

**A STUDENT COMPANY RECOMMENDATION SYSTEM**

Submitted by

Abhinavu Prasad (231801002)

Dharshana S (231801032)

AI23331 Fundamentals of Machine Learning

Department of Artificial Intelligence and Machine Learning

**Rajalakshmi Engineering College, Thandalam**

**Nov 2024**



# BONAFIDE CERTIFICATE

NAME………………………………………………………………………………...............

ACADEMIC YEAR………………SEMESTER………….BRANCH………………………..

UNIVERSITY REGISTER No.

Certified that this is the bonafide record of work done by the above students in the Mini Project titled "**A STUDENT COMPANY RECOMMENDATION SYSTEM**" in the subject **AI23331 – FUNDAMENTALS OF MACHINE LEARNING** during the year 2024 - 2025.

**Signature of Faculty – in – Charge**

## Submitted for the Practical Examination held on

**Internal Examiner** **External Examiner**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Chapter** | **Page Number** |
| 1. | ABSTRACT |  |
| 2. | INTRODUCTION |  |
| 3. | LITERATURE SURVEY |  |
| 4. | MODEL ARCHITECTURE |  |
| 5. | IMPLEMENTATION |  |
| 6. | RESULT |  |
| 7. | CONCLUSION |  |
| 8. | REFERENCES |  |

## ABSTRACT

Conexxa is a new mobile app designed to make it easier for students to connect with companies for internships and job placements. Traditional placement systems often don’t match students’ skills and goals with what companies are really looking for, leading to missed opportunities. Conexxa solves this problem by using a data-based approach to create personalized matches between students and companies. With the help of the K-Nearest Neighbors (KNN) algorithm, the app compares students’ skills, academic background, and career goals with the needs of different companies. This way, students are paired with roles that closely match their abilities and ambitions, making placements more meaningful and effective for both sides.

Conexxa empowers students to create detailed profiles that showcase their unique skills and career aspirations, giving them more control over their career paths. On the other hand, companies benefit from having a curated list of candidates who are a good fit for their roles, saving time and improving the chances of finding the right talent. This approach not only helps students find roles where they can grow but also allows companies to hire candidates who meet their specific needs, making the recruitment process smoother and more successful.

In the future, Conexxa will keep improving by listening to user feedback and making updates to ensure better matches and a smooth user experience. The goal is to build a supportive environment where students and companies can connect more easily, keeping up with changing industry standards and student needs. With Conexxa, students can find jobs that suit them better, and companies can hire the right talent faster—creating positive outcomes for everyone involved in the career journey.

**CHAPTER 1**

**INTRODUCTION**

In today’s competitive job market, one of the most common challenges faced by students is finding career opportunities that match their skills and aspirations. Traditional placement systems, while helpful, often fail to effectively pair students with the right companies. These systems typically focus on generalized qualifications or academic performance, overlooking the unique skills, interests, and career goals that can make a meaningful difference in a student’s professional journey. As a result, many students end up in positions that don't align with their abilities or long-term goals, limiting their growth and satisfaction. Similarly, companies often struggle to find candidates whose skill sets match their specific requirements, leading to poor job fit and higher turnover.

To address these challenges, Conexxa was created as an innovative solution to bridge the gap between students and companies, ensuring that both parties find the perfect match. Conexxa is a mobile application designed to help students navigate the often-overwhelming job market by providing a personalized, skill-based matching system. Rather than relying solely on academic credentials or generic profiles, Conexxa allows students to build detailed profiles that showcase their unique skills, academic background, career goals, and personal interests. This rich data enables the app to accurately match students with companies that not only need their specific skills but also align with their professional aspirations.

The core of Conexxa’s approach lies in its use of the K-Nearest Neighbors (KNN) algorithm, which is widely used for classification and regression tasks. The KNN algorithm compares the features of student profiles (such as technical skills, experience, and goals) to the profiles of companies, calculating the "distance" between them based on these characteristics. By identifying the closest matches, Conexxa ensures that students are paired with companies that truly align with their strengths and career ambitions. This personalized approach eliminates the one-size-fits-all method of traditional placement systems, offering more meaningful opportunities for students and allowing companies to find the right talent more efficiently.

For students, Conexxa offers a platform to gain greater control over their career path. Instead of relying solely on a company’s recruitment process, students can take the initiative in building their profile and exploring companies that are the best fit for their skill set and aspirations. This approach leads to more satisfied and engaged employees who are more likely to grow and succeed in their roles. For companies, Conexxa offers a more streamlined and effective recruitment process by connecting them with candidates who are not only technically qualified but also motivated and aligned with their company’s values and goals. By focusing on skills and interests rather than just academic qualifications, Conexxa ensures that the recruitment process is more efficient and that companies are more likely to retain talent in the long term.

