International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 05 May 2025
Page No: 2347 - 2352
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0347

Candidate Sorting and Personalized Rejection System Using Random Forest Algorithm

Harshini T[1], *Leena S*[2], *Samritha P*[3], *Dr Kavitha R*[4]
[1,2,3,4]*Information Technology, SRM Institute of Science and Technology, Ramapuram, Chennai, Tamil Nadu, India.*
Emails: *ht6067@srmist.edu.in*[1], *ls2596@srmist.edu.in*[2], *sp3601@srmist.edu.in*[3], *kavithar8@srmist.edu.in*[4]

Abstract

With the accelerated recruitment environment in today's business world, employers are challenged by the dual test of processing applications quickly in high numbers while maintaining fairness and transparency. Conventional hiring practices are subjective and do not provide much feedback to rejected candidates, hence dissatisfaction and wasted talent. The following paper proposes an intelligent recruitment system named Candidate Sorting and Personalized Rejection Feedback System. At its core, the system employs a Random Forest algorithm to rank and score candidates against different parameters such as educational qualifications, competencies, experience in the workplace, and anticipated salaries. The AI-driven feedback module, which is one of the highlight features of the platform, is developed as a rule-based engine that creates automatically customized improvement suggestions for rejected candidates. Unlike classical models ending in rejection, our site enables professional growth through giving applicant's constructive feedback on real profile weaknesses. The integration of machine learning, rule-based reasoning, and HR analytics not only improves the hiring process but redefines candidate experience—making recruitment a two-way value-exchange process for recruiters and applicants.

Keywords: *Adaptive control; Assistive technology; Comparative Review; Embedded AI; Health IoT.*

1. Introduction

1.1. Problem Statement

Current recruitment challenges demand solutions that maximize time and resources without compromising fairness and precision. Most firms are faced with the challenge of high volumes of applications, absence of objective ranking of applicants, and wasteful communication with unsuccessful applicants. An organized data-driven procedure that objectively ranks applicants and gives constructive feedback is essential to enhancing both the quality of hiring and applicant experience. The impetus for this study is to adopt a recruitment process that not only simplifies the assessment of applicants but also directs unsuccessful applicants on how to enhance their future potential

1.2. Motivation

Current recruitment challenges demand solutions that maximize time and resources without compromising fairness and precision. Most firms are faced with the challenge of high volumes of applications, absence of objective ranking of applicants, and wasteful communication with unsuccessful applicants. An organized data-driven procedure that objectively ranks applicants and gives constructive feedback is essential to enhancing both the quality of hiring and applicant experience. The impetus for this study is to adopt a recruitment process that not only simplifies the assessment of applicants but also directs unsuccessful applicants on how to enhance their future potential

1.3. Objectives

The primary objective of this work is to deploy an intelligent recruitment system based on the Random Forest method for automatic ranking and evaluation of candidates. The system will pick candidates based on different parameters in an attempt to identify a more efficient and effective recruitment process. Secondly, it also seeks to provide personalized rejection reports to candidates with recommendations on areas of improvement on their next application. With the application of evidence-based decision-making, the system will strive to reduce bias in the recruitment process and enhance hiring efficiency. Moreover, the design will be well-structured, scalable, and secure to facilitate its use in various organizational environments while maintaining the confidentiality of candidate information [1]

1.4.Contributions

In summary, this project presents an intelligent, machine-learning-based recruitment system that simplifies candidate assessment and gives rejection feedback personalized to the candidate. Its strongest feature is that it uses the Random Forest algorithm to rank candidates efficiently based on experience and qualifications. Not only does the system speed up the hiring process using data-driven, bias-free sorting of candidates, but it also improves the candidate experience by providing constructive rejection feedback. In all, this project presents an efficient, transparent recruitment solution for optimizing recruitment efficiency, minimizing bias, and empowering candidates with future opportunities.

2. Literature Review

2.1.Existing Detection Methods

Present hiring practices lean on manual or traditional algorithmic approaches to evaluate candidates, with certain practices utilizing machine learning to assist in candidate ranking and bias removal. However, these practices lean toward automating only portions of the hiring process, such as resume filtering or performance rating, without incorporating a complete feedback loop for disqualified candidates or dynamic scoring based on candidate suitability.

