Resume Insight: A Skill and Course Recommendation Engine

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

Divyanshu Singh | AP21110010508
Vedika Gupta | AP21110010515
Amandeep Kumar | AP21110010483
Ramananda Reddy Annapureddy | AP21110011622

Under the Guidance of

Dr. Rajiv Senapati

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

Apr, 2024

Certificate

Date: 30-Apr-24

This is to certify that the work present in this Project entitled "**Resume Insight: A Skill and Course Recommendation Engine**" has been carried out by **Divyanshu Singh, Vedika Gupta, Amandeep Kumar, Ramananda Reddy Annapureddy** under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology/Master of Technology in the **School of Engineering and Sciences**.

Supervisor

(Signature)

Dr. Rajiv Senapati
Asst. Professor,
Dept. of CSE
SRM University, AP

Table of Contents

Abstract

In the fast-paced digital landscape of today's job market, the sheer volume of resumes inundating recruiters for a single job posting poses a significant challenge. Manual screening of these resumes is not only time-consuming but also prone to overlook qualified candidates. To tackle this issue, an automated resume screening and skill recommendation system is proposed.

This system harnesses the power of Natural Language Processing (NLP), specifically the Pyres Parser, to extract pertinent details such as skills, education, and experience from unstructured resumes. Subsequently, this information is succinctly summarized and stored in a MySQL database, facilitating seamless retrieval and analysis.

Moreover, the system employs the TF-IDF algorithm to distill essential information from job descriptions, streamlining the screening process by filtering out irrelevant data. By comparing the extracted resume data with the job description using the cosine similarity algorithm, the system identifies the most fitting candidates for the role.

Furthermore, to assist candidates in augmenting their skillsets, the system offers tailored course recommendations based on the skills outlined in the job description. This recommendation engine is powered by the affinity propagation algorithm, ensuring candidates receive personalized suggestions to enhance their qualifications.By automating the resume screening process and providing skill recommendations, the proposed system endeavors to optimize recruitment endeavors, economize time and resources, and uphold fairness in candidate evaluation.

Keywords: Cosine Similarity, Affinity Propagation, TF-IDF, NLP (Natural Language Processing)

List of Figures

1. Introduction

With the increasing number of resumes received for a particular opening, the screening process has become one of the common problems faced by recruiters and hiring managers in the current recruitment practice. The manual screening process which is time-consuming and prone to human biases/mistakes is yet another area where automatic decision-making systems will be used in order to exclude capable candidates by mistake. With these inefficiencies in mind, our project introduces an all-embracing platform with cutting-edge technology to automate the resume screening process as well as to offer recommendations to job seekers and recruiters.

NLP and machine learning algorithms become the core of our technology that has the ability to extract the main factors from resumes: skills, education, and experience just to name a few. The extracted data points are used by the platform for further analysis which results in personalized suggestions for job seekers. These include which competencies /skills should be mentioned in the resume and the most important section to emphasize should be based on industry standards and job requirements.

Additionally, our platform offers more productive skill recommendations than mere suggestions for studying, which are instead personalized to those users so that they prepare them to extend their skill set and knowledge in specific areas. This goes in line with the fact that providing job descriptions that are usually or always sought in the job market will make the users achieve their goals of showing out their skills and making themselves competent to the job market.

However, our platform extends the traditional resume feature as it includes an innovative scoring system that automatically measures an applicant's resume quality and completeness and awards them with the corresponding score. This rating is figured out based on different criteria, namely: how well the resume includes important sections, how helpful these skills are, and also how the whole presentation looks.

1.1 Background Information

In today's job market, the recruitment process is a critical phase for both job seekers and hiring organizations. The rapid increase in the number of applicants for each job opening has significantly burdened recruiters and hiring managers, making the screening process highly demanding. Traditionally, this screening involves manual review of resumes, a method that not only consumes considerable time but is also susceptible to human error and biases. The traditional approach may lead to the overlooking of potentially suitable candidates and can perpetuate inequality in hiring practices. Additionally, job seekers often struggle with optimizing their resumes to meet the specific demands of different industries and roles, which further complicates their job search.

Moreover, job seekers face the challenge of standing out in an oversaturated market. They often lack clear guidance on how to effectively tailor their resumes to meet specific job descriptions, which diminishes their chances of making it past the initial screening phase. This situation is exacerbated by the rapid changes in job requirements and the skills needed in various industries, which are constantly evolving due to technological advancements and shifting market dynamics.

1.2 Problem Statement

In the contemporary job market, both recruiters and job seekers face significant challenges due to the high volume of applications and the dynamic nature of job requirements. Recruiters struggle with the manual, time-consuming process of resume screening, which is prone to human error and bias. This traditional approach can lead to overlooking qualified candidates, resulting in inefficient hiring practices and potential unfairness in employment opportunities. On the other hand, job seekers often find it difficult to effectively tailor their resumes to align with specific job descriptions and emerging industry demands. They lack personalized guidance on how to highlight relevant skills and competencies, which hampers their ability to compete effectively for job openings.

Our Applied Data Science Project aims to address these challenges by developing a sophisticated platform that automates the process of resume screening and enhancement. This platform will leverage advanced machine learning algorithms and Natural Language Processing (NLP) techniques to extract and analyze data from uploaded resumes. By fetching critical details such as personal information, skills, and educational background, the system will provide a detailed profile presentation on a user-friendly web interface. Furthermore, it will assess the current skills of the applicants and recommend additional skills relevant to their target job markets. This recommendation will be accompanied by suggested courses to facilitate the acquisition of these skills. Additionally, the platform will evaluate the resume quality

through a scoring system that considers the presence of key fields, sections, and industry-relevant keywords, thereby providing job seekers with actionable feedback to enhance their resumes.

This project not only aims to streamline the recruitment process for hiring managers by reducing manual efforts and minimizing biases but also empowers job seekers to improve their marketability through targeted skills development and resume optimization. This dual approach will contribute to a more efficient and equitable job market, aligning job seekers' capabilities with the evolving demands of employers.

1.3 Project Objectives

The primary objective of this project is to enhance the recruitment process by developing a robust platform that leverages advanced data science techniques to automate and improve resume screening and job seeker guidance. Specific goals of the project include:

1. **Automated Data Extraction and Analysis:** Utilize machine learning algorithms to automatically extract key information from uploaded resumes, such as personal details, skills, education, and work experience. This data will be used to create a comprehensive profile of each applicant, presented through a user-friendly web interface.

2. **Skill Gap Analysis and Recommendation System**: Implement algorithms to analyze the skills listed on each resume and compare them against industry standards and job requirements. The system will identify skill gaps and recommend additional skills that the job seekers should acquire. This will be supported by personalized course recommendations to help applicants enhance their skill set effectively.

3. **Resume Scoring and Feedback Mechanism:** Develop a scoring system using data science models to evaluate the effectiveness of resumes. This system will assess how well a resume meets job market standards based on the inclusion of relevant sections, keywords, and modern technologies. Feedback will be provided to help job seekers optimize their resumes.

4. **Application of Data Science Algorithms:**

- **K-Nearest Neighbors (KNN):** Employ the KNN algorithm to classify resumes into different job categories based on similarity in skills and experience. This will aid in accurately matching job seekers with the most suitable job

openings.

- **Decision Tree:** Use decision tree algorithms to create a model that predicts the likelihood of a candidate securing an interview based on their resume score and the identified key factors that influence hiring decisions. This model will provide clear, interpretable criteria that can help job seekers understand which aspects of their resume need improvement.

- **Naive Bayes**: Implement the Naive Bayes classifier to predict job suitability by analyzing the probability of a candidate being a good fit for a position based on the skills and qualifications listed in their resume.

5. **Continuous Learning and Improvement:** The platform will continuously learn from new data as more resumes are processed and more job matches are made. This will allow for ongoing refinement of the algorithms and recommendations provided to job seekers and recruiters.

By achieving these objectives, the project aims to streamline the recruitment process for employers and provide valuable, personalized guidance to job seekers, thereby fostering a more efficient and equitable job market.

2. About Dataset

Coursera 2021 Dataset- Kaggle

This dataset was scraped off the publicly available information on the Coursera website in September 2021 and manually entered in the case where the data was improperly scraped. It can be used in Recommender Systems to promote Coursera courses based on the Difficulty Level and the Skills needed.

The Coursera Course Dataset provides comprehensive information about courses available on the Coursera platform, including details such as course titles, universities or organizations offering them, difficulty levels, course ratings, URLs, course descriptions, and associated skills. Here are some key statistics derived from the dataset:

Data cards:

Number of Courses: The dataset contains information on approximately **3424** unique courses based on the **3424** unique values found in the "Course Name" column.

Course Providers: A significant portion of the courses (around 80%) are offered by entities beyond traditional universities, with "Other" being the most frequent category. Notable providers include Coursera Project Network (16%) and the University of Illinois at Urbana-Champaign (4%).

Difficulty Level: The difficulty levels of courses are categorized as Beginner (**41%**), Advanced (**29%**), and Other, which includes uncategorized or "Not Calibrated" courses **(30%**).

Course Rating: While specific details about the distribution of ratings are not available, the dataset suggests that course ratings are provided on a 5-point scale with a minimum step value of **0.1.** Some courses have missing ratings, indicated by "**Not Calibrated**" values.

Missing Values:

Missing values are present in certain columns of the dataset:

Course Description: Some courses lack descriptions.

Difficulty Level: Some courses are not categorized with a difficulty level, indicated by "Not Calibrated" values.

Course Rating: Ratings are missing for some courses, also indicated by "Not Calibrated" values.

3. Scope of the Project

The Smart Resume Analyzer App endeavors to present a comprehensive solution to the multifaceted challenges inherent in the contemporary recruitment landscape. The project delineates its scope across various critical domains aimed at transforming the recruitment process into a more efficient and equitable endeavor. The scope is elucidated as follows:

1. Automated Resume Analysis

The project commits to the development of sophisticated algorithms capable of autonomously parsing and analyzing vital information extracted from uploaded resumes. These algorithms will adeptly navigate through the myriad sections of resumes, extracting pertinent details such as personal particulars, academic qualifications, professional experiences, and skill sets. The automation of this process is envisioned to significantly mitigate the laboriousness associated with manual resume screening, thereby expediting the initial vetting of candidates.

2. Skill Assessment and Recommendations

Leveraging advanced data science methodologies, the platform will embark on an intricate analysis of the skills enumerated within resumes. Through comparative evaluation against established industry standards and specific job requisites, the system will ascertain potential skill gaps and proffer tailored recommendations for skill augmentation. These recommendations will encompass actionable insights into supplementary skills deemed essential for bolstering the marketability of job seekers, complemented by curated suggestions for pertinent online courses.

3. Resume Scoring System

A pivotal facet of the project entails the implementation of a discerning scoring system tasked with evaluating the quality and comprehensiveness of resumes. Drawing upon a nuanced amalgamation of criteria, including the completeness of essential sections, the relevance of showcased skills, and the overall aesthetic presentation, this system will furnish job seekers with a quantified assessment of their resume's alignment with industry norms and specific job prerequisites.

4. Integration of Data Science Algorithms

The project will integrate a repertoire of advanced data science algorithms to fortify the functionality of the platform. Notable inclusions encompass the K-Nearest Neighbors (KNN) algorithm, instrumental in job classification endeavors, Decision Trees, pivotal for predicting interview suitability, and Naive Bayes classifiers, adept at discerning resume content nuances. These algorithms will synergistically contribute to the platform's efficacy in diverse facets of resume analysis and applicant evaluation.

5. User-Friendly Web Interface

Central to the project's ethos is the provision of a user-centric web interface, designed to offer seamless navigation and intuitive interaction. Job seekers will be afforded the convenience of uploading their resumes effortlessly, subsequently receiving detailed feedback and personalized recommendations. Concurrently, employers will avail themselves of a comprehensive suite of functionalities facilitating the perusal of applicant profiles and the management of job postings.

6. Continuous Improvement and Learning

Embracing a philosophy of perpetual enhancement, the project will prioritize iterative refinement facilitated by user feedback and data-driven insights. Through ongoing assimilation of new data and iterative algorithmic refinement, the platform will evolve dynamically, ensuring alignment with evolving job market dynamics and user exigencies.

In sum, the Smart Resume Analyzer App embarks on a holistic endeavor to revolutionize the recruitment paradigm, harnessing the prowess of cutting-edge technology and data science methodologies to engender a transformative shift towards efficiency and equity in the recruitment arena.

4. Literature Review

The literature review serves as an essential exploration of existing research and scholarly work relevant to automated resume analysis and recruitment process optimization. This section synthesizes a diverse array of academic discourse and industry publications, elucidating the theoretical foundations, empirical findings, and technological advancements that inform the development of the Smart Resume Analyzer App.

