ML-Based Resume Ranking & Personalized Interview System

Devarsh Patel
Machine Learning
Department of Applied Data Science
SID: 018208996
devarshnatvarlal.patel@sjsu.edu

Smit Ardeshana
Machine Learning
Department of Applied Data Science
SID: 018190016
smitarvindkumar.ardeshana@sjsu.edu

Dharmitkumar Patel
Machine Learning
Department of Applied Data Science
SID: 018189457
dharmitkumarsureshbhai.patel@sjsu.edu

Shatayu Thakur
Machine Learning
Department of Applied Data Science
SID: 017451096
shatayu.thakur@sjsu.edu

Lovely Priya
Machine Learning
Department of Applied Data Science
SID: 017625309
lovely@sjsu.edu

Mansi Tanna
Machine Learning
Department of Applied Data Science
SID: 018201755
mansisanjaybhai.tanna@sjsu.edu

Sai Swetha Madapati
Machine Learning
Department of Applied Data Science
SID: 018199974
saiswetha.madapati@sjsu.edu

***Abstract*—In the modern era of digitization, organizations are dealing with a deluge of job applications, rendering the recruitment process even more inefficient and time-consuming. Manual resume screening tends to result in delays, biases, and errors, thereby adding fuel to the troubles of HR professionals. Additionally, developing personalized interview questions from a candidate's resume is still a time-consuming and subjective task. This project needs an automated system that not only ranks resumes based on their relevance to job openings but also generates personalized interview questions for each applicant. The arrival of machine learning (ML) and natural language processing (NLP) has brought in an opportunity to enhance and automate various facets of the recruitment process. In spite of the advances made, there exists a gap in developing an effective AI system that can effectively screen resumes and, at the same time, develop personalized interview questions in real-time. The impetus for this project lies in building an intelligent system that has the capability to automate these two processes, thereby enabling human resource experts with a system that improves the precision of hiring, accelerates the recruitment process, and removes human bias.**

I. Introduction

Recruitment is one of the most labor and time-intensive activities for HR professionals today. With the number of resumes that arrive for every job posting so huge, to screen these applications manually is a time-inefficient and formidable task. Resume screening by hand is not just time-consuming but also susceptible to prejudice and human error, whereby potential candidates may be inadvertently overlooked. Besides, the task of creating individualized interview questions for each applicant from their resumes is also subjective and time-consuming. In light of the recent progress in Artificial Intelligence (AI) and Machine Learning (ML), the hiring process can be greatly improved with the use of automation, which can be used to make the process efficient, equitable, and objective. This project aims to address these issues by designing an AI-powered system that will be able to automate resume ranking along with generating customized interview questions depending on the individual's qualifications and job description.

The overall objective of this project is to create an end-to-end AI-Powered Resume Ranker and Interview Question Generator. The system will rank the resumes based on how relevant they are to job descriptions, using semantic analysis techniques powered by Natural Language Processing (NLP). By sorting resumes against the skills, experience, and qualifications mentioned in the job advertisement, the system will allow HR professionals to quickly pinpoint the most qualified candidates. In addition to ranking the resumes, the system will offer tailored interview questions with an NLP model like GPT or T5 so that each question is maximally relevant to the candidate's qualification and job requirements. This dual nature of the system will not only relieve some of the pressure on HR departments but also improve the accuracy and objectivity of the hiring process.

It is driven by the need to enhance the recruitment process to be more efficient and objective. With automation of resume ranking and interview question generation, HR professionals will have more time for high-level decision-making rather than spending time on low-level tasks. The system will leverage

advanced NLP and machine learning techniques, including semantic similarity scores and machine learning classification models, to score resumes and generate interview questions. It will also feature SHAP (Shapley Additive Explanations) to provide explainability in the decision-making process so that HR professionals can understand the basis of the model's predictions.

The end result of this project is expected to be a strong, data-driven platform that streamlines the recruitment process, reduces bias, and enhances the candidate assessment experience. By automating these vital processes of hiring, the system will allow organizations to save time, increase hiring accuracy, and attain a more unbiased way of selecting candidates. The scalability of the system and its applicability across different job fields and industries will also add to its worth, particularly in mass hiring situations where screening by hand and interview preparation are not feasible.