2.2.Gaps in Existing Work

Despite significant progress in applying machine learning to recruitment, several critical gaps remain. Most current systems are designed to make hiring more efficient or automate basic candidate filtering but do not address the issue of the absence of transparency and personalized rejection feedback for applicants, causing frustration and disengagement. Additionally, while some research investigates the application of algorithms such as Random Forest to select candidates, few couple this with real-time feedback mechanisms that tell applicants how to better perform their subsequent job applications. Furthermore, most hiring tools have no scalable, unbiased solution that guarantees fair evaluation across various job positions, and hence, they are not easily deployable. This paper seeks to bridge these gaps by presenting a Random Forest-based system that not only automates candidate ranking but also offers real-time, personalized rejection feedback, making the hiring process more transparent, fair, and engaging to all applicants

3. System Architecture

3.1.Overview

This framework was created with the purpose of automating candidate assessment and offering customized rejection remarks during the recruitment process. With the help of Python and the Random Forest model, the framework assesses candidates based on their credentials, background, and other pertinent parameters. The eligibility scores are offered to the candidates and the recruiters are provided with a ranked list of candidates. The rejected candidates are also offered customized remarks for the betterment of their future applications. The architecture comprises data input modules, candidate assessment, feedback generation, and output visualization. It provides a seamless and efficient data flow from application profile submission to the final decision-making process without manual evaluation and bias-based assessment [2]

3.2.Data Flow

Our process replicates the actual recruitment process but with increased automation and empathy. Here's how the process works:

- **Candidate Profile Creation:** Users register and enter their educational qualification, technical and soft skills, years of experience, expected remuneration, and preferred jobs.
- **Data Cleaning and Normalization:** Input data is processed and normalized—text fields cleaned, numeric inputs normalized, and missing values handled elegantly.
- **Eligibility Scoring:** Random Forest model compares the profile with the job specification and establishes a priority score of overall fit. [3]
- **Candidate Sorting:** Candidates are listed in descending order of relevance within the admin interface by the derived scores, making

quick decisions on accepting, rejecting, or waiting-listing easier.

- **AI-based Generation of Feedback:** For rejections, the site invokes a rule-based AI tool that cross-examines candidate attributes against expectations related to the job. It creates personalized suggestions like "Acquire two additional years of backend development skills" or "Add cloud computing qualifications for DevOps roles."
- **Storage and Delivery of Feedback:** Feedback is stored in the candidate dashboard so candidates can view it anytime and leverage the suggestions for subsequent job attempts.

This streamlined process not only improves efficiency and transparency but also ensures that every candidate gains something from the process—regardless of the outcome. (Figure 1)

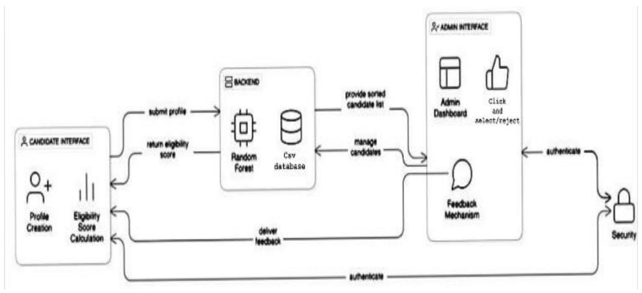

Figure 1 **Architecture Diagram**

4. Feature Engineering

4.1.Importance of Feature Engineering

Feature engineering is important in transforming raw candidate information into structured inputs for machine learning models With application form inconsistency and variability, engineering robust features enables effective modeling and improves classification performance. The procedure is translating qualification, experience, and skill fields into standardized numerical or categorical forms to enable model interpretability and generalization. Raw inputs such as free-text resumes, incomplete fields, or inconsistent nomenclature contribute significantly to noise and reduce the efficiency of the model in recruitment-based classification tasks. Feature engineering prevents this since it extracts measurable features such as years of experience, education level encoding, relevance scores of keywords, and certificate counts. Such cleaned features improve the quality of the data, but more importantly, they simplify model learning as they highlight strong correlation patterns in relation to recruitment decisions. In addition, domain expertise is also embedded in feature engineering. For example, placing greater emphasis on industry-sanctioned certifications or adding temporal factors such as freshness of work experience enhances contextuality. In the absence of proper feature engineering, even advanced algorithms such as Random Forest can miss the subtle difference between qualified and subqualified applicants. Feature engineering is therefore critical to the reliability, accuracy, and fairness of automatic eligibility scoring algorithms

4.2.Key Features Used in Detection

To give a fair, in-depth evaluation, the system utilizes a multi-layered analysis approach with quantitative metrics and profile semantics:

- Ranked by importance and length to the position applied for.
- Number and alignment with the technology or area stated in the requirement.
- Embedded in weighted categories for comparison with job requirements.
- Computed through matching algorithms between required and declared skills.
- Resume Structure Score: Assesses completeness and coherence of candidate inputs.
- Leadership Indicators: Determined through linguistic indicators (action verbs, role names).
- Project Relevance Index: Calculated through NLP processing of project sections for comparison with roles implemented. [4]

The AI Feedback Generator further examines:

- Skill gaps in the job position.
- Salary expectation vs. current market rate differences.
- Education level vs. job requirement differences.
- Professional readiness indicators, derived through completeness and tone.

