large number of resume, then Industry specific model can be developed by utilizing the proposed approach. By involving the domain experts like HR professional would help to build a more accurate model, feedback of the HR professional helps to improve the model iteratively.

Chapter 13

Limitations and Future Enhancements

There are few limitations to the model design as of now, but these can be overcome by having more data to train the model. The current limitation of the model are i) Model takes CVs in CSV format, but in the real world, the CVs are either in .doc, .pdf, etc format. Due to the limitation of the data set, the model could not be enhanced to take .doc or .pdf as input, but using a library "textract" this can be achieved. The library can read varied file format and convert them into a single format which can be used as input to the model, ii) Generation of a summary using "genism" library might have caused loss of important information due to implicit compression of the text due to summarization. There is the scope of fine-tuning this summarization process to ensure minimal information loss, for example, important features of data like candidate skill and experience are not lost.

References

- **https://thecleverprogrammer.com/2020/12/06/resume-screening-with-python/**
- **https://www.kdnuggets.com/2019/02/nlp-spacy-data-science-resumes.html**
- **https://www.sciencedirect.com/science/article/pii/S187705092030750X**
- **https://www.kaggle.com/gauravduttakiit/resume-screening-using-machine-learning**
- **https://bit.ly/2X8srUh**