II. Project Background and Execute Summary

A. *Project Background*

Recruitment is likely the most critical but most resource-intensive function within an organization. In traditional hiring, HR managers would typically be faced with the issue of screening scores of resumes for a single job post, which is a labor-wasting, boring, and even biased activity. Furthermore, crafting customized interview questions for each candidate based on his/her credentials is also labor-intensive and calls for expertise. As technology advanced with Artificial Intelligence (AI) and Machine Learning (ML), there's a possibility of automating such manpower-intensive jobs, thereby rendering the recruitment more efficient and better hiring outcomes. AI-based software can shortlist resumes based on relevance and auto-create interview questions, and hence offer a product that is time-efficient, devoid of bias, and enhances efficiency and objectivity in the hiring process.

It is in this context that the AI-Powered Resume Ranker and Interview Question Generator was conceived. The project seeks to develop an automated system of machine learning where resumes would be ranked according to how well they match job specifications and interview questions created from the resume of the candidate and job specifications. Through the incorporation of Natural Language Processing (NLP) techniques of text analysis and machine learning techniques of resume classification, this system will automate two critical aspects of recruitment: interview question generation and resume screening. The incorporation of SHAP (Shapley Additive Explanations) will ensure that the decision-making process of the system is transparent, enabling HR professionals to realize how the model arrived at its conclusions.

B. *Executive Summary*

This project will develop an AI-Powered Resume Ranker and Interview Question Generator to automate and optimize the recruitment process. The system will first rank resumes based on their relevance to a given job description using semantic analysis via NLP techniques such as Sentence-BERT or Cross-Encoder BERT to determine semantic similarity between resumes and job descriptions. This will allow HR professionals to identify the most suitable candidates in a short time. Apart from that, the system will generate personalized interview questions for each shortlisted candidate using an NLP model like GPT or T5 that will be tailored based on the skills and qualifications mentioned in the resume and match them up with the job posting.

III. Project Requirements

To efficiently build the AI-Powered Resume Ranker and Interview Question Generator, there are various functional, non-functional, and data requirements that must be met so that the system is efficient, secure, and scalable.

Some of the key functional requirements are a series of processes to filter resumes and job ads, rank candidates in relevance, and generate tailored interview questions. As the initial step, the system will have to be capable of accepting data ingestion from resumes and job descriptions in their various forms like CSV, PDF, or Word. These data subsequently will have to undergo preprocessing where the text cleaning and normalization must be done, eliminating stop words, special characters, and punctuation. Feature extraction from text follows next, which involves semantic similarity between resumes and job descriptions, skills matching, experience score, and education matching. These will be processed through NLP techniques, using a model like Sentence-BERT or Cross-Encoder BERT to convert the text into embeddings that are matched against for similarity.

Once the data is preprocessed and features are extracted, machine learning algorithms like Logistic Regression, Random Forest, XGBoost, and SVM must be implemented to classify resumes as appropriate or inappropriate for a specific job on the basis of the extracted features. The system must also have the capability of fine-tuning these models by utilizing GridSearchCV or RandomizedSearchCV in order to optimize the performance. It also needs to integrate SHAP (Shapley Additive Explanations) so that the model predictions are made explainable and transparent to HR professionals so that they can understand why a particular resume was shortlisted or rejected. The second major functionality of the system will be creating customized interview questions using an NLP model like GPT or T5, which will read through the resume and job description to create specific and relevant questions based on the candidate's profile.

Non-functional requirements are also essential in making the system reliable, secure, and scalable. The system needs to scale with large datasets, especially in cases of high-volume hiring, without significant degradation in performance. Ranking of the resumes and generating questions must take place

within a reasonable time so that HR teams are not left with any downtime. Security is utmost priority because the system will be dealing with sensitive candidate data. Therefore, data must be encrypted and anonymized when stored or in transit. Also, the system should be user-friendly, with a minimal interface, especially if the web application is developed with Streamlit.

From a data perspective, the system will require high-quality resumes and job descriptions to train upon. Datasets must be diverse in terms of a wide range of job titles, industries, and formats to allow the model to generalize to different hiring scenarios. Resume data must include primary sections such as skills, experience, education, and certifications, while the job descriptions should clearly outline the qualifications, duties, and skills required for each job.

Hardware and software-wise, the system should be compatible with widely used operating systems such as macOS, Windows, and Linux, and it should be Python 3.x compatible. Major libraries such as scikit-learn, spaCy, pandas, and nltk will be used for data preprocessing, feature extraction, and machine learning processes. TensorFlow or PyTorch will be used for implementing deep learning models for NLP operations, while SHAP will be used for model explainability. The system should be compatible with Python-based deployment environments, including cloud environments if necessary.