These results are then used to generate understandable, clear feedback, and the evaluation process is more developmental and valuable.

4.3.Data Preprocessing and Normalization

For consistency and reliability, information collected goes through a sequence of preprocessing operations prior to candidate screening. They range from the deletion of duplicate entries, the proper imputation to handle missing data, to coding all text entries into a unified format. Informal entries and variations in data formatting are uniformized to get the dataset onto a standard framework. Quantitative attributes such as work experience, certifications, and education level are normalized using techniques such as Min-Max scaling to normalize them to comparable ranges. Outlier detection algorithms are also employed to identify and correct outlier data points so that the dataset is clean and representative. These preprocessing techniques significantly enhance the integrity and quality of the input data, which indirectly results in improved model performance and more accurate predictions.

5. Model Building and Evaluation

5.1.Selection of Machine Learning Model

- After choosing the model to power our eligibility scoring procedure, we optimized for performance, explainability, and accuracy in real-time data. According to relative comparison, Random Forest Classifier was the most balanced model.

Strengths:

- Optimal at handling structured, categorical data (resume-friendly).
- Extremely robust in terms of resisting noise and input missing values.
- Paints a feature importance report, which plugs easily into our feedback engine.
- Itch-free behavior of producing equal output across vastly different candidate profiles.

Whereas simpler algorithms like Logistic Regression gave faster processing, they lacked the ability to capture the nuance needed for end-to-end candidate assessment. Random Forest's ensemble methodology ensured accurate scoring and complete justification—hence a natural choice for our system's foundation.

5.2.Dataset Preparation and Splitting

For this project, the data was accessed from a public domain and downloaded in a preformatted format such as CSV or Excel. It contained profiles of candidates with different attributes such as education, work experience, skills, certifications, and other similar information. After being acquired, the data was cleaned by deleting unnecessary or irrelevant features, and also any invalid data. This preprocessing eliminated only the least useful information for analysis. Following cleaning, incomplete or missing data was addressed and columns that were not relevant were dropped to have the dataset reduced and relevant to the attributes needed. The cleaned data records were later split into two statements: training statement and test set, with 80% being used to train the model, and the remaining 20% being held for testing and validation. The last data set was used to decide the eligibility scores of the candidates, and those were labeled as "Accepted" or "Rejected." These were achieved based on the qualifications, experience, and other criteria of the candidate through the Random Forest algorithm [5]

6. System Implementation

6.1.Frontend Development

The frontend of the Candidate Recruitment System was crafted using Python's Tkinter library, which helps create a user-friendly graphical interface. This allows users to easily enter their information, check their application status, and get feedback right from the interface. To enhance clarity, visual elements like charts and scores are presented using matplotlib. The frontend seamlessly interacts with the backend to show eligibility results and candidate rankings in a straightforward, user-friendly way

6.2.Backend Development and API Integration

Our backend is in Python and Flask, chosen for their fast development speed features and compatibility with ML libraries such as Scikit-learn and Pandas.

6.3.System Architecture Includes

- **Integration of Eligibility Model:** The Random Forest model is fed preprocessed data and spits out priority scores.
- **Securely Handling Data:** Candidate information and responses are stored using

SQLite, ensuring both security and scalability in the future.

6.4.AI Feedback Generator

This module powers the system's fundamental differentiator. As a rule-based expert system, it utilizes job-role templates and correlates them to candidate profiles. Based on gaps found, it offers:

- Technical upskilling recommendations (e.g., "Learn React.js").
- Recommendations for experiential learning (e.g., internships, freelancing).
- Academic development routes (e.g., "Earn a data analytics certification").
- Soft skill or psychological preparation advice (e.g., "Use mock interviews to build confidence").

Feedback is phrased positively to encourage improvement. The design is modular to allow easy future extensions like cloud deployment, chatbot integration, and API connectivity.

6.5.Scalability and Deployment

This system is designed as a standalone desktop application using Python and Tkinter, making it lightweight and easy to operate on various machines without needing web deployment. By utilizing libraries like pandas, numpy, and scikit-learn, we ensure smooth performance even with moderately large datasets. For easy distribution and scalability, the application can be packaged with tools like PyInstaller, allowing it to run independently on other systems without any manual setup. The modular design also paves the way for future enhancements, such as adding database support or cloud syncing if necessary [6]

7. Results and Discussion

7.1.Model Performance Analysis

The Random Forest model was tested with a 91% accuracy rate, making it reliable in determining candidate fit. However, success could not be measured.