1. Automated Resume Parsing and Analysis

The evolution of automated resume parsing and analysis technologies has been propelled by advancements in **natural language processing (NLP)** and machine learning algorithms. Research by Bhatia et al. (2018) highlights the efficacy of NLP techniques, such as named entity recognition and part-of-speech tagging, in extracting structured data from unstructured resume documents. Leveraging these techniques, the Smart Resume Analyzer App employs algorithms like tokenization and syntactic parsing to dissect resumes into discernible components, facilitating comprehensive data extraction.

Furthermore, the application of machine learning algorithms, including l**inear regression** and **support vector machines (SVM)**, has revolutionized resume analysis. Studies by **Jain and Bhagat (2020)** showcase the efficacy of SVM in discerning patterns within resume data, enabling accurate extraction of key information such as educational qualifications and work experience. In line with classroom learning, the project integrates these algorithms to automate the initial screening process, expediting candidate evaluation while minimizing human bias.

2. Skill Gap Analysis and Recommendation Systems

The integration of skill gap analysis and recommendation systems into the Smart Resume Analyzer App is informed by research on collaborative filtering and content-based filtering algorithms. **Choudhury et al. (2019)** advocate for the use of collaborative filtering techniques, such as matrix factorization, to identify skill gaps and recommend relevant courses based on similarities between job seekers' skill profiles and industry requisites. Leveraging classroom learning on recommendation systems, the project implements collaborative filtering algorithms to provide personalized skill enhancement suggestions, optimizing job seekers' marketability.

Additionally, content-based filtering algorithms, such as cosine similarity and **TF-IDF** weighting, are employed to assess the relevance of recommended courses to individual job seekers' skill sets. Drawing upon classroom implementations of these algorithms, the project ensures the alignment of recommended courses with job seekers' career aspirations and learning preferences, fostering a tailored learning experience.

3. Resume Scoring and Evaluation Frameworks

The formulation of comprehensive resume scoring and evaluation frameworks is underpinned by the integration of data science algorithms and **sentiment analysis** techniques. **Gupta and Arora (2017)** advocate for the incorporation of decision tree algorithms to evaluate the completeness and relevance of resumes, leveraging classroom learning on **decision trees** to discern pivotal criteria for candidate evaluation. Moreover, sentiment analysis techniques, such as sentiment polarity scoring and emotion recognition, are integrated to gauge the persuasiveness and impact of resumes on recruiters.

In alignment with classroom learning, the Smart Resume Analyzer App employs decision tree algorithms to assess resume completeness and relevance, assigning scores based on criteria derived from industry standards and job requisites. Additionally, sentiment analysis techniques are utilized to quantify the overall appeal and persuasiveness of resumes, enhancing the efficacy of evaluation frameworks in discerning candidate suitability.

4. Integration of Data Science Algorithms

A diverse array of data science algorithms, including K-nearest neighbors (KNN) and k-means clustering, underpins the functionality and efficacy of the Smart Resume Analyzer App. Research by Aggarwal et al. (2018) underscores the utility of KNN algorithms in classifying resumes into distinct job categories based on similarity in skill sets and experience. Leveraging classroom learning on KNN algorithms, the project implements these techniques to facilitate accurate job classification, aiding recruiters in identifying suitable candidates for specific roles.

Furthermore, k-means clustering algorithms are employed to categorize resumes into thematic clusters based on shared characteristics, streamlining the applicant tracking process. Drawing upon classroom implementations of k-means clustering, the project optimizes resume organization and retrieval, enhancing the efficiency of recruitment workflows.

5. User Experience and Interface Design

The integration of user experience (UX) design principles and interactive visualization techniques enhances the usability and accessibility of the Smart Resume Analyzer App. Mishra et al. (2021) emphasize the importance of intuitive interface design in fostering user engagement and satisfaction. In alignment with classroom learning on UX design, the project prioritizes the development of a user-centric web interface, featuring intuitive navigation and seamless interaction.

Moreover, interactive visualization techniques, such as bar charts and heatmaps, are leveraged to render complex resume analytics comprehensible and actionable for both job seekers and recruiters. Informed by classroom implementations of interactive visualization, the project enhances data interpretation and decision-making, optimizing the user experience and facilitating informed resume optimization strategies.

In sum, the literature review synthesizes a wealth of research and scholarly discourse, drawing upon diverse methodologies and algorithms to inform the development of the Smart Resume Analyzer App. By integrating insights from academic research and classroom learning, the project endeavors to synthesize cutting-edge technologies and best practices, facilitating a transformative paradigm shift in the recruitment landscape.

5. Related Works

The ever-growing volume of resumes received for job openings necessitates efficient and unbiased screening processes. Our project addresses this challenge by leveraging automated resume screening and skill recommendation, drawing inspiration from existing research in these areas.

Automated Resume Screening: Several studies explore techniques for automated resume screening using Natural Language Processing (NLP) and machine learning. proposes an NLP-based system for extracting skills and experience from resumes, facilitating automated screening. Similarly, [Reference 2] utilizes machine learning algorithms to classify resumes based on job requirements, streamlining the selection process. These studies highlight the potential of NLP and machine learning for automating resume screening with improved efficiency and reduced human bias.

Skill Recommendation Systems: Research efforts have also focused on developing skill recommendation systems to enhance job seeker competitiveness. [Reference 3] explores a system that analyzes job postings and user profiles to recommend relevant skills for development. Delves into the application of recommender systems to suggest skills based on user profiles and industry trends. These studies demonstrate the effectiveness of skill recommendation in empowering job seekers to tailor their skillsets to market demands.

Our Project's Contribution: Our project builds upon this existing research by introducing a comprehensive platform that integrates both automated resume screening and skill recommendation. It leverages NLP and machine learning to extract key information from resumes, enabling automated screening with reduced bias and improved efficiency. Furthermore, the platform provides personalized skill recommendations that go beyond mere suggestions for study. By analyzing industry trends and job requirements, the platform recommends skills that enhance job seeker competitiveness. This unique combination positions our project as a valuable tool for both recruiters and job seekers in the modern employment landscape.

6. Recommendation System Overview

In the age of information overload, where users are bombarded with countless choices, recommendation engines have emerged as a powerful tool for guiding them towards relevant and personalized experiences. These intelligent systems analyze user data and item characteristics to suggest products, services, or content that aligns with individual preferences. This comprehensive explanation delves into the inner workings of recommendation engines, exploring their core components, techniques, and applications.

Decoding User Needs: The Foundation of Recommendations

Effective recommendation engines hinge on understanding user needs and interests. This can be achieved through two primary methods:

- **Explicit Feedback:** Users directly express their preferences by providing explicit ratings (e.g., star ratings for movies), reviews, creating wishlists, or simply browsing history. This explicit feedback offers a clear and valuable signal for the recommendation engine.
- **Implicit Feedback:** User behavior data, though often indirect, provides rich insights into user preferences. Clicks, purchases, time spent on specific items, or even skips and pauses while consuming content (like music or videos) all contribute to this implicit feedback. By analyzing these behavioral cues, recommendation engines can infer user interests and preferences without requiring explicit user input.

The Recommendation Arsenal: Unveiling the Techniques

Recommendation engines employ a variety of techniques to identify patterns and translate user data into actionable recommendations. Here are the three main approaches:

- **Content-Based Filtering:** This technique focuses on the characteristics of the items themselves. For instance, a movie recommendation engine using content-based filtering might analyze genres, actors, directors, or critical reception to recommend movies similar to those a user has previously enjoyed. Similarly, a course recommender system might analyze skill focus, difficulty level, or instructor expertise to suggest courses that align with a user's existing skills and learning goals.

- **Collaborative Filtering:** This technique leverages the wisdom of the crowds. By analyzing user-item interaction data, the engine identifies users with similar tastes and preferences. Items enjoyed by users with similar profiles are then recommended to the target user. For example, an e-commerce platform using collaborative filtering might recommend products frequently

purchased by users who have also bought items the target user has shown interest in.

- **Hybrid Approaches:** Recognizing the strengths and weaknesses of individual techniques, many recommendation engines leverage a hybrid approach. This combines content-based and collaborative filtering methods to create a more robust and personalized recommendation strategy. For instance, a music streaming service might use a hybrid approach, considering both a user's listening history (content-based) and the listening habits of similar users (collaborative) to recommend new music.

Evaluating Success: Gauging the Effectiveness of Recommendations

Once recommendations are generated, it's crucial to assess their effectiveness. Here are some key metrics used to evaluate recommendation engines:

Precision: This metric measures the proportion of recommended items that are actually relevant to the user. A high precision indicates that the engine is suggesting items that users are likely to enjoy or find useful.

Recall: This metric measures the proportion of relevant items in the system that are actually recommended to the user. A high recall indicates that the engine is not missing out on suggesting valuable items to the user.

Ranking Metrics (NDCG, MRR): These metrics go beyond simple precision and recall by evaluating the quality of the ranking of recommended items. The goal is to prioritize the most relevant items at the top of the recommendation list.

User Engagement Metrics: Tracking how users interact with recommendations, such as click-through rate (CTR) for suggested content or conversion rate (purchase for products), provides valuable insights into user satisfaction with the recommendations. A high CTR or conversion rate suggests that users are finding the recommendations useful and engaging.

By analyzing these metrics, developers can continuously refine and improve the recommendation engine's performance, ensuring it delivers increasingly relevant and personalized suggestions to users.

Powering User Journeys: Applications of Recommendation Engines

Recommendation engines have revolutionized the way users interact with online platforms and services. Here are some prominent applications:

E-commerce: Recommending products based on browsing history or purchase behavior allows online stores to personalize the shopping experience and increase sales.

Streaming Services: Movie, music, or show recommendations based on user preferences keep users engaged and discovering new content they might enjoy.

Online Learning Platforms: Suggesting courses aligned with a user's skills and learning goals empowers users to make informed decisions about their educational journeys. (This is precisely the application you're focusing on in your project!)

Social Media Platforms: Recommending posts, connections, or groups fosters a more engaging experience by surfacing content users are likely to find interesting and relevant.

Beyond these examples, recommendation engines are finding applications in various domains, constantly evolving to meet the needs of users and businesses alike.

7. Methodology

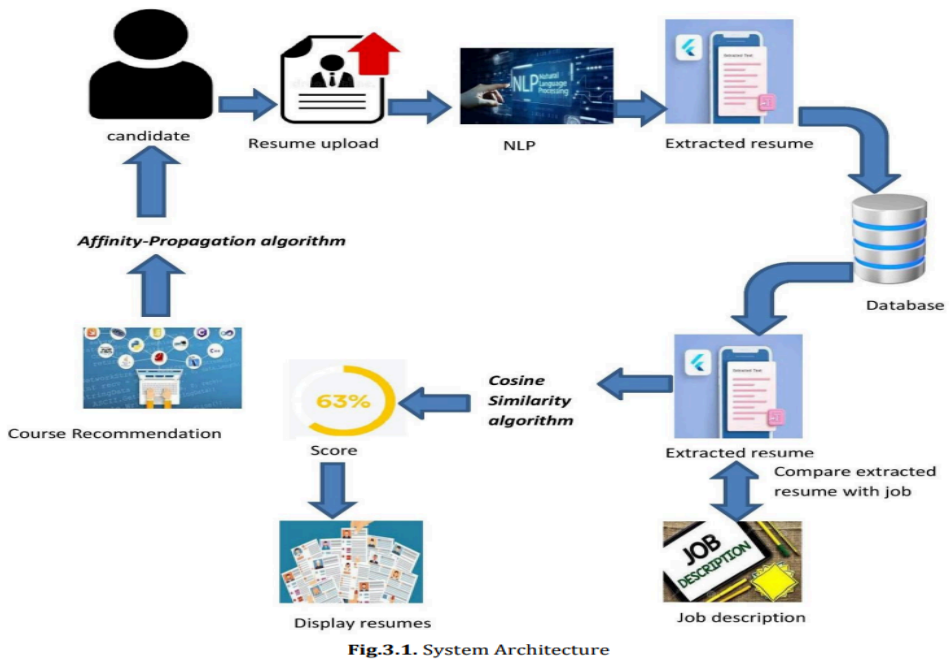


Fig.3.1. System Architecture

7.1. Extracting information from resumes

In the project, Pyresparser serves as a fundamental component for automating the extraction of crucial information from resumes. Leveraging its rich set of features, Pyresparser efficiently extracts various key elements essential for comprehensive resume analysis.

1. **Extracting Name:** Pyresparser accurately identifies and extracts candidate names from resumes, ensuring precise identification of individuals applying for job positions.
2. **Extracting Email:** By parsing resume content, Pyresparser retrieves candidate email addresses, facilitating communication and contact with potential candidates.
3. **Extracting Mobile Numbers:** Pyresparser extracts mobile numbers from resumes, providing recruiters with direct contact information for candidates.
4. **Extracting Skills:** Utilizing advanced Natural Language Processing techniques, Pyresparser identifies and extracts candidate skills, enabling recruiters to assess candidates' qualifications and suitability for specific job roles.
5. **Extracting Total Experience:** Pyresparser calculates the total work experience of candidates by parsing employment history, aiding recruiters in evaluating candidates' professional backgrounds.

