A Project Report on

PYTHON POWERED RESUME SCREENING AND ANALYSIS

Submitted in Fulfillment of Requirement for The Award of The Degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

K. VASANTHI (21NP5A0504)

M. LOHITA (20NP1A0591)

M. SAI DRUGA (20NP1A0592)

T. MAHA LAKSHMI (20NP1A05B2)

Under The Esteemed Guidance Of

Ms. SHALINI TAMMANA, M. Tech

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VIJAYA INSTITUTE OF TECHNOLOGY FOR WOMEN

(Affiliated To J.N.T.U.K Kakinada, Approved By A.I.C.T.E, New Delhi)

Enikepadu, Vijayawada-521108,

2023-2024

VIJAYA INSTITUTE OF TECHNOLOGY FOR WOMEN

(An ISO 9001:2015 Certified Institute, Approved by AICTE, New Delhi and Affiliated to JNTU- Kakinada, A.P, COLLEGE CODE: NP, Phone: 0866-2844444,

Email: vijayatechfw@gmail.com,

Web: vitw.edu.in

Enikepadu, Vijayawada-521108

CERTIFICATE

This is to certify that the dissertation entitled "**PYTHON POWERED RESUME SCREENING AND ANALYSIS**" is a bonafide work done by K. VASANTHI (21NP5A0504), M. LOHITA (20NP1A0591), M. SAI DURGA (20NP1A0592), T. MAHALAKSHMI (20NP1A05B2), under my guidance and supervision and is submitted to JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA in fulfillment for the award of the degree of "Bachelor of Technology" in CSE is a record of confined work carried out by her under my guidance and supervision during the academic year 2023-2024 and it has been found worthy of acceptance according to the requirements of the university.

PROJECT GUIDE	**SIGNATURE OF HOD**
Ms. T. Shalini, M. Tech	**Dr. P. Subbaiah,** BE, MBA, M. Tech, Ph. D

EXTERNAL EXAMINER

DECLARATION

Here by we declare that this project work entitled "**PYTHON POWERED RESUME SCREENING**" is a genuine work carried out by us, for the fulfillment of Bachelor of Technology to the Department. of Computer Science & Engineering during the academic year 2023-2024 under the supervision of my project guide **Ms. T. SHALINI** and submitted to the Department of CSE, **Vijaya Institute of Technology for Women** and that it has not formed the basis for the award of any Degree/ or other similar title to any candidate of the university.

TEAM MEMBERS

K. VASANTHI	**21NP5A0504**
M. LOHITA	**20NP1A0591**
M. SAI DURGA	**20NP1A0592**
T. MAHALAKSHMI	**20NP1A05B2**

ACKNOWLEDGEMENT

I extend my heartfelt gratitude to **Dr. G. CHENCHAMMA**, **PRINCIPAL** for her unwavering support and for providing the necessary resources and facilities that are instrumental in the successful completion of this project.

I am deeply appreciative for the encouragement provided by **Dr. P. SUBBAIAH**, **Professor and Head of the Department**, whose expertise and mentorship greatly contributed to the project's development.

I am also grateful to my project guide, **Ms. SHALINI TAMMANA**, **Project Guide** for her invaluable assistance and continuous support throughout the duration of this endeavor. Her insights and feedback were invaluable in shaping the direction of the project.

Furthermore, I would like to thank our external project guide, **Mr. Md. UMARUL FARUK ALI** for his expertise and valuable contributions, which enriched the project's outcomes. His insights and suggestions were invaluable in broadening the project's scope and depth.

The provision of facilities and support from each of these individuals have been instrumental in the successful execution of this project, and I am truly thankful for their guidance, encouragement, and belief in my abilities.

Special thanks are due to my family for their unwavering encouragement and understanding during this challenging time.

TEAM MEMBERS

K. VASANTHI	**21NP5A0504**
M. LOHITA	**20NP1A0591**
M. SAI DURGA	**20NP1A0592**
T. MAHALAKSHMI	**20NP1A05B2**

INDEX

LIST OF FIGURES

ABSTRACT

ABSTRACT

Resume screening is the process of analyzing the resumes where the candidates apply for the different types of jobs where the company feel the tedious job to find the appropriate candidate due to the complexity in resumes formats since it has different styles. As a result, selecting applicants for the appropriate job within a company is a difficult task for recruiters. We can extract the key information from the CV using NLTK, Natural Language Processing (NLP) techniques to save time and effort. This system could work with many resumes for classifying the right categories using different classifiers like KNN. Furthermore, this system attempts to find the accuracy and performance of the proposed methodology and incorporate it in the IT firms and other regulations for the prevention of manual screening and establish a safe allocation of resources for the companies.

Index Terms—Resume, CV, NLTK, NLP, KNN, ONE-VS-REST CLASSIFIER

TALENT SPOT: PYTHON POWERED RESUME SCREENING AND ANALYSIS

INTRODUCTION

CHAPTER-1

INTRODUCTION

The Python-powered resume screening and analysis tool is designed to help recruiters and hiring managers quickly and efficiently screen resumes and identify the most qualified candidates for a job posting. The tool uses NLP and ML techniques to analyze the text in resumes and job descriptions, and ranks resumes based on their relevance to the job posting.

In today's fast-paced job market, organizations are inundated with countless resumes for each job posting. Sorting through this deluge of applications can be a daunting and time-consuming task for recruiters. However, with the advent of Python-powered resume screening systems, the process has become significantly more efficient and effective. This essay explores the fundamentals of resume screening using Python and how it revolutionizes the hiring process. At its core, resume screening involves analyzing resumes to identify candidates who possess the requisite skills and qualifications for a job position.

Traditionally, this task was carried out manually, requiring human recruiters to meticulously review each resume. However, this approach is not only labor-intensive but also prone to biases and inconsistencies. Python-powered resume screening systems offer a viable solution to these challenges by automating the process and enabling objective evaluation based on predefined criteria.

Hiring the right talent is a challenge for all businesses. This challenge is magnified by the high volume of applicants if the business is labor intensive, growing, and facing high attrition rates. An example of such a business is that IT departments are short of growing markets. In a typical service organization, professionals with a variety of technical skills and business domain expertise are hired and assigned to projects to resolve customer issues.

This task of selecting the best talent among many is known as Resume Screening. Typically, large companies do not have enough time to open each CV, so they use machine learning algorithms for the Resume Screening task and by this unemployment rate [8] also reduced with efficient hiring. Machine learning is a field in which we train a model with data to anticipate the intended outcome when new data is submitted.

Natural language processing (NLP) is a commonly used to screen resumes. Natural language refers to how humans communicate with one another. In the NLP the system enables us to find the text based on the English dictionary in the same way as humans.

NLP combines statistical, machine learning, and deep learning models of human language with computational linguistics-based rule-based modeling, here we need to check for the data from different formats which are either in the form of the document or either in the form of the audio data and understanding the whole meaning of it. The number of applications is in the millions, making it a time-consuming chore to sort through them.

With the help of advanced algorithms and natural language processing, we're revolutionizing the way we evaluate candidates. Instead of manually sifting through piles of resumes, our system intelligently analyzes each document to identify top talents efficiently and accurately. By harnessing the power of Python, we ensure a streamlined process that saves time, reduces bias, and ultimately helps us build the strongest team possible. Let us dive in and discover the future of recruitment together.

Candidates, imagine your resume not just being glanced at, but truly understood. Our Python-powered resume screening system goes beyond simple keyword matching. It delves deep into the content, extracting meaningful insights about your skills, experiences, and potential cultural fit.

This cutting-edge technology allows us to identify the best candidates faster and more effectively, giving you a fair chance to showcase your talents. Gone are the days of resumes lost in a sea of applicants. With our system, your qualifications shine brighter than ever before. Join us as we embrace innovation and reimagine the hiring process for the better.

Employers, welcome to a new era of recruitment efficiency and accuracy. Our Python-powered resume screening solution offers you unparalleled insights into candidate suitability. By leveraging state-of-the-art machine learning algorithms, we not only match keywords but understand context, relevance, and potential.

This means fewer missed opportunities and more qualified candidates in your pipeline. With Python at the helm, we are not just automating tasks; we are enhancing decision-making with data-driven intelligence. Say goodbye to tedious manual screening and hello to a smarter, faster, and more reliable hiring process. Let us unlock the full potential of your talent acquisition strategy together.

Job seekers, imagine a recruitment process that sees beyond the surface of your resume, one that truly comprehends your unique skills and experiences. With our Python-powered resume screening, that vision becomes a reality. By harnessing the power of Python, we are not only saving time but ensuring fairness and accuracy in candidate evaluation.

Our system does not just scan for keywords; it understands the nuances of your achievements and capabilities, giving you a genuine opportunity to shine. So, whether you are a seasoned professional or just starting your career journey, rest assured that your potential won't go unnoticed. Join us as we redefine the standards of recruitment and empower you to showcase your true worth.

Our Python-powered resume screening solution offers you a panoramic view of candidate suitability. By leveraging advanced machine learning models, we go beyond surface-level keyword matches, diving deep into each applicant's qualifications, experiences, and potential cultural fit.

With Python driving the analysis, we are not just automating tasks; we are elevating decision-making with data-driven intelligence. Say goodbye to the inefficiencies of manual resume screening and hello to a streamlined, data-powered approach that ensures you are only considering the best-fit candidates for your team. Let us embark on a journey where hiring becomes not just a process, but a strategic advantage.

WHAT IS MACHINE LEARNING?

Machine learning (ML) is a subset of artificial intelligence (AI) that enables computer systems to automatically learn and improve from experience without being explicitly programmed. ML algorithms use statistical models to analyze and draw inferences from patterns in data, allowing the system to make predictions or decisions based on new data.

Machine learning methods enable computers to operate autonomously without explicit programming. ML applications are fed with new data, and they can independently learn, grow, develop, and adapt.

Machine learning derives insightful information from large volumes of data by leveraging algorithms to identify patterns and learn in an iterative process. ML algorithms use computation methods to learn directly from data instead of relying on any predetermined equation that may serve as a model.

The performance of ML algorithms adaptively improves with an increase in the number of available samples during the 'learning' processes.

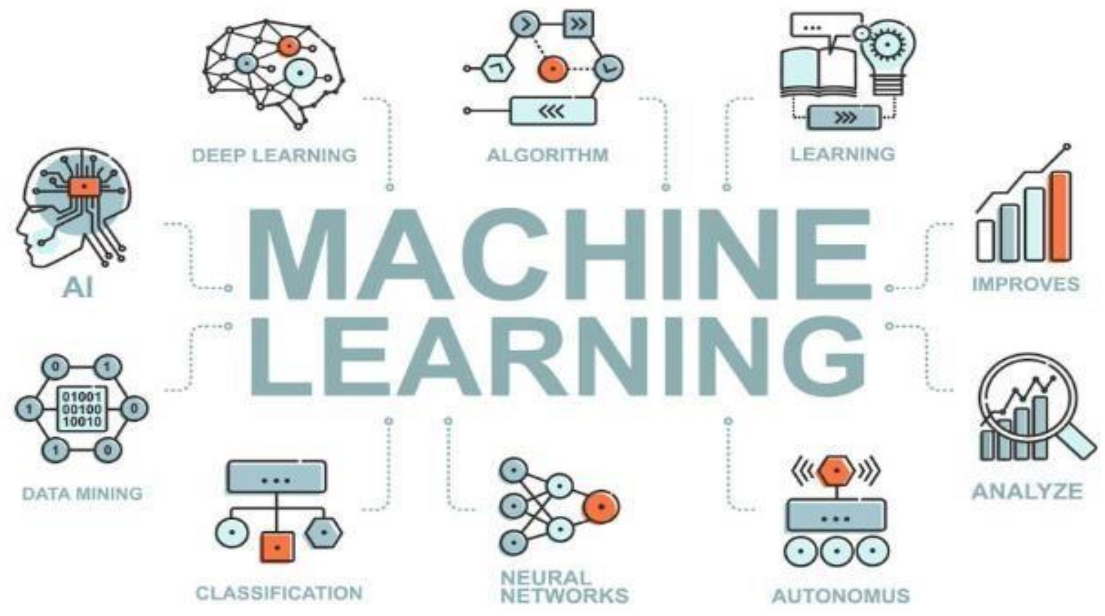


Fig 1.1: INTRODUCTION TO MACHINE LEARNING

HOW DOES MACHINE LEARNING WORK?

Machine learning is the brain where all the learning takes place. The way the machine learns is like the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if it is feed a previously unseen example, the machine has difficulties to predict.

The core objective of machine learning is the learning and inference. First, the machine learns through the discovery of patterns. This discovery is made thanks to the data. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a feature vector. You can think of a feature vector as a subset of data that is used to tackle a problem. The machine uses some fancy algorithms to simplify the reality and transform this discovery into a model. Therefore, the learning stage is used to describe the data and summarize it into a model.

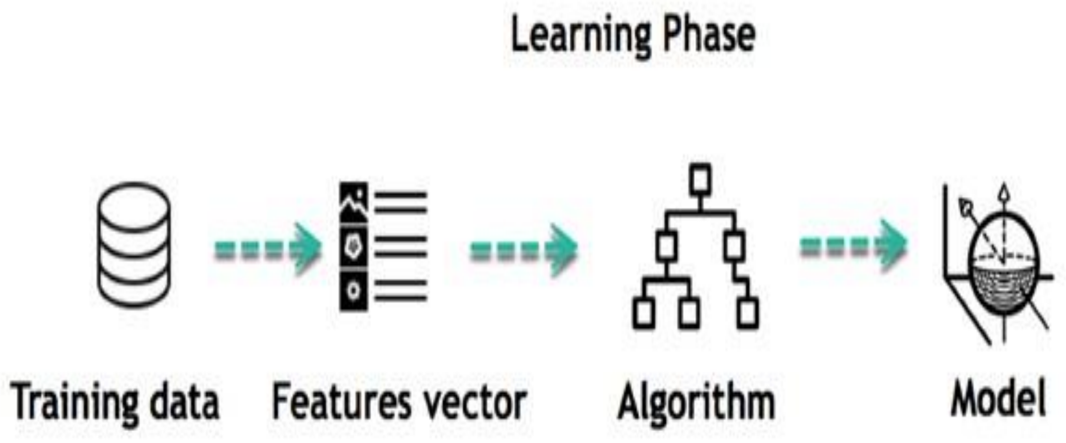


Fig 1.2: HOW DOES MACHINE LEARNING WORKS

INFERRING

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model, and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

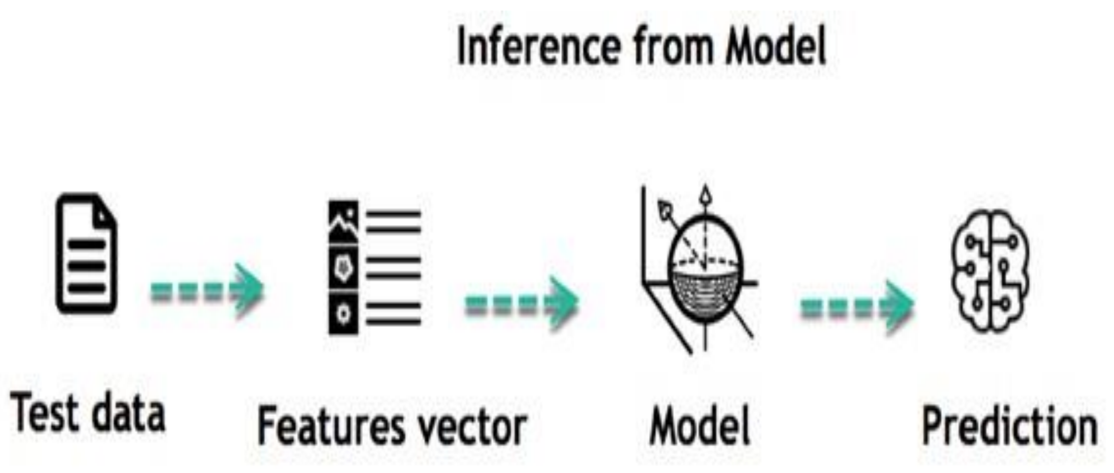


Fig 1.3: INFERENCE MODEL

The life of Machine Learning programs is straightforward and can be summarized in the following points:

1. Define a question
2. Collect data
3. Visualize data
4. Train algorithm
5. Test the Algorithm
6. Collect feedback
7. Refine the algorithm
8. Loop 4-7 until the results are satisfying
9. Use the model to make a prediction

Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data.