By and large, the project requires rigorous synergy of NLP practices, machine learning, and data engineering to automate the interview preparation process and resume screening and make it scalable, user-friendly, as well as secure.

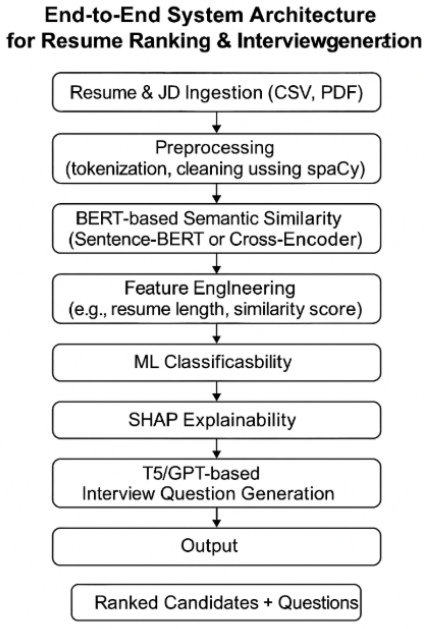

Fig. 1. Methodology

IV. Technology and Solution Survey

The development of the AI-Powered Resume Ranker and Interview Question Generator entails bringing together several cutting-edge technologies in Natural Language Processing (NLP), Machine Learning (ML), and data engineering. These will be used to automate resume screening, generate tailored interview questions, and render machine learning models explainable and transparent. In this survey, we will present the key technologies and solutions used in the project to achieve these tasks.

One of the core components of this solution is Natural Language Processing (NLP), which is solely responsible for processing and understanding text data from resumes and job postings. NLP facilitates text data cleaning, preprocessing, and analysis, resulting in relevant features for resume ranking and interview questions generation. Certain of the most critical NLP techniques, such as tokenization, lemmatization, and stopword removal, are employed to pre-process the data to be evaluated further. Sentence-BERT and Cross-Encoder BERT are employed for semantic text matching, which is a crucial part of analyzing whether a resume is relevant to a job posting. These transformer models leverage pre-trained word vectors to represent text in dense representations and allow the system to compute cosine similarity between resumes and job descriptions. Using these models ensures semantic analysis of high quality, which further enables the system to evaluate candidate suitability accurately and efficiently.

Apart from NLP techniques, Machine Learning (ML) models form the backbone of the classification process. After identifying key features from the resumes and job descriptions, they are fed into various ML classifiers for resume ranking and shortlisting. Support Vector Machines (SVM), Logistic Regression, Random Forest, and XGBoost are utilized to classify a resume as being either suitable or not suitable for a specific role. Such algorithms are best suited for classification problems, and their performance can be optimized with the tuning of hyperparameters through techniques such as GridSearchCV or RandomSearchCV. Through the use of these ML algorithms, it can be ensured that the system will be able to make accurate, data-driven decisions on which candidates to interview.

To make the predictions of the ML models more understandable and interpretable for HR professionals, SHAP (Shapley Additive Explanations) is incorporated into the system. SHAP is a powerful technique for model interpretability of machine learning models, providing a notion of what features contributed most to a model's decision. By explaining these contributions through SHAP summary plots and waterfall plots, the system allows for interpretation of the reasons behind resume shortlisting and rejection, which allows for trust in the automated process as well as fairness. This allows HR professionals to view

the rationale behind the AI-suggested rankings and interview questions.Generative Pre-trained Transformers (GPT) or T5 models are utilized by the system to come up with personalized interview questions. They are trained to generate coherent and contextually relevant text based on input prompts. The input here is the cleaned job description and resume text, and the output is a list of interview questions based on the skills and qualifications of the candidate. GPT and T5 are perfectly applicable for this task since they are capable of producing human-like text and can learn to adapt to various contexts, which makes them ideal for creating customized interview questions according to the requirements of each job.

Technically, the project is built on Python 3.x, which boasts an extensive collection of libraries for machine learning, NLP, and data processing. Data manipulation and handling of large data are accomplished with the help of libraries like pandas and NumPy, while text processing and NLP operations are carried out with spaCy and nltk. scikit-learn is employed for the machine learning component to easily implement and evaluate the classifiers. TensorFlow or PyTorch frameworks are employed for deep learning applications, particularly with NLP models. Streamlit is used in the development of the web interface (wherever applicable), providing an easy-to-use interface for HR professionals for resume uploading, ranking display, and interaction with the system.