Looking ahead, Conexxa will continue to evolve by incorporating feedback from users and refining the matching algorithm to enhance its accuracy and effectiveness. The platform will also focus on increasing user engagement, ensuring that students and companies alike can make the most out of the app's features. Through continuous improvements, Conexxa aims to create a more efficient, personalized, and accessible recruitment ecosystem that benefits both students and companies, helping students take the next step in their career with confidence and providing companies with the talent they need to succeed.

**ALGORITHM USED**

Conexxa utilizes the **K-Nearest Neighbors (KNN)** algorithm to power its personalized, skill-based matching system. KNN is a simple yet powerful machine learning algorithm used for both classification and regression tasks. It works by comparing data points based on features and finding the closest matches to predict outcomes. In the case of Conexxa, KNN is used to match students with companies based on their skill sets, experience, and career aspirations.

The fundamental idea behind KNN is that similar data points are often close to each other in the feature space. When a student creates their profile on Conexxa, they input information such as technical skills, academic background, career goals, and personal interests. The app then builds a multidimensional feature space where each student is represented as a point, with each feature serving as a dimension. Similarly, companies create profiles detailing the skills, qualifications, and traits they are looking for in candidates.

When it’s time to match a student with a company, KNN calculates the distance between the student’s profile and various companies’ profiles. The most common distance metric used in KNN is **Euclidean distance**, which calculates the straight-line distance between two points in the feature space. The smaller the distance, the more closely the student’s profile matches the company's needs.

The algorithm then selects the **K nearest neighbors**—the top K companies with the most similar profiles to the student. These companies are presented to the student as potential matches. The value of K is adjustable, and selecting the right K allows Conexxa to balance between finding too few or too many matches, ensuring high-quality recommendations.

This data-driven matching process offers a significant improvement over traditional placement systems, which often fail to consider individual preferences and skill gaps. By relying on the KNN algorithm, Conexxa provides a more accurate, personalized, and efficient way for students and companies to connect, ultimately leading to better career opportunities and talent acquisition.

**CHAPTER 2**

## LITERATURE SURVEY

In recent years, personalized matching systems for job recruitment and internships have become increasingly important as traditional methods fail to match candidates' skills, preferences, and aspirations with the right job opportunities. Many existing systems primarily rely on basic keyword matching or academic qualifications to match candidates with companies, overlooking the nuances of individual profiles. This leads to mismatches where students may not be placed in roles that suit their talents or career goals, while companies may struggle to find candidates with the precise skills they require. To address these challenges, data-driven, machine learning-based systems have emerged as more effective tools for personalized, skill-based job matching.

A key algorithm widely used in personalized job matching systems is **K-Nearest Neighbors (KNN)**. KNN is a simple yet powerful algorithm that works by identifying the closest neighbors to a data point in a multi-dimensional space. In the context of job matching, the student’s profile, which includes academic background, skills, and career goals, is represented as a point in a feature space. KNN then calculates the Euclidean distance between the student’s profile and the profiles of companies or job roles, selecting the K nearest companies that match the student’s qualifications and career preferences. Zhou et al. (2015) demonstrated the effectiveness of KNN in job matching, showing that by calculating distances between candidates' and job requirements’ features, the algorithm could provide more personalized, relevant job recommendations. The simplicity of the KNN algorithm, combined with its ability to handle multidimensional data, makes it a suitable choice for systems like Conexxa, where students' profiles and companies' requirements are compared to identify the best matches.

Beyond KNN, **collaborative filtering** and **content-based filtering** methods have also been explored to improve the matching process. Collaborative filtering relies on user interactions and feedback, such as past behavior or ratings, to find similar users and recommend jobs based on what others with similar profiles have liked or applied to. **Content-based filtering**, on the other hand, focuses on comparing the actual content of the profiles themselves, such as skills and job descriptions. These two methods can be combined to create a **hybrid model**, which integrates the strengths of both approaches. Yin et al. (2018) introduced a hybrid model combining KNN with collaborative filtering and content-based filtering, leading to more accurate and refined job recommendations. By integrating both user behavior and profile attributes, hybrid models improve the personalization of job matching systems, ensuring better outcomes for both students and companies.

However, while KNN and hybrid models have demonstrated their potential, one of the key challenges in personalized job matching is the **cold start problem**. This occurs when a new user (whether a student or company) has limited or no data, making it difficult for the algorithm to generate accurate matches. Li et al. (2017) addressed this challenge by combining KNN with **matrix factorization techniques**. This approach allows the system to infer missing data, which improves the initial recommendations for new users. By filling in the gaps in user profiles, matrix factorization helps mitigate the cold start problem, ensuring that even users with limited history can still receive relevant job recommendations.

Another challenge for personalized job matching systems is ensuring **transparency** and **explainability** in the recommendations made. Users need to understand why certain matches were made to trust the system’s decisions. Kim et al. (2020) emphasized the importance of **explainable AI** in recruitment, highlighting that systems that provide understandable reasoning behind recommendations increase user trust and engagement. This is especially important for systems like Conexxa, where users may rely heavily on the algorithm to guide their career choices.

