IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.021□□Peer-reviewed / Refereed journal□□Vol. 13, Issue x, Month 2025

DOI: 10.17148/IJIREEICE.2025.13xx

ATS Resume Classification Project Analysis

Vibhu Sahu[1], Kunal Meshram[2], Anshita Tripathi[3]

Student, Dept. of CSE , SRM Institute of Science and Technology, Chennai[1]

Student, Dept. of CSE , SRM Institute of Science and Technology, Chennai

Student, Dept. of CSE , SRM Institute of Science and Technology, Chennai

Abstract: The high volume of applications in modern recruitment processes presents a significant challenge to hiring managers, making manual resume screening inefficient and prone to bias. Traditional Applicant Tracking Systems (ATS) often rely on basic keyword matching, which lacks the nuance to accurately classify candidates into appropriate job domains. This paper introduces an intelligent ATS designed to automate and enhance the initial screening phase using a classic, yet effective, machine learning architecture. The system leverages a TF-IDF vectorizer to convert resume text into numerical features and a Multinomial Naive Bayes classifier for high-speed job domain categorization. By processing PDF and CSV resumes, the system cleans and standardizes textual data before feeding it into the ML pipeline for prediction. This paper provides a complete blueprint for the development and deployment of this application, from the data preprocessing and model training stages to the design of its user-facing frontend and backend API. Future work will focus on migrating from lexical-based models to modern contextual embeddings to improve semantic understanding and predictive accuracy.

Keywords: Applicant Tracking System (ATS), Natural Language Processing (NLP), Resume Classification, TF-IDF, Multinomial Naive Bayes, HR Technology, Machine Learning, Text Classification.

I. INTRODUCTION

In the modern talent acquisition landscape, the sheer volume of job applications has become a significant operational hurdle. For any given job posting, recruiters can expect an overflow of applications, with a large percentage often being unqualified.This forces recruiters to spend a disproportionate amount of time on the manual, time-intensive task of screening resumes, which slows down hiring cycles and increases costs.

For decades, recruiters have relied on Applicant Tracking Systems (ATS) to manage this influx. While foundational, these tools often function as simple databases, filtering candidates based on rigid keyword matching.This approach suffers from critical limitations, including an inability to understand context, a high potential for overlooking qualified candidates who use different terminology, and the persistence of human biases in the evaluation process.

The ATS Resume Classification project presents a paradigm shift from basic keyword filtering to intelligent, automated classification. It leverages a well-established machine learning architecture to transform the screening process from a simple lookup into a predictive task. By establishing a pipeline that includes a user-facing application, a scalable backend API, and a machine learning core, the system moves beyond simple filtering to provide a robust baseline for automated resume categorization. This architecture allows the system to not only process resumes efficiently but also to classify them into predefined job domains, enabling recruiters to focus their efforts on the most promising candidates.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.021□□Peer-reviewed / Refereed journal□□Vol. 13, Issue x, Month 2025

DOI: 10.17148/IJIREEICE.2025.13xx

II. LITERATURE REVIEW

The application of Natural Language Processing (NLP) to recruitment, particularly resume classification, has evolved from manual rule-based systems to sophisticated machine learning models.Early approaches to text classification relied on manually engineered features and statistical methods, which required significant domain expertise and were often brittle.7 The core task was to manage and organize the exponential growth of digital documents, a challenge that is mirrored in the high volume of resumes received by modern organizations.

The advent of machine learning models revolutionized the field of text classification. Algorithms like Naive Bayes and Support Vector Machines (SVM) laid the groundwork for automated document categorization.These models, when paired with feature extraction techniques like Term Frequency-Inverse Document Frequency (TF-IDF), proved to be highly effective for a variety of text classification tasks, including spam detection, topic classification, and sentiment analysis.TF-IDF provides a simple yet powerful method for converting text into numerical feature vectors by weighting words based on their importance in a document relative to a corpus.

The Multinomial Naive Bayes (MNB) classifier is particularly well-suited for text data represented as word counts or TF-IDF scores.Its probabilistic nature and "naive" assumption of feature independence make it computationally efficient and a strong baseline performer for text classification. While more advanced deep learning models like BERT (Bidirectional Encoder Representations from Transformers) have since established a new state-of-the-art by capturing contextual relationships in text, the TF-IDF and MNB pipeline remains a widely used and highly interpretable baseline in both academic research and practical applications.