MACHINE LEARNING ALGORITHMS AND WHERE THEY ARE USED

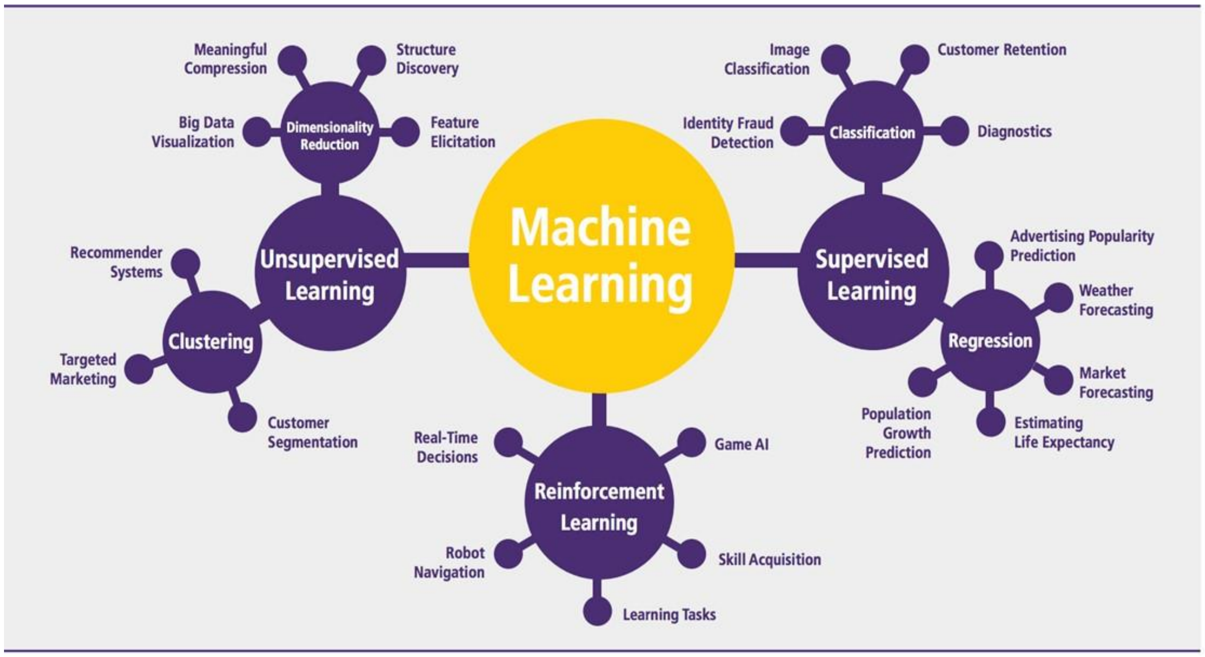


Fig 1.4: MACHINE LEARNING ALGORITHMS

Machine learning algorithms are computational models that allow computers to understand patterns and forecast or make judgments based on data without the need for explicit programming. These algorithms form the foundation of modern artificial intelligence and are used in a wide range of applications, including image and speech recognition, natural language processing, recommendation systems, fraud detection, autonomous cars etc., there are many other algorithms.

- SUPERVISED
- UNSUPERVISED

SUPERVISED LEARNING

An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans. You can use supervised learning when the output data is known. The algorithm will predict new data. There are two categories of supervised learning:

- Classification task
- Regression task

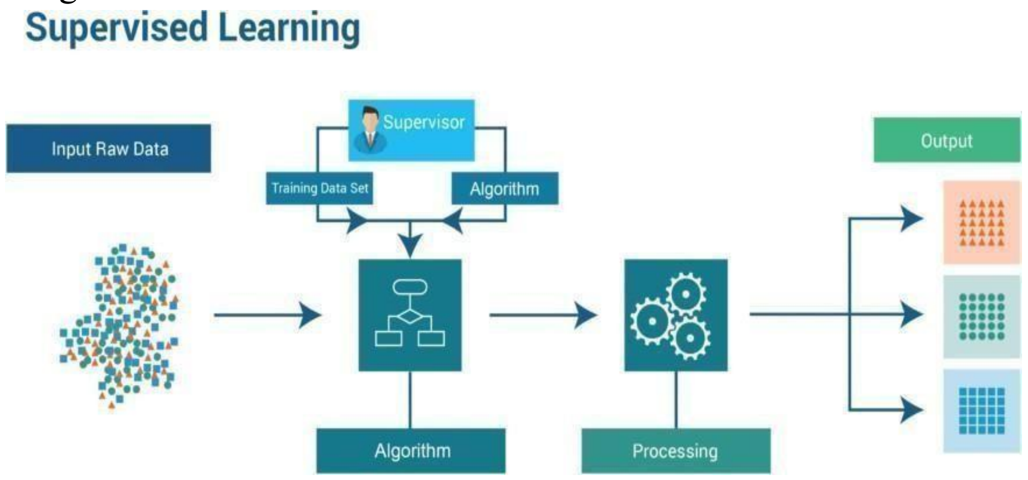


Fig 1.5: SUPERVISED LEARNING

In supervised learning, the computer is provided with example inputs that are labeled with their desired outputs. The purpose of this method is for the algorithm to be able to "learn" by comparing its actual output with the "taught" outputs to find errors, and modify the model accordingly. Supervised learning therefore uses patterns to predict label values on additional unlabeled data.

A common use case of supervised learning is to use historical data to predict statistically likely future events. It may use historical stock market information to anticipate upcoming fluctuations, or be employed to filter out spam emails. In supervised learning, tagged photos of dogs can be used as input data to classify untagged photos of dogs.

UNSUPERVISED LEARNING

In unsupervised learning, data is unlabeled, so the learning algorithm is left to find commonalities among its input data. As unlabeled data are more abundant than labeled data, machine learning methods that facilitate unsupervised learning are particularly valuable.

The goal of unsupervised learning may be as straightforward as discovering hidden patterns within a dataset, but it may also have a goal of feature learning, which allows the computational machine to automatically discover the representations that are needed to classify raw data. Unsupervised learning is commonly used for transactional data.

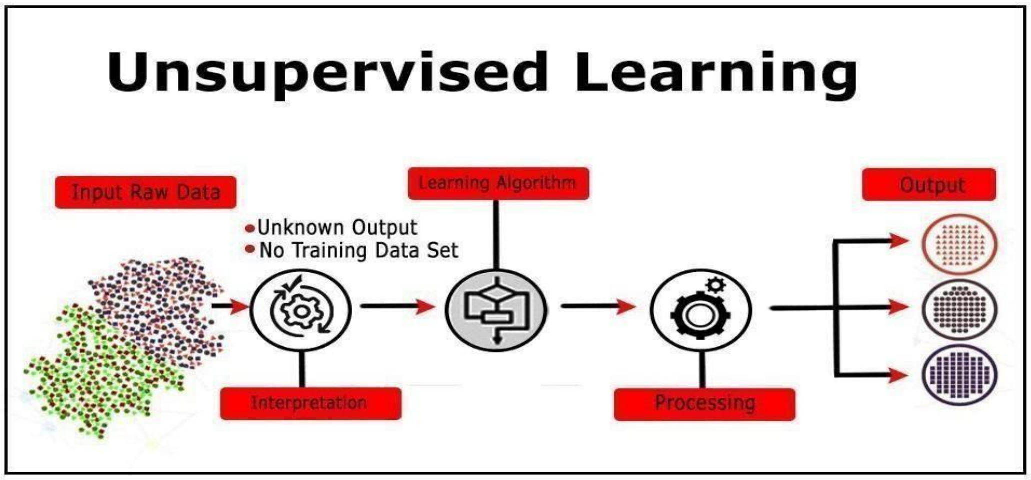


Fig 1.6: UNSUPERVISED LEARNING

Without being told a “correct” answer, unsupervised learning methods can look at complex data that is more expansive and seemingly unrelated to organize it in potentially meaningful ways. Unsupervised learning is often used for anomaly detection including for fraudulent credit card purchases, and recommender systems that recommend what products to buy next.

APPLICATION OF MACHINE LEARNING:

- Face Recognition
- Social Media Services
- Virtual Personal Assistants
- Online Fraud Detection
- Product Recommendations
- Self-Driving Cars
- Stock Market Trading

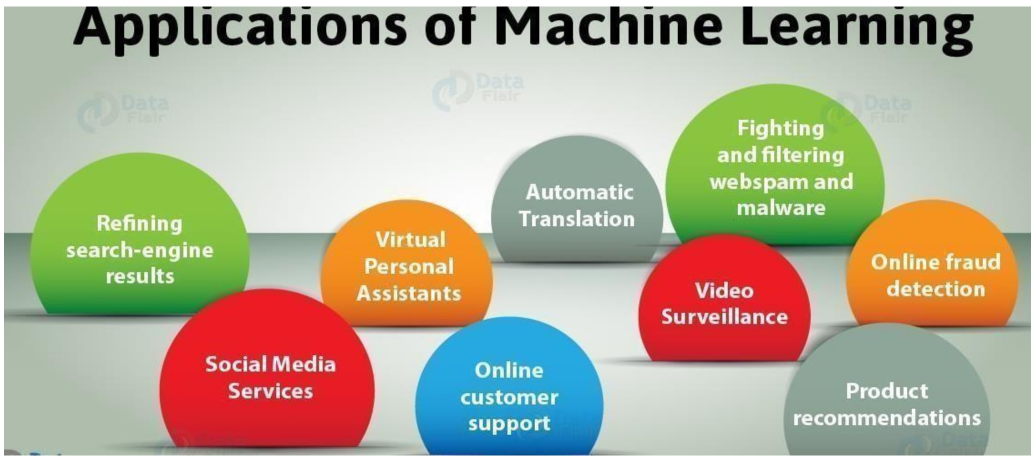

Applications of Machine Learning
Refining search-engine results
Social Media Services
Virtual Personal Assistants
Online customer support
Automatic Translation
Fighting and filtering webspam and malware
Video Surveillance
Online fraud detection
Product recommendations

Fig 1.7: APPLICATIONS OF MACHINE LEARNING

MACHINE LEARNING ALGORITHMS USED

1. K-NEAREST NEIGHBOUR CLASSIFIER

The KNN algorithm is used to measure the distance between the given test instance and all the instances in the data set, this is done by choosing the ‘k’ closest instances and then predict the class value based on these nearest neighbors. The ‘k’ is assigned as number of neighbors voting on the test instance. As such KNN is often referred to as case-based learning or an instance-based learning where each training instance is a case from the problem domain. KNN is also referred to as a lazy learning algorithm since there is no learning of the model required and all the computation works happen at the time a prediction is requested. KNN is a non-parametric machine learning algorithm as it makes no assumptions about the functional form of the problem being solved. Each prediction is made for a new instance (x) by searching through the entire training set for the ‘k’ nearest instances and applying majority voting rule to determine the prediction outcome.

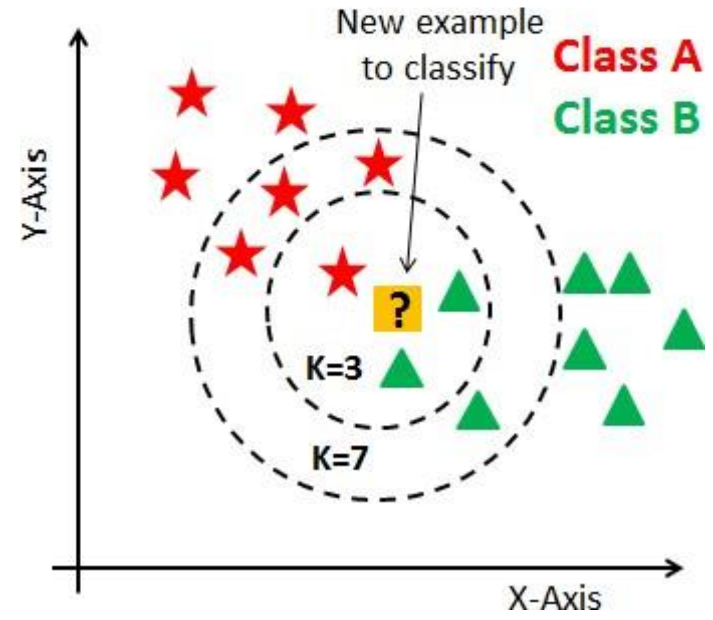


Fig 1.8: K-NN CLASSIFIER

A variety of distance functions are available in KNN which include Euclidean, Manhattan, Minkowski, and Hamming. The Euclidean distance function is probably the most used in any distance-based algorithm. It is defined as

$$D\ (x, y) = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$$

(1) Where, x and y are two data vectors and k are the number of attributes. The Manhattan distance function is defined as

$$\mathbf{D\ (x, y)} = (\sum_{i=1}^{k} |x_i - y_i|)$$

(2) The Minkowski distance function is defined as

$$\mathbf{D\ (x, y)} = (\sum^{k} (|x_i - y_i|)^p)^{1/p}$$

(3) The distance functions for Euclidean, Manhattan and Minkowski are used for numerical attributes. The Hamming distance function is defined as

$$\mathbf{D\ (x, y)} = {}^{k}\ |x_i - y_i|$$

x = y => D = 0 x ≠

y => D = 1

(4) Hamming distance is usually used for categorical attributes. The Hamming distance between two data vectors is the number of attributes in which they differ.

Advantages of KNN Algorithm:

- It is simple to implement.
- It is robust to the noisy training data • It can be more effective if the training data is large.

Disadvantages of KNN Algorithm:

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the datapoints for all the training samples.

WHY DO WE NEED A KNN ALGORITHM?

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:

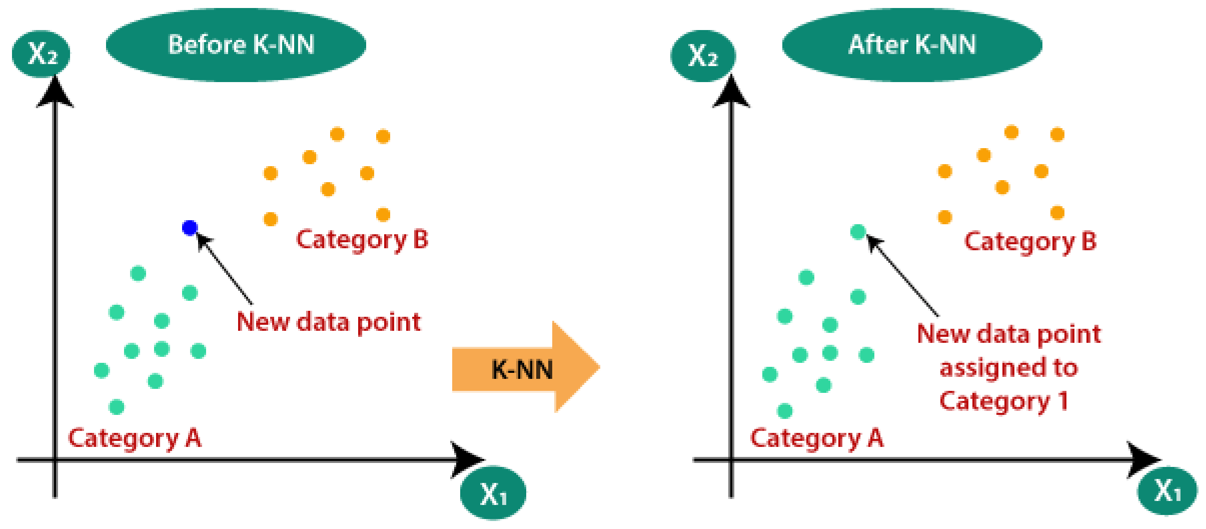


Fig 1.9: BEFORE AND AFTER KNN

HOW DOES KNN WORK?