The system must also have a robust backend infrastructure for handling the data, model training, and making real-time predictions. The system can be deployed on local machines or cloud systems, depending on the scope of the implementation. For larger datasets and more intensive processing, cloud-based solutions like AWS or Google Cloud can be utilized in order to make it scalable and performant. The system must ensure that sensitive candidate data is protected, employing encryption and anonymization processes to provide data privacy and comply with security requirements.

With the application of such technologies, the AI-Powered Resume Ranker and Interview Question Generator intends to offer an extremely efficient, transparent, and scalable solution for the new-age recruitment practices. With the application of NLP, ML, and Generative Models, the HR departments can automate processes, reduce human bias, and at last make smarter and data-driven recruitment decisions.

V. Data and Project Management Plan

A. *Data Management Plan*

*1) **Data Collection**:*

- Resumes: The data frame contains 15000 rows of resumes under 29 categories such as ”Data Science” and potentially more (based on the ”Category” column).
- Job Descriptions: Job descriptions are not available in this specific data frame, but you can merge this data frame with job description data so that you can match resumes for ranking.

*2) **Data Storage**:*

- The data is stored in a CSV file (resumes_dataset.csv) for easy access and processing.
- Storage Location: The dataset will be stored securely, potentially in a cloud storage system such as AWS S3 or Google Cloud Storage for scalability and data access management.

*3) **Data Access**:*

- The data set will be accessed through Python and libraries such as pandas to read and preprocess data.
- Access to the data set will be restricted to maintain privacy, especially if sensitive data (e.g., personal information) is involved.

*4) **Data Processing**:*

- Resume text will be preprocessed to normalize, eliminate stopwords, and extract helpful features such as skills, experience, and education. Tokenization is included in this.
- Preprocessing can include converting the entire text to lowercase, removing special characters, and addressing encoding issues.

B. *Project Development Methodology (CRISP-DM)*

Business Understanding The goal is to develop an AI-powered resume ranking and interview question generation system that uses resumes and job descriptions (when provided) to streamline the hiring process.

*1) **Data Understanding:** :* We will begin by examining the dataset (resumes_dataset.csv), performing exploratory data analysis (EDA) to understand the contents and structure of the resumes and their categories.

*2) **Data Preparation:** :* Data will be cleaned, and text features such as skills, experience, and education will be extracted from the Resume column.

*3) **Text preprocessing**:* **I**t will include tokenization, lemmatization, and removal of stopwords.

*4) **Modeling:** :* Machine learning models (e.g., Logistic Regression, Random Forest, SVM) and deep learning models (e.g., BERT) will be used to rank resumes based on job descriptions and generate interview questions.

*5) **Evaluation:** :* Model performance will be evaluated based on metrics such as accuracy, precision, recall, and F1-score. SHAP will be used to explain the model predictions.

*6) **Deployment:** :* The final model will be deployed in a web application (using Streamlit) or integrated into an HR tool for real-time resume evaluation and interview question generation.

C. *Project Schedule*

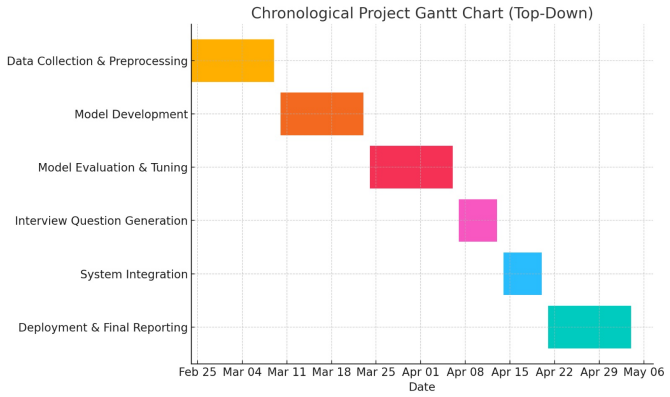

Fig. 2. Gantt Chart

VI. Model Development

A. *Model Comparison and Justification*

Feature	Logistic Regression	Random Forest	XGBoost	LightGBM
Type	Linear, binary classifier	Ensemble (bagging)	Boosted trees	Histogram-based boosting
Data Types	Numerical & categorical	Numerical & categorical	Numerical & categorical	Numerical & categorical
Robustness	Weak to noise, irrelevant feats	Good, via averaging	Strong, handles missing values	Strong, handles noise & missing values
Scalability & Data Size	Small datasets only	Medium to large datasets	Large datasets, distributed OK	Excellent scalability, low ``memory use
Generalization Issues	Overfits, underfits w/ few feats	Rarely over/underfits	Rarely over/underfits	Handles overfitting, avoids underfitting
Interpretability	High (simple math)	High (feature importances)	Medium (complex but explainable)	Medium (interpretable feature scores)
Performance	Good baseline	Strong for structured data	Best for complex interactions	High accuracy, fast training