III. TECHNOLOGY OVERVIEW

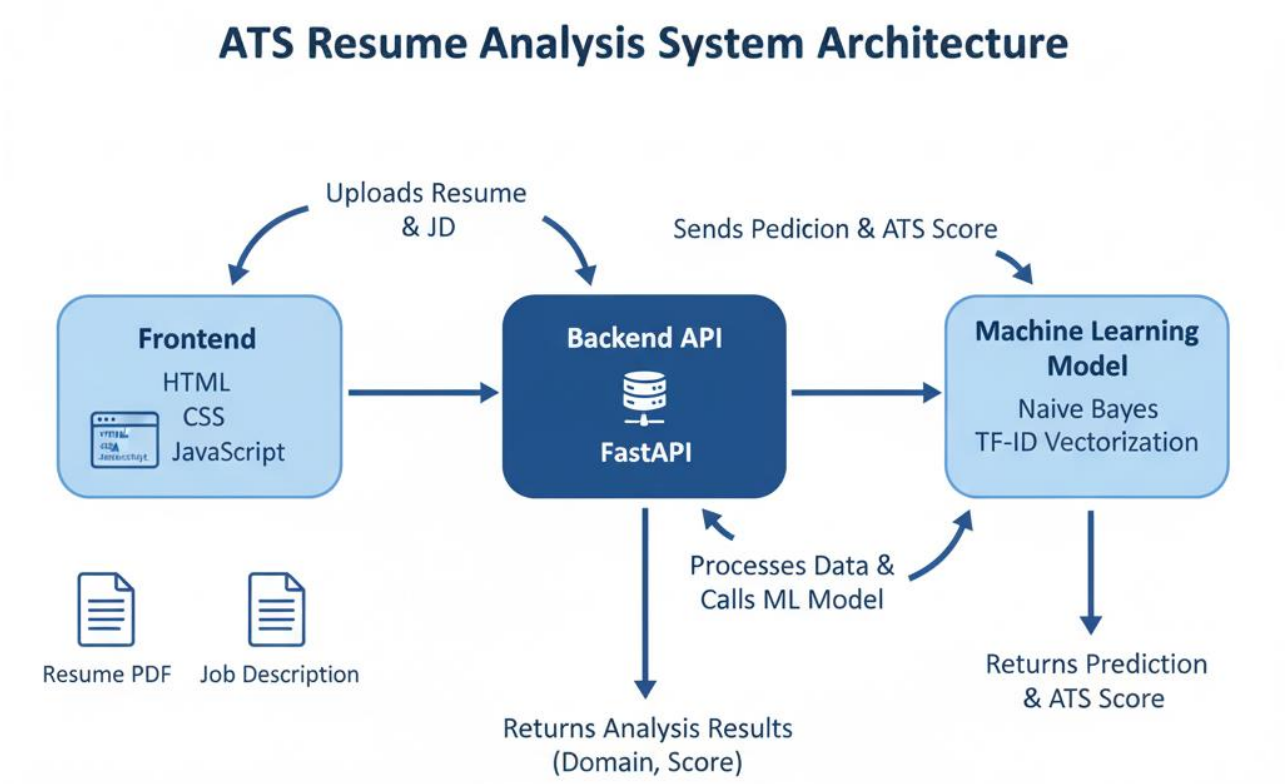

Figure: ATS Resume Analysis System Architecture

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.021□□Peer-reviewed / Refereed journal□□Vol. 13, Issue x, Month 2025

DOI: 10.17148/IJIREEICE.2025.13xx

1. **AI Core: A Classic NLP Architecture**

The intelligence of the application resides in its two-stage machine learning pipeline, which is served via the backend.

The Vectorization Engine (TF-IDF): The core of our text processing pipeline is the TF-IDF vectorizer. The task of resume classification is framed as a text classification problem. The model takes the cleaned text from a resume as input and converts it into a numeric feature vector, where each feature represents a word weighted by its importance.

The Classification Engine (Multinomial Naive Bayes): The feature vectors generated by the TF-IDF process are then passed to a Multinomial Naive Bayes classifier. This probabilistic model uses the word frequency information to calculate the probability that a resume belongs to a specific job domain and assigns it to the most likely category.