The K-NN working can be explained based on the below algorithm:

- Step-1: Select the number K of the neighbors
- Step-2: Calculate the Euclidean distance of K number of neighbors
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbor is maximum. o Step-6: Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:

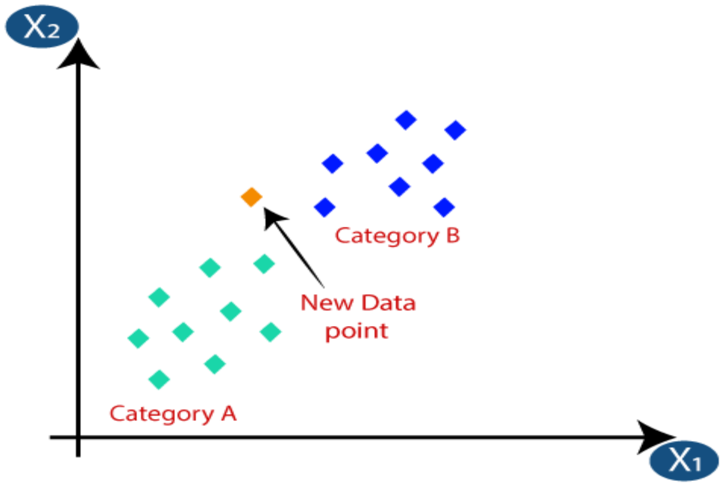


Fig 2.0: KNN CATEGORICAL DISTRIBUTION

- Firstly, we will choose the number of neighbors, so we will choose the k=5.
- Next, we will calculate the Euclidean distance between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:

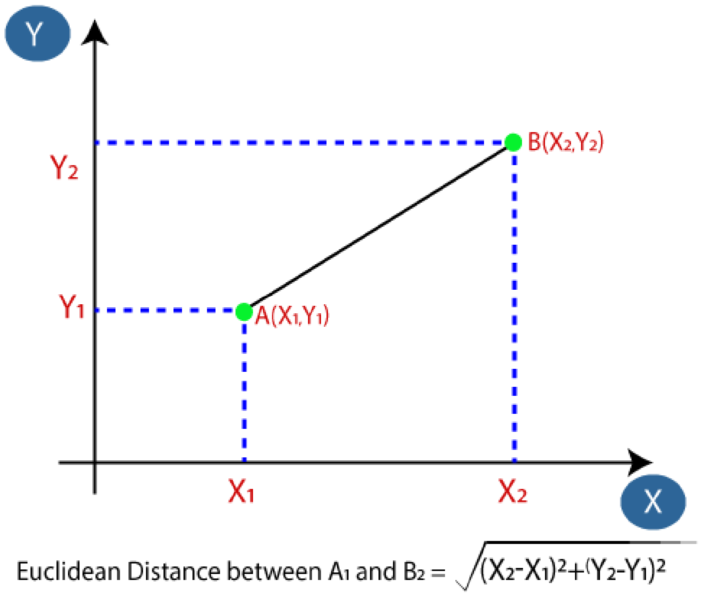

Y
X
Y_2
Y_1
X_1
X_2
$B(X_2,Y_2)$
$A(X_1,Y_1)$
Euclidean Distance between A_1 and B_2 = $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

Fig 2.1: EUCLIDEAN DISTANCE

- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:

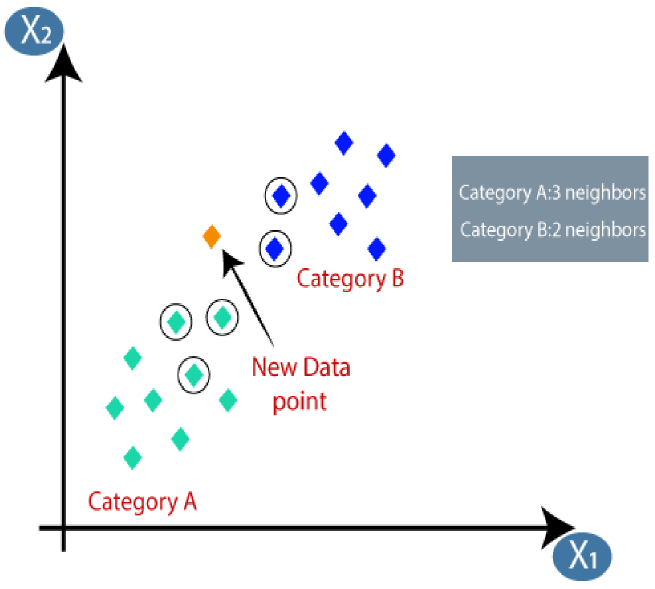


Fig 2.2: NEAREST NEIGHBORS

- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

2. ONE VS REST CLASSIFIER

One-vs-the-rest (O v R) multiclass strategy. Also known as one-vs-all, this strategy consists in fitting one classifier per class. For each classifier, the class is fitted against all the other classes. In addition to its computational efficiency

(only n classes classifiers are needed), one advantage of this approach is its interpretability. Since each class is represented by one and one classifier only, it is possible to gain knowledge about the class by inspecting its corresponding classifier. This is the most used strategy for multiclass classification and is a fair default choice. One Vs Rest Classifier can also be used for multilabel classification. To use this feature, provide an indicator matrix for the target y when calling. fit.

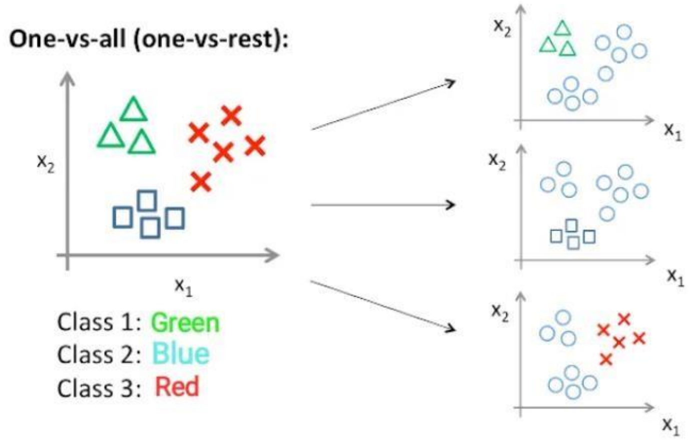


Fig 2.3: ONE VS REST CLASSIFIER

LITERATURE SURVEY

CHAPTER-2
LITERATURE SURVEY

A. Machine Learning approach for automation of Resume Recommendation System

Choosing the best candidates from the pool here to perform these types of tasks different NLP techniques such as bigram trigram and n gram and text classification are used, this model used Machine Learning to perform the classification using the algorithm.

B. Skill Finder: Automated Job-Resume Matching System

API for web services [9]. This information is then utilized to score the students' resumes based on the skills required for the job, using Named Entity Recognition (NER) software such as Apache Open NLP [10] and Stanford Name Entity Recognizer [11].

C. Web Application for Screening Resume

The goal was to create a web application for resume screening using 220 resumes, 200 of which were utilized for training and 20 for testing, and the web application was separated into three sections.

- Job Applicant side
- Server-Side
- Recruiter Side

The applicant will supply his or her résumé on the applicant side, which will be processed on the server side and then trained using the NLP Pipeline, which uses Spa Cy, an NLP framework [6]. On the recruiter's side, the resume rank list will be displayed, which was determined using a score calculator, so that the recruiter may choose the best candidate for the job.

D. Differential Hiring using Combination of NER and Word Embedding:

The NER model is used to extract useful entities from documents, which is enhanced by the word2vec model by making the system more generic and the similarity is calculated using the cosine similarity algorithm [7].

E. Al-Otaibi et al., [12] provided a detailed survey of job recommendation service. They discussed the steps involved in the recruiting process used by any organization. How the e-recruitment portal is helping to the organization, what factor of the candidate may lead to getting selected and many other relevant recruitment processes are explained.

F. Design and Development of Machine Learning based Resume Ranking System

The system proposes a technique in which the candidate submits his or her resume following an interview here the face-based technique was used. After the resume is submitted, NLP techniques are used to extract the necessary abilities from the resume, then TF- IDF vectorization is used to transform the words into vectors so the machine can interpret them. The KNN algorithm [5] is used to identify the resume that most closely fits the JD provided by the recruiter. The system has a parsing accuracy of 85 percent on average [4].

Here are some specific examples of NLP-based resume screening systems that have been proposed in the literature:

- Deep Learning-Based Resume Screening Model for Efficient Candidate Selection (Ali et al., 2023)
- Resume Screening with Transformer-Based Models (Kumar et al., 2023)
- Resume Screening with Multi-Task Learning (Liu et al., 2022)
- Resume Screening with BERT (Gupta et al., 2022)
- Resume Screening with Robert a (Khan et al., 2022)
- Resume Screening with XL Net (Singh et al., 2022)
- Resume Screening with ALBERT (Khan et al., 2023)

SYSTEM ANALYSIS AND DESIGN

CHAPTER-3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In the existing systems like Application Tracking software (Ats) it gives the score based on the keywords not the entire resume. These websites typically based on the keyword matching, semantic analysis.
One existing system for Python-powered resume screening and analysis is the "Resume Screening" library. This library provides a simple and efficient way to screen resumes using ML algorithms. It includes pre-processing steps such as tokenization, stop words removal, and stemming, as well as ML algorithms such as Naive Bayes, Logistic Regression, and Support Vector Machines.

Another existing system is the "Resume Analyzer" library, which uses NLP techniques to extract relevant information from resumes. It includes features such as keyword extraction, skill detection, and education level detection. The library also includes a ranking algorithm that ranks resumes based on their relevance to a job posting.

In addition, there are several commercial products that use Python-powered resume screening and analysis. For example, the "Talent Lyft" platform uses ML algorithms to screen resumes and rank candidates based on their relevance to a job posting.

DISADVANTAGES OF EXISTING SYSTEM:

- HUMAN JUDGMENT: Recruiters bring intuition and nuanced understanding to candidate assessment.
- FLEXIBILITY: Recruiters can adapt quickly to changing requirements or circumstances.
- Complex Decision-making: Recruiters consider multifaceted factors beyond resume content.
- It has a less accuracy
- It requires keywords.

3.2 PROPOSED SYSTEM

In the proposed system the resumes are short listed based on the company's requirements of skills but not on specific keywords and ranking. The accuracy is more and it is efficient and can handle huge amount of data. The model's concept will be trained with existing data gathered from the Kaggle open platform. The first model, either K-Nearest Neighbor and one-vs-rest classifier will help us predict what kind of job role our resume is best suited for, while the second model, cosine similarity, will check the user's input of what job role they want, and the recommendation system will provide it based on that. The following is the control flow: At the front end, the candidate uploads their resume; the resume is then passed to the resume parser, which is a pipeline of NLP algorithms that extracts important information. information from the resume; and finally, adding more value to the overall extracted data from vectors and providing it to the Machine learning Model for tagging.

ADVANTAGES OF PROPOSED SYSTEM:

- It is faster and more efficient than comparing to existing system
- It has a more accuracy
- It is scalable to handle large amount of data.

3.3 FEASIBILITY STUDY

In the fast-paced world of recruitment, the ability to efficiently screen resumes and identify top talent is crucial. Traditional methods often fall short in handling the sheer volume of applicants, leading to time-consuming processes and potential oversights. However, with advancements in technology, particularly in the realm of artificial intelligence and natural language processing, there lies an opportunity to revolutionize this aspect of recruitment. This essay presents a feasibility study on the development and implementation of a Python-powered resume screening and analysis system, aimed at streamlining the recruitment process and enhancing candidate selection.

Three key considerations involved in the feasibility analysis are:

1. ECONOMICAL FEASIBILITY
2. TECHNICAL FEASIBILITY
3. OPERATIONAL FEASIBILITY

3.3.1 ECONOMICAL FEASIBILITY

A thorough economic analysis would assess the costs associated with development, deployment, and maintenance of the system, weighed against potential benefits such as time savings and improved candidate selection accuracy. Calculating the return on investment and payback period would provide insight into the economic viability of the project, guiding decision-making.

3.3.2 TECHNICAL FEASIBILITY

The feasibility of this project hinges on leveraging Python, along with machine learning and natural language processing libraries, to develop an automated system capable of parsing and analyzing resumes. With the availability of open-source tools and libraries such as TensorFlow, scikit-learn, NLTK, and spa Cy, the technical foundation for such a system is well-established. Challenges such as scalability, integration with existing systems, and computational resources required will need to be carefully addressed through robust architecture and infrastructure planning.

3.3.3 OPERATIONAL FEASIBILITY

Operationally, the system would need to seamlessly integrate into existing recruitment workflows, with clear delineation of stakeholder roles and responsibilities. Training and onboarding of recruitment staff on the use of the system would be essential to ensure its effective adoption. Additionally, potential impacts on existing processes must be considered, with strategies in place to mitigate any disruptions.

3.3.4 LEGAL AND ETHICAL FEASIBILITY

From a legal and ethical standpoint, the system must adhere to data privacy regulations such as GDPR and CCPA. Ensuring fairness and mitigating biases in the algorithm are paramount to maintaining ethical integrity throughout the recruitment process. Transparency in the system's decision-making process is essential to instill trust among both candidates and recruiters.

3.4 SYSTEM REQUIREMENTS

3.4.1 SOFTWARE REQUIREMENTS

- Operating System: Windows 10
- Programming Language Used: Python

3.5 SYSTEM DESIGN

3.5.1 SYSTEM ARCHITECTURE

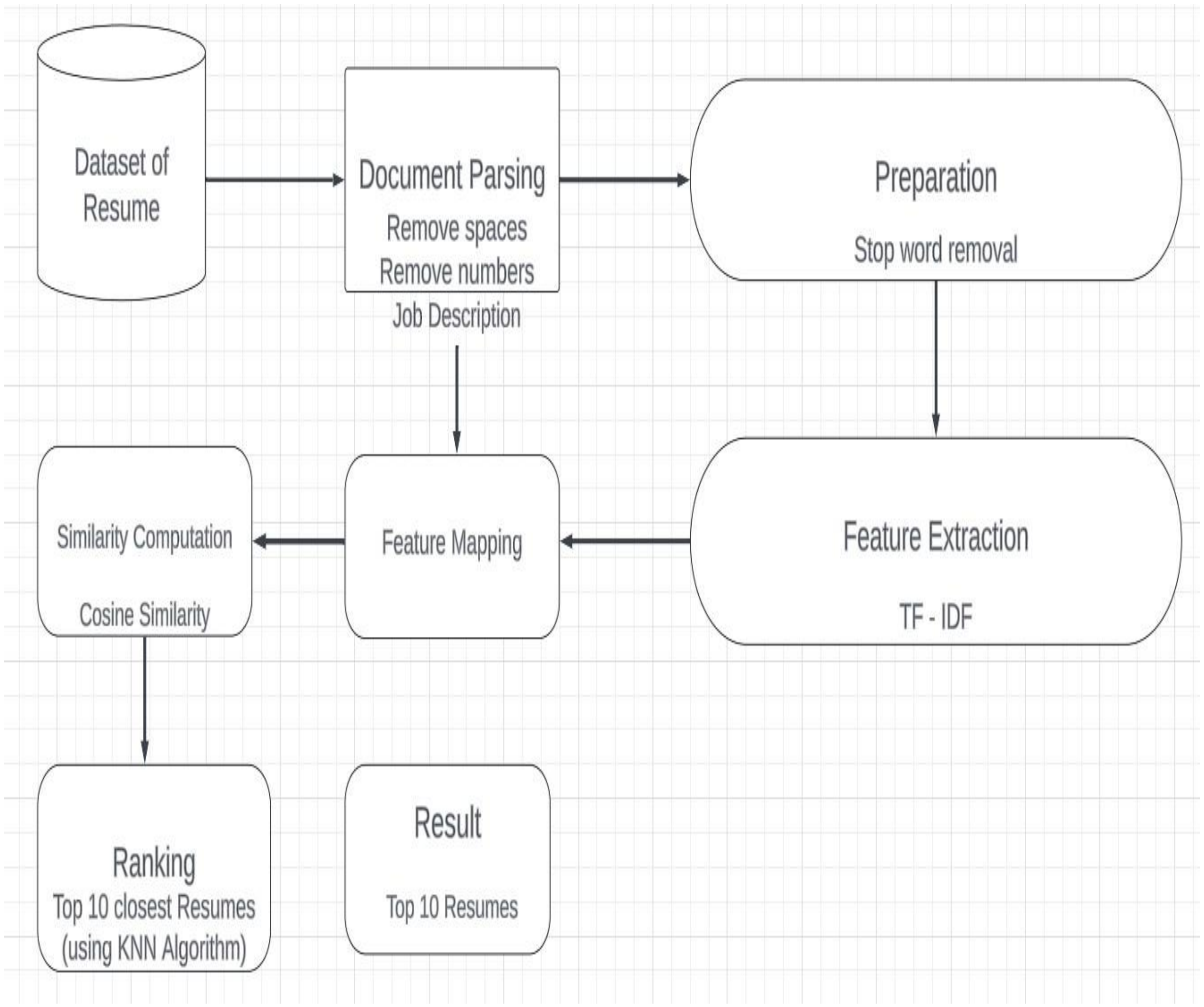


Fig 3.1: System Architecture

3.5.2 DATA FLOW DIAGRAMS

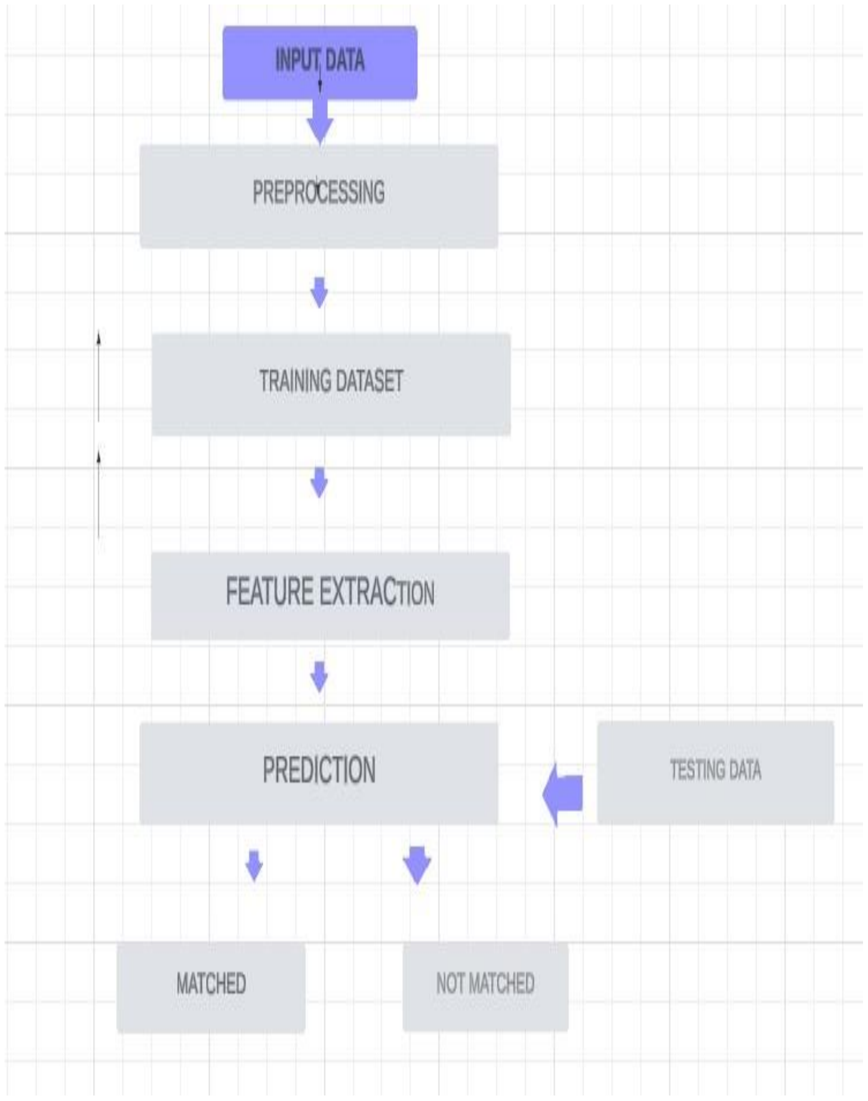


Fig 3.2: DATA FLOW DIAGRAMS

3.6 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

3.6.1 CLASS DIAGRAM

In this diagram, Resume screening is the main class that handles the screening and analysis of resumes. It uses NPL models and Machine Learning Model to process and analyze the resumes. The NPL Model class

is responsible for natural language processing tasks such as tokenization, parsing, and vectorization. The Machine Learning Model class is responsible for training on resume data and predicting a score for each candidate. The resume, Job Description, and Candidate classes represent the data structures used in the system.