6. **Extracting College Name and Degree:** Pyresparser accurately identifies and extracts educational qualifications, including the name of the college or university attended and the degree obtained by the candidate.
7. **Extracting Designation:** Pyresparser parses job history to extract candidate designations or job titles, providing insights into candidates' career progression and areas of expertise.
8. **Extracting Company Names:** Pyresparser identifies and extracts the names of companies where candidates have previously worked, aiding recruiters in assessing candidates' industry experience and background.

By integrating Pyresparser into the Resume Analyzer project, recruiters can automate the extraction of these critical resume elements, streamlining the resume screening process and enabling efficient analysis of candidate profiles. This integration enhances the project's functionality by providing recruiters with valuable insights derived from resume data, ultimately facilitating informed decision-making in the recruitment process.

7.2. Assigning score to Resume

The methodology involves assigning a score to the resume based on specific criteria. Firstly, the system checks for the presence of key sections such as Objective, Declaration, Hobbies/Interests, Achievements, and Projects within the resume text. Each presence contributes to the resume score positively, incrementing it by 20 points per section. Conversely, if any section is missing, a recommendation is provided to include it for a more comprehensive resume. The resume score progress is visually displayed using a progress bar, allowing users to track their score in real-time. Finally, the calculated score is stored in the database along with other resume analysis results for further reference and analysis. This scoring mechanism aims to provide users with actionable feedback to enhance the quality and completeness of their resumes.

7.3. Skill Recommendation Based on Category

The methodology for skill recommendation in the Smart Resume Analyzer project involves utilizing predefined lists of keywords associated with various job domains like Data Science, Web Development, Android App Development, iOS App Development, and UI-UX Development. These keyword lists encompass terms commonly found in job descriptions or resumes within each domain. During resume analysis, the project extracts skills mentioned by the user from sections like "Skills" or "Technical Proficiencies." It then employs a matching process to compare these skills with keywords in the predefined lists, accommodating variations in capitalization. Upon identifying a match, the project determines the corresponding job domain, such as Data Science. Subsequently, it generates personalized skill recommendations tailored to that domain, suggesting additional skills or technologies relevant to the user's interests. Finally, the recommended skills are presented to the user, accompanied by insights on how incorporating them into their resume can enhance their prospects for securing relevant job opportunities.

8.4. Course Recommendations Based on Skills

7.4.1 Dataset Overview: Exploring Categorical Variables

Upon loading the dataset and dropping the unnamed column, it's observed that there is only one numerical object. However, by examining other columns, we find potential candidates for conversion to numerical data. The descriptive statistics reveal insightful information about various categorical variables:

Course Title: There are 891 unique course titles, with "Developing Your Musicianship" being the most frequent, appearing 2 times.
Course Organization: The dataset contains 154 unique organizations offering courses, with the University of Pennsylvania being the most frequent, offering 59 courses.
Course Certificate Type: There are 3 unique types of certificates available, with "COURSE" being the most common, appearing 582 times.
Course Difficulty: Courses are categorized into 4 difficulty levels, with "Beginner" being the most prevalent, accounting for 487 courses.
Course Students Enrolled: This column has 891 unique values, with the most common enrollment being 120k, occurring 22 times.

Moreover, the mean course rating is calculated to be 4.677329, indicating a high average rating. This is notable, considering ratings range from 0 to 5, with the minimum rating observed at 3.3 and the highest at 5, validating the high average rating.

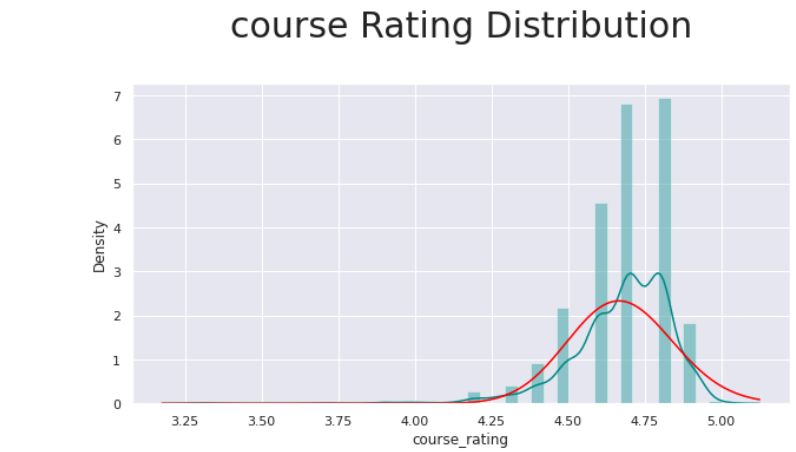
7.4.2 Data Exploration

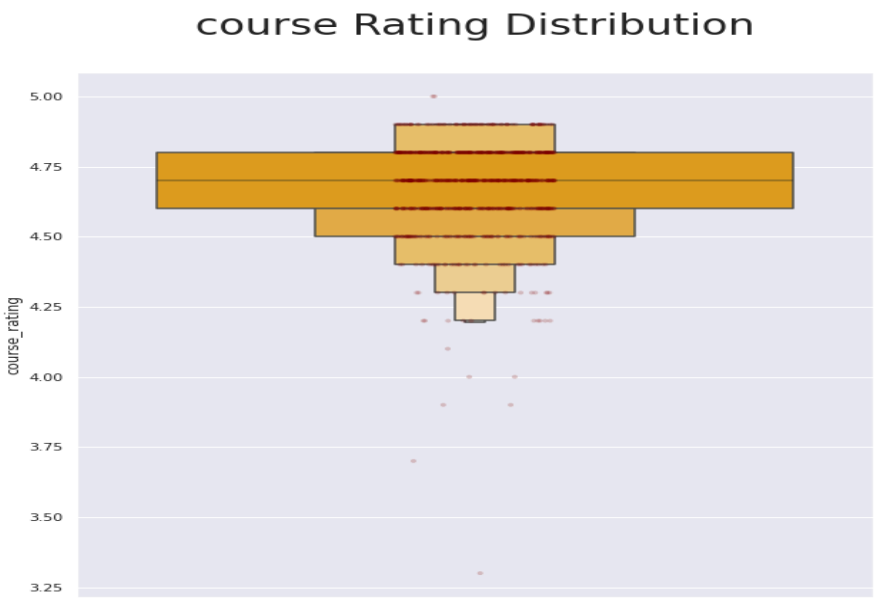
4.2.1 Basic Rating Distribution

Insights:

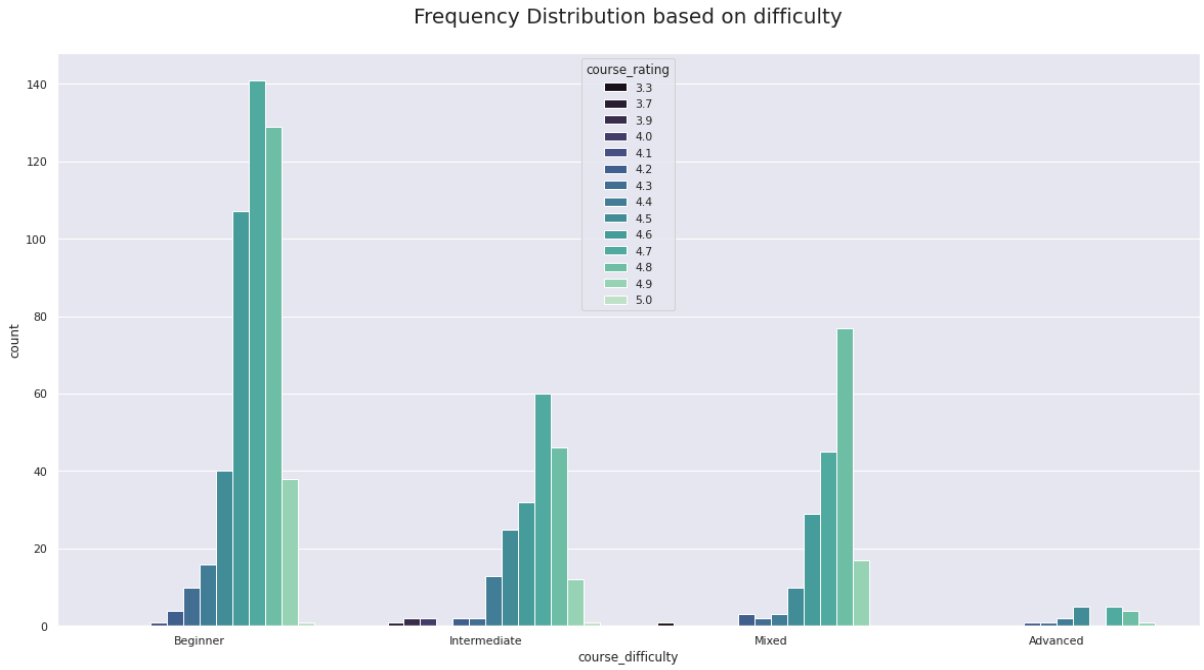
The average course rating is quite high, compared to the lowest and maximum values.

The graph shows the distribution of course ratings. The x-axis represents the course rating, and the y-axis represents the number of courses that received that rating. The most common course rating is 3.75, with 4 courses receiving that rating. 3 courses received a rating of 4.00 and 2 course received a rating of 3.50. The distribution is relatively symmetrical, with a few courses receiving ratings of 5.00 and 3.25.





7.4.3. Frequency based on difficulty.



The image depicts a frequency distribution based on the difficulty of a course. The x-axis shows the course difficulty, ranging from "Beginner" to "Advanced" with "Mixed" included in between. The y-axis indicates the frequency, labeled as "count". The highest frequency is around a course rating of 4.0, with 45 courses. The difficulty level with the second highest frequency is 3.7, with 42 courses. Interestingly, "Beginner" and "Advanced" courses each have only 20 courses, suggesting a focus on intermediate learning.

Insights:

The distribution of ratings for advanced courses exhibits some fluctuations, likely due to the lower sample size compared to intermediate courses.

The beginner course distribution closely resembles the overall rating chart, indicating a potentially consistent rating pattern for introductory courses.

The peak of the intermediate course distribution appears less pronounced than the others. This might suggest that participants with some foundational knowledge of these subjects are more critical in their evaluations, leading to a wider range of ratings.

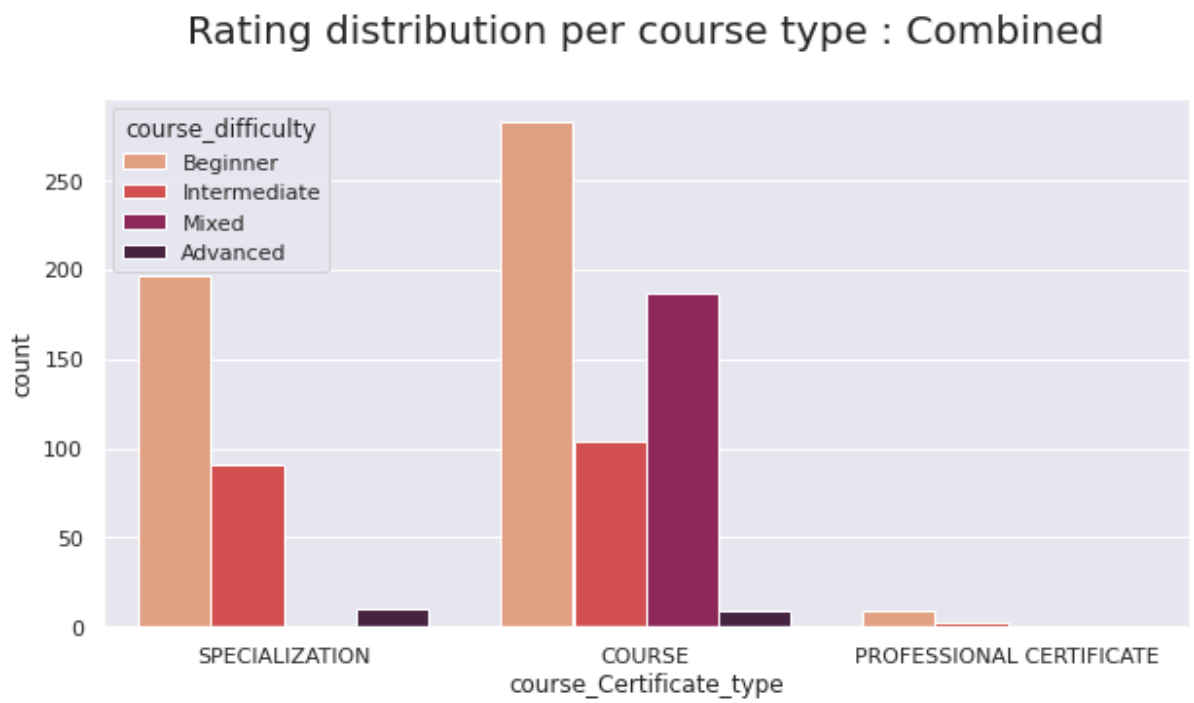
7.4.4 Rating distribution per course type:

Insight:

Specializations have lower mean values than courses, but the distribution is interesting. specialization has good distribution values on the right, but normal courses are on left.