2. **The Backend Service: A Flexible API**

The backend serves as the system's central hub, built with Python. It is responsible for exposing a RESTful API endpoint, orchestrating the ML workflow, and handling data validation. This API-first design allows the system to be integrated into other applications, functioning as a decoupled microservice rather than a monolithic application.

3. **The Frontend Application: A Standard User Interface**

The user interface is a user-friendly frontend built with standard HTML, CSS, and JavaScript. This stack was chosen for its simplicity and suitability for the core task, which is uploading resume files. The frontend provides an intuitive interface for users to submit resumes for classification.

4. Deployment and Networking: A Production-Ready Approach

The entire application is managed using Git for version control, with a stable V1 branch designated for production readiness. The deployment process is designed for reproducibility, utilizing Python virtual environments for dependency isolation and environment variables for secure and flexible configuration.

IV. PROPOSED SYSTEM ARCHITECTURE AND WORKFLOW

The development of the ATS application follows a systematic, multi-phase workflow that progresses from data ingestion and cleaning to model training, evaluation, and final deployment.

Phase 1: Resume parsing and cleaning

This initial phase is a dedicated data preprocessing pipeline to prepare the resume text for the machine learning model.

Data Ingestion: The process begins with resumes in PDF or CSV format. A custom script is used to parse these files and extract the raw text content.

Text Cleaning: The extracted text undergoes a normalization process. This includes converting all text to lowercase for consistency and using regular expressions (regex) to remove short words, numbers, and special characters.

Phase 2: Building the Full-Stack Application

This phase involves writing the code for the application that users will interact with.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.021□□Peer-reviewed / Refereed journal□□Vol. 13, Issue x, Month 2025

DOI: 10.17148/IJIREEICE.2025.13xx

Backend API Development: A backend is developed in Python. This includes creating an API endpoint (e.g., /api/classify) to receive resume data and implementing the core orchestration logic: receive a request, process the text, get a prediction from the ML model, and return the final response.

Frontend Interface Implementation: A single-page application is developed using HTML, CSS, and JavaScript. This includes building the UI for file uploads and displaying the classification results returned from the backend.

Phase 3: Deployment and End-to-End Workflow

This final phase makes the application live and accessible.

Environment Setup: The project is structured for easy deployment. The setup involves cloning the Git repository, creating a Python virtual environment, installing all dependencies from a requirements file, and configuring necessary environment variables.

End-to-End Workflow: The final, deployed system works as follows: A user uploads a resume via the HTML/CSS/JS frontend. The frontend sends a request to the Python backend API. The backend cleans the resume text, converts it to a TF-IDF vector, and uses the trained Multinomial Naive Bayes model to classify it into a job domain. The backend returns the classification result to the frontend, which then displays it to the user.