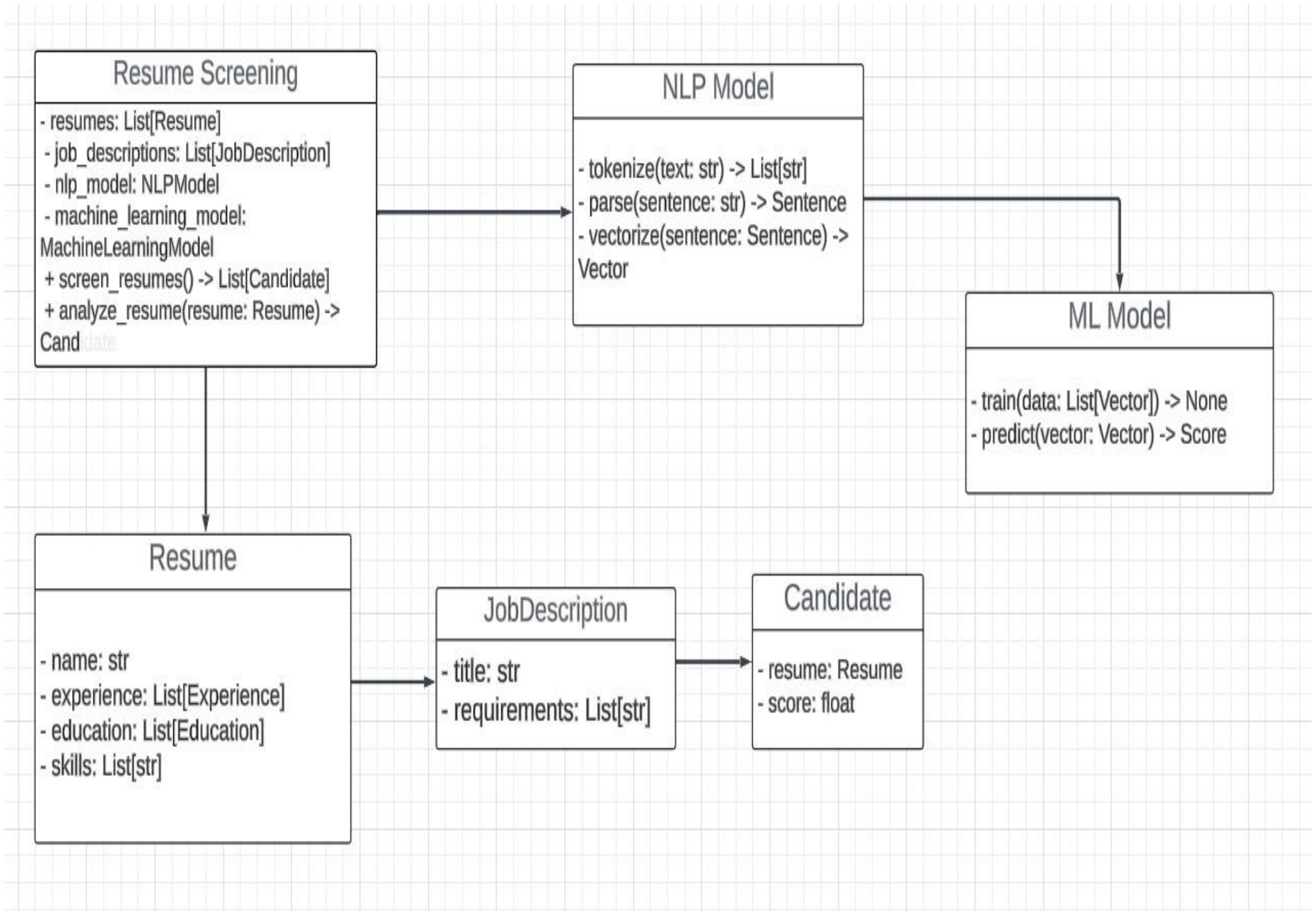

Resume Screening
- resumes: List[Resume]
- job_descriptions: List[JobDescription]
- nlp_model: NLPModel
- machine_learning_model: MachineLearningModel
+ screen_resumes() -> List[Candidate]
+ analyze_resume(resume: Resume) -> Cand
NLP Model
- tokenize(text: str) -> List[str]
- parse(sentence: str) -> Sentence
- vectorize(sentence: Sentence) -> Vector
ML Model
- train(data: List[Vector]) -> None
- predict(vector: Vector) -> Score
Resume
- name: str
- experience: List[Experience]
- education: List[Education]
- skills: List[str]
JobDescription
- title: str
- requirements: List[str]
Candidate
- resume: Resume
- score: float

Fig 3.3: CLASS DIAGRAM

3.6.2 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event construct of a Message Sequence Chart. Sequence diagrams are sometimes called event The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the

communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching

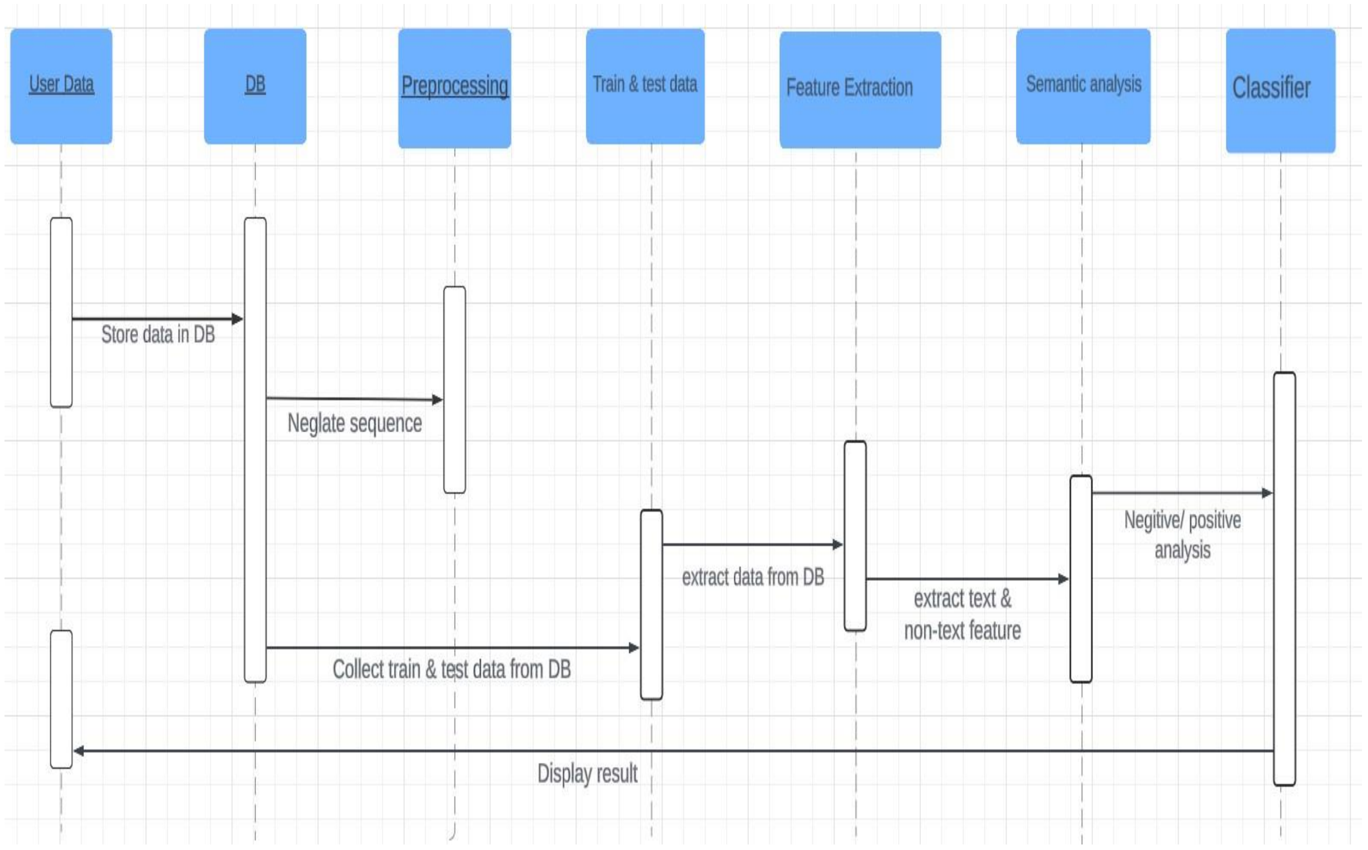


Fig 3.4: SEQUENCE DIAGRAM

3.6.3 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration, and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control. Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

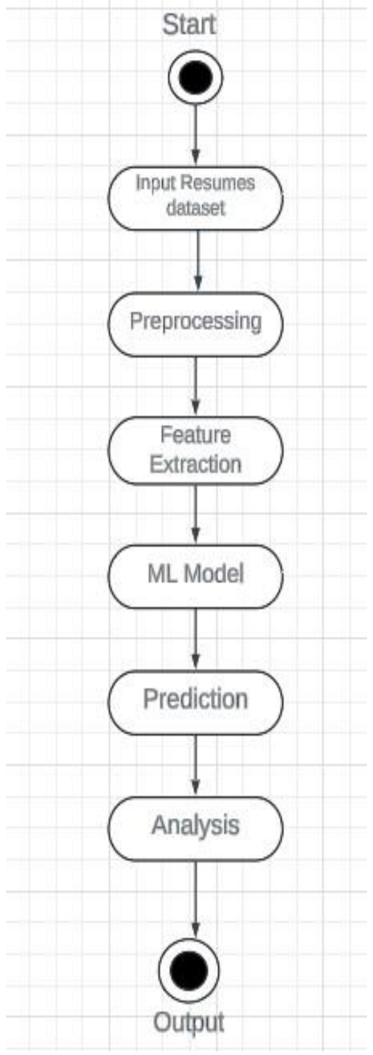


Fig 3.5: ACTIVITY DIAGRAM

3.7 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the directly

into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps, and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy.

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow.

3.8 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and the hard copy output. It is the most important and direct source information to

the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, status, or projections of the Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

IMPLEMENTATION

CHAPTER-4
IMPLEMENTATION

4.1 MODULES

- Dataset
- Data Pre-processing
- Data Ingestion
- Data Visualization
- Model Selection
- Analyze and prediction
- Accuracy on test set

Dataset Description:

The dataset consists of 962 rows and 3 columns, each representing different aspects of job descriptions and resumes collected from various individuals.

1.Rows:

The dataset contains 962 rows, each representing a unique entry comprising job descriptions and accompanying resumes.

2.Columns:

• **Column 1:** Category: This column contains descriptions of different job positions, each providing details about the roles, responsibilities, and requirements associated with various employment opportunities.

• **Column 2:** Resume: This column comprises resumes submitted by individuals applying for the respective job positions outlined in Column 1. Each resume provides information about the candidate's qualifications, skills, experiences, and educational background.

- **Column 3:** Cleaned Resumes: This column contains preprocessed and cleaned versions of the resumes provided in Column 2. The "cleaned _ resume" column may involve text preprocessing steps such as removing irrelevant

information, standardizing text formats, and handling formatting inconsistencies to facilitate further analysis.

Data Preprocessing:

This module involves cleaning and preparing the collected data for further analysis. It may include tasks like removing irrelevant information, standardizing data formats, handling missing values, and converting data into a suitable format for analysis. The preprocessing step involves cleaning and normalizing the text data. We used the following techniques to preprocess the data.

Text Cleaning:

- **Remove Special Characters:** Remove non-alphanumeric characters, punctuation marks, and symbols that do not contribute to the analysis.
- **Lowercasing:** Convert all text to lowercase to ensure consistency in text analysis.
- **Remove stop words:** Remove common stop words (e.g., "a," "an," "the") that do not carry significant meaning for resume screening.
- **Spell Checking:** Correct spelling errors to improve the accuracy of keyword extraction and analysis.

Tokenization:

- **Split Text into Tokens:** Tokenize the resume text into individual words or phrases to facilitate further analysis.
- **Word Tokenization:** Split text into individual words.
- **Sentence Tokenization:** Split text into individual sentences.

Normalization:

- **Stemming:** Reduce words to their base or root form to consolidate variations of the same word (e.g., "running" and "runs" both become "run").
- **Lemmatization:** Like stemming, lemmatization reduces words to their base form, but it considers the context of the word to ensure meaningful transformations (e.g., "ran" becomes "run").

Entity Recognition:

- **Named Entity Recognition (NER)**: Identify and categorize named entities such as names of people, organizations, locations, etc. This information can be useful for extracting relevant information from resumes, such as candidate names, companies, or educational institutions.

Keyword Extraction:

- **Identify Relevant Keywords**: Extract keywords or key phrases relevant to the job requirements, skills, experiences, and qualifications mentioned in the job descriptions.
- **TF-IDF (Term Frequency-Inverse Document Frequency):** Calculate the importance of words in resumes relative to their occurrence in the entire corpus of resumes. This helps identify keywords that are more specific and relevant to individual resumes.

Feature Engineering:

- **Create Additional Features:** Generate additional features based on specific criteria relevant to the screening process, such as years of experience, education level, relevant skills, etc.
- **Skills Matching:** Assess the match between candidate skills mentioned in resumes and the required skills specified in job descriptions.

Data Transformation:

- **Vectorization:** Convert textual data into numerical representations (e.g., Bag-of-Words, TF-IDF vectors) suitable for machine learning algorithms.
- **Dimensionality Reduction:** Reduce the dimensionality of the data while preserving important information to improve computational efficiency and model performance.

Data Integration:

- **Combine Preprocessed Data**: Integrate the preprocessed resume data with other relevant datasets or features for comprehensive analysis.

Libraries commonly used for data preprocessing:

- **Pandas:** Pandas provides powerful tools for data manipulation and analysis. It can be used for tasks like data cleaning, filtering, and transformation.
- **NumPy:** NumPy offers support for numerical operations and array processing. It can be used for tasks like handling missing values and numerical conversions.
- **Regular Expressions (Reg Ex):** Reg Ex can be used for pattern matching and text processing tasks, such as extracting specific information from resumes or standardizing text formats.
- **NLTK (Natural Language Toolkit):** NLTK provides tools for natural language processing tasks, such as tokenization, stemming, and part-of-speech tagging. It can be useful for processing textual data in resumes.