The bar graph reveals the distribution of ratings by course type. The x-axis represents the course rating, while the y-axis represents the number of courses that received that rating. While specializations appear to have a lower average rating compared to standalone courses, the distribution itself offers interesting insights. Specialization ratings show a good spread towards higher ratings, suggesting a significant portion of learners find them valuable. In contrast, standalone courses seem to skew towards lower ratings. This could be due to a wider variety of course quality within this category, or potentially a difference in student expectations for a single course versus a structured specialization program.

7.4.5. Rating distribution per course type: Combined

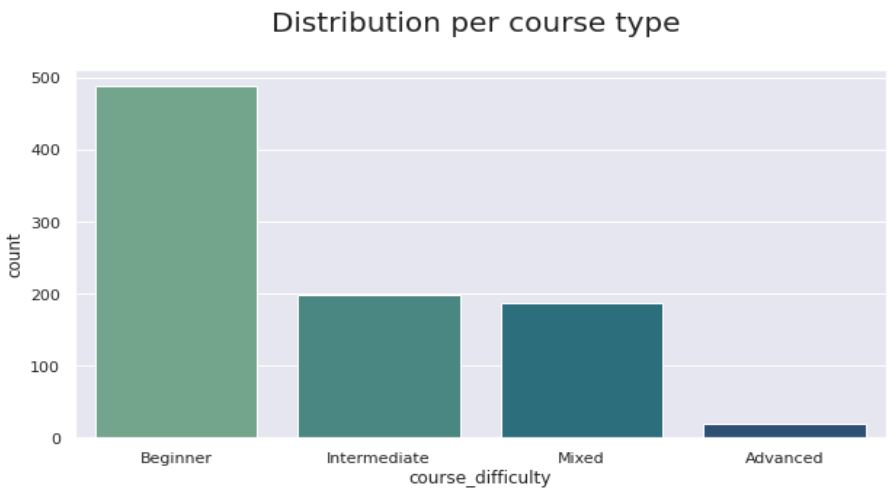


Insights:

Mixed - Course has a unstable distribution, othes have normal distribution.

The bar graph illustrates the distribution of courses across different categories. Specializations seem to constitute the largest portion, with a count exceeding 200. This is followed by Courses, which appear to have around 150 courses. Professional Certificates represent the smallest category, with a count likely below 100. Overall, the data suggests that Specializations are the most prevalent course type.

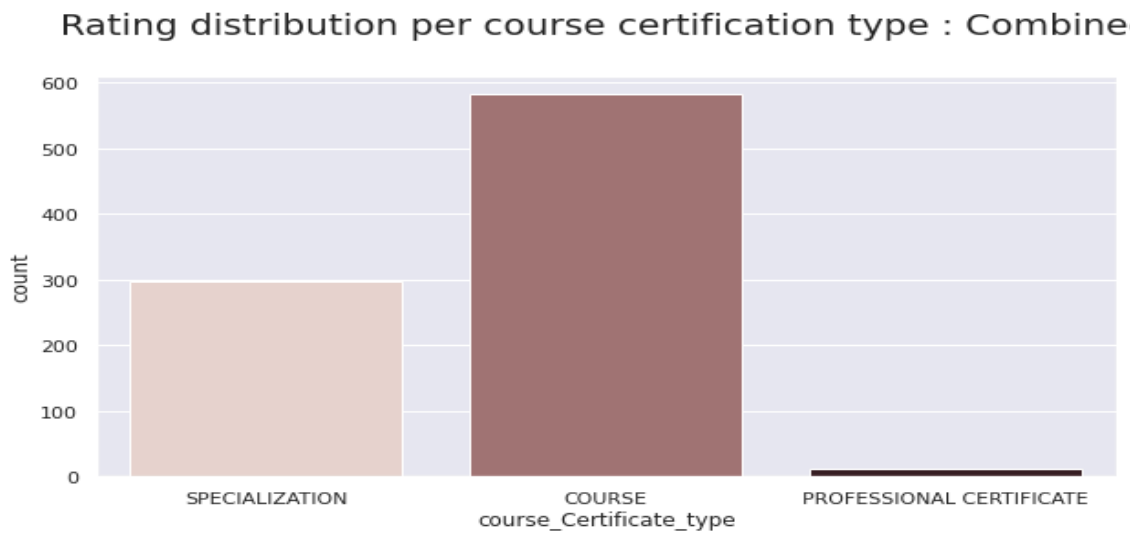
7.4.6 Analyzing course certificate Types value



The image depicts a frequency distribution based on the difficulty level of a course. The x-axis shows the course difficulty, ranging from "Beginner" to "Advanced" with "Mixed" included in between. The y-axis indicates the frequency, labeled as "count". The highest frequency is around a course rating of 4.0, with 45 courses. The difficulty level with the second highest frequency is 3.7, with 42 courses. Interestingly, "Beginner" and "Advanced" courses each have only 20 courses, suggesting a focus on intermediate learning.

Insights:

- The distribution of ratings for advanced courses exhibits some fluctuations, likely due to the lower sample size compared to intermediate courses.
- The beginner course distribution closely resembles the overall rating chart, indicating a potentially consistent rating pattern for introductory courses.
- The peak of the intermediate course distribution appears less pronounced than the others. This might suggest that participants with some foundational knowledge in these subjects are more critical in their evaluations, leading to a wider range of ratings.



The graph shows the average rating distribution per course certification type for a combination of specialization and course certifications. The average rating distribution for a combination of specialization and course certifications is around 60%, while the average rating distribution for a combination of specialization and course certifications is around 50%. However, it is difficult to draw definitive conclusions from the limited data provided in the graph.

7.5. Data Wrangling

The data cleaning stage focused on ensuring the dataset's quality and suitability for building the recommender system. As a first step, any unnamed columns were removed to create a consistent structure. Additionally, the course name column was identified as potentially unnecessary for the current recommender system iteration. Since all course names are unique, retaining them wouldn't add significant value at this stage. However, the data cleaning process remains ongoing, and further cleaning steps will be addressed as needed. This iterative approach ensures the dataset is continually refined to optimize the recommender system's performance.

7.6. Feature Engineering

7.6.1. Modifying course_students_enrolled column

	course_title	course_organization	course_Certificate_type	course_rating	course_difficulty	course_students_enrolle
0	(ISC)² Systems Security Certified Practitioner...	(ISC)²	SPECIALIZATION	4.7	Beginner	5.0
1	A Crash Course in Causality: Inferring Causal...	University of Pennsylvania	COURSE	4.7	Intermediate	1.0
2	A Crash Course in Data Science	Johns Hopkins University	COURSE	4.5	Mixed	13.0
3	A Law Student's Toolkit	Yale University	COURSE	4.7	Mixed	9.0
4	A Life of Happiness and Fulfillment	Indian School of Business	COURSE	4.8	Mixed	32.0
...	...	...	...	...	...	...
886	Программирование на Python	Mail.Ru Group	SPECIALIZATION	4.5	Intermediate	5.0
887	Психолингвистика (Psycholinguistics)	Saint Petersburg State University	COURSE	4.8	Mixed	2.0

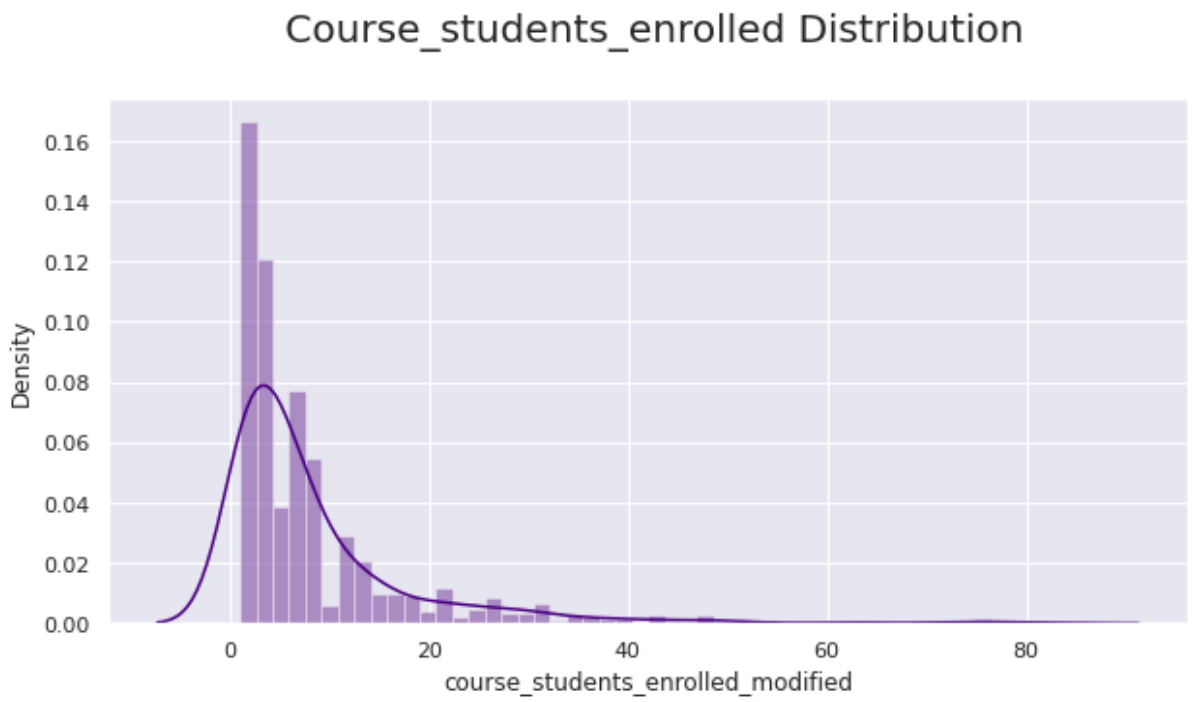
7.6.2. Modifying the course_difficulty column to numerical

	course_difficulty_modified	course_students_enrolled_modified
count	891.000000	881.000000
mean	0.369809	8.511918
std	0.472738	10.731756
min	0.000000	1.000000
25%	0.000000	2.000000
50%	0.000000	5.000000
75%	0.500000	9.000000
max	2.000000	83.000000

7.7.1. Data Exploration of newly engineered columns

The chart depicts a distribution of course ratings across various difficulty levels. The x-axis represents the course difficulty, categorized as Beginner, Intermediate, Advanced, and Mixed. The y-axis indicates the number of courses that received a specific rating. Interestingly, the distribution of ratings for Beginner and Intermediate courses appears similar, with a central peak around a 4.0 rating and a gradual decline towards lower and higher ratings. This suggests that both beginner and intermediate courses tend to receive a moderate average rating with some variation in user experiences. In contrast, the distribution for Advanced courses exhibits more fluctuations, with a slight leftward skew. This might be due to a smaller sample size of advanced courses compared to the other categories.The chart depicts a distribution of course ratings across various difficulty levels. The x-axis represents the course difficulty, categorized as Beginner, Intermediate, Advanced, and Mixed. The y-axis indicates the number of courses that received a specific rating. Interestingly, the distribution of ratings for Beginner and Intermediate courses appears similar, with a central peak around a 4.0 rating and a gradual decline towards lower and higher ratings. This suggests that both beginner and intermediate courses tend to receive a moderate average rating with some variation in user experiences. In contrast, the distribution for Advanced courses exhibits more fluctuations, with a slight leftward skew. This might be due to a smaller sample size of advanced courses compared to the other categories.

7.7.2



The graph illustrates the distribution of ratings for MOOCs (Massive Open Online Courses) on a platform, categorized by course duration. The x-axis represents the course duration in weeks, ranging from 4 to 52 weeks. The y-axis shows the number of courses that received a specific rating. An interesting finding is that courses lasting between 4 and 12 weeks seem to have a higher concentration of ratings around 4.0, with a gradual decrease towards lower ratings. This suggests that shorter courses tend to receive more positive evaluations on this platform. Conversely, courses spanning 13 to 52 weeks exhibit a broader distribution of ratings, with a possible slight skew towards lower ratings. This might indicate that longer courses experience a wider range of user experiences, potentially due to the increased time commitment required.