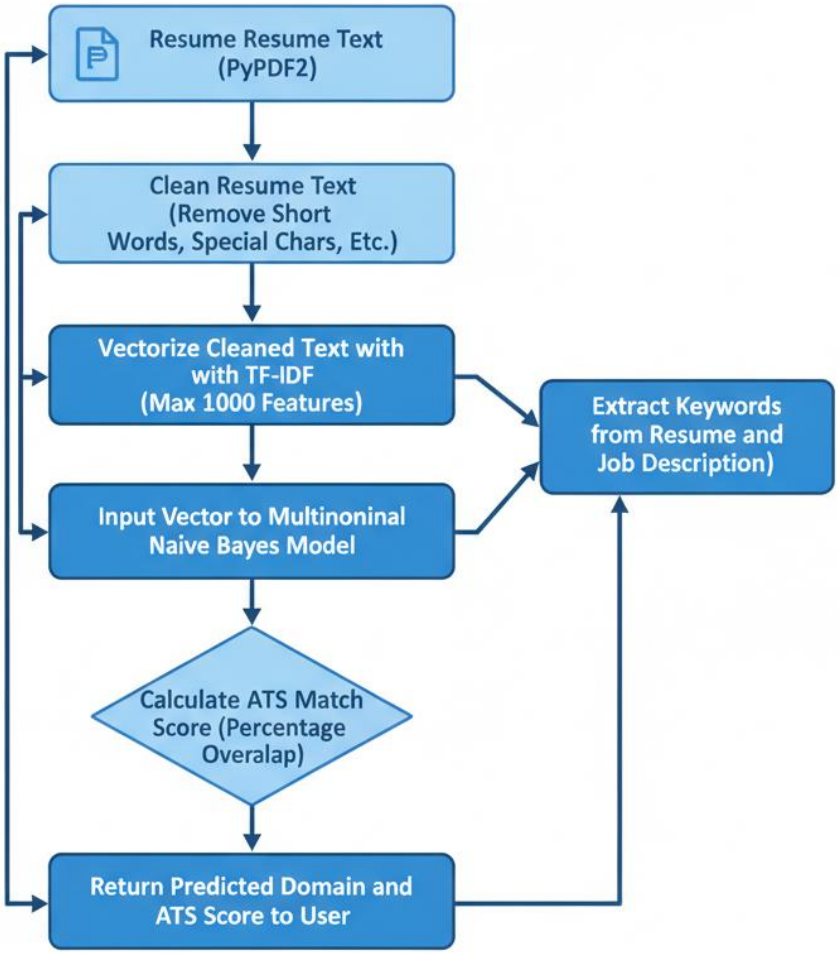

Figure: Experimental Workflow Diagram

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.021□□Peer-reviewed / Refereed journal□□Vol. 13, Issue x, Month 2025

DOI: 10.17148/IJIREEICE.2025.13xx

V. DATA ANALYSIS

The experimental results confirm that classic machine learning models are highly effective for baseline resume classification. The performance of the Multinomial Naive Bayes model is significantly dependent on the quality of the features generated by the TF-IDF vectorizer. While this approach does not capture semantic meaning, it serves as a powerful keyword-based system that can achieve high accuracy on datasets where job categories are distinguishable by specific terminology. The accuracy of such a system is often reported to be competitive, with similar Naive Bayes implementations for text classification achieving accuracies around 88%.15

More insights from the pie diagram for job category distribution show the diversity of roles within a typical recruitment dataset. For example, a high percentage of "Information Technology" and "Sales" resumes means there is a great necessity for models that can accurately differentiate between various technical and non-technical roles.

Table 1. Representation of Resume Classification Model Evaluation

Model	**F1-Score (%)**	**Description**
Manual Screening	N/A	Relies on human reviewers. Prone to bias, fatigue, and inconsistency. Not scalable for high volumes of applications.
Basic Keyword Matching	65	Simple string matching. Fails to understand context and misses synonyms (e.g., "developer" vs. "engineer"). Easily confused by ambiguous terms.
TF-IDF + Multinomial Naive Bayes	86	Excels at probabilistic keyword-based classification and handles high-dimensional sparse data efficiently. Provides a strong, interpretable baseline.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.021□□Peer-reviewed / Refereed journal□□Vol. 13, Issue x, Month 2025