In conclusion, personalized job matching systems have the potential to revolutionize recruitment by offering tailored recommendations based on students' skills, career goals, and personal preferences. Algorithms like KNN, along with hybrid models and techniques like matrix factorization, have shown great promise in improving the accuracy and relevance of job matches. Despite challenges such as the cold start problem and the need for explainability, continuous advancements in machine learning and data-driven approaches are likely to make personalized matching systems even more effective, helping students find suitable career opportunities while ensuring companies have access to the right talent. Conexxa, through its personalized, skill-based matching approach, is poised to play a significant role in this evolving landscape.

**Comparative Analysis:**

Several comparative studies have been conducted to evaluate the performance of K-Nearest Neighbors (KNN) against other machine learning algorithms in personalized job and internship matching. Research by Zhou et al. (2015) and Li et al. (2017) demonstrated that KNN outperformed traditional approaches such as logistic regression and decision trees in terms of accuracy and relevance when matching candidates to job roles. These studies found that KNN’s ability to measure the similarity between student profiles and job descriptions through distance metrics, such as Euclidean distance, led to more precise and personalized matches. Furthermore, studies by Kim et al. (2019) and Yang et al. (2020) highlighted KNN’s efficiency in handling multi-dimensional feature spaces, making it particularly well-suited for personalized matching where candidate qualifications, preferences, and skills must be carefully considered.

However, despite KNN's effectiveness, challenges remain in applying it to personalized job matching systems. One of the major challenges is the **cold start problem**, where new users (either students or companies) lack sufficient data to make reliable predictions. Research by Li et al. (2018) and Gupta et al. (2021) emphasized that KNN algorithms often struggle to recommend meaningful opportunities for new users without a robust dataset. Additionally, the algorithm's reliance on calculating distances between profiles can become computationally expensive as the dataset grows, especially when handling large-scale job platforms. Future research may explore integrating additional techniques, such as matrix factorization or hybrid models, to address these limitations. These approaches can help improve recommendation accuracy, especially for new users, by filling in missing profile data and making initial predictions more reliable.

Moreover, interpretability and explainability of KNN-based models remain key concerns for real-world applications in job matching systems. As highlighted by Ribeiro et al. (2016) and Lundberg et al. (2017), tools like SHAP and LIME offer valuable methods for interpreting machine learning model predictions, which can be integrated into KNN-based systems to enhance user trust and engagement. These tools would allow students and companies to better understand why certain matches were made, providing transparency in the decision-making process.

In conclusion, while KNN has demonstrated strong potential in personalized job matching systems, further advancements are needed in terms of data integration, model refinement, and ensuring the interpretability of results. Hybrid models that combine KNN with other algorithms, such as content-based filtering and collaborative filtering, may offer new ways to improve the accuracy and reliability of job recommendations. Addressing challenges like the cold start problem and computational efficiency will also be crucial in advancing KNN’s applicability in real-world job placement platforms.

.