Fig. 3. Model Comparison and Justification

B. *Model Evaluation Methods*

These models in the project will be gauged using several key metrics to identify their performance. They include Accuracy, Precision, Recall, F1-Score, and Support. Each of these has a unique role in ascertaining the performance of the model, especially for imbalanced data like resumes where some categories may be underrepresented.

- **Accuracy:** $\frac{\text{Correct Predictions}}{\text{Total Predictions}}$
  - **Explanation:** It calculates the ratio of correct predictions to overall predictions. It is an easy and intuitive measure of model performance.
  - **Impact on Model:** Accuracy is misleading in the case of imbalanced datasets as it will favor the majority class.
- **Precision:** $\frac{\text{TP}}{\text{TP} + \text{FP}}$
  - **Explanation:** Precision is about the number of the correctly identified positive instances. It is particularly useful when there is a high penalty for false positives.
  - **Impact on Model:** High precision means fewer false positives but at the cost of recall (missing some true positives).
- **Recall:** $\frac{\text{TP}}{\text{TP} + \text{FN}}$
  - **Explanation:** Recall is the proportion of actual positives that were correctly identified by the model. It is crucial when it is costly to miss relevant instances (false negatives).
  - **Impact on Model:** High recall ensures most positives are discovered, but may increase the number of false positives.
- **F1-Score:** $2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
  - **Explanation:** F1-Score is the harmonic mean of precision and recall, yielding a balance between the two metrics. It is helpful particularly when dealing with imbalanced datasets in which precision and recall are in contention.
  - **Impact on Model:** High F1-Score indicates high precision and high recall, which is ideal for testing models where balancing the two is needed.
- **Support:** Number of actual instances in the dataset.
  - **Explanation:** It is the number of actual instances per class. High support guarantees that metric values are more reliable, as the model is being tested on a larger number of instances.
  - **Impact on Model:** Higher support guarantees that the evaluation metrics are reliable and representative of the model's general performance.

Model	Key Insights	Strengths	Constraints
Logistic Regression	Excellent precision (1.0) but very low recall (0.19); high accuracy (99.15%) due to class imbalance.	Simple, interpretable baseline. Fast training and inference.	Poor recall leads to many missed shortlisted candidates. Not suitable for imbalanced classification.
Random Forest	Strong balance between precision (0.74) and recall (0.95); F1: 0.83. Very few false negatives.	Ensemble model handles non-linearity and imbalanced data well. High overall performance.	Precision lower than accuracy suggests some false positives. Can be resource-intensive for many trees.
XGBoost	High recall (0.96) and decent F1 (0.70); more aggressive in predicting positives than RF.	Handles feature interactions well. Strong recall ensures few missed resumes.	Precision (0.55) is lowest—more false positives. Slightly lower F1 than Random Forest.
LightGBM	High recall (0.96) with better precision (0.67) than XGBoost. F1: 0.79. Balanced and robust.	Efficient training, even with large data. Best balance of recall, precision, and runtime.	Some false positives remain. Requires tuning for optimal performance.

Fig. 4. Qualitative Metrics

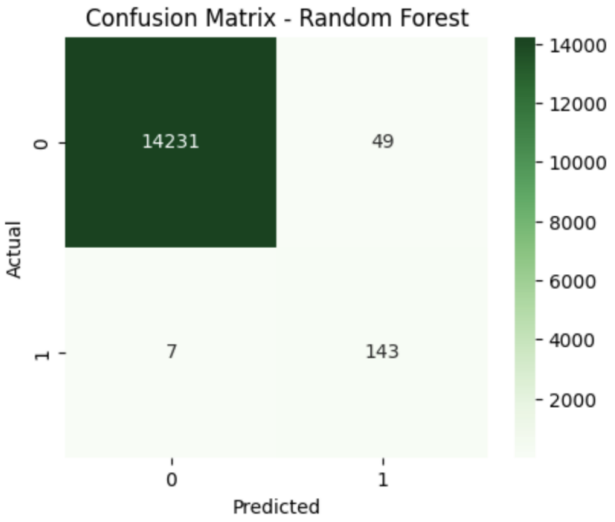

Fig. 6. Confusion Matrix of Random Forest

C. *Model Validation and Evaluation Results*

1) Logistic Regression: Logistic regression model is very accurate for class 0 but struggles to classify class 1 resumes, correctly classifying only 28 of them. It also has 122 false negatives, reflecting low sensitivity and class imbalance issues. Such a model will not be suitable in scenarios where it is critical to rank highly qualified resumes (class 1) accurately.