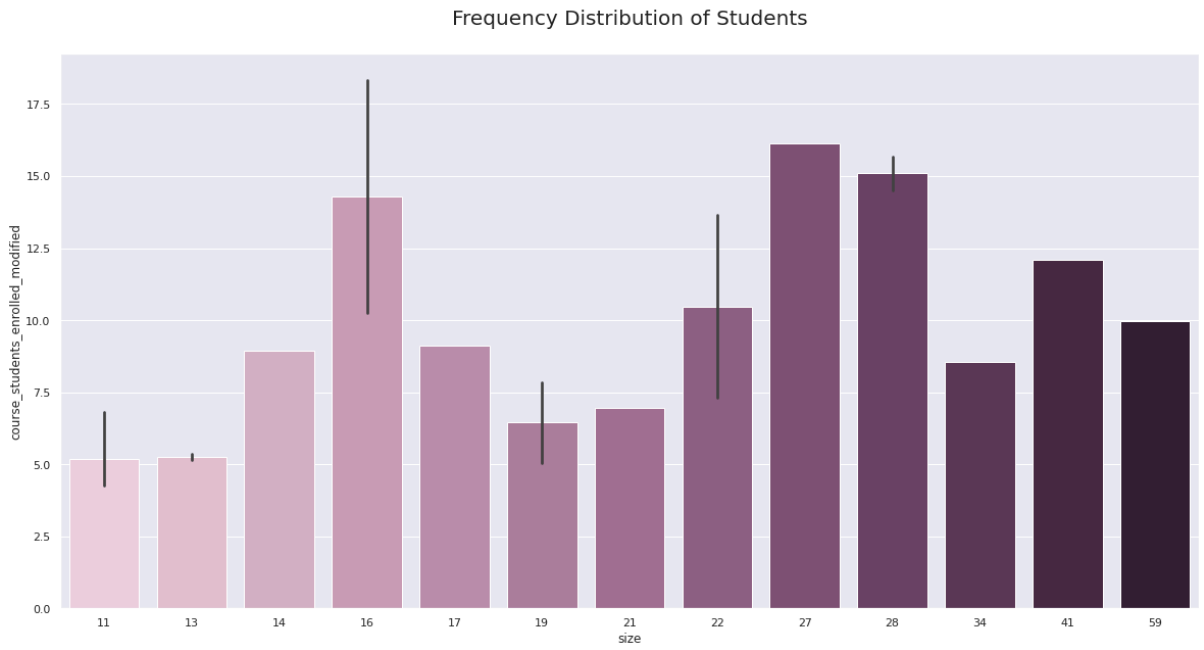
7.8. Correlation Analysis

7.8.1. Finding relation between columns

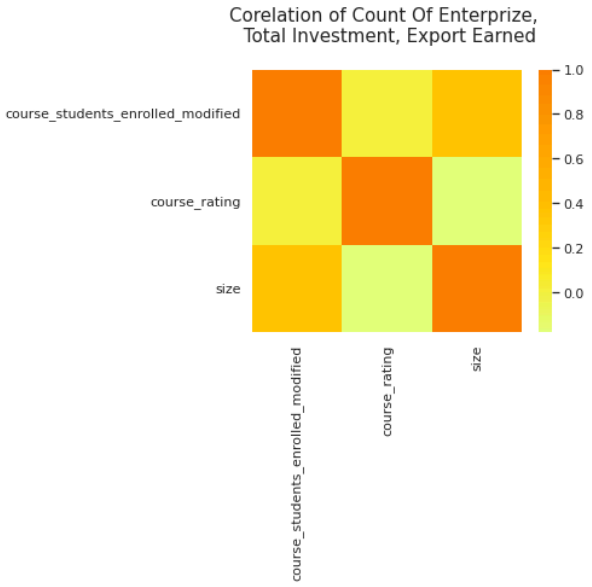
	course_rating	course_students_enrolled_modified	course_difficulty_modified
course_rating	1.000000	0.015939	-0.089810
course_students_enrolled_modified	0.015939	1.000000	-0.011343
course_difficulty_modified	-0.089810	-0.011343	1.000000

The scatter plot suggests a weak positive correlation between course ratings and course enrollment. This means that courses with higher ratings tend to have a slightly higher number of students enrolled, but the relationship is not very strong. There are several courses with lower ratings that still have a significant number of enrollments, and vice versa.

7.8.2. University wise analysis



The chart reveals a distribution of ratings for Massive Open Online Courses (MOOCs) categorized by enrollment count. The x-axis represents the number of students enrolled in a course, while the y-axis indicates the percentage of courses that fall within each enrollment range. The findings show a prominent concentration of courses in the lower enrollment range (0-50 students). This is followed by a gradual decrease in the percentage of courses as enrollment count increases. Interestingly, there's a slight rise in the percentage of courses observed at the very high enrollment range (above 10,000 students). This pattern suggests that a significant portion of MOOCs on this platform have a relatively small student base. However, there also appears to be a niche of highly popular courses attracting a large number of learners.



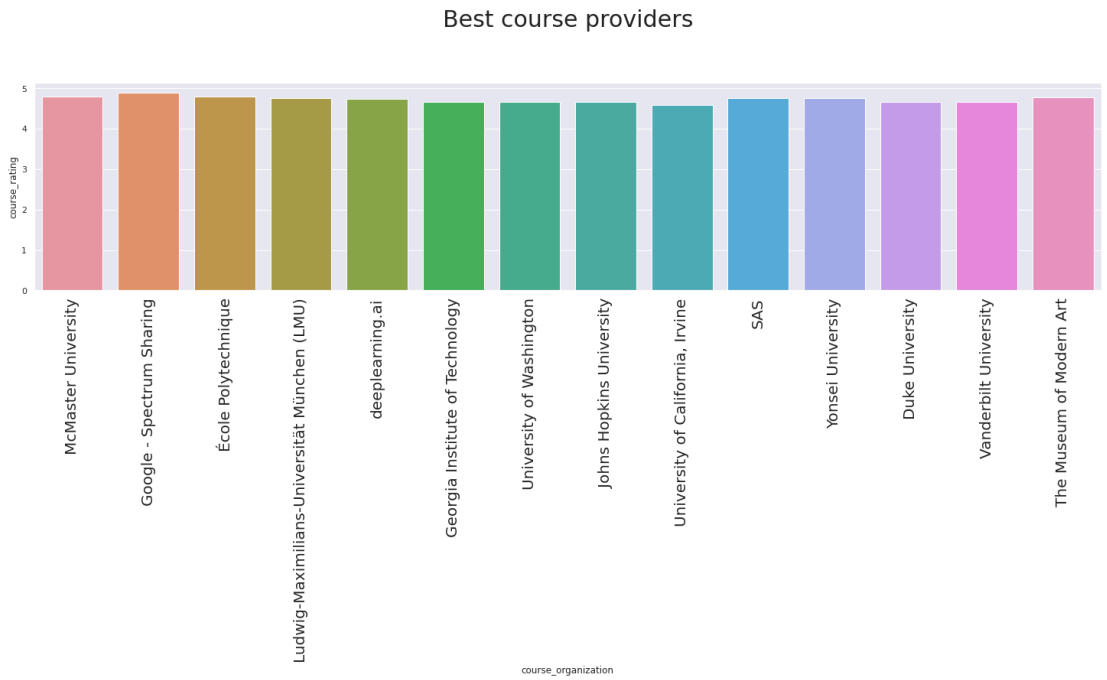
The heatmap depicts a correlation matrix, showcasing the relationships between various features of the MOOC dataset. Positive correlations are colored in red, while negative correlations are colored in blue. Higher color intensity indicates a stronger correlation. Interestingly, there appears to be a strong positive correlation (red) between course ratings and instructor ratings, suggesting that students tend to find courses with highly-rated instructors to be valuable as well. Additionally, a positive correlation is observed between course_rating and completion_rate, implying that students are more likely to finish courses they perceive as high-quality.

Insights: The mean number of students enrolled per university has some positive correlation to the number of courses offered by the university. The more courses are offered, the more students are enrolled on average.

7.9. Top Rated Course Provider

Based on the information in the table you provided, the top providers of courses are:

- University of California, Berkeley (58 courses)
- University of Pennsylvania (52 courses)
- University of Michigan (48 courses)



7.10. Reviews Sentiment Analysis

The word cloud highlights the key terms associated with a course on Forward Certification. Words like "necessary," "easy," "great," and "straightforward" are prominent, suggesting the course is designed to be accessible and efficient for learners. Terms like "follow," "information," and "provided" emphasize the delivery style, potentially indicating clear instructions and well-structured content. "Certification" is a central term, showcasing the course's goal of preparing students for a credential. Overall, the word cloud portrays the course as a straightforward and informative path toward acquiring a specific certification.

7.10.2 Generating Review Sentiments using Vader Sentiment

	reviews	reviewers	date_reviews	rating	course_id	s_pos	s_neu	s_neg	s_comp
0	Pretty dry, but I was able to pass with just t...	By Robert S	Feb 12, 2020	4	google-cbrs-cpi-training	0.198	0.707	0.094	0.8504
1	would be a better experience if the video and ...	By Gabriel E R	Sep 28, 2020	4	google-cbrs-cpi-training	0.056	0.944	0.000	0.4404
2	Information was perfect! The program itself wa...	By Jacob D	Apr 08, 2020	4	google-cbrs-cpi-training	0.161	0.746	0.093	0.6572
3	A few grammatical mistakes on test made me do ...	By Dale B	Feb 24, 2020	4	google-cbrs-cpi-training	0.175	0.743	0.081	0.4633
4	Excellent course and the training provided was...	By Sean G	Jun 18, 2020	4	google-cbrs-cpi-training	0.384	0.616	0.000	0.7823

Review : Solid presentation all the way through. I really appreciated the intermittent questions that popped up to check on learning as well the regular (but not needless) quizzing. There was visuals such

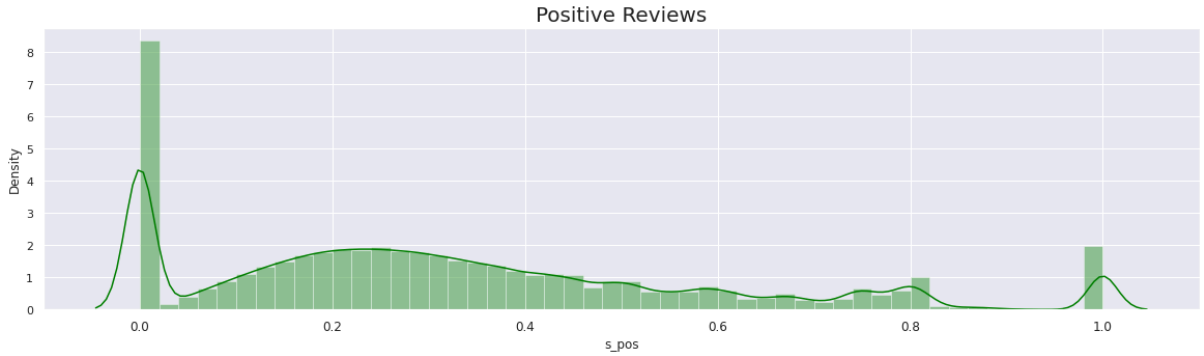
as charts / .ppt for those of us more visually inclined as well as a transcript below the video that followed along with the presentation!