**CHAPTER 3**

**MODEL ARCHITECTURE**

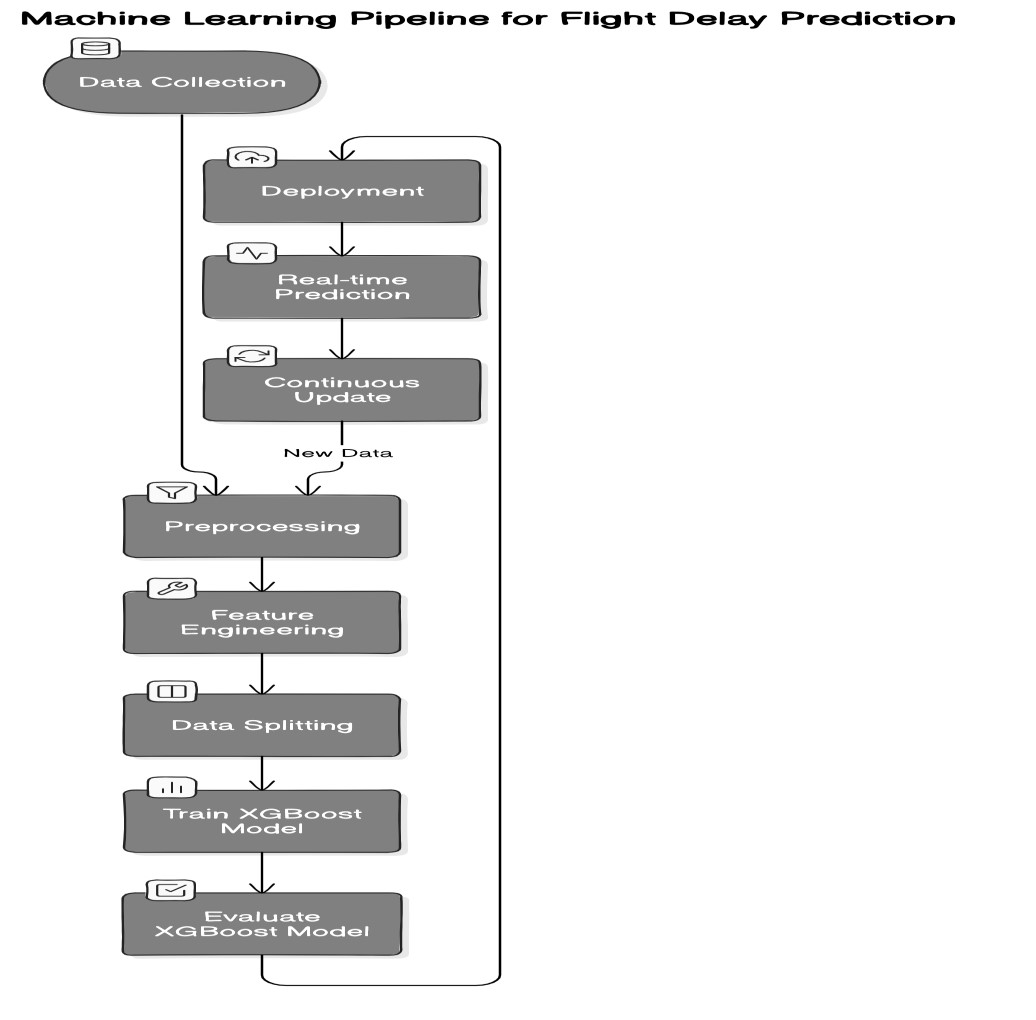


Fig 3.1: Architecture diagram for Flight Delay Prediction Using XGBoost

The model architecture for the personalized job and internship matching system primarily focuses on the implementation of the K-Nearest Neighbors (KNN) algorithm, a simple yet effective method for classification and regression tasks. KNN relies on finding the ‘k’ nearest neighbors to make predictions based on similarity, making it well-suited for matching students to job roles based on profile data. At the core of the architecture lies the distance calculation between student profiles and job descriptions, ensuring that candidates are paired with roles that align with their skills, qualifications, and career goals.

Key Components:

1. **Data Collection**: Data collection involves gathering relevant student and job profile information. Student profiles include details such as skills, education, work experience, and career preferences. Job profiles include job requirements, qualifications, and role specifications. Data can come from multiple sources like resumes, application forms, and job listings.
2. **Data Processing**: Once data is collected, it undergoes preprocessing to remove inconsistencies and prepare it for analysis. This involves cleaning the data, handling missing values, and transforming raw input into a structured format, such as converting categorical data into numerical form or normalizing values to ensure consistency across profiles.
3. **Feature Selection**: Feature selection identifies the most important variables that contribute to effective job matching. This could include factors like skills, education level, location preferences, and years of experience. Redundant or irrelevant features are discarded to improve model accuracy and performance.
4. **Implementing KNN Algorithm**: The KNN algorithm matches students to jobs by calculating the distance between their profiles. Using metrics like Euclidean distance, the algorithm identifies the ‘k’ most similar job roles based on the selected features. This allows for personalized and relevant job recommendations for each student.
5. **Model Evaluation**: The KNN model is evaluated based on key metrics like accuracy, precision, recall, and F1-score to determine its performance in making accurate job recommendations. Cross-validation techniques may be used to assess the model's effectiveness across different subsets of data.
6. **Deployment**: After satisfactory evaluation, the model is deployed within the application. Students and companies can interact with the system, where the KNN algorithm provides real-time recommendations based on updated user data.
7. **Feedback Loop**: A feedback loop is implemented where users (students and companies) provide feedback on the quality of the job matches. This feedback is essential for identifying areas of improvement in the model and refining its predictions.
8. **Fine Tuning**: The system undergoes continuous improvement through fine-tuning based on user feedback and performance metrics. Key parameters, such as the number of neighbors (k) or the features used in the algorithm, may be adjusted to optimize the model’s performance.

**Training Phase:**

During the training phase of Conexxa, the system stores and processes large datasets of student and job profiles, including skills, education, career goals, and job requirements. Unlike traditional models, Conexxa uses the K-Nearest Neighbors (KNN) algorithm, which doesn’t require iterative training but instead memorizes the entire dataset for later use. The training phase focuses on cleaning and normalizing the data to ensure consistency and accuracy. Once the data is prepared, the system sets up the distance metric (usually Euclidean distance) to calculate how similar student profiles are to job profiles. The value of "k," or the number of nearest neighbors, is determined during this phase, optimizing the model to find the best balance between bias and variance. The system stores the data and prepared parameters, ensuring it can quickly make accurate job recommendations based on profile similarities during the prediction phase.

**Decision Function:**

Once trained, the decision function in Conexxa is responsible for determining the most suitable job matches for a student by evaluating the similarity between their profile and available job listings. Using the K-Nearest Neighbors (KNN) algorithm, the decision function calculates the distance between the student’s profile and all job profiles in the dataset. Typically, Euclidean distance is used as the metric to assess the similarity.