Data Ingestion:

- This module involves loading the preprocessed data into the analysis environment or data storage system. It ensures that the data is accessible

  and ready for manipulation and analysis. Data ingestion is the process of collecting, importing, and transferring data from various sources into a storage or processing system, where it can be further analyzed, processed, or utilized.
- Data ingestion transforms resume data into a structured format suitable for analysis. This may involve converting unstructured text data into structured formats like JSON, XML, or CSV, making it easier to process and analyze using machine learning algorithms.
- Data ingestion supports keyword matching and filtering of resumes based on specific job requirements and criteria. This helps in prioritizing resumes that closely match the desired qualifications and skills.
- Overall, data ingestion is critical for powering predictive analytics and machine learning models in resume screening and analysis. By efficiently processing and analyzing resume data, organizations can streamline their recruitment processes, identify top talent more effectively, and make data-driven hiring decisions.

Libraries or tools commonly used for data ingestion:

- **Pandas:** Pandas can also be used for data ingestion tasks, such as reading data from various file formats (e.g., CSV, Excel, JSON) or connecting to databases.

Data Visualization:

- Data visualization is crucial for presenting insights from the resume data in a clear and understandable manner. Matplotlib, Seaborn, and

Plotly are popular libraries for creating various types of visualizations such as bar charts, histograms, scatter plots, or interactive dashboards.

- Data visualization is the graphical representation of data to help people understand and interpret patterns, trends, and insights within the data more easily and intuitively. It involves creating visual representations such as charts, graphs, maps, and dashboards to communicate complex information visually. Visualizations can help recruiters or hiring

  managers identify trends, patterns, and correlations in candidate skills, experiences, or qualifications.
- The primary purpose of data visualization is to present data in a visually appealing and informative manner, making it easier for users to understand complex datasets, identify patterns, and gain insights. Visualizing the distribution of skills mentioned in resumes can help recruiters identify the most mentioned skills among candidates. This can help in understanding the skill sets that are in high demand or are most relevant to the job role.
- Visualizing the frequency of keywords or phrases mentioned in resumes can help recruiters identify common themes or trends among candidates. This can be particularly useful for identifying key qualifications, experiences, or attributes that are highly sought after for the job role. Visualizing candidate data across different job roles or departments can help recruiters compare candidate profiles. This can be useful for identifying commonalities or differences in candidate qualifications and experiences across different positions within the organization.
- Overall, data visualization can enhance the resume screening prediction and analysis process by providing recruiters with visual insights into candidate data, enabling them to make more informed decisions and identify top candidates more effectively.

Libraries are commonly used in Python for data visualization:

- **Matplotlib:** Matplotlib is a widely-used plotting library that provides a MATLAB-like interface for creating static, interactive, and animated visualizations in Python. It supports various types of plots, including line plots, scatter plots, bar plots, histograms, and more.

- **Seaborn:** Seaborn is built on top of Matplotlib and provides a high-level interface for creating attractive and informative statistical graphics. It simplifies the process of creating complex visualizations such as heatmaps, violin plots, and pair plots by providing easy-to-use functions with sensible defaults.

- **Plotly:** Plotly is a versatile library that supports creating interactive plots and dashboards in Python. It allows you to create interactive visualizations that can be embedded in web applications or notebooks. Plotly supports a wide range of plot types, including scatter plots, line plots, bar plots, 3D plots, and choropleth maps.

Data Manipulation:

- Data manipulation involves performing operations on the resume data to extract relevant information, derive insights, or prepare it for visualization. Libraries like Pandas are commonly used for data manipulation tasks such as filtering, sorting, grouping, joining, or transforming data. Machine learning techniques may also be applied for feature extraction, classification, or clustering of resumes based on specific criteria.

- Feature extraction involves selecting or creating relevant features from the resume data that can be used for predictive modeling. Features may include skills, experience level, education background, job titles, industry keywords, and any other information that may be predictive of candidate suitability. Feature extraction techniques can vary depending on the specific requirements of the prediction model.

- Before training predictive models, the dataset is typically split into training, validation, and test sets. Data manipulation is used to ensure that the splits are representative and maintain the distribution of key features across the sets. Overall, effective data manipulation is essential for preparing resume data for analysis and model training in resume screening prediction and analysis. It ensures that the data is clean, structured, and enriched with relevant features, leading to more accurate and reliable predictive models.

Libraries commonly used for data manipulation tasks:

- **Pandas:** Once again, Pandas is a powerful tool for data manipulation. It offers functions for tasks like data filtering, grouping, merging, and reshaping.
- **NumPy:** NumPy provides functions for array manipulation and mathematical operations. It can be used for tasks like numerical computations and array transformations.

Model Selection:

While creating a machine learning model, we need two datasets, one for training and other for testing. But now we have only one. So, lets split this in two with a ratio of 80:20. We will also divide the data frame into feature column and label column.

Here we imported train _ test _ split function of sk learn. Then use it to split the dataset. Also, test _ size = 0.2, it makes the split with 80% as train dataset and 20% as test dataset.

The random _ state parameter seeds random number generator that helps to split the dataset.

The function returns four datasets. Labelled them as train _x, train _y, test _x, test _y. If we see shape of this datasets we can see the split of dataset.

We will use KNN, which fits multiple decision tree to the data. Finally, I train the model by passing train _x, train _y to the fit method.

Once the model is trained, we need to Test the model. For that we will pass test _x to the predict method.

The K-Nearest Neighbors (KNN) algorithm is a popular machine learning technique used for classification and regression tasks. It relies on the idea that similar data points tend to have similar labels or values.

During the training phase, the KNN algorithm stores the entire training dataset as a reference. When making predictions, it calculates the distance between the input data point and all the training examples, using a chosen distance metric such as Euclidean distance.

Next, the algorithm identifies the K nearest neighbors to the input data point based on their distances. In the case of classification, the algorithm assigns the most common class label among the K neighbors as the predicted label for the input data point. For regression, it calculates the average or weighted average of the target values of the K neighbors to predict the value for the input data point.

Now we will analyze the data using bar plot and pie chart We will plot the data categories vs number of categories and he pie chart number percent of each categorical data. Now the next step in the process is to train a model for the task of Resume Screening. Here I will use the one vs the rest classifier; K Neighbor Classifier. For this task, I will first split the data into training and test sets and we will train the model where will get the accuracy on training set and test set.

Saving the Trained Model:

Once you are confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or. pkl file using a library like pickle.

Make sure you have pickle installed in your environment.
Next, let us import the module and dump the model into. pkl file

Analyze and Prediction:

- **Data Preparation:**

Preprocess the resumes in your dataset using the techniques mentioned earlier, including text cleaning, tokenization, and feature engineering.

- **Model Training:**

Split your preprocessed dataset into training and testing sets. Train the selected model(s) on the training data using appropriate algorithms and techniques.
Tune hyperparameters and optimize the model's performance using techniques like cross-validation.

- **Model Evaluation:**

Evaluate the trained model(s) on the testing data using relevant performance metrics, such as accuracy, precision, recall, F1-score, and AUC.
Analyze the model's performance and identify any areas for improvement or refinement.

Accuracy on test set:

- We got an accuracy of 0.99% on test set.

4.2 SOFTWARE ENVIRONMENT

4.2.1 Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is like PERL and PHP.

- **Python is Interactive** − You can sit at a Python prompt and interact with the interpreter directly to write your programs.

- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language** − Python is a great language for the beginner level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

4.2.2History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.
Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and Unix shell and other scripting languages.
Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).
Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.
Python Features

Python's features include −

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let us understand how to set up our Python environment.

Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org.
Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to https://www.python.org/downloads/.
- Follow the link for the Windows installer python-XYZ. misfile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

First Python Program

Let us execute programs in different modes of programming.

Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt −

```
$ python

Python2.4.3(#1, Nov112010,13:34:43)

[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2

Type "help", "copyright", "credits" or "license" for more information.
```

```
>>>
```

Type the following text at the Python prompt and press the Enter −

```
>>> print "Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");** However, in Python version 2.4.3, this produces the following result − Hello, Python!

Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**. Type the following source code in a test.py file −

```
Print "Hello, Python!"
```

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows −

```
$ python test.py
```

This produces the following result –

```
Hello, Python!
```

Python Libraries:

- **Pandas:** Pandas is a powerful data manipulation and analysis library. It can be used for handling structured data, such as resumes, and performing various data processing tasks.

- **NumPy:** NumPy is a fundamental package for scientific computing in Python. It provides support for large multi-dimensional arrays and matrices, which can be useful for numerical operations and data manipulation.

- **NLTK (Natural Language Toolkit):** NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources, such as WordNet. NLTK can be used for text processing and analysis tasks, such as parsing resumes for relevant keywords and phrases.

- **Scikit-learn:** Scikit-learn is a machine learning library for Python. It provides simple and efficient tools for data mining and data analysis. Scikit-learn can be used for building machine learning models to classify resumes or extract relevant features from them.

- **Matplotlib and Seaborn:** These are plotting libraries for Python. They can be used for creating visualizations to analyze trends and patterns in the data, such as the distribution of skills or experience levels in resumes.

What is Python?

Python is a popular programming language. It was created in 1991 by Guido van Rossum.
It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics. Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax like the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way, or a functional way.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, PyCharm, NetBeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

- Python was designed to for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions, and classes. Other programming languages often use curly brackets for this purpose.

Python Installation

Many PCs and Macs will have python already installed.
To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

```
C:\Users\Your Name>python --version
```

To check if you have python installed on a Linux or Mac, then on Linux open the command line or on Mac open the Terminal and type:

```
python --version
```

If you find that you do not have python installed on your computer, then you can download it for free from the following website: https://www.python.org/

Python Quick start

Python is an interpreted programming language; this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

```
C:\Users\Your Name>python helloworld.py
```

Where "helloworld.py" is the name of your python file.
Let us write our first Python file, called helloworld.py, which can be done in any text editor.

```
helloworld.py print ("Hello, World!")
```

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run: C:\Users\Your Name>python helloworld.py The output should read:

```
Hello, World!
```

Congratulations, you have written and executed your first Python program.

The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.
Type the following on the Windows, Mac, or Linux command line:
C:\Users\Your Name>python
From there you can write any python, including our hello world example from earlier in the tutorial:

C:\Users\Your Name>python Python 3.6.4 (v3.6.4: d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32 Type "help", "copyright", "credits" or "license" for more information. >>> print ("Hello, World!") Which will write "Hello, World!" in the command line:
C:\Users\Your Name>python Python 3.6.4 (v3.6.4: d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32 Type "help", "copyright", "credits" or "license" for more information. >>> print ("Hello, World!") Hello, World!
Whenever you are done in the python command line, you can simply type the following to quit the python command line interface: exit ()
Execute Python Syntax
As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:
>>> print ("Hello, World!") Hello, World!
Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:
C:\Users\Your Name>python myfile.py

Python Indentations

Where in other programming languages the indentation in code is for readability only, in Python the indentation is very important. Python uses indentation to indicate a block of code.

Example
if 5 > 2: print ("Five is greater than two!")
Python will give you an error if you skip the indentation:
Example
if 5 > 2: print ("Five is greater than two!")

Comments

Python has commenting capability for the purpose of in-code documentation.

Comments start with a #, and Python will render the rest of the line as a comment:
Example
Comments in Python:
#This is a comment. print ("Hello, World!")
Docstrings
Python also has extended documentation capability, called docstrings.
Docstrings can be one line, or multiline.
Python uses triple quotes at the beginning and end of the docstring:
Example
Docstrings are also comments:
"""This is a multiline docstring."""print ("Hello, World!")

CODING

CHAPTER-5
CODING

```
#Loading Libraries import warnings warnings. filterwarnings ('ignore') import
numpy as np import pandas as pd import matplotlib. pyplot as plt import seaborn
as sns from matplotlib. gridspec import GridSpec import re from
sklearn.preprocessing import LabelEncoder from sklearn.model_selection
import train_test_split from sklearn.feature_extraction.text import
TfidfVectorizer from scipy.sparse import hstack from sklearn.multiclass import
OneVsRestClassifier from sklearn.neighbors import KNeighborsClassifier from
sklearn import metrics

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
from sklearn.naive_bayes import MultinomialNB
from sklearn.multiclass import OneVsRestClassifier
from sklearn import metrics
from sklearn.metrics import accuracy_score

from pandas.plotting import scatter_matrix
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics

resumeDataSet=pd.read_csv('UpdatedResumeDataSet.csv')
resumeDataSet['cleaned_resume']=' '
resumeDataSet.head()
print(resumeDataSet['Category'].unique())#unique values

print(resumeDataSet['Category'].count())
print(resumeDataSet['Category'].value_counts())

import seaborn as sn
plt.figure(figsize=(15,15))
plt.xticks(rotation=90)
sn.countplot(y='Category',data=resumeDataSet)

from matplotlib.gridspec import GridSpec # gridspec- used to create the
subplots by representing the geometric figure
targetCounts = resumeDataSet['Category'].value_counts() #value_counts: key
value pairs, how many times a value repeated in the column
targetLabels = resumeDataSet['Category'].unique() #generating unique values
from the dataset
#Make square figures and axes
plt.figure(1, figsize=(18,18))# creating the dimension, for matplot lib we should
have to create the size
the_grid = GridSpec(2, 2)# 2 rows and 2 columns this is a row column matrix

cmap = plt.get_cmap('coolwarm')#plt=pyplot, get_cmap- colormap, coolwarm
is the color in the get_cmap function
colors = [cmap(i) for i in np.linspace(0, 1, 3)]# linspace- start position is 0
ending is 1 and 3 is giving the 3 colors


plt.subplot(the_grid[0,1], aspect=1, title='CATEGORY DISTRIBUTION')#
aspect-1 is ration i.e 360 degrees in a counter clock wise, title-title of the pie
chart

source_pie = plt.pie(targetCounts, labels=targetLabels, autopct='%1.1f%%',
shadow=True, colors=colors)# autopct is autopercentage of the each and every
value of the column 1.1 first 1 is for my integer part and second 1 is my decimal
part f is floating point, shadow=True creATES shadow and colors is displaying
colors
plt.show()# displays the pie chart
```

```
import re# re= regular expressions a library
def cleanResume(resumeText):# cleanResume
    resumeText = re.sub('http\S+\s*', ' ', resumeText)  # remove URLs replacing with space sub-substitute
    resumeText = re.sub('RT|cc', ' ', resumeText)  # remove RT and cc RT-subject in the mail cc-copy of the text and reciever mail
    resumeText = re.sub('#\S+', '', resumeText) #replacing it with empty string # remove hashtags #-to indicate and tags the person in which the person is searching for the domain
    resumeText = re.sub('@\S+', '  ', resumeText)  # remove mentions *-multiple number of values may be present or may not be present it will identify it
    resumeText = re.sub('[%s]' % re.escape("""!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~"""), ' ', resumeText)  # remove punctuations escape-backspacing same symbol using inside the same symbol
    resumeText = re.sub(r'[^\x00-\x7f]',r' ', resumeText) #removing Aschee and non aschee values +-it should have atleast one character
    resumeText = re.sub('\s+', ' ', resumeText)  # remove extra whitespace %s-it should be start with s ^- mentioned characters should not be there
    return resumeText

resumeDataSet['cleaned_resume'] = resumeDataSet.Resume.apply(lambda x: cleanResume(x))


import nltk #natural language tool kit-nltk
from nltk.corpus import stopwords
import string
from wordcloud import WordCloud
nltk.download('stopwords')

nltk.download('punkt')

oneSetOfStopWords = set(stopwords.words('english')+['``',"''"])
totalWords =[]
```

```
Sentences = resumeDataSet['Resume'].values
cleanedSentences = ""
for i in range(0,160):
    cleanedText = cleanResume(Sentences[i])
    cleanedSentences += cleanedText
    requiredWords = nltk.word_tokenize(cleanedText)
    for word in requiredWords:
        if word not in oneSetOfStopWords and word not in string.punctuation:
            totalWords.append(word)

wordfreqdist = nltk.FreqDist(totalWords)
mostcommon = wordfreqdist.most_common(50)
print(mostcommon)

wc = WordCloud().generate(cleanedSentences)
plt.figure(figsize=(15,15))
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.show()

resumeDataSet['Category'].unique()

from sklearn.preprocessing import LabelEncoder


var_mod = ['Category']
le = LabelEncoder()
for i in var_mod:
    resumeDataSet[i] = le.fit_transform(resumeDataSet[i])

resumeDataSet.head()

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
requiredText = resumeDataSet['cleaned_resume'].values
requiredTarget = resumeDataSet['Category'].values

word_vectorizer = TfidfVectorizer(
    sublinear_tf=True,
    stop_words='english',
    max_features=1500)
word_vectorizer.fit(requiredText)
WordFeatures = word_vectorizer.transform(requiredText)

print ("Feature completed .....")

