ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

AI-POWERED JOB MATCHING SYSTEM TO CLASSIFY JOB CANDIDATES AND MATCH THEM WITH SUITABLE JOB OPENINGS

Uradi Rakesh,
UG Student, Department of CSE,
St. Martin's Engineering College,
Secunderabad, Telangana, India
rakeshuradi48@gmail.com

Mr. N. Balaraman,
Assistant Professor, Department of CSE,
St. Martin's Engineering College,
Secunderabad, Telangana, India
nbalaramancse@smec.ac.in

Abstract- ***The AI-powered Job Matching System that classifies job candidates and matches them with suitable job openings, optimizing the recruitment process for both job seekers and employers. The system is designed to efficiently process and analyze large volumes of job data and candidate profiles, delivering accurate and personalized job recommendations while improving the quality of hires. The system gathers data from job postings, which include details such as job titles, required skills, experience, education, location, and job type, alongside candidate data, including resumes, job history, qualifications, and preferences. Natural Language Processing (NLP) techniques are employed to preprocess text data, such as cleaning, tokenization, and embedding generation. This structured data is then used to extract relevant features from both candidates and job descriptions, enabling efficient matching. To match candidates with suitable job openings, the system utilizes a combination of machine learning models, including Gradient Boosting, K-Nearest Neighbors (KNN), and Multi-Layer Perceptron (MLP). The first step of the matching process applies rule-based filters to quickly narrow down candidates based on basic criteria such as location, experience, and job type preferences. Subsequently, the Gradient Boosting model is used to rank candidates based on their overall fit for a particular job.***

Keywords: Gradient Boosting, K-Nearest Neighbors (KNN), and Multi-Layer Perceptron (MLP

I. INTRODUCTION

AI-powered job matching systems are transforming the recruitment process by automating candidate screening and job matching through machine learning algorithms. These systems analyze vast datasets, including resumes, job descriptions, and candidate profiles, to match candidates to jobs based on skills, experience, and preferences. This approach improves the quality of hires, reduces recruitment time, and enhances user experience. AI models like Gradient Boosting, KNN, and MLP ensure that job seekers are matched with jobs that best suit their qualifications and preferences, improving job satisfaction and retention. The system's ability to learn from data continuously makes it adaptable and capable of offering personalized recommendations. As the job market grows, especially in India, such AI-driven systems offer immense potential in optimizing recruitment efforts for both employers and job seekers, bringing precision and scalability to the process.

The motivation behind this research stems from the need to optimize the recruitment process using AI technologies. With the ever-growing volume of job seekers and openings in India, traditional hiring methods are no longer efficient enough to keep up with the demands of the job market. The current recruitment systems are slow, prone to human error, and do not always lead to the best matches between candidates and jobs. This research aims to create an automated system that improves the speed, accuracy, and fairness of candidate-job matching. By utilizing machine learning algorithms, this project explores how AI can significantly reduce hiring time, improve match quality, and remove human bias from the recruitment process. Furthermore, it opens new avenues for personalization, allowing job seekers to find roles more closely aligned with their qualifications and career goals.

The need for this research arises from the inefficiencies and limitations of traditional recruitment methods, especially in the context of the fast-paced and expanding job market in India. With millions of job seekers and a rapidly changing labor market, employers and recruiters struggle to identify the most suitable candidates quickly and accurately. AI-powered job matching systems can automate the initial screening process, significantly reducing manual effort and improving the speed of hiring. For job seekers, such systems offer personalized recommendations based on their qualifications, experience, and preferences, helping them find more relevant job opportunities. Furthermore, AI reduces human biases in candidate selection, leading to fairer and more diverse recruitment. This paper, therefore, addresses critical pain points in the recruitment process, improving efficiency for both employers and candidates.

II. RELATED WORK

AI-driven job matching systems have become essential in modern recruitment, streamlining candidate selection, optimizing hiring processes, and improving job-market efficiency. Traditional job matching relied on keyword-based searches and rule-based filtering, which often failed to capture the nuanced alignment between candidate qualifications and job requirements. Early approaches utilized basic Natural Language Processing (NLP) techniques and structured databases, but these systems struggled with contextual understanding, skill variations, and semantic ambiguities in job descriptions and resumes.