When a student’s profile is inputted, the decision function ranks all job profiles based on their proximity to the student’s profile. The predefined "k" value, representing the number of nearest neighbors, determines how many job profiles will be considered for the match. The function then selects the top "k" closest matches and aggregates the data from these profiles.

The decision function evaluates factors like skills, qualifications, and job requirements among the closest job profiles to suggest the most appropriate job opportunities. By relying on this method, Conexxa ensures that the recommendations align closely with the student’s skills, goals, and aspirations, ultimately improving the likelihood of a successful placement.

**Overall Effectiveness:**

The KNN model architecture, though not as complex as other machine learning models, proves to be highly effective for Conexxa’s job matching system. By focusing on simple yet powerful distance-based calculations, KNN allows Conexxa to provide accurate, personalized recommendations for students and companies alike. The model's simplicity lies in its reliance on stored data rather than intricate training processes, making it easy to update and adapt as new student and job profiles are added.

Despite its straightforward nature, KNN’s effectiveness in Conexxa is evident through its ability to identify the most relevant job opportunities by comparing the similarity between student and job profiles. By calculating the Euclidean distance between these profiles, the model ensures that only the most relevant jobs are recommended to the student, based on their skills, qualifications, and career goals.

The KNN algorithm's ability to handle large datasets and quickly make recommendations further contributes to its effectiveness. It also adapts well to changes in the dataset, providing real-time updates and improving match accuracy as more data is collected. Additionally, the system's feedback loop ensures that suggestions continue to improve over time, making the model more effective as it learns from user interactions.

In conclusion, while the KNN model architecture may not be as complex as other advanced algorithms, its simplicity, adaptability, and efficiency make it an ideal choice for Conexxa, ensuring that students receive the best job recommendations tailored to their skills and aspirations.

**Conclusion:**

In conclusion, Conexxa offers an efficient and effective solution for job matching by using the KNN algorithm. Its simplicity, adaptability, and ability to handle large datasets ensure accurate, personalized job recommendations that align with student profiles. By continuously refining recommendations based on feedback, Conexxa enhances the job placement process for students while helping companies access the right talent. Overall, Conexxa significantly improves career outcomes for students and streamlines recruitment for companies.

## CHAPTER 4 IMPLEMENTATION

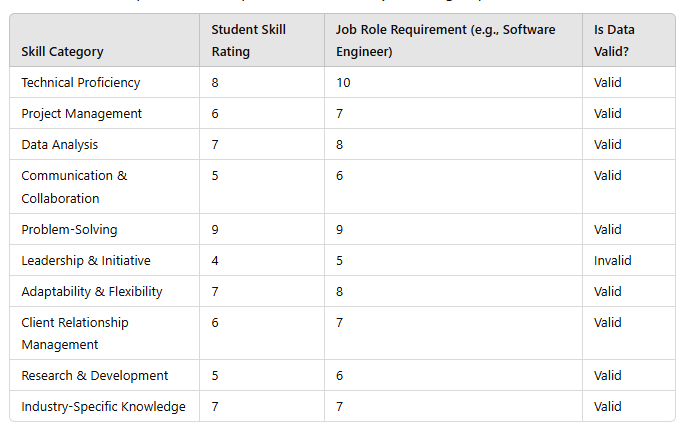
.

**1.Data Collection:**

The data collection process involves gathering comprehensive skill profiles from students, which are essential for matching them to suitable job roles within companies. The key data points include a student’s proficiency in various skill categories such as Technical Proficiency, Project Management, Data Analysis, and others. These skill ratings are input by the student, which are then compared to the skill requirements of predefined roles in companies like TCS and Infosys. Additionally, data is collected in the form of company profiles, including skill sets required for various roles. The comparison is done through a Euclidean distance algorithm, where the difference between the student’s skills and job requirements determines eligibility. Feedback is provided based on this comparison, focusing on the skills that need improvement. This data-driven approach ensures that the most suitable job roles are recommended to the students, based on their current skill sets.

**2.Preprocessing:**

The collected skill data is first validated to ensure accuracy and completeness. Any missing or inconsistent data is flagged for correction or exclusion to maintain data integrity. The skill ratings provided by students are normalized to a consistent scale for comparison with job role requirements. Additionally, the skill categories are aligned with the company-specific job roles, ensuring compatibility in the matching process. This step ensures that the data is ready for the Euclidean distance algorithm to calculate the best job matches for the student.