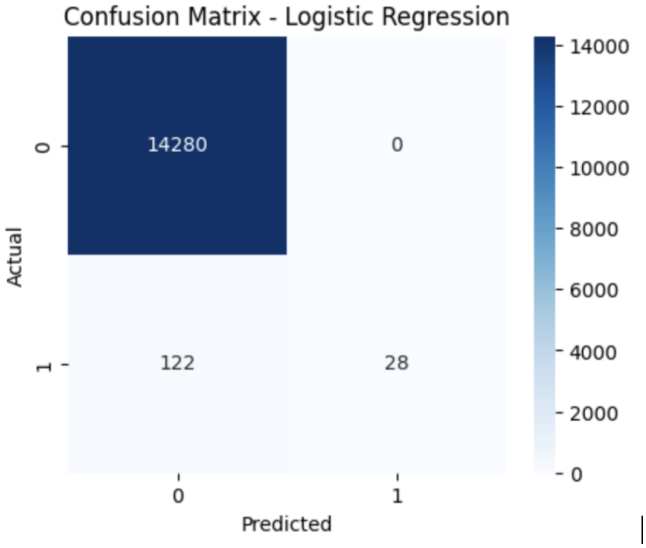

Fig. 5. Confusion Matrix of Logistic Regression

2) Random Forest Classifier: Random forest model works considerably better, with 7 false negatives and 49 false positives. It accurately predicts in 143 instances that are of class 1 and reflects a well-balanced performance. This signifies good generalization and stability and is a better-fitting choice compared to logistic regression.

3) XGBoost: XGBoost does great, correctly classifying 144 of 150 class 1 resumes and keeping false negatives as low as 6. It has 120 false positives, but its superior recall makes it a good fit for resume shortlisting scenarios where missing a good candidate comes with high opportunity cost.

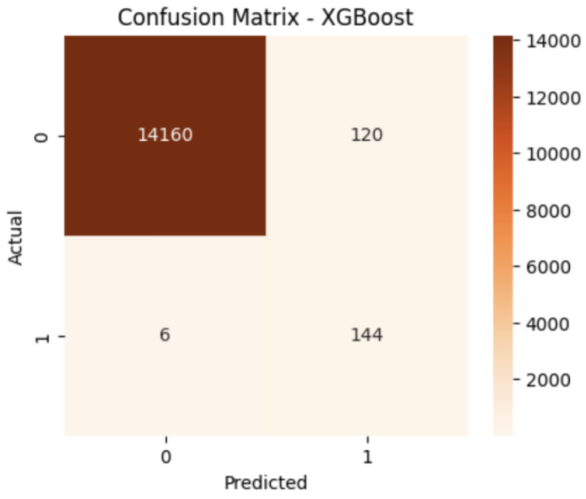

Fig. 7. Confusion Matrix of XGBoost

4) LightGBM Classifier: LightGBM is similar to XGBoost in accurately classifying all but 6 resumes in class 1 and performing slightly better with fewer misclassifications (70). It displays a great balance of precision and recall, and it is an outstanding performer for this classification task.

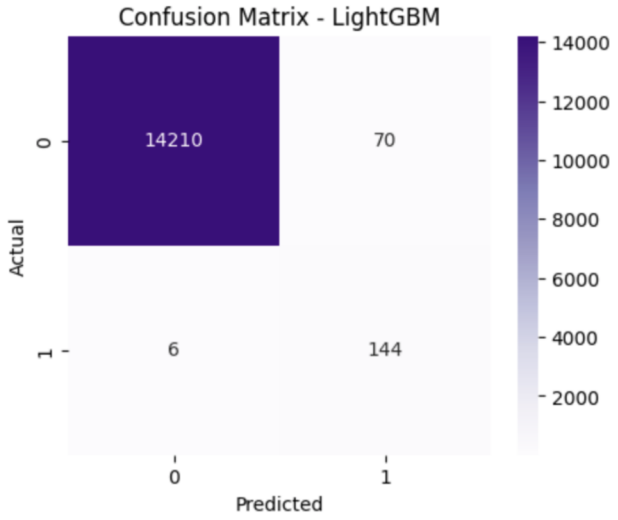

Fig. 8. Confusion Matrix of LightGBM Classifier

D. Model Validation and Evaluation Results

Each model will undergo a thorough evaluation process using the metrics above, and the results will be validated based on their performance on a separate test set. Here's the process:

*1) **Hyperparameter Tuning**:* GridSearchCV or RandomizedSearchCV will be used for hyperparameter tuning. These methods search for the best combination of hyperparameters that lead to the highest performance for each model.

- GridSearchCV exhaustively tests all combinations of hyperparameters in a specified grid.
- RandomizedSearchCV samples random combinations of hyperparameters, offering a faster alternative to GridSearchCV.

*2) **SHAP for Model Explainability**:* To enhance transparency, SHAP (Shapley Additive Explanations) will be used to explain the predictions made by each model. SHAP values indicate the contribution of each feature (e.g., skills, experience) to the final prediction.

- SHAP Visualizations: The visual output, such as waterfall plots, will show how each feature influenced the model's decision for each individual resume. This helps ensure the model is making predictions based on relevant factors.

*3) **Conclusion of Model Evaluation**:*

- Based on the accuracy, precision, recall, F1-Score, and SHAP analysis, the model that performs best will be selected for deployment in the final system.
- Random Forest is most balanced and reliable model. It provides excellent recall, strong precision, and the least trade-off between missing good candidates or recommending bad ones.

E. Model Comparison Summary

Model	Accuracy	Precision	Recall	F1 Score	TN	FP	FN	TP
Logistic Regression	0.9915	1.0000	0.1867	0.3146	14280	0	122	28
Random Forest	0.9961	0.7448	0.9533	0.8363	14231	49	7	143
XGBoost	0.9913	0.5455	0.9600	0.6957	14160	120	6	144
LightGBM	0.9947	0.6729	0.9600	0.7912	14210	70	6	144