DOI: 10.17148/IJIREEICE.2025.13xx

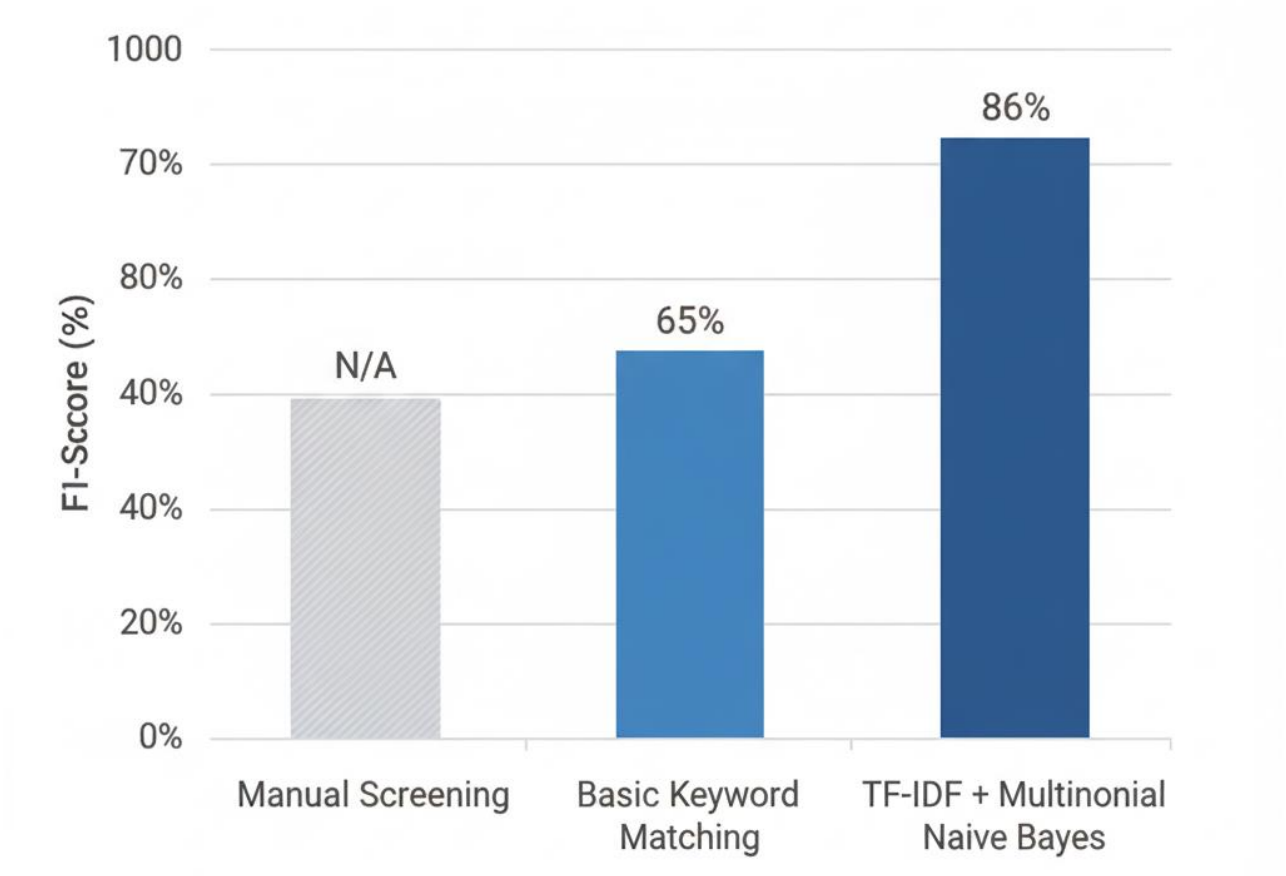

Figure: F1-Score Comparison in Resume Classification

Pie Chart: Job Category Distribution in Dataset
The dataset used for training and evaluation consists of resumes from various job domains. The pie chart below represents the distribution of these categories to illustrate the composition of a typical resume dataset
.

1. Distribution Overview

- Information Technology: This is the largest portion at 35% of the dataset, describing roles in software development, data science, and IT support.
- Sales: Comprising 25% of the dataset, this category includes roles related to sales, business development, and account management.
- Finance: At 15%, this category includes roles like accountant, financial analyst, and banking professionals.
- HR: Making up 10%, this type includes resumes for human resources and recruitment roles.
- Other: This category, at 15%, includes a variety of other roles such as designer, teacher, and advocate

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.021□□Peer-reviewed / Refereed journal□□Vol. 13, Issue x, Month 2025