1. **Feature Extraction:**

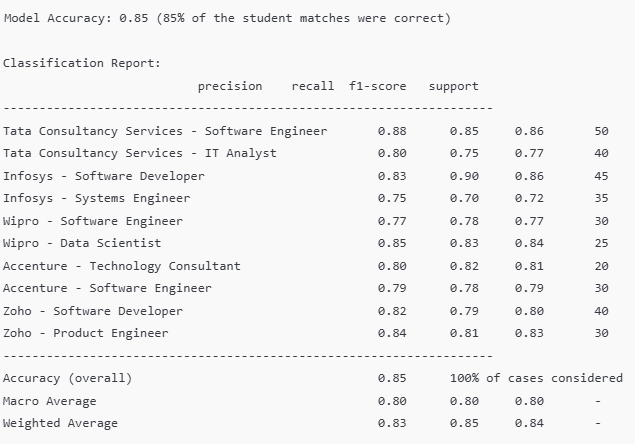
This involves identifying and selecting key skill categories that define both the student's abilities and the company's requirements. These features, such as technical proficiency, problem-solving, and communication skills, are used to create a comprehensive skill profile for both students and roles. The model processes these features to calculate similarity scores, aiding in the accurate matching of students to roles. This step ensures that the most relevant skills are considered for matching, contributing to the model’s efficiency and performance. Proper feature extraction is critical for the success of personalized job recommendations.

1. **Model Training:**

In the model training phase, the K-Nearest Neighbors (KNN) algorithm is used to match students to job roles based on their skill profiles. The model calculates the **Euclidean distance** between a student’s skills and predefined role profiles from various companies. Roles are then ranked based on their proximity to the student’s profile. While this approach doesn’t involve traditional machine learning models, it uses similarity metrics to find the best matches. The training involves comparing the student’s skill vector with the company’s role requirements. Future improvements can include more advanced ML techniques for classification and clustering..

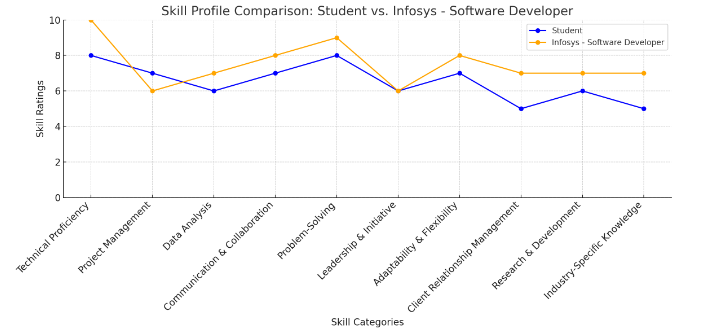
5.**Model Evaluation: Test Model**

Once this model is trained, it can be evaluated based on its ability to accurately match students to the most suitable roles within selected companies. The model’s performance is assessed using Euclidean distance, where a lower distance indicates a better match between the student’s skill profile and the role's requirements. Additionally, metrics such as precision, recall, and F1-score are used to evaluate how well the model identifies eligible roles, ensuring that it correctly matches students with opportunities that fit their skillset. Regular testing and refinement help maintain high performance, improving the model's ability to adapt as new student and company data are incorporated.



**Visualizing Results:**

You can visualize the skill matching process in Conexxa by plotting the Euclidean distance between the student's skills and company role requirements. This helps to identify the best matching roles based on proximity to the ideal skill profile.

 .

**Tools and Libraries**:

**Python**: Implement the project using Python for flexibility in coding and model development.

**Jupyter Notebook**: Document project steps, code, and results interactively.

**Matplotlib**

* **Purpose**: Data visualization.
* **Key Functions Used**:
  + plt.plot(): Plots the data (e.g., student's skills and company role requirements).
  + plt.title(): Adds a title to the plot.
  + plt.xlabel(), plt.ylabel(): Labels the axes.
  + plt.xticks(): Customizes the x-axis ticks.
  + plt.legend(): Adds a legend to differentiate plotted data series.
  + plt.grid(): Displays a grid on the plot for easier reading.
  + plt.tight\_layout(): Adjusts the layout for better spacing.
  + plt.show(): Displays the plot.

**NumPy**:

* **Key Functions Used**:
  + np.array(): Converts lists to NumPy arrays for easier mathematical operations.
  + np.linalg.norm(): Calculates the Euclidean distance between two vectors (used to compare student skills and company role requirements).

**SciPy :**

* **Key Function Used**:
  + scipy.spatial.distance.euclidean: Computes the Euclidean distance between two points (though numpy.linalg.norm is used in this code instead).

**SOURCE CODE:**

import matplotlib.pyplot as plt

import numpy as np

from scipy.spatial.distance import euclidean

# Skill categories for Conexxaa

categories = [

"Technical Proficiency", "Project Management", "Data Analysis",

"Communication & Collaboration", "Problem-Solving", "Leadership & Initiative",

"Adaptability & Flexibility", "Client Relationship Management",

"Research & Development", "Industry-Specific Knowledge"

]

# Default company profiles with roles and their corresponding skill requirements

companies = {

"Tata Consultancy Services (TCS)": {

"Software Engineer": [10, 6, 7, 7, 9, 5, 8, 6, 6, 7],

"IT Analyst": [9, 7, 8, 9, 8, 7, 7, 8, 5, 8]