Fig. 9. Comparison of Classification Metrics Across ML Models

VII. Screenshots

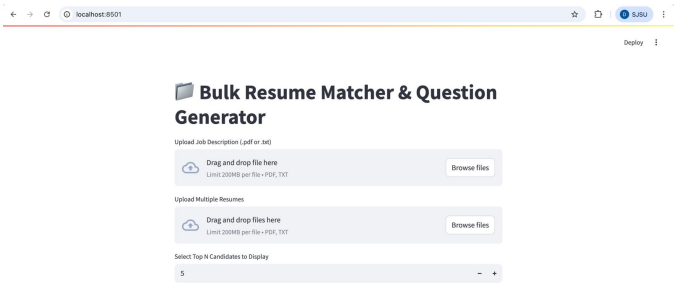

Fig. 10. Main Screen

Fig. 11. Upload JD

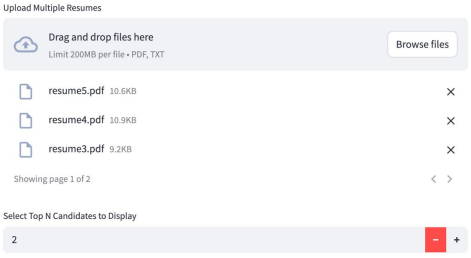

Fig. 12. Upload Resume

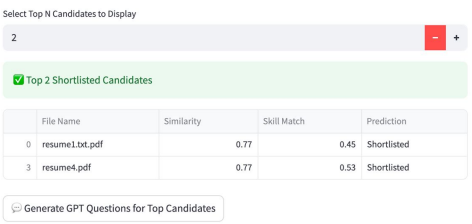

Fig. 13. Shortlisted Candidates

Fig. 14. Main Screen 2

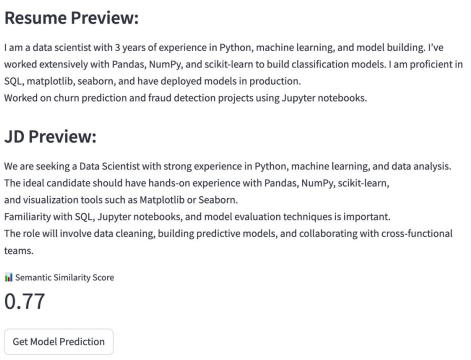

Fig. 15. Preview and Semantic Similarity Score

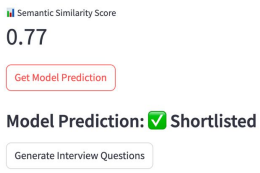

Fig. 16. Final Prediction

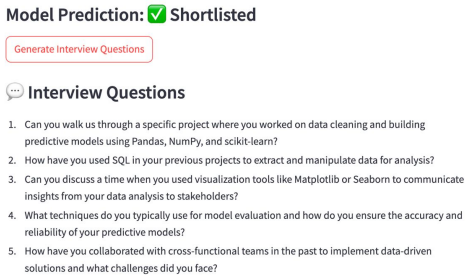

Fig. 17. Generate Interview Questions

CONCLUSION

*1) **Summary**:* In this project, we built a machine learning system to automate resume shortlisting by comparing resumes to job descriptions. We evaluated traditional and ensemble models—Logistic Regression, Random Forest, XGBoost, and LightGBM—to determine which best balances precision, recall, and efficiency. Our analysis revealed that Random Forest consistently delivered the most reliable results, providing a strong balance between catching suitable candidates and minimizing irrelevant ones. Unlike transformer models like BERT, which were not used here, our approach focuses on engineered features and classic ML techniques for transparency, speed, and explainability.

*2) **Benefits and Shortcomings**:* The system's key strength is scalability—it can process thousands of resumes in seconds, removing manual effort and reducing human bias. Random Forest and LightGBM were particularly effective, handling imbalanced data well and providing robust accuracy without requiring the heavy compute resources of deep learning models.

However, even the best models had trade-offs:

- Logistic Regression was too conservative.
- XGBoost was too aggressive.
- LightGBM and Random Forest hit the right balance but still require fine-tuning to minimize false positives.

*3) **Real-World Applications**:* This system is ideal for bulk hiring scenarios where speed and fairness matter most. Recruiters can instantly access a ranked list of top candidates, saving time while maintaining consistent evaluation standards. Beyond hiring, this model could be extended to other selection tasks, such as university admissions, internship filtering, or scholarship reviews.

*4) **Lesson Learned**:*

- Data quality is crucial: Cleaning and normalizing resume content had a major impact on model performance.
- Model trade-offs matter: Simpler models are faster, but ensemble models perform better.
- Explainability helps: Adding interpretability tools like SHAP gave insight into model decisions, improving trust

and usability.

*5) **Recommendations for Future Work**:*

- Extend support for multilingual resumes and domain-specific job roles (e.g., healthcare, legal).
- Implement active learning using recruiter feedback to refine predictions over time.
- Combine models—e.g., use a light model for initial filtering and Random Forest for final ranking.
- Fully integrate GPT-based interview generation for a seamless candidate experience.

*6) **Broader Impact**:* This system highlights how AI can augment hiring, not replace human judgment. By making resume screening faster, fairer, and more consistent, we can improve access to opportunities for applicants and reduce the workload on recruiters. It's a step toward more ethical, transparent, and data-driven talent acquisition.

7) : **GitHub Repository**

REFERENCES

[1] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," *arXiv preprint arXiv:1908.10084*, 2019. [Online]. Available: https://arxiv.org/abs/1908.10084

[2] S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, vol. 30, pp. 4765–4774, 2017.

[3] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

[4] T. Wolf, L. Debut, V. Sanh, J. Chaumond, et al., "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on EMNLP: System Demonstrations*, pp. 38–45, 2020.

[5] A. Vaswani, N. Shazeer, N. Parmar, et al., "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[6] Kaggle, "Resume dataset for skill extraction," 2020. [Online]. Available: https://www.kaggle.com/datasets

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[8] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

[9] Google Research, "T5: Exploring the limits of transfer learning with a unified text-to-text transformer," *arXiv preprint arXiv:1910.10683*, 2020. [Online]. Available: https://arxiv.org/abs/1910.10683

[10] The SHAP Library, "SHAP documentation," 2023. [Online]. Available: https://shap.readthedocs.io

[11] Explosion AI, "spaCy: Industrial-strength NLP in Python," [Online]. Available: https://spacy.io/

[12] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," *arXiv preprint arXiv:1603.02754*, 2016. [Online]. Available: https://arxiv.org/abs/1603.02754

[13] Hugging Face, "Transformers library documentation," [Online]. Available: https://huggingface.co/docs/transformers

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://scikit-learn.org/