X_train,X_test,y_train,y_test                                                    =
train_test_split(WordFeatures,requiredTarget,random_state=0, test_size=0.2)
print(X_train.shape)
print(X_test.shape)

clf = OneVsRestClassifier(KNeighborsClassifier())
clf.fit(X_train, y_train)
prediction = clf.predict(X_test)
print('Accuracy    of    KNeighbors    Classifier    on    training    set:
{:.2f}'.format(clf.score(X_train, y_train)))



print('Accuracy    of    KNeighbors    Classifier    on    test    set:
{:.2f}'.format(clf.score(X_test, y_test)))

print("\n  Classification  report  for  classifier  %s:\n%s\n"  %  (clf,
metrics.classification_report(y_test, prediction)))
```

SYSTEM TESTING

CHAPTER-6
SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

6.1 SYSTEM TESTING

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields.

Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- **Valid Input:** identified classes of valid input must be accepted.
- **Invalid Input:** identified classes of invalid input must be rejected.
- **Functions:** identified functions must be exercised.
- **Output:** identified classes of application outputs must be exercised.
- **Systems/Procedures:** interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure, and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

6.1.1. Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

6.1.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.
The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.1.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.1.4 Functional testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

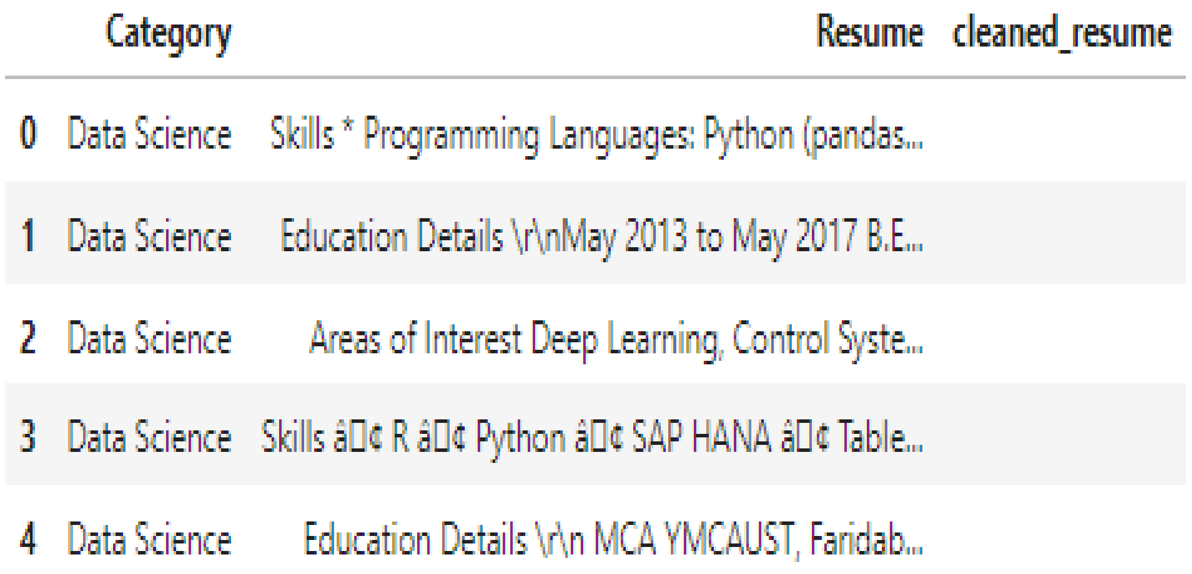
- **Valid Input**: identified classes of valid input must be accepted.
- **Invalid Input**: identified classes of invalid input must be rejected.
- **Functions**: identified functions must be exercised.
- **Output**: identified classes of application outputs must be exercised.
- **Systems/Procedures**: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

OUTPUT SCREENS

CHAPTER-7
OUTPUT SCREENS

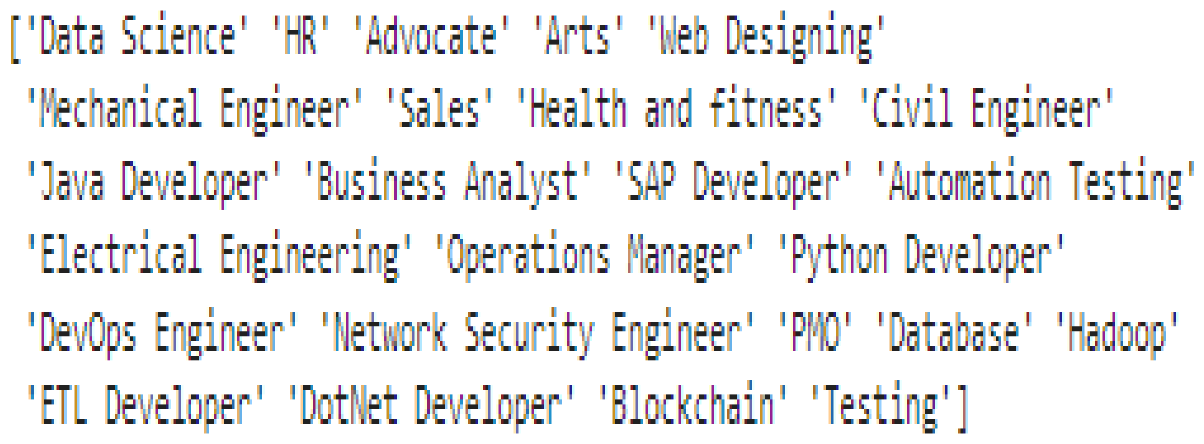
1. Data set that we are taking



	Category	Resume	cleaned_resume
0	Data Science	Skills * Programming Languages: Python (pandas...	
1	Data Science	Education Details \r\nMay 2013 to May 2017 B.E...	
2	Data Science	Areas of Interest Deep Learning, Control Syste...	
3	Data Science	Skills â¢ R â¢ Python â¢ SAP HANA â¢ Table...	
4	Data Science	Education Details \r\n MCA YMCAUST, Faridab...	

Fig 7.1: Resume Data Set

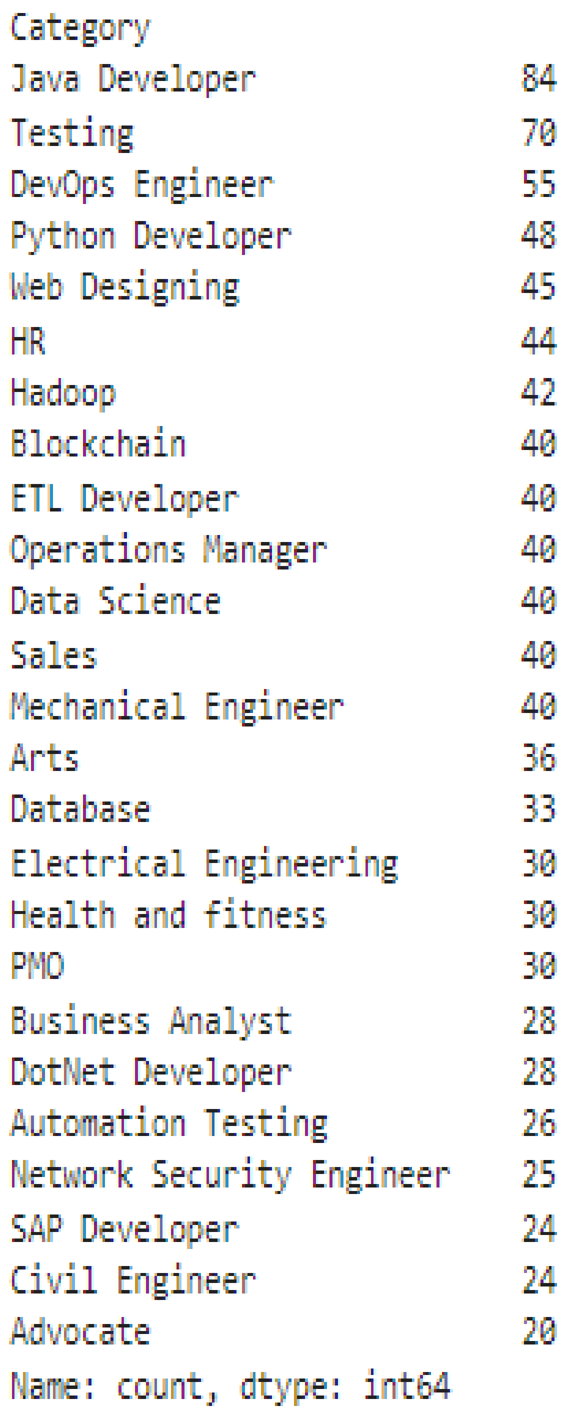
2. Unique categories of the data set



```
['Data Science' 'HR' 'Advocate' 'Arts' 'Web Designing'
 'Mechanical Engineer' 'Sales' 'Health and fitness' 'Civil Engineer'
 'Java Developer' 'Business Analyst' 'SAP Developer' 'Automation Testing'
 'Electrical Engineering' 'Operations Manager' 'Python Developer'
 'DevOps Engineer' 'Network Security Engineer' 'PMO' 'Database' 'Hadoop'
 'ETL Developer' 'DotNet Developer' 'Blockchain' 'Testing']
```

Fig 7.2: Unique categorical values

3. Value count of unique categories



```
Category
Java Developer               84
Testing                      70
DevOps Engineer              55
Python Developer             48
Web Designing                45
HR                           44
Hadoop                       42
Blockchain                   40
ETL Developer                40
Operations Manager           40
Data Science                 40
Sales                        40
Mechanical Engineer          40
Arts                         36
Database                     33
Electrical Engineering       30
Health and fitness           30
PMO                          30
Business Analyst             28
DotNet Developer             28
Automation Testing           26
Network Security Engineer    25
SAP Developer                24
Civil Engineer               24
Advocate                     20
Name: count, dtype: int64
```

Fig 7.3: value count of unique categories

4. Representing the value counts in bar graph

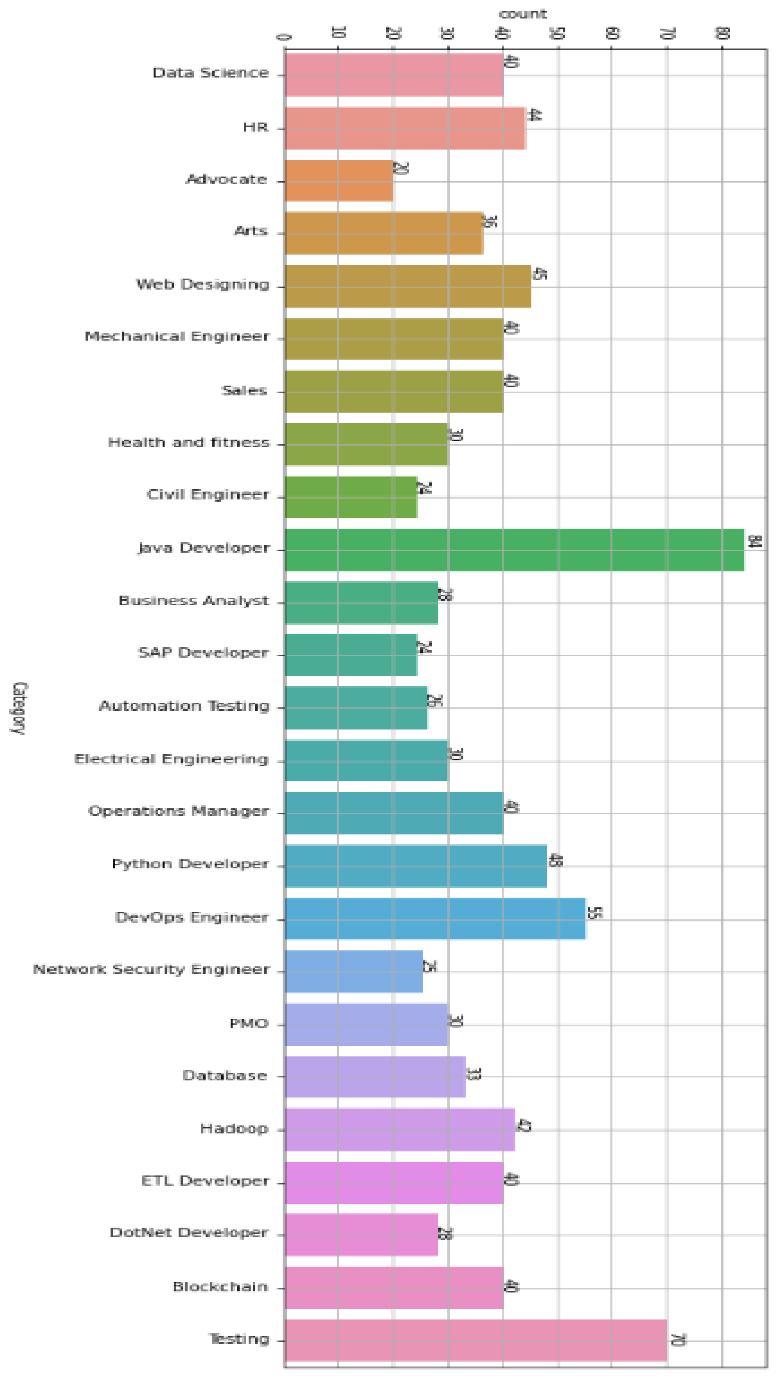
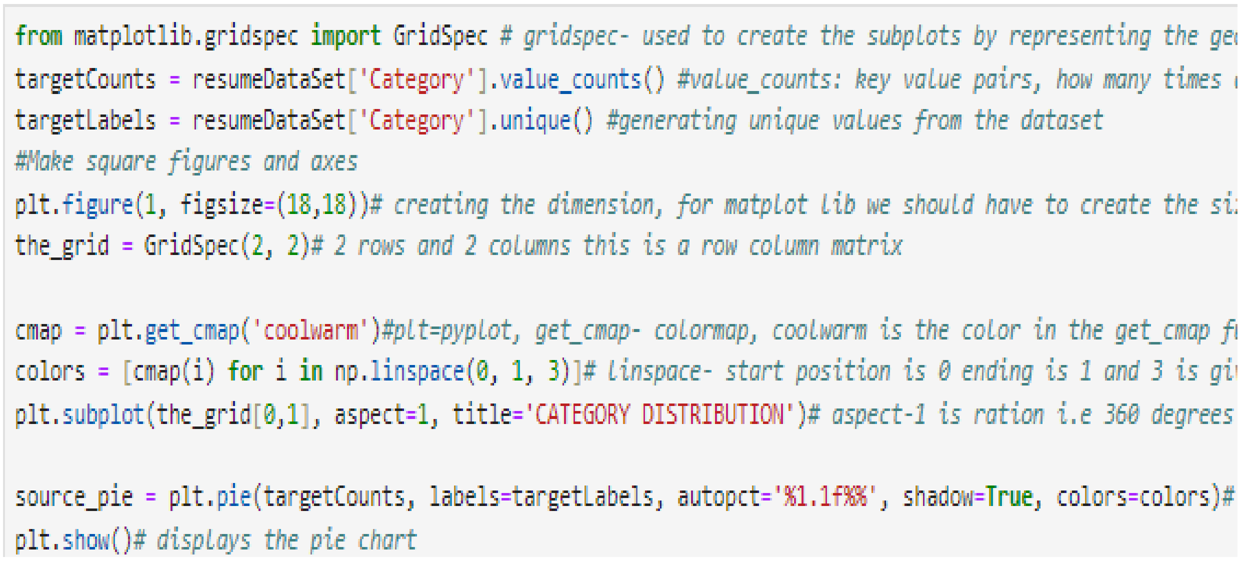


Fig 7.4: value counts in bar graph

5. It shows the target count and target labels through decimal form in this pie chart.