The emergence of deep learning and advanced NLP models, such as Bidirectional Encoder Representations from Transformers (BERT), Recurrent Neural Networks (RNNs), and Transformer-based architectures, has significantly enhanced job classification accuracy. BERT and similar models excel in understanding contextual relationships in textual data, improving candidate-job matching. Machine learning algorithms, including Random Forests, Support Vector Machines (SVMs), and K-Nearest Neighbors (KNN), have also been applied to analyze structured features like experience, skills, and industry relevance. Additionally, hybrid models integrating CNNs with LSTMs have demonstrated effectiveness in extracting relevant job-related features from unstructured text, further improving classification and ranking mechanisms.

Despite these advancements, several challenges persist in real-world applications. The lack of standardized resume formats, variations in job descriptions across industries, and the subjectivity of skill relevance pose significant hurdles in achieving consistent matching performance. Moreover, bias in training data, stemming from historical hiring patterns, can lead to unfair or discriminatory outcomes, necessitating bias-mitigation techniques. Another challenge lies in the dynamic nature of job markets, where skill demands evolve rapidly, making it difficult for static models to adapt without continuous learning mechanisms. Additionally, the scalability of AI-powered matching systems remains a concern, particularly in handling large-scale job boards with millions of candidates and job listings in real time.

Future research should focus on developing robust datasets that encompass diverse job sectors, enhancing fairness in AI models by implementing bias-mitigation techniques, and improving adaptive learning strategies that continuously update models based on evolving job trends. Additionally, integrating AI-driven job matching with Graph Neural Networks (GNNs) and Knowledge Graphs could enhance the representation of candidate-job relationships by considering complex interdependencies beyond simple text matching. Furthermore, deploying these systems on edge computing platforms and cloud-based architectures would facilitate scalable, real-time job recommendations. By addressing these challenges, AI-powered job matching can revolutionize hiring processes, improve workforce efficiency, and create a more intelligent and inclusive job market.

III. PROPOSED WORK

Step 1: Dataset

The process begins by gathering a job dataset that contains essential details about job openings, such as job titles, required skills, qualifications, and other relevant attributes. This dataset also includes information about job candidates, such as resumes, work experience, education, skills, and preferences. The dataset serves as the foundation for the matching system, enabling the system to understand the relationship between candidate profiles and job openings. A proper dataset with a diverse set of job categories, roles, and candidates is crucial for the performance and scalability of the system.

Step 2: Dataset Preprocessing

The next step involves data preprocessing, where raw data undergoes several transformation procedures. First, null values in the dataset are identified and handled, either by removing or imputing missing entries. The dataset is then split into two parts: features (X) and target (y). The features (X) represent the independent variables such as skills, experience, and location, while the target variable (y) represents the job fit score. Following this, the dataset is divided into training and test sets to ensure that the model learns from one portion of the data and is validated on another. This split ensures that the model generalizes well to unseen data.

Step 3: Existing GBR Regressor (Algorithm):

To facilitate machine learning, the categorical labels associated with each sound file are transformed into numerical values through label encoding. This process involves mapping each category to a unique integer, allowing the algorithms to interpret the labels numerically. This step is essential for effectively training classification models, as machine learning algorithms require numerical input.

Step 4: Existing KNN Regressor (Algorithm):

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

The second algorithm used is the K-Nearest Neighbors Regressor (KNN). KNN is a simple yet effective method that evaluates the job fit score of candidates by analyzing the proximity between the candidate's features and the job requirements. This model considers a predefined number of nearest neighbors and averages their outputs to make predictions. The KNN model is trained and evaluated using the same training and test sets. Performance metrics are computed, similar to GBR, to understand how well this model matches candidates to job openings based on their attributes.

Step 5: Proposed MLP Regressor (Algorithm):
The proposed approach in the system is the Multi-Layer Perceptron Regressor (MLP). MLP is a type of neural network that can model complex patterns in data. This algorithm is particularly effective for capturing intricate relationships between candidates' characteristics and job requirements. The MLP model consists of several layers of neurons that transform input data into output predictions. The model is trained using the same training set, and its predictions are evaluated on the test set. The performance of the MLP model is compared with GBR and KNN to assess improvements in job matching accuracy.

Step 6: Performance Comparison Graph: After training and predicting with the GBR, KNN, and MLP models, the performance of each algorithm is compared. The performance metrics for each model—such as R-squared (R^2), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE)—are displayed in graphical form. These graphs visually represent how each model performed, allowing for an easy comparison of their effectiveness in predicting job fit scores. This comparison helps in selecting the most suitable model for deployment.