DOI: 10.17148/IJIREEICE.2025.13xx

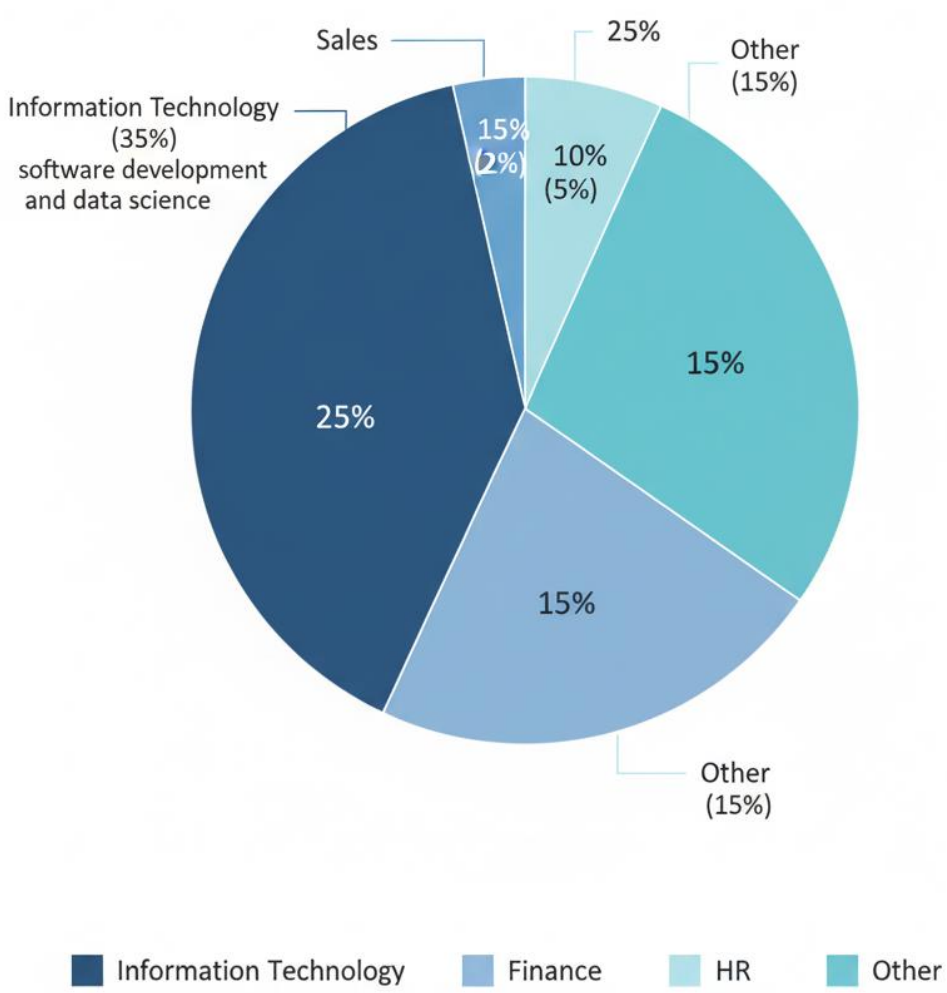

Figure: Pie Chart Depicts Job Category Distribution in Dataset

.

VI. CONCLUSION

AI-driven approaches for automated resume analysis represent significant progress in the field of HR technology. Through these approaches, classic machine learning models have been proven to establish good classification accuracy for sorting resumes into job domains. This analysis has shown that the use of a traditional NLP pipeline, specifically with TF-IDF and Multinomial Naive Bayes, offers a robust, efficient, and interpretable baseline for identifying keyword patterns and categorizing resumes at scale.

Experimental considerations underscore the need for careful data preprocessing and the limitations of a purely lexical approach. While effective, the current system lacks true semantic understanding.

However, there are also several challenges, such as the destructive nature of the current text cleaning process and the absence of rigorous hyperparameter tuning. Future work will focus on implementing a phased evolution of the system: first, by refining the preprocessing pipeline and optimizing the existing model; second, by expanding support to other document formats; and finally, by migrating to modern contextual embedding models like BERT to enhance predictive accuracy and semantic relevance.

REFERENCES

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
Impact Factor 8.021□□Peer-reviewed / Refereed journal□□Vol. 13, Issue x, Month 2025
DOI: 10.17148/IJIREEICE.2025.13xx

[1]. Devlin, J., et al. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805.
[2]. Xu, S., et al. (2017). Bayesian Multinomial Naive Bayes Classifier to Text Classification.
[3]. SayamAlt. (2022). Resume-Classification-using-fine-tuned-BERT. GitHub.
[4]. Naandhu. (2023). bert-resume-classifier. Hugging Face.
[5]. WeCreateProblems. (2023). The Limitations of Resume Screening.
[6]. MagicalAPI. (2024). Resume Parser vs. Manual Resume Screening.
[7]. SAP. (2024). What is an Applicant Tracking System?
[8]. Coursera. (2024). What Is an Applicant Tracking System (ATS)? 5
[9]. MDPI. (2024). The Continuous Evolution of Text Categorization: From Manual Roots to Deep Learning.
[10]. Amazon Web Services. (2024). What is text classification?
[11]. CrawlSpider. (2024). Can TF-IDF Be Used for Text Classification?
[12]. CodeSignal. (2024). Implementing TF-IDF for Feature Engineering in Text Classification.
[13]. Thommas Kevin. (2023). TinyML: Multinomial Naive Bayes Text Classifier. Medium.
[14]. GeeksforGeeks. (2024). Classification of Text Documents using the approach of Naive Bayes.
[15]. Cureus. (2025). A Hybrid Naive Bayes Framework for Real-Time News Text Classification.