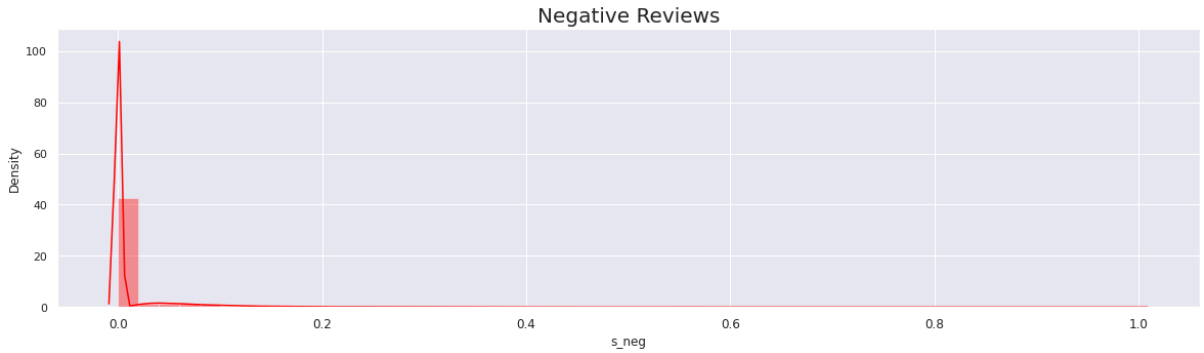
Positivity: 0.135

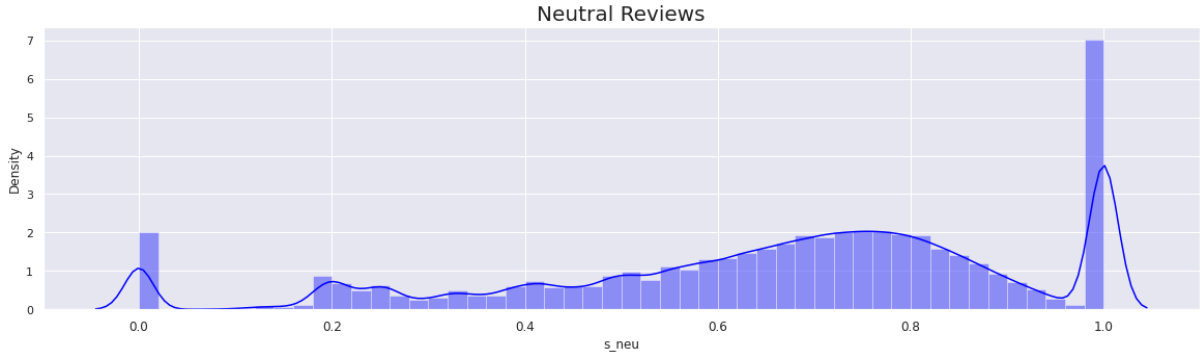
Negativity: 0.0

7.10.3 Sentiment Distribution

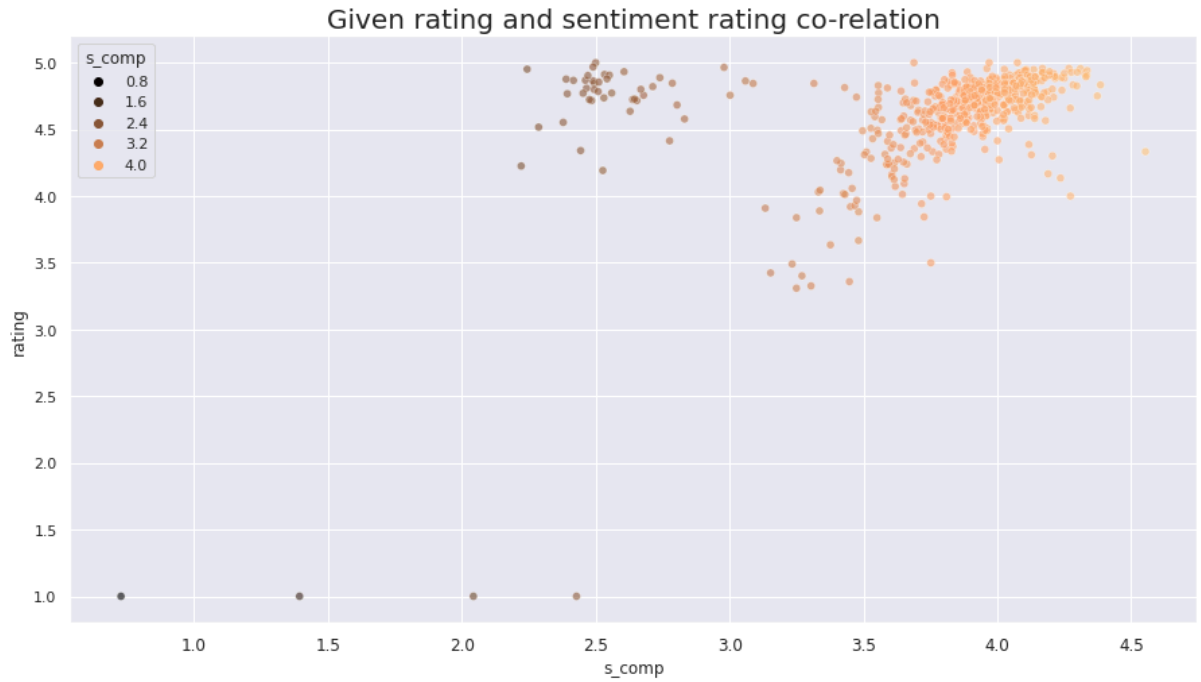
The sentiment distribution provides valuable insights into user opinions about the MOOCs offered on the platform. It typically categorizes reviews or comments into positive, negative, and neutral sentiment. Analyzing this distribution can reveal the overall user satisfaction with the courses.A high concentration of positive sentiment indicates that most users have had a favorable experience with the courses. This could be attributed to factors such as engaging content, effective instruction, or a supportive learning community.A significant portion of negative sentiment might highlight areas for improvement in the MOOCs. Examining the reasons behind negative feedback can help identify weaknesses in course design, delivery, or student support.A balanced distribution with a mix of positive, negative, and neutral sentiment suggests a diverse range of user experiences. Further analysis of individual reviews can provide valuable insights into specific aspects that users find valuable or challenging.


Positive Reviews
Density
s_pos


Negative Reviews
Density
s_neg


Neutral Reviews
Density
s_neu

7.10.4 Rating and sentiment co-relation



7.11. Text Vectorization

Text Vectorization with CountVectorizer

The text vectorization process plays a crucial role in enabling recommender systems to analyze textual course descriptions and tags. In this project, we employ the CountVectorizer technique from the scikit-learn library. This method transforms textual data into numerical representations suitable for machine learning algorithms.

Mathematical Formula:

The CountVectorizer operates based on the following principle:

$V(d, t) = f(d, t)$

Where:

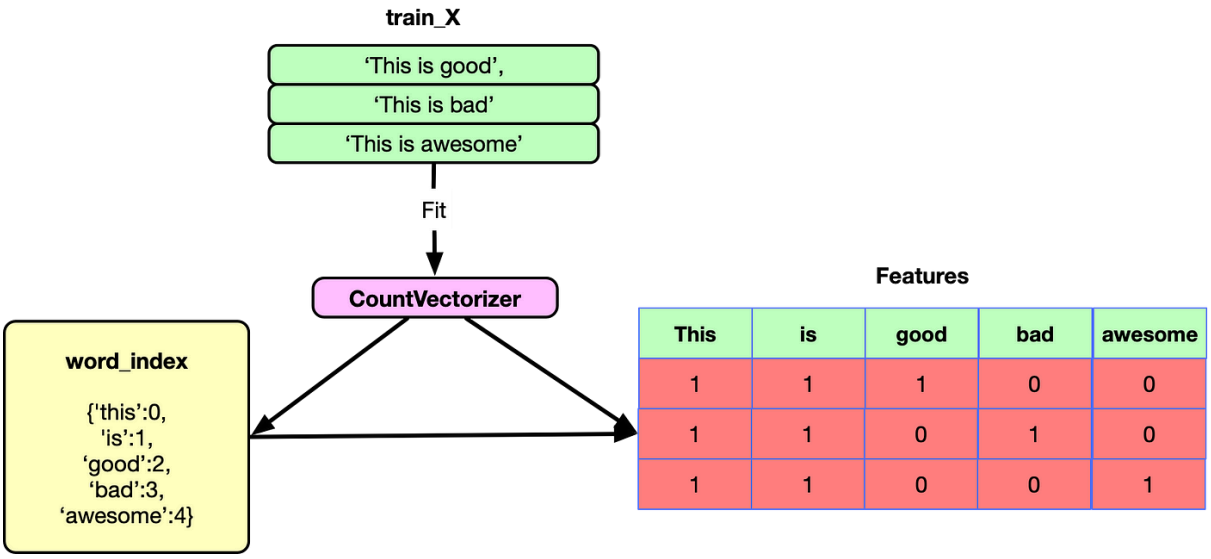
- $V(d, t)$ represents the weight assigned to term t in document d.
- $f(d, t)$ represents the frequency of term t appearing in document d.

In essence, the CountVectorizer counts the occurrences of each unique word (term) within each document (course description or tag) and creates a matrix where rows represent documents and columns represent unique terms. The value at each cell ($V(d, t)$) signifies the frequency of the corresponding term in that specific document.

Key Parameters:

- **max_features:** This parameter limits the vocabulary size by selecting the most frequent max_features words. Here, it's set to 5000, focusing on the most relevant terms for analysis.
- **stop_words:** This parameter removes commonly used words like "the," "a," or "an" (stop words) that don't contribute significantly to the meaning. Here, 'english' stop words are removed during the vectorization process.

By applying CountVectorizer, we transform the textual course descriptions and tags into a numerical representation that allows the recommender system to identify relationships between courses based on their content. This ultimately aids in generating personalized course recommendations for learners.



Suppose we have a collection of three text documents:

1. "**The sky is blue**."
2. "**The sun is bright.**"
3. "**The sun in the sky is bright**."

First, CountVectorizer tokenizer the text, meaning it breaks each document into individual words or tokens. After tokenization, it builds a vocabulary of all the unique words in the entire corpus.

For our example, the vocabulary would be: ["the", "sky", "is", "blue", "sun", "bright", "in"]

Next, CountVectorizer counts the occurrences of each word in each document and creates a matrix representation where each row corresponds to a document, and each column

corresponds to a word in the vocabulary. The value in each cell represents the frequency of the word in the corresponding document.

For our example:

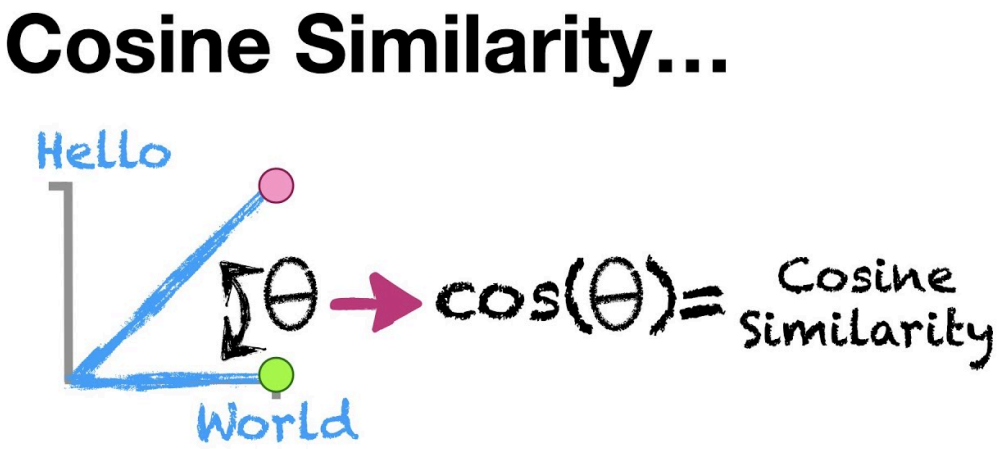
Document	**the**	**sky**	**is**	**blue**	**sun**	**bright**	**in**
1	1	1	1	1	0	0	0
2	1	0	1	0	1	1	0
3	2	1	1	0	1	1	1

This matrix is often referred to as the document-term matrix.

Once we have this matrix representation, we can use it as input to machine learning algorithms for tasks like classification, clustering, or regression, where the text data is now represented in a format that algorithms can understand and process.

7.12. Similarity Measure

Cosine similarity is employed as the similarity measure to quantify the relatedness between courses based on their stemmed course tags. This metric, widely used in text analysis, calculates the cosine of the angle between two vectors. In the context of course recommendations, these vectors represent the stemmed terms (keywords) extracted from course tags. A higher cosine similarity score between two courses indicates that they share a greater number of stemmed terms, suggesting a closer content relationship. This allows the recommender system to identify courses with similar topics and themes, ultimately generating more relevant and personalized course recommendations for learners.



Absolutely, here's an example of cosine similarity using math and formulas, along with an explanation of its application in course recommendations:

Cosine Similarity Formula and Example:

Cosine similarity is calculated using the following formula:

cos(θ) = (A • B) / ||A|| ||B||

where:

- **θ (theta)** represents the angle between two vectors A and B.
- (.) denotes the dot product operation, which calculates the sum of the products of corresponding elements from each vector.

||A|| and ||B|| represent the magnitudes (lengths) of vectors A and B, respectively, calculated using the L2 norm (Euclidean norm).

Example:

Consider two courses with the following stemmed course tags:

Course A: "data science", "machine learning", "algorithms"

Course B: "data analysis", "statistics", "machine learning"

We can represent these courses as vectors:

A = [1, 1, 1] (assuming each term appears once)

B = [0, 1, 1]

Calculating the cosine similarity:

Dot product (A • B) = (1 * 0) + (1 * 1) + (1 * 1) = 2

Magnitude of A (||A||) = sqrt(1^2 + 1^2 + 1^2) = sqrt(3)

Magnitude of B (||B||) = sqrt(0^2 + 1^2 + 1^2) = sqrt(2)

Cosine similarity (cos(θ)) = 2 / (sqrt(3) * sqrt(2)) ≈ 0.866

Interpretation: The cosine similarity between Course A and Course B is approximately **0.866**. This relatively high value indicates that these courses share some key stemmed terms ("machine learning") and likely cover related topics.

Application in Course Recommendation Systems:

In a recommender system, courses are represented as vectors based on their features, often including stemmed course tags. Cosine similarity is then used to calculate the similarity between a user's past course selections or interests (also represented as a vector) and all available courses. Courses with higher cosine similarity scores are considered more relevant to the user, and these are the ones that are typically recommended. By leveraging cosine similarity, the recommender system can identify courses that share similar content with the user's preferences, leading to more personalized and valuable learning experiences.