Step 7: Prediction of Output from Test Images with MLP Regressor Algorithm Trained Model: The final step is the prediction phase, where the MLP Regressor model, having been trained on the job candidate and job opening data, is applied to new, unseen test data. Test candidates are processed, and their job fit scores are predicted using the trained MLP model. The system outputs a list of predicted job fit scores, providing tailored recommendations for job seekers. The predictions are displayed, and the results can be further analyzed to optimize the matching process, allowing both candidates and employers to make informed decisions.

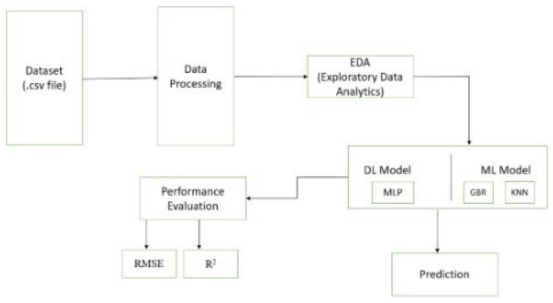


Figure 1: Architectural Block Diagram

4.2 Workflow Data Preprocessing:

Data preprocessing is a crucial step to ensure the dataset is clean and structured for machine learning models. In this project, the raw dataset, which contains job and candidate-related information, is cleaned and transformed to make it suitable for model training. The following preprocessing steps are implemented:

Handling Missing Values: Before any analysis, missing values are checked within the dataset. Missing values may occur in various columns like candidate qualifications, skills, or job descriptions. These missing values are either imputed (filled with the mean, median, or mode for numerical features or the most frequent value for categorical features) or removed based on their importance and the impact on the model's accuracy.

Encoding Categorical Features: Many features in the dataset, such as job title, skills, or location, may be categorical in nature (e.g., "Software Engineer," "Manager," "Remote"). These categorical variables need to be converted into numerical format to be processed by machine learning models. Common methods like one-hot encoding or label encoding are used to convert these textual data into numerical representations.

Scaling/Normalization of Features: Some features in the dataset, such as years of experience or job salary, may have different ranges and units. To avoid models being biased towards variables with larger values or ranges, these numerical features are scaled or normalized. Techniques like Min-Max Scaling or Standardization are used to ensure that all features are on a similar scale, which helps improve model performance, especially for algorithms like KNN and MLP.

Feature Selection: Irrelevant or redundant features can lead to overfitting or increase the complexity of the model without improving its performance. Feature selection techniques are applied to retain only the most relevant features (such as skills, education, and experience) that contribute to predicting job fit scores.

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

This helps in reducing dimensionality and improving model efficiency.

Text Data Processing: Since job descriptions and candidate profiles may contain free text (such as job titles, skills, and qualifications), Natural Language Processing (NLP) techniques are employed. Textual features are cleaned by removing stop-words, tokenizing words, and converting text into a numerical format using techniques like TF-IDF (Term Frequency-Inverse Document Frequency) or Word Embeddings, which are fed into the models.

Data Splitting Once the data is pre-processed and cleaned, it is divided into two main parts: features (X) and target (y).

Features (X): The features represent the input variables or independent variables that describe the job candidates and job openings. These may include attributes like skills, years of experience, education level, job title, location, etc. These features are used by machine learning algorithms to identify patterns and make predictions.

Target (y): The target variable, or dependent variable, is the output that the model aims to predict. In this project, the target is the "Job Fit Score," which represents how well a candidate matches a specific job opening. This score is a continuous variable and is predicted by the model based on the features in the dataset.

Train-Test Split: To evaluate the performance of the machine learning models effectively, the dataset is split into two sets: the training set and the test set. The training set is used to train the machine learning models, while the test set is used to evaluate the models' performance on unseen data. Typically, a 80-20 or 70-30 split is used, where 80% (or 70%) of the data is used for training, and the remaining 20% (or 30%) is reserved for testing.

Shuffling the Data: Before splitting the data, shuffling is done to ensure that the data is randomly distributed. This avoids any potential bias that may arise if the data is ordered in a particular way (e.g., all candidates from the same region appearing together). Shuffling helps the model generalize better to new, unseen data.

IV.RESULTS AND DISCUSSION

4.1 Implementation Description

Step 1: Job Dataset Upload

The research begins by uploading the job dataset, which contains the details of job candidates and job openings. The dataset includes various features, such as qualifications, skills, experience, and other relevant information, along with the job fit scores that indicate how well candidates align with job openings. The user is prompted to upload the dataset through a graphical interface, and the dataset is loaded into the system for further processing.