```
from matplotlib.gridspec import GridSpec # gridspec- used to create the subplots by representing the ge
targetCounts = resumeDataSet['Category'].value_counts() #value_counts: key value pairs, how many times
targetLabels = resumeDataSet['Category'].unique() #generating unique values from the dataset
#Make square figures and axes
plt.figure(1, figsize=(18,18))# creating the dimension, for matplot lib we should have to create the si
the_grid = GridSpec(2, 2)# 2 rows and 2 columns this is a row column matrix

cmap = plt.get_cmap('coolwarm')#plt=pyplot, get_cmap- colormap, coolwarm is the color in the get_cmap f
colors = [cmap(i) for i in np.linspace(0, 1, 3)]# linspace- start position is 0 ending is 1 and 3 is gi
plt.subplot(the_grid[0,1], aspect=1, title='CATEGORY DISTRIBUTION')# aspect-1 is ration i.e 360 degrees

source_pie = plt.pie(targetCounts, labels=targetLabels, autopct='%1.1f%%', shadow=True, colors=colors)#
plt.show()# displays the pie chart
```

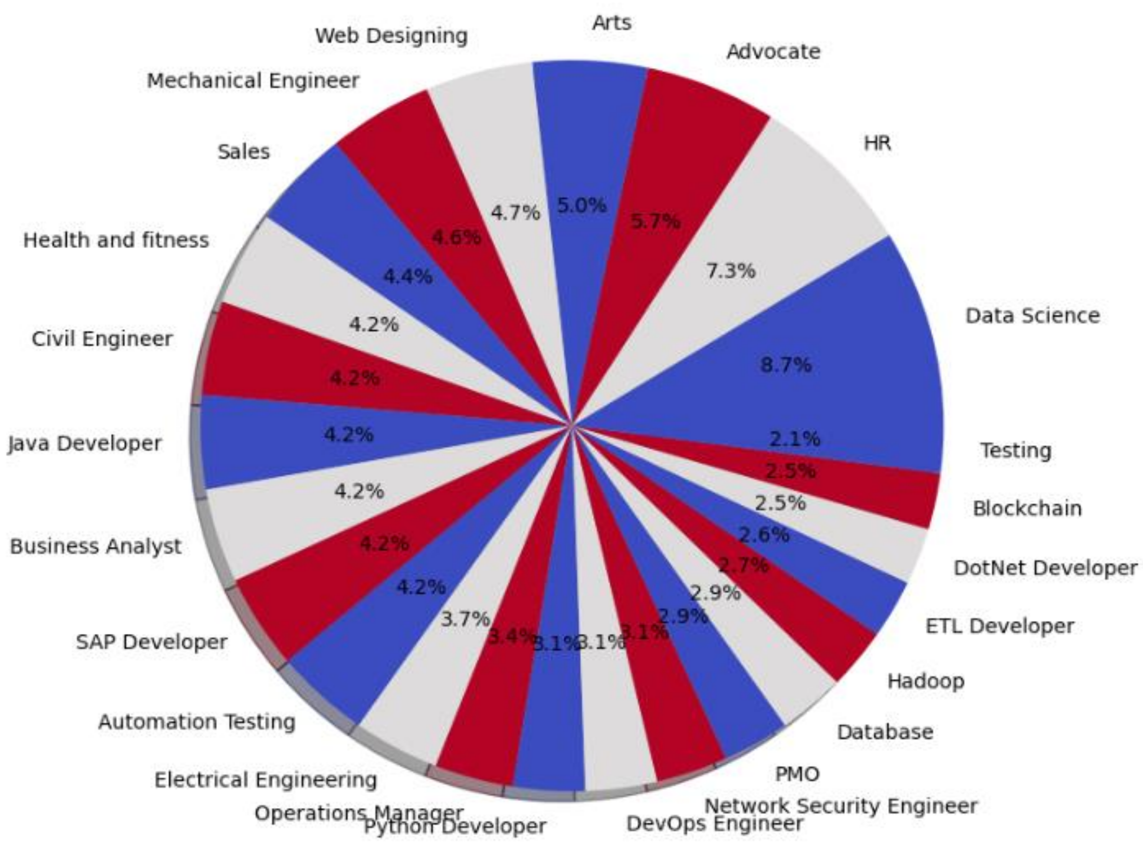
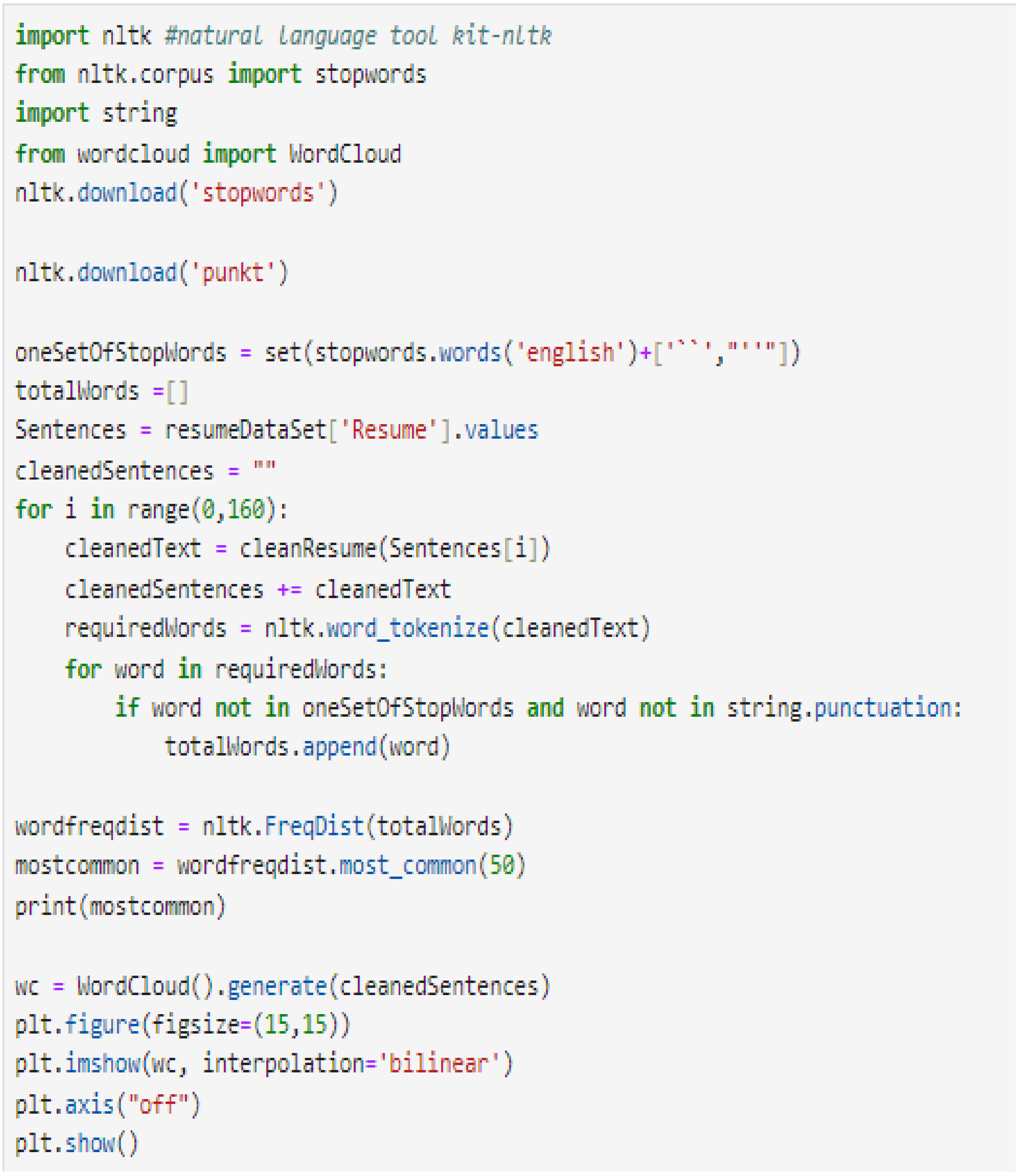


Fig 7.5: target count target labels in pie chart

6. These world cloud shows the highest value count in bigger size and lowest value count in smaller size



```python
import nltk #natural language tool kit-nltk
from nltk.corpus import stopwords
import string
from wordcloud import WordCloud
nltk.download('stopwords')

nltk.download('punkt')

oneSetOfStopWords = set(stopwords.words('english')+['``',"''"])
totalWords =[]
Sentences = resumeDataSet['Resume'].values
cleanedSentences = ""
for i in range(0,160):
    cleanedText = cleanResume(Sentences[i])
    cleanedSentences += cleanedText
    requiredWords = nltk.word_tokenize(cleanedText)
    for word in requiredWords:
        if word not in oneSetOfStopWords and word not in string.punctuation:
            totalWords.append(word)

wordfreqdist = nltk.FreqDist(totalWords)
mostcommon = wordfreqdist.most_common(50)
print(mostcommon)

wc = WordCloud().generate(cleanedSentences)
plt.figure(figsize=(15,15))
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.show()
```

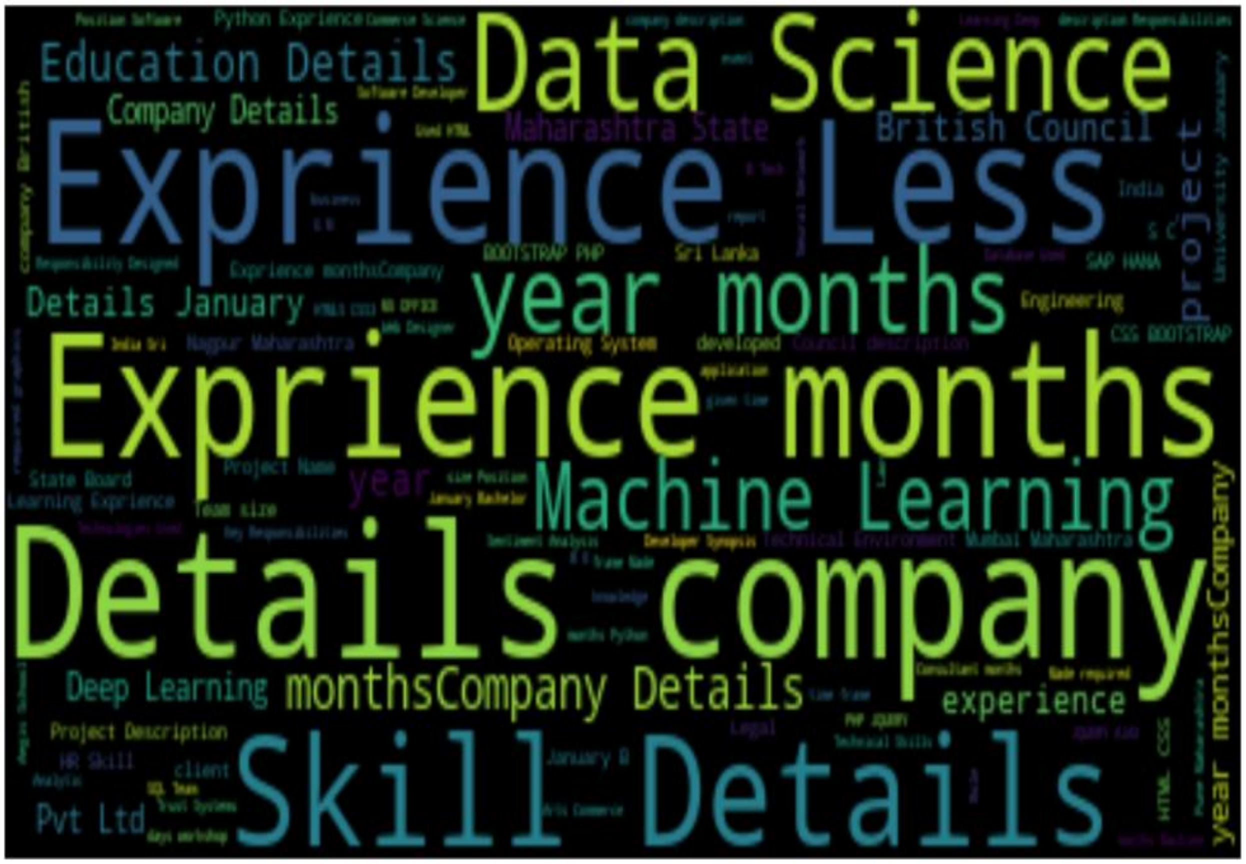
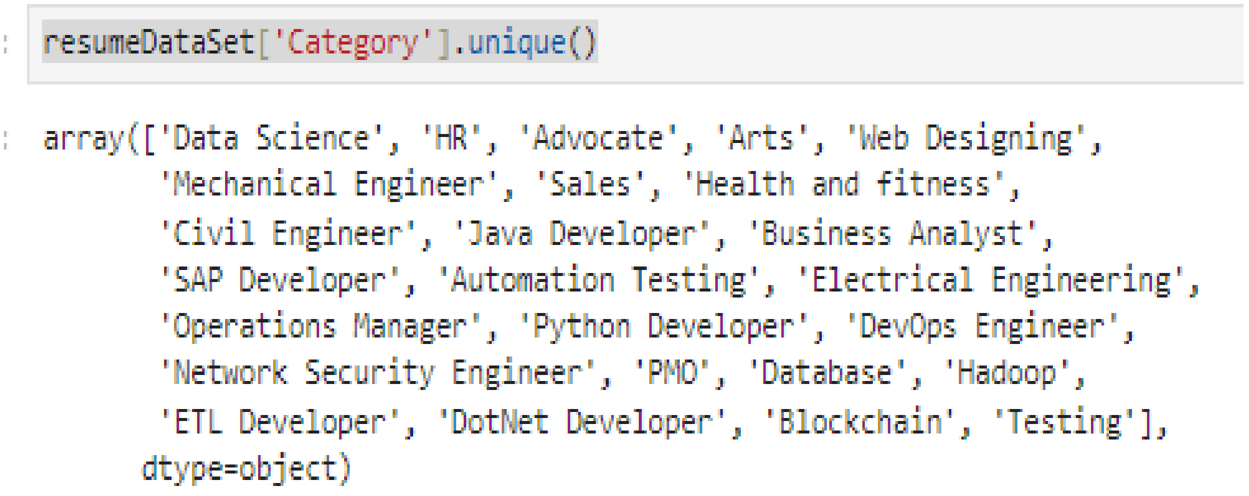


Fig 7.6: Highest Value Counts

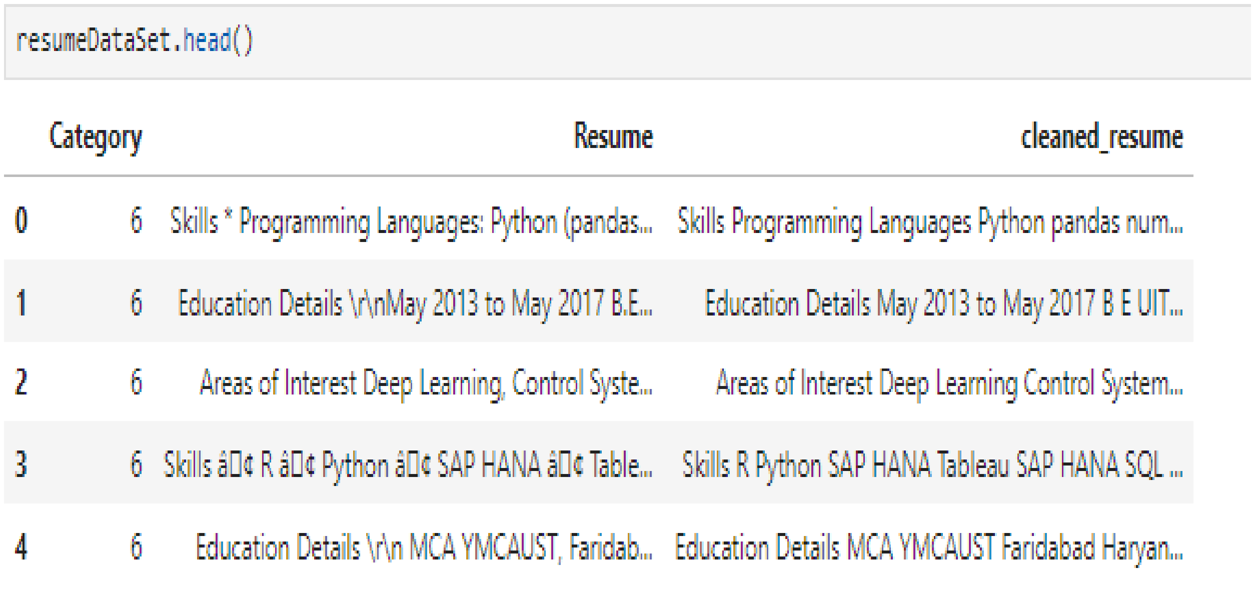
7. It shows the unique categories from the dataset.