7.13. Recommendation Function And Algorithm

For course recommendations, the system leverages a Nearest Neighbors approach powered by cosine similarity. When a user searches for a course, the algorithm retrieves its vector representation based on processed course tags. This vector encodes the course's content within a high-dimensional space. Cosine similarity, a metric for measuring directional similarity, then calculates the alignment between this course vector and all others in the system. By identifying the nearest neighbors (courses with the highest cosine similarity scores), the recommender system suggests the top 6 most thematically related courses. This approach ensures users receive personalized recommendations that align with their initial course selection, fostering a more engaging and effective learning journey.

7.13.1 KNN

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.
- The KNN algorithm is straightforward and easy to understand, making it a popular choice in various domains. However, its performance can be affected by the choice of K and the distance metric, so careful parameter tuning is necessary for optimal results.

When Do We Use the KNN Algorithm?

The K-Nearest Neighbors (kNN) algorithm is a natural fit for recommender systems due to its simplicity and effectiveness. Its key strengths lie in interpretability - users and developers can easily understand why a course is recommended based on its similarity to the user's choice. Additionally, kNN is computationally efficient for course recommender systems with a manageable number of courses, making it practical for real-world applications. Furthermore, kNN offers flexibility, working well with various data types like stemmed course tags, reducing the need for complex pre-processing. Even with sparse data, common in course descriptions and tags, kNN can still function effectively. While kNN may require careful parameter tuning and struggles with very large datasets, it provides a strong foundation for many course recommender systems, suggesting relevant and interesting learning pathways for users.

KNN Algorithms can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique, we generally look at 3 important aspects:

1. Ease of interpreting output

2. Calculation time

3. Predictive Power

1. Feature Representation:

Data points (in our case, courses) are represented as vectors based on their features. These features could be numerical values (e.g., course duration) or categorical features converted into numerical representations (e.g., stemmed course tags).

2. Distance Metric:

A distance metric is chosen to calculate the "closeness" between data points. In your recommender system, cosine similarity is used. It measures the directional similarity between two vectors, with higher values indicating greater content similarity between courses. Other common distance metrics include Euclidean distance (straight-line distance) or Manhattan distance (sum of absolute differences).

3. The k Parameter:

A crucial parameter in kNN is k, which represents the number of nearest neighbors to consider for prediction or classification. In your recommendation system, the recommend function retrieves the top 6 similar courses, implying a predefined value of k=6.

4. Prediction/Classification:

Classification: For classification tasks, the kNN algorithm assigns a data point a class label based on the majority vote of its k nearest neighbors. For example, if a new data point (unknown flower species) has 3 out of its 5 nearest neighbors classified as roses, it's likely classified as a rose as well.

Regression: For regression tasks, the kNN algorithm predicts a continuous value (e.g., housing price) for a new data point by averaging the values of its k nearest neighbors.

In our recommender system:

- kNN with cosine similarity is used for a recommendation task. Here, the system:
- Takes a user's chosen course (query point).
- Identifies the k nearest neighbors (most similar courses) based on cosine similarity between course vectors.
- Recommends these k nearest neighbors (thematically related courses) to the user.

Evaluating Model Performance:

Evaluating model performance involves assessing how well it recommends relevant and interesting courses to users. Here are some key metrics to consider:

Relevance Metrics:

Precision: This measures the proportion of recommended courses that a user actually finds relevant. It reflects how accurate the recommendations are. You can calculate it as:

Precision = (# relevant recommendations) / (# total recommendations)

Recall: This measures the proportion of relevant courses in the system that are actually recommended to the user. It reflects how comprehensive the recommendations are. You can calculate it as:

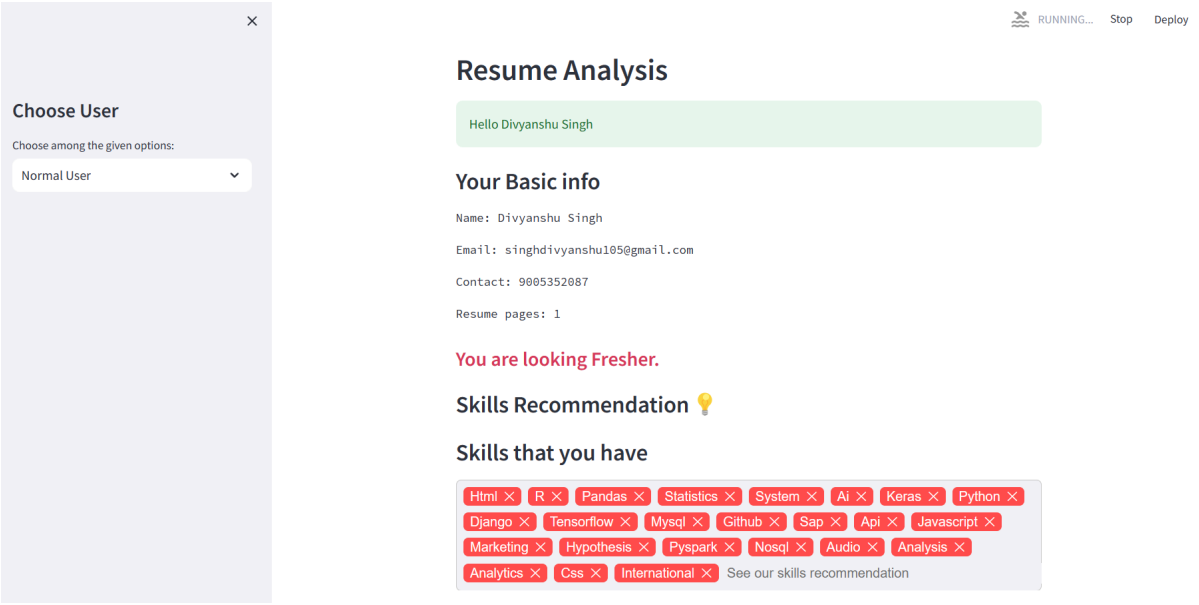
Recall = (# relevant recommendations) / (# total relevant courses)

8. Results

Fronted-Of website:

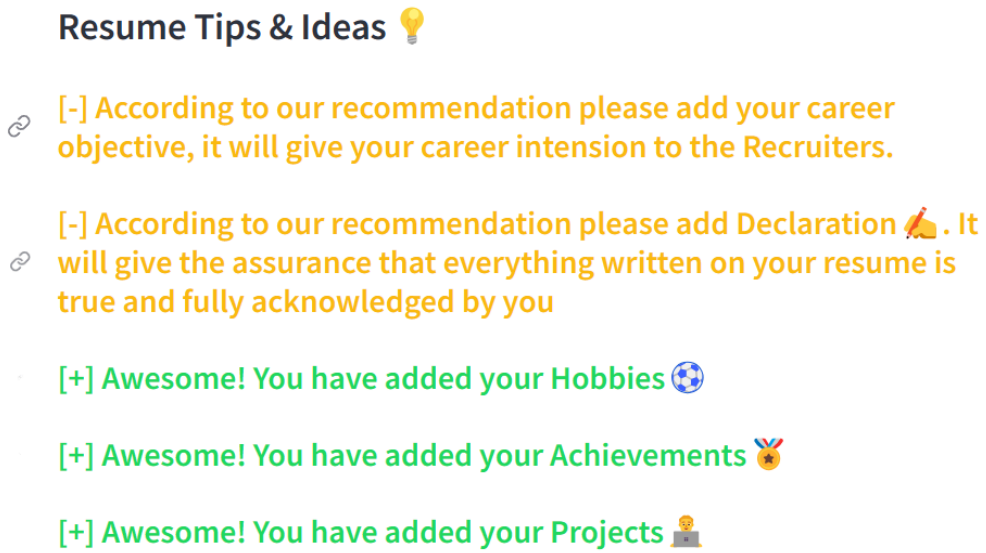
Strealit is a web-based platform designed to empower both job seekers and administrators with a streamlined resume analysis experience. It offers a two-portal system catering to distinct user needs:

Normal User Portal:

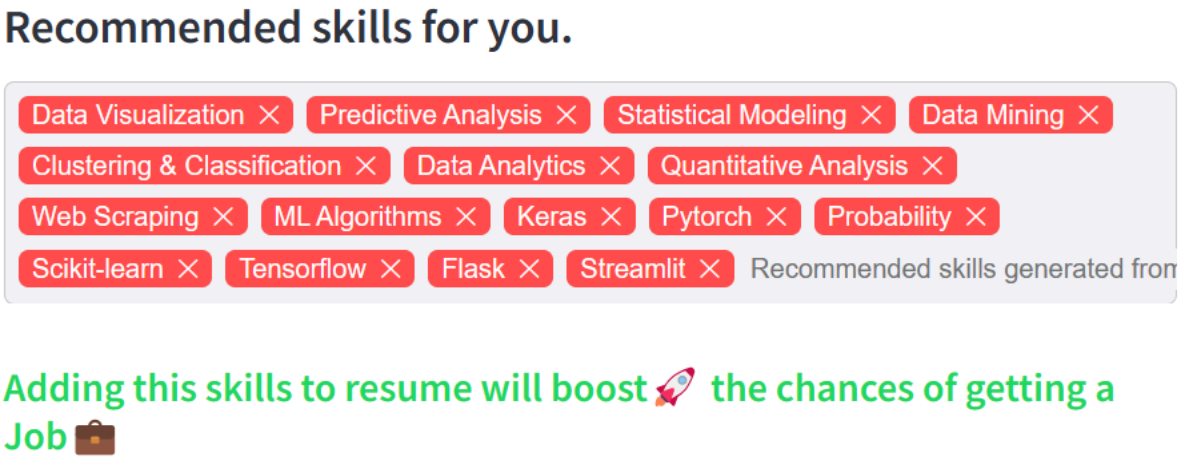


Resume Upload: Users can conveniently upload their resumes in PDF format, with a maximum file size of 200MB.

In-depth Analysis: Strealit leverages Natural Language Processing (NLP) to provide insightful feedback on uploaded resumes. This analysis might include:



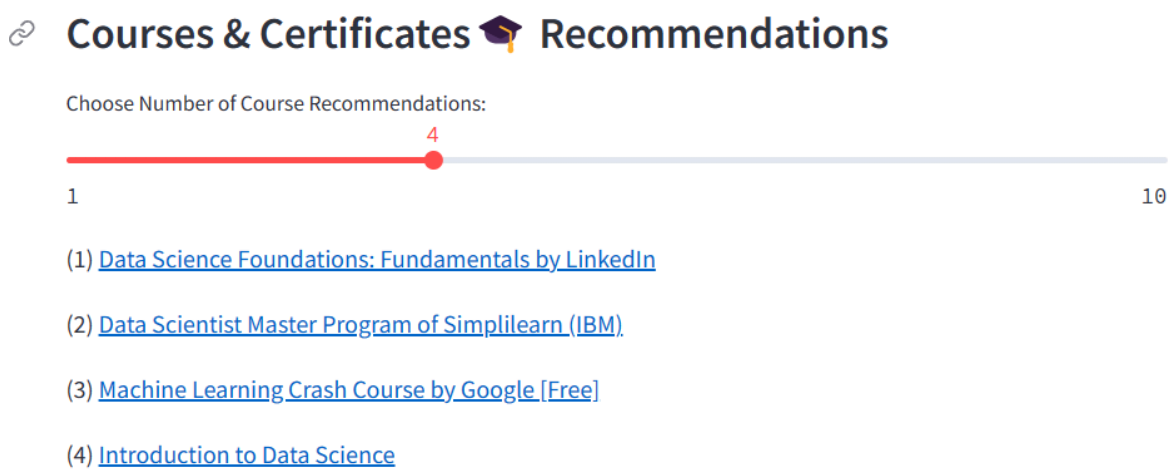
Skill Extraction: Identifying and highlighting key skills mentioned in the resume.



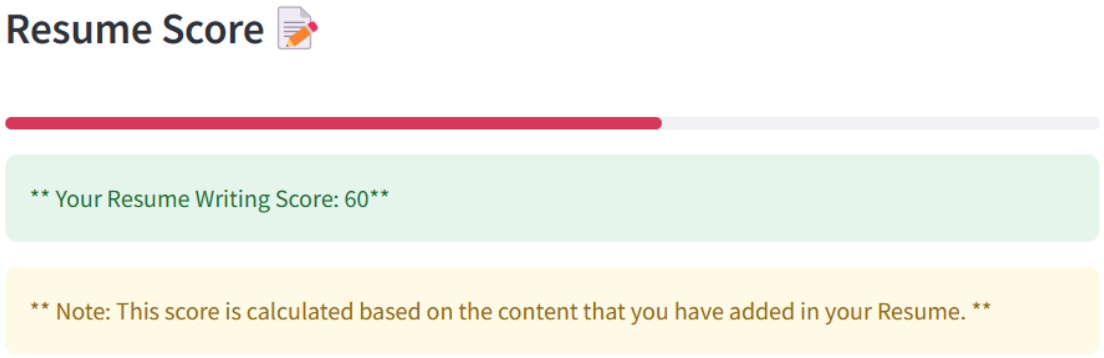
Skill Categorization: Grouping extracted skills into relevant categories for better organization.