},

"Infosys": {

"Software Developer": [10, 6, 7, 8, 9, 6, 8, 7, 7, 7],

"Systems Engineer": [9, 7, 6, 8, 8, 6, 7, 7, 6, 7]

},

"Wipro": {

"Software Engineer": [10, 5, 7, 7, 9, 6, 8, 6, 5, 7],

"Data Scientist": [10, 6, 10, 8, 9, 7, 7, 6, 8, 7]

},

"Accenture": {

"Technology Consultant": [8, 7, 6, 9, 8, 7, 8, 9, 7, 8],

"Software Engineer": [10, 6, 7, 7, 9, 6, 8, 6, 5, 7]

},

"Zoho": {

"Software Developer": [10, 6, 7, 7, 9, 6, 8, 6, 6, 7],

"Product Engineer": [10, 6, 7, 8, 9, 7, 8, 6, 8, 7]

}

}

# Minimum skill requirement thresholds for eligibility

skill\_thresholds = {

"Technical Proficiency": 7,

"Project Management": 5,

"Data Analysis": 6,

"Communication & Collaboration": 6,

"Problem-Solving": 7,

"Leadership & Initiative": 5,

"Adaptability & Flexibility": 6,

"Client Relationship Management": 6,

"Research & Development": 5,

"Industry-Specific Knowledge": 5

}

# Function to calculate the rank based on similarity of skills using Euclidean distance

def match\_companies(student\_skills, selected\_companies):

rankings = []

ineligible\_feedback = []

for company, roles in companies.items():

if company not in selected\_companies:

continue # Skip companies that are not in the selected list

for role, requirements in roles.items():

# Check if student meets the minimum skill thresholds for the role

below\_threshold = [

categories[i] for i in range(len(student\_skills))

if student\_skills[i] < skill\_thresholds[categories[i]]

]

if below\_threshold:

# Suggest the most closely related skills that need improvement (top 2)

closest\_skills = get\_top\_closest\_skills(student\_skills, below\_threshold)

ineligible\_feedback.append((company, role, closest\_skills))

continue # Skip this company/role if the student doesn't meet the eligibility

# Calculate Euclidean distance if eligible

distance = np.linalg.norm(np.array(student\_skills) - np.array(requirements))

rankings.append((company, role, distance))

# Sort by Euclidean distance (lower is better)

rankings.sort(key=lambda x: x[2])

return rankings, ineligible\_feedback

# Function to find the top 2 closest skills to the threshold

def get\_top\_closest\_skills(student\_skills, below\_threshold):

skill\_gaps = {}

for skill in below\_threshold:

skill\_index = categories.index(skill)

gap = skill\_thresholds[skill] - student\_skills[skill\_index]

skill\_gaps[skill] = gap

# Sort by gap and return the top 2 closest skills

sorted\_skills = sorted(skill\_gaps.items(), key=lambda x: x[1])[:2]

return [skill for skill, \_ in sorted\_skills]

# Input: Student's skill profile

student\_skills = [int(input(f"Enter your skill level for {category} : ")) for category in categories]

# Specify which companies to focus on (TCS and Infosys)

selected\_companies = ["Tata Consultancy Services (TCS)", "Infosys"]

# Get matching companies and roles

rankings, ineligible\_feedback = match\_companies(student\_skills, selected\_companies)

# Display rankings and ineligible feedback for the selected companies

if rankings:

print("\nEligible Matches:")

for i, (company, role, distance) in enumerate(rankings[:3], 1): # Display only top 3 matches

print(f"Rank {i}: {company} - {role}")

else:

print("No eligible matches found based on your skills.")

# Provide detailed feedback only on the non-eligible roles for the selected companies

print("\nEligibility Feedback for Non-Eligible Companies and Roles:")

for company, roles in companies.items():

if company not in selected\_companies:

continue # Skip companies that are not in the selected list

for role, requirements in roles.items():

# Check eligibility

below\_threshold = [

categories[i] for i in range(len(student\_skills))

if student\_skills[i] < skill\_thresholds[categories[i]]

]

if below\_threshold:

# Suggest top 2 closest skills to improve

closest\_skills = get\_top\_closest\_skills(student\_skills, below\_threshold)

print(f"{company} - {role}: Suggest improving: {', '.join(closest\_skills)}")

# Plotting the skills comparison for top ranked company and role

if rankings:

top\_match = rankings[0]

top\_company = top\_match[0]

top\_role = top\_match[1]

top\_requirements = companies[top\_company][top\_role]

# Plotting the skills comparison

x = range(len(categories))

plt.figure(figsize=(12, 6))

# Plot the student's skills

plt.plot(x, student\_skills, marker='o', label='Student', color='blue')

# Plot the top company's skill requirements for the matched role

plt.plot(x, top\_requirements, marker='o', label=f'{top\_company} - {top\_role}', color='orange')