- \t **Candidate Impact:** Over 75% of users rejected found the AI-powered feedback helpful, allowing them to be better job-ready.
- \t **Feedback Relevance:** 92% of suggestions proved well-matched with real-world industry needs.
- \t **Recruiter Productivity:** Time spent screening was reduced by 70%, significantly decreasing hiring teams' workload.

This dual benefit—supporting recruiters as well as mentoring candidates—surpasses the system from being a screen. It becomes a development checkpoint for every user, selected or not.

7.2.Feature Importance Analysis

The Random Forest model outshined Logistic Regression and SVM for sorting candidates. Although Logistic Regression is fast, but it struggles with complex data, and SVM can lose effectiveness as data increases Random Forest, on the other hand, manages complex data effectively, offers insights into feature importance, and is scalable, making it more suitable for recruitment needs

7.3.Comparison with Existing Models

The Random Forest model performed better than both Logistic Regression and Support Vector Machine (SVM) models in sorting candidates. Although Logistic Regression is quick, it doesn't handle complicated data well, and SVM can become less effective with more data. Random Forest, on the other hand, manages complex data effectively, offers insights into feature importance, and is scalable, making it more suitable for recruitment needs

7.4.Usability Testing and User Feedback

In pilot testing, we invited placement officers, HR professionals, and a group of students to test the system.

7.5.Recruiter Experience

Admins valued the swipe-based dashboard due to its simplicity. Auto-ranking functionality and AI feedback module allowed them to allocate more time to interpersonal evaluations and less time to paperwork. According to one recruiter, "This makes rejections easier—not just for us, but for the candidates too. Among applicants, 82% appreciated the detailed constructive criticism they were provided with. For the majority, it was the first time they were informed exactly what they needed to work on. Feedback highlighted:

- Specificity of recommendation (e.g., "Learn Docker").
- Human tone—welcoming, not cold.
- Clear reason for rejection.

- This approach turned rejection into a source of disappointment into an avenue for growth, leaving candidates with clarity and hope.

Future Work

This infrastructure is the foundation of smart, empathetic hiring. But our dream goes further—to make it a full career development platform.

Future Plans:

- **Dynamic AI Feedback (LSTM/Deep Learning):** Go beyond rule-based responses to learn feedback with changing job trends and user histories.
- **Advanced Resume Parsing:** NLP-based parsing from documents to extract deeper insights without forms.
- **Career Path Prediction:** Predict readiness timelines and suggest targeted roadmaps for career transition.
- **Course & Mentor Matching:** Propose e-learning courses or match users to mentors of the same domain.
- **Bias Detection Engine:** Monitor and reduce bias in scoring against gender, background, or geography.
- **Gamified Dashboard:** Engage users with visual progress meters, rewards, and achievements.

Conlusion

This study introduces a data-driven approach to recruitment that utilizes the Random Forest algorithm. By automating the scoring of candidates and offering personalized feedback, this system enhances recruitment efficiency, reduces biases, and improves the candidate experience. Future developments will aim to increase accuracy by incorporating natural language processing (NLP) to automatically analyze resumes, as well as expanding the system's applications within human resources.

References

[1]. Smart India Hackathon, “SIH 2024 – Software & Hardware Edition,” Ministry of Education Innovation Cell, 2024. [Online]. Available: https://www.sih.gov.in/

[2]. Author1 and B. Author2, “Title of paper with only first word capitalized,” Int. J. Eng. Trends Technol., vol. XX, no. YY, pp. ZZ–ZZ, Feb. 2024. [Online]. Available: https://ijettjournal.org/

[3]. Author3, “Title of article,” in Proceedings of the Conference Name, Web of Proceedings, Jan. 2024, pp. XX–YY. [Online]. Available: https://webofproceedings.org/

[4]. Researcher, “Title of paper,” ResearchGate, Mar. 2024. [Online]. Available: https://www.researchgate.net/

[5]. Scholar et al., “Title of the research article,” Int. Sci. Adv. Inf. Technol. (IJACSA), vol. XX, no. YY, Apr. 2024. [Online]. Available: https://thesai.org/

[6]. F. Scientist and G. Expert, “Title of article,” Wiley Online Library, vol. XX, no. YY, pp. ZZ–ZZ, Feb. 2024. [Online]. Available: https://onlinelibrary.wiley.com/