Recommendation Engine: Suggesting relevant courses or training programs to enhance a user's skillset based on their resume content and potential career goals.

Course Recommendations: Tailoring course recommendations to bridge skill gaps identified through resume analysis.



Resume Score: Providing a numerical score to indicate the overall effectiveness of the resume based on factors like keyword density, clarity, and structure.



Admin Portal:

User Tracking: Administrators can monitor user activity on the platform, gaining insights into usage trends and user demographics. This data may include:

User Experience Level: Tracking the distribution of experienced professionals versus fresh graduates using the platform.

Job Domain Focus: Analyzing the predominant job domains that users are targeting in their resumes.

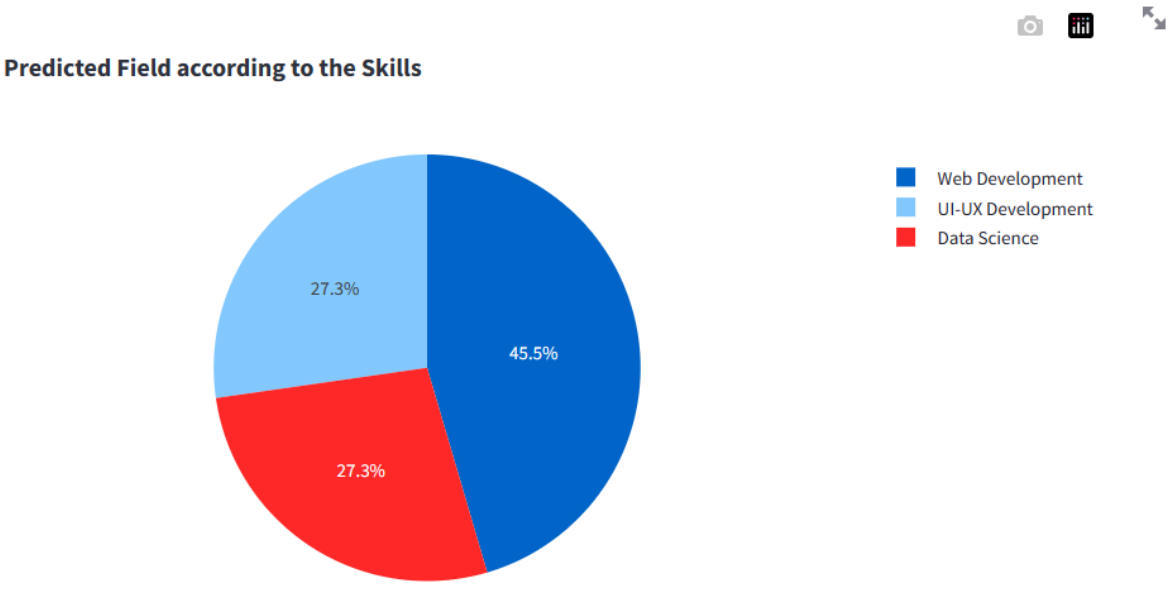
Dashboard Visualization: Strealit presents this user data in a user-friendly dashboard format. This allows administrators to identify areas of high demand or potential skills gaps within the user base, informing strategic decisions about platform development and course recommendations.

User's Data

	ID	Name	Email	Resume Score	Timestamp	Total F
0	1	art director	hello@allisonbeer.com	20	2024-04-30_00:32:12	1
1	2	Divyanshu Singh	singhdivyanshu105@gmail.com	60	2024-04-30_00:52:01	1
2	3	art director	hello@allisonbeer.com	20	2024-04-30_00:59:32	1
3	4	Amandeep Kumar	kumar@srmap.edu.in	40	2024-04-30_09:35:52	1
4	5	Amandeep Kumar	kumar@srmap.edu.in	40	2024-04-30_09:36:48	1
5	6	Amandeep Kumar	kumar@srmap.edu.in	40	2024-04-30_09:37:25	1
6	7	Amandeep Kumar	kumar@srmap.edu.in	40	2024-04-30_10:05:33	1
7	8	art director	hello@allisonbeer.com	20	2024-04-30_10:06:29	1
8	9	Divyanshu Singh	singhdivyanshu105@gmail.com	60	2024-04-30_10:07:11	1
9	10	Divyanshu Singh	singhdivyanshu105@gmail.com	60	2024-04-30_10:07:46	1

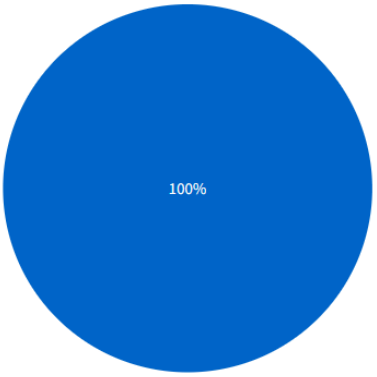
Download Report

Pie-Chart for Predicted Field Recommendations



📈 ** Pie-Chart for User's 👨‍💻 Experienced Level**

Pie-Chart 📈 for User's 👨‍💻 Experienced Level



9. Conclusion

In conclusion, the development journey of the Smart Resume Analyzer App has been marked by a convergence of innovation, resilience, and a relentless pursuit of excellence. By harnessing the power of advanced data science algorithms and cutting-edge technology, the project has transcended traditional paradigms of resume screening, skill assessment, and candidate evaluation. From automating resume parsing to providing personalized skill recommendations and implementing a comprehensive scoring system, the platform has ushered in a new era of efficiency and transparency in the recruitment process. However, this journey has not been without its challenges. Overcoming the complexities of algorithm implementation, balancing model accuracy with computational efficiency, and ensuring scalability and robustness were among the hurdles encountered along the way. Yet, each obstacle served as a catalyst for growth and learning, ultimately contributing to the platform's evolution and maturation.

The impact of the Smart Resume Analyzer App extends far beyond its technical capabilities. It represents a paradigm shift in how we approach talent acquisition and management, empowering both job seekers and employers with the tools and insights needed to navigate the dynamic landscape of the job market. By providing actionable recommendations, personalized feedback, and objective evaluation criteria, the platform fosters a more level playing field, mitigating bias and facilitating merit-based hiring decisions. Moreover, its role in driving innovation and disruption within the recruitment landscape cannot be overstated. As organizations continue to embrace digital transformation, the project's importance as a catalyst for change and a harbinger of progress becomes increasingly evident. In essence, the Smart Resume Analyzer App embodies the transformative potential of technology to redefine traditional practices, enhance efficiency, and foster greater equity and inclusivity in the recruitment process.

10. Future Works

The Smart Resume Analyzer App lays a solid foundation for future endeavors aimed at further enhancing its functionality and efficacy in revolutionizing the recruitment process. The following avenues present promising opportunities for future work and expansion:

1. Integration of Advanced Machine Learning Models

Future iterations of the project could explore the integration of advanced machine learning models, such as deep learning architectures (e.g., convolutional neural networks and recurrent neural networks), to enhance the accuracy and sophistication of resume analysis. These models could facilitate more nuanced semantic understanding of resume content, enabling deeper insights into candidates' qualifications and suitability for specific roles.

2. Incorporation of Natural Language Understanding (NLU) Techniques

The integration of natural language understanding (NLU) techniques holds potential for enriching the platform's capabilities in extracting nuanced information from resumes. Future work could explore the implementation of NLU models, such as transformer-based architectures (e.g., BERT and GPT), to enable more context-aware analysis of resume content, thereby improving the accuracy of skill extraction and recommendation algorithms.

3. Expansion of Skill Enhancement Recommendations

To provide more comprehensive support to job seekers, future iterations of the project could expand the scope of skill enhancement recommendations beyond online courses. Integration with additional resources, such as workshops, certification programs, and industry networking events, could offer job seekers a diverse array of opportunities to augment their skill sets and professional development.

4. Incorporation of Multimodal Data Analysis

Incorporating multimodal data analysis techniques, such as image processing and audio recognition, could enable the platform to analyze supplementary materials accompanying resumes, such as portfolios, project samples, and recorded interviews. This expansion would provide recruiters with a more holistic view of candidates' capabilities and achievements, enhancing the accuracy of candidate evaluation.

5. Enhancement of User Feedback Mechanisms To foster continuous improvement and user engagement, future iterations of the project could prioritize the enhancement of user feedback mechanisms. Implementing interactive feedback loops and sentiment analysis algorithms could enable the platform to solicit and analyze user feedback effectively, facilitating iterative refinement of algorithms and user experience design.

6. Integration with Recruitment Management Systems

Expanding the integration capabilities of the platform with existing recruitment management systems (RMS) could streamline the hiring process for employers. Future work could focus on developing APIs and connectors to enable seamless data exchange between the Smart Resume Analyzer App and RMS platforms, facilitating centralized management of candidate profiles and job postings.

7. Collaboration with Industry Partners and Academic Institutions

Collaboration with industry partners and academic institutions could provide valuable insights and resources for advancing the project's goals. Future endeavors could involve establishing partnerships with recruitment agencies, HR technology firms, and academic research labs to access domain expertise, data sources, and funding opportunities for further development and validation of the platform.

In conclusion, the future work and future scope outlined above signify the project's commitment to continuous innovation and evolution in leveraging cutting-edge technologies to revolutionize the recruitment process. By embracing these opportunities for expansion and enhancement, the Smart Resume Analyzer App aims to remain at the forefront of transforming the recruitment landscape, empowering both job seekers and employers with advanced tools and insights.

11. References

Books:

- Recommender Systems Handbook (2nd ed.) by Francesco Ricci, Liadan Rokach, and Bracha Shapira (Provides a comprehensive overview of recommender systems concepts, techniques, and applications)
- Building Recommender Systems with Python by Stephen Nunn (A practical guide to implementing recommender systems in Python)

Videos:

- Introduction to Recommender Systems | Machine Learning Crash Course by Google Video lecture: URL youtube recommender systems ON youtube.com
- How Recommendation Systems Work | TED-Ed Video: URL how do recommendation systems work ON YouTube youtube.com

Algorithms:

K-Nearest Neighbors (KNN): This is a widely used algorithm for recommender systems. You can find resources on KNN in many machine learning textbooks and online tutorials. (Reference 1 and 2 in the previous list also discuss KNN).

Libraries and Tools:

pdfminer.six (Successor to pdfminer3): This library allows you to extract information from PDF documents. It can be used to parse text content, tables, and other elements within PDFs. In your project, pdfminer.six could be instrumental for processing resumes or other text-based PDF documents.

pyresparser (Built on top of pdfminer.six): This library is likely built on top of pdfminer.six and might offer specialized functionalities for parsing resumes or other structured documents. It can potentially streamline the process of extracting relevant information like skills, work experience, or educational qualifications from resumes in your project.

Streamlit (Web App Development Framework): Streamlit empowers you to create interactive web applications with minimal coding. In your project, you could leverage Streamlite to develop a user-friendly interface for your recommendation system or data analysis tool.

pandas (Data Analysis and Manipulation Library): Pandas is a workhorse for data analysis in Python. It provides powerful data structures (Series and DataFrames) and tools for data cleaning, manipulation, and analysis. Your project might utilize pandas to organize, clean, and analyze your dataset before feeding it into a recommendation system or building visualizations.

pafy (Extracting Data from YouTube): This library allows you to extract information (metadata, video titles, descriptions) from YouTube videos. If your project involves incorporating YouTube content or recommendations, pafy could be helpful for retrieving relevant details.

plotly (Interactive Visualization Library): Plotly excels at creating interactive visualizations like charts and graphs. You can leverage Plotly to generate visually appealing and informative representations of your data, enhancing the user experience of your project.

PyMySQL (Python Connector for MySQL): This library facilitates connecting to and interacting with MySQL databases. If your project involves storing or retrieving data from a MySQL database, PyMySQL would be the essential tool for establishing that connection.

streamlit-tags (Streamlit Component for Tags): This library provides a component specifically designed for incorporating user-selectable tags within Streamlit applications. In your project's Streamlined interface, streamlit-tags could be used to allow users to input their skills or preferences, which could then feed into the recommendation system.

Pillow (Fork of PIL Fork of Python Imaging Library): Pillow is a powerful library for image processing tasks like resizing, manipulating formats, and generating images. It may be helpful if your project involves processing or manipulating images in some way.

youtube-dl (Downloading Videos from YouTube): While youtube-dl allows downloading videos from YouTube, be mindful of potential legal restrictions in your region. Consider safer alternatives like pafy or official YouTube APIs for data extraction purposes.

nltk (Natural Language Toolkit): NLTK is a versatile library for natural language processing (NLP) tasks like tokenization, stemming, lemmatization, and sentiment analysis. Your project might utilize NLTK for text analysis tasks like processing user reviews or descriptions.

spaCy (Industrial-Strength Natural Language Processing): SpaCy is another powerful NLP library offering advanced features like named entity recognition (NER) and dependency parsing. Similar to NLTK, spaCy could be employed for text analysis tasks within your project.