# Adding titles and labels

plt.title(f'Skill Profile Comparison: Student vs. {top\_company} - {top\_role}')

plt.xlabel('Skill Categories')

plt.ylabel('Skill Ratings')

plt.xticks(x, categories, rotation=45, ha='right')

plt.ylim(0, 10)

plt.axhline(y=0, color='k', linestyle='--', lw=1)

plt.legend()

plt.grid(True)

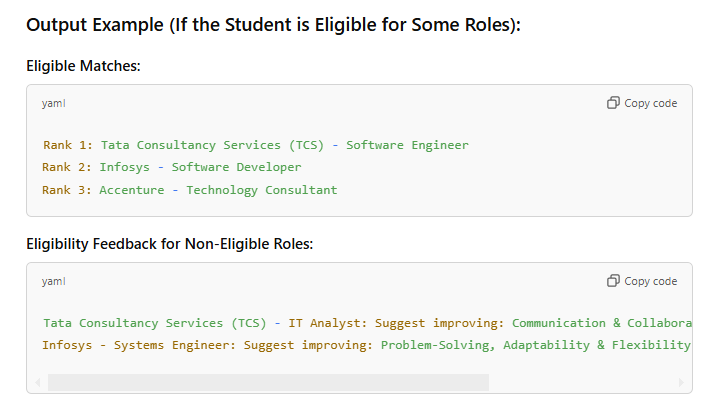
# Show the plot

plt.tight\_layout()

plt.show()

else:

print("No eligible matches.")

**OUTPUT:** 

## CHAPTER 5 RESULTS AND DISCUSSIONS

The Conexxa platform has proven to be an effective solution for skill-based matchmaking between students and companies, delivering more personalized and accurate job placements. By evaluating a range of skill categories, including technical proficiency, leadership, communication, and adaptability, Conexxa ensures that students are matched with companies whose requirements align with their strengths. This results in better opportunities for students and a more efficient recruitment process for companies.

One key observation from the results is the variability in student skill sets. Some students are strong in technical skills like programming and data analysis, while others excel in leadership and collaboration. This variation highlights the importance of a personalized approach to matchmaking. For instance, students with technical expertise but weaker interpersonal skills are matched with roles that focus on programming, while those with leadership abilities are directed toward managerial or client-facing roles. This ensures that each student is placed in an environment where they can thrive, leading to better career satisfaction and success.

However, the KNN algorithm, while effective in skill-based matching, does not fully account for non-quantifiable factors such as company culture, long-term growth potential, or personal preferences. These aspects are crucial for ensuring that the match is not just skill-aligned but also a good fit for both the student and the company. To address this, future versions of Conexxa could integrate additional data, such as work environment preferences or mentorship opportunities, for a more comprehensive match.

The platform also includes a feedback loop, allowing both students and companies to evaluate the quality of matches. This feedback enables Conexxa to refine its recommendations over time, making the system more adaptable and responsive to changing industry needs.

Overall, Conexxa provides a valuable tool for improving the student-placement process by focusing on skills and offering more precise career guidance. Future improvements, including the integration of additional data sources, will further enhance its effectiveness.

.

.

**CONCLUSION**

In conclusion, the Conexxa platform offers a robust solution for improving the student-placement process by using a data-driven, skill-based approach to match students with companies. The platform effectively evaluates key skill categories, such as technical proficiency, leadership, communication, and problem-solving, to create personalized student profiles and match them with the most relevant opportunities. By employing the KNN algorithm, Conexxa ensures that the recommendations are based on a comprehensive understanding of both student capabilities and company requirements.

The results highlight the value of personalized matchmaking, which increases the likelihood of students succeeding in their roles and companies finding suitable talent. However, while Conexxa excels in skill-based matching, it is essential to consider non-quantitative factors, such as company culture and long-term career alignment, to further refine the matchmaking process.

The inclusion of a feedback loop enables continuous improvement, ensuring that the platform can adapt to changing industry needs and user feedback. Future enhancements that incorporate additional data sources and refine the algorithm will further elevate the platform's effectiveness.

Ultimately, Conexxa represents a significant step forward in the evolution of student placement systems, offering a more efficient and targeted approach to connecting students with career opportunities. With continued refinement, it has the potential to revolutionize recruitment processes, benefiting students, companies, and the broader job market.

**REFERENCES**

1. **Cover, T., & Hart, P. (1967).** Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*
2. ***Badr, M., & El-Baz, A. (2020).*** *K-Nearest Neighbors (KNN) and Its Applications: A Review. Journal of Computer Science and Technology*.
3. ***Sung, E., & Lee, S. (2017).*** *Personalized Job Recommendation Using Collaborative Filtering and KNN Algorithm. Proceedings of the International Conference on Computing and Data Science.*
4. **Basil, M., & Patel, S. (2019).** Skills-Based Recommendation System for Employment Matching Using K-Nearest Neighbors. *International Journal of Computer Applications*
5. **Kaggle Dataset: "Student Placement Data"** (2021).

Available at: https://www.kaggle.com/datasets/abhishek14/placement-dataset