```
resumeDataSet['Category'].unique()
```

```
array(['Data Science', 'HR', 'Advocate', 'Arts', 'Web Designing',
       'Mechanical Engineer', 'Sales', 'Health and fitness',
       'Civil Engineer', 'Java Developer', 'Business Analyst',
       'SAP Developer', 'Automation Testing', 'Electrical Engineering',
       'Operations Manager', 'Python Developer', 'DevOps Engineer',
       'Network Security Engineer', 'PMO', 'Database', 'Hadoop',
       'ETL Developer', 'DotNet Developer', 'Blockchain', 'Testing'],
      dtype=object)
```

Fig 7.7: Unique values

8. Head function shows the first value counts.

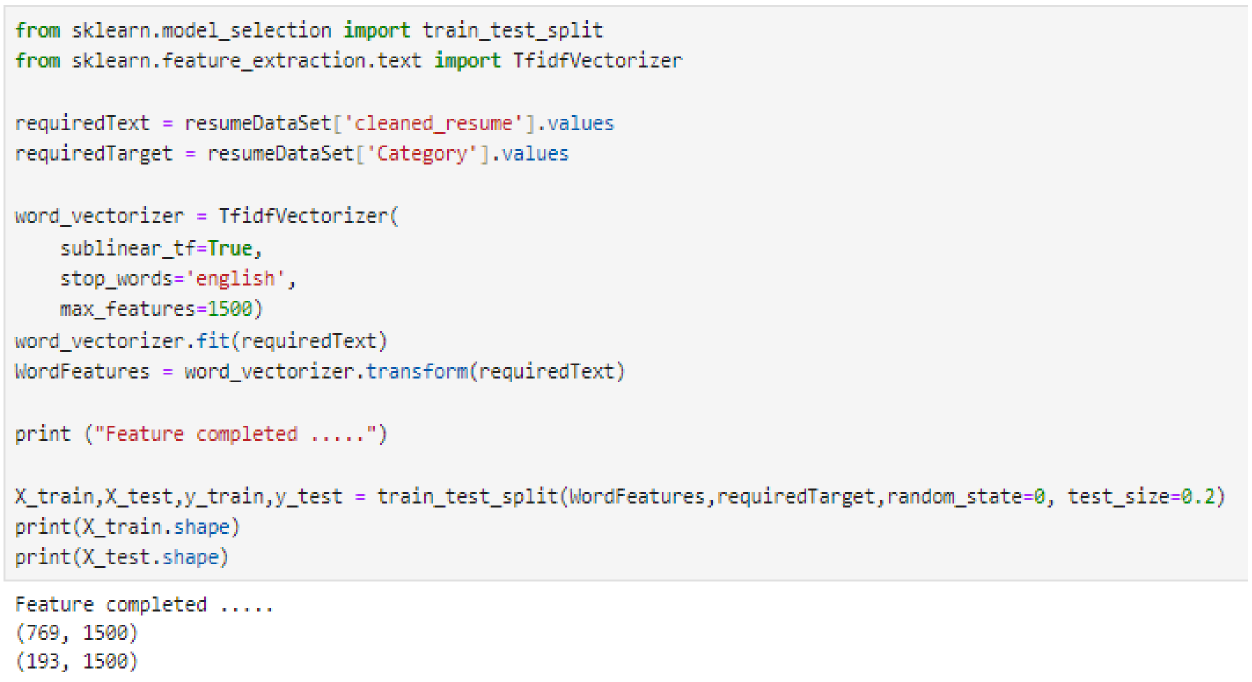


```
resumeDataSet.head()
```

	Category	Resume	cleaned_resume
0	6	Skills * Programming Languages: Python (pandas...	Skills Programming Languages Python pandas num...
1	6	Education Details \r\nMay 2013 to May 2017 B.E...	Education Details May 2013 to May 2017 B E UIT...
2	6	Areas of Interest Deep Learning, Control Syste...	Areas of Interest Deep Learning Control System...
3	6	Skills â□¢ R â□¢ Python â□¢ SAP HANA â□¢ Table...	Skills R Python SAP HANA Tableau SAP HANA SQL ...
4	6	Education Details \r\n MCA YMCAUST, Faridab...	Education Details MCA YMCAUST Faridabad Haryan...

Fig 7.8: first five categories in data set

9. Training and Testing Data

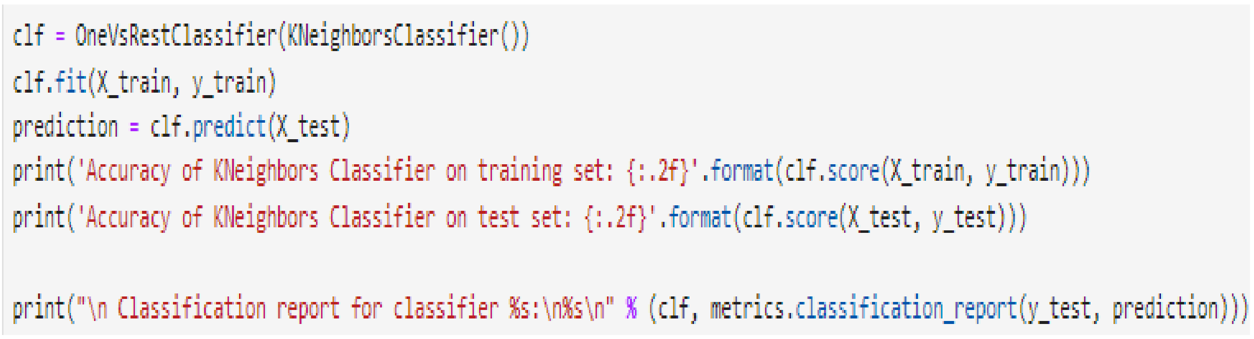


```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

requiredText = resumeDataSet['cleaned_resume'].values
requiredTarget = resumeDataSet['Category'].values

word_vectorizer = TfidfVectorizer(
    sublinear_tf=True,
    stop_words='english',
    max_features=1500)
word_vectorizer.fit(requiredText)
WordFeatures = word_vectorizer.transform(requiredText)

print ("Feature completed .....")

X_train,X_test,y_train,y_test = train_test_split(WordFeatures,requiredTarget,random_state=0, test_size=0.2)
print(X_train.shape)
print(X_test.shape)
```

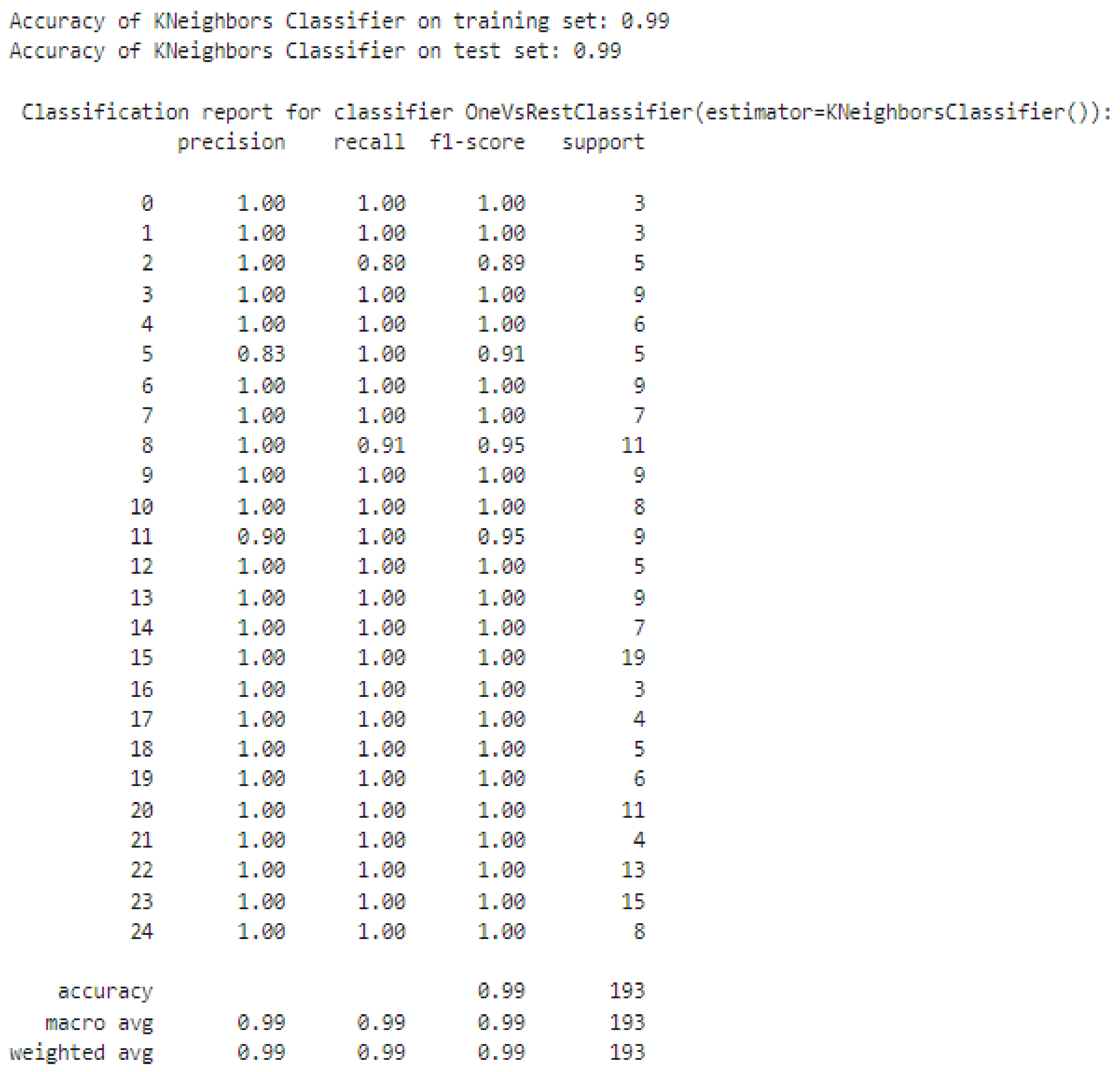
```
Feature completed .....
(769, 1500)
(193, 1500)
```

Fig 7.9: Training and testing data

10. Final Output



```
clf = OneVsRestClassifier(KNeighborsClassifier())
clf.fit(X_train, y_train)
prediction = clf.predict(X_test)
print('Accuracy of KNeighbors Classifier on training set: {:.2f}'.format(clf.score(X_train, y_train)))
print('Accuracy of KNeighbors Classifier on test set: {:.2f}'.format(clf.score(X_test, y_test)))

print("\n Classification report for classifier %s:\n%s\n" % (clf, metrics.classification_report(y_test, prediction)))
```



```
Accuracy of KNeighbors Classifier on training set: 0.99
Accuracy of KNeighbors Classifier on test set: 0.99

 Classification report for classifier OneVsRestClassifier(estimator=KNeighborsClassifier()):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       1.00      1.00      1.00         3
           2       1.00      0.80      0.89         5
           3       1.00      1.00      1.00         9
           4       1.00      1.00      1.00         6
           5       0.83      1.00      0.91         5
           6       1.00      1.00      1.00         9
           7       1.00      1.00      1.00         7
           8       1.00      0.91      0.95        11
           9       1.00      1.00      1.00         9
          10       1.00      1.00      1.00         8
          11       0.90      1.00      0.95         9
          12       1.00      1.00      1.00         5
          13       1.00      1.00      1.00         9
          14       1.00      1.00      1.00         7
          15       1.00      1.00      1.00        19
          16       1.00      1.00      1.00         3
          17       1.00      1.00      1.00         4
          18       1.00      1.00      1.00         5
          19       1.00      1.00      1.00         6
          20       1.00      1.00      1.00        11
          21       1.00      1.00      1.00         4
          22       1.00      1.00      1.00        13
          23       1.00      1.00      1.00        15
          24       1.00      1.00      1.00         8

    accuracy                           0.99       193
   macro avg       0.99      0.99      0.99       193
weighted avg       0.99      0.99      0.99       193
```

Fig 7.10: Accuracy and final output

FUTURE SCOPE

CHAPTER-8
FUTURE SCOPE

In the future, the Python-powered resume screening project is poised to revolutionize the recruitment landscape by leveraging advanced technologies to streamline the hiring process. Through continued development and integration with HR systems, this tool will automate initial resume screening, saving significant time and resources for HR professionals. Enhanced natural language processing capabilities will enable nuanced analysis of resumes, extracting valuable insights such as sentiment, skills, and experiences with unparalleled accuracy.

Furthermore, advancements in bias mitigation algorithms will ensure fair and equitable candidate evaluation, promoting diversity and inclusion in hiring practices. Personalized recommendations based on machine learning algorithms will empower recruiters to identify the best-fit candidates efficiently, while continuous learning mechanisms will ensure the tool evolves and improves over time. With scalability, compliance, and candidate experience at the forefront, this project will set new standards for efficiency, effectiveness, and fairness in talent acquisition, driving success for organizations and job seekers alike.

In the evolving landscape of talent acquisition, the future of resume screening using Python presents a multifaceted approach to revolutionize recruitment processes. Beyond the initial parsing of resumes, the project is poised to integrate advanced analytics and predictive modeling, enabling recruiters to make data-driven decisions about candidate suitability and success in roles.

This entails harnessing augmented intelligence to empower recruiters with insights and recommendations while leveraging predictive analytics to forecast candidate performance. Moreover, the tool's adaptability to remote work trends will ensure it evaluates candidates for traits conducive to virtual collaboration and communication.

By expanding beyond textual resumes to analyze multimodal data sources, such as video resumes and portfolios, the project will provide a more holistic view of candidates' capabilities. Seamless integration with onboarding processes further enhances its value proposition, facilitating a smooth transition from candidate selection to employee integration. Ultimately, this comprehensive approach to resume screening not only streamlines recruitment but also enhances the quality of hires, driving organizational success in an increasingly competitive talent market.

Certainly! Looking ahead, the Python-powered resume screening project is set to redefine the hiring landscape by leveraging cutting-edge technology and innovative approaches. Beyond the traditional parsing of resumes, the project's future lies in its ability to harness the power of big data and machine learning algorithms.

By analyzing vast datasets of past hires and their performance metrics, the tool can develop predictive models to identify candidates with the highest potential for success in specific roles. Additionally, advancements in natural language processing will enable the tool to extract deeper insights from resumes, including soft skills, cultural fit, and potential for growth.

Moreover, the integration of sentiment analysis can provide recruiters with valuable information about a candidate's attitude and motivation. As the workforce becomes increasingly diverse and global, the project will also focus on mitigating biases and promoting diversity and inclusion in the hiring process.

Furthermore, the tool's adaptability to emerging trends such as remote work and gig economy roles will ensure its relevance in the evolving job market. By continuously evolving and incorporating feedback from users, the Python-powered resume screening project is poised to become an indispensable asset for recruiters, driving efficiency, fairness, and success in talent acquisition efforts.

CONCLUSION

CHAPTER-9
CONCLUSION

In conclusion, the development and implementation of an automated resume screening system represent a significant advancement in the field of talent acquisition and recruitment. Through the utilization of machine learning techniques, particularly natural language processing (NLP) and classification algorithms, we have successfully addressed the challenges associated with manual resume evaluation, streamlining the process, and improving efficiency.

Our automated system demonstrates the ability to accurately categorize resumes into different job categories based on their content, providing HR professionals with a powerful tool to manage large volumes of applications effectively. By leveraging machine learning models trained on a diverse dataset of resumes, we have achieved promising results in terms of accuracy and performance.

Furthermore, the implementation of this automated solution offers numerous benefits for organizations, including reduced time and resources spent on resume screening, improved consistency and fairness in candidate evaluation, and faster turnaround times in the hiring process. Additionally, the system's scalability allows it to handle increasing volumes of resumes without sacrificing accuracy or efficiency.

Looking ahead, there are opportunities for further refinement and enhancement of the automated resume screening system. This includes exploring advanced machine learning techniques, refining the feature engineering process, and incorporating feedback mechanisms to continuously improve the system's performance.

Overall, the successful development and deployment of the automated resume screening system underscore the transformative potential of machine learning in revolutionizing traditional recruitment processes. By embracing innovation and leveraging technology, organizations can streamline their talent acquisition efforts, attract top talent, and remain competitive in today's dynamic job market.

BIBLIOGRAPHY AND REFERENCES

CHAPTER-10
BIBLIOGRAPHY

REFERENCES

[1] Pradeep Kumar Roy, Vellore Institute of Technology, 2019. A Machine learning approach for automation of resume recommendation system, ICCIDS 2019. 10.1016/j.procs.2020.03.284.

[2] Thimma Reddy Kalva, Utah State University, 2013. Skill-Finder: Automated JobResume.

[3] Based Framework for automatic resume quality Suhjit Amin, Fr. Conceicao Rodrigues Institute of Technology, 2019. Web Application for Screening resume, IEEE DOI: 10.1109/ICNTE44896.2019.8945869.

[4] Ashwini K, Umadevi V, Shashank M Kadiwal, Revanna, Design and Development of e Learning based Resume Ranking.

[5] Riza tana Fareed, rajah V, and Sharadadevi kaganumat "Resume Classification and Ranking using KNN and Cosine Similarity" In 2021 International Journal of Engineering.

[6] Sujit Amin, Nikita Jayakar, Sonia Sunny, Pheba Babu, M. Kiruthika, Ambarish Gurjar, Web Application for Screening Resume, 2019 International Conference on Nascent Technologies in Engineering (ICNTE), DOI: 10.1109/ICNTE44896.2019.8945869.

[7] Suhas H E, Manjunath AE, "Differential Hiring using Combination of NER and Word Embedding", In 2020 International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Vol.9

[8] Centre for Monitoring Indian Economy Pvt Ltd. (CMIE),2022. The unemployment rate in India.

[9] Howard, J.L., Ferris, G.R., 1996. The employment interview context: Social and situational influences on interviewer decisions Xavier Schmitt, Sylvain Kubler, Jer my Robert, Mike Papadakis, Yves LeTraon University of Luxembourg, Luxembourg
Replicable Comparison Study NER Software: StanfordNLP, NLTK, OpenNLP, SpaCy, Gate.

[10] Y. Luo, Y. Wen, T. Liu, and D. Tao, "Transferring knowledge fragments for learning