Step 2: Data Preprocessing and Data Splitting

Once the dataset is loaded, the next step is data preprocessing. The dataset is first examined for null values, and any missing values are handled accordingly, either by removal or imputation. After cleaning the dataset, it is divided into input features (X) and the target variable (y), which is the "Job_Fit_Score." The dataset is then split into training and testing subsets, typically using an 80-20 split, where 80% of the data is used for training the machine learning models, and 20% is reserved for evaluating the model's performance.

Step 3: Existing Gradient Boosting Regressor (GBR) Model

The Gradient Boosting Regressor (GBR) algorithm is applied to the training dataset. GBR builds an ensemble of weak learners (decision trees), where each new learner corrects the errors of the previous one. The model is trained on the training data, and the predictions are made on the test data. The performance of the GBR model is evaluated using standard regression metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2). These metrics provide insights into how well the GBR model predicts the job fit scores.

Step 4: Existing K-Nearest Neighbors Regressor (KNN) Model

Next, the K-Nearest Neighbors (KNN) Regressor is implemented. The algorithm determines the "K" nearest neighbors to a given test point and predicts the job fit score by averaging the job fit scores of those neighbors. Similar to GBR, the KNN model is trained using the training dataset, and predictions are made on the test dataset. The model's performance is also evaluated using MAE, MSE, RMSE, and R^2 to assess its predictive accuracy.

Step 5: Proposed Multi-Layer Perceptron (MLP) Regressor Model

The proposed model, a Multi-Layer Perceptron (MLP) Regressor, is applied to the data. MLP is a type of neural network with multiple layers of neurons, where each layer processes the input data through an activation function to capture complex non-linear relationships. The model is trained using the training data, and the predictions for the job fit scores are generated for the test data. The MLP model's performance is evaluated using the same regression metrics as GBR and KNN, allowing a comparison of the results.

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

Step 6: Performance Comparison

After training all three models (GBR, KNN, and MLP), the next step involves comparing their performance. A comparison is made based on the regression metrics: MAE, MSE, RMSE, and R². A graphical representation, such as a scatter plot or bar graph, is generated to visualize the performance of each model. This helps in determining which model provides the best prediction accuracy for the job fit scores.

Step 7: Job Fit Prediction with MLP Regressor

Finally, the trained MLP Regressor model is used to predict the job fit scores for new, unseen data. The user can upload test data containing job candidates' details, and the MLP model will generate the predicted job fit scores based on the trained model. The results are displayed in the interface, showing the predicted scores along with the corresponding input data, providing valuable insights for job matching.

4.2 Dataset Description

The dataset used in this project contains various features related to job candidates and job openings. The key columns in the dataset are as follows:

Candidate_Skills_Score
This column represents the score assigned to a candidate based on their skillset. It is a numerical value that evaluates how well a candidate's skills align with the job requirements.

Candidate_Experience
This column indicates the number of years of work experience a candidate has. It is a continuous numerical variable, where higher values suggest more experienced candidates.

Education_Level
The education level column categorizes the candidate's highest level of education. This can be a categorical feature with values such as "High School," "Undergraduate," "Graduate," or "Postgraduate," reflecting the educational qualifications of the candidate.

Certifications_Score
This column reflects the score assigned to a candidate based on the number and relevance of certifications they hold. It is a numerical value, where higher scores indicate more relevant or specialized certifications.

Job_Requirements_Complexity
This feature describes the complexity of the job requirements, representing how challenging or specialized the job is. It is a numerical value, where higher values correspond to more complex job roles.

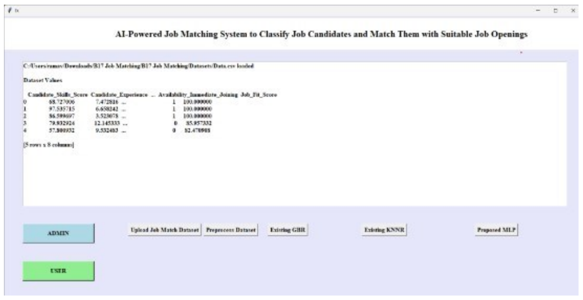
Job_Location_Match
The job location match column measures how well the candidate's preferred job location aligns with the job's location. It is a numerical value, with higher values indicating a better match between the candidate's preferred location and the job's location.

Availability_Immediate_Joining
This column indicates whether the candidate is available for immediate joining. It is a binary feature, with values such as 1 for candidates who are ready to join immediately and 0 for candidates who have a delayed availability.

Job_Fit_Score
The job fit score is the target variable that indicates how well a candidate is suited for a particular job opening. This score is a numerical value and represents the overall match between the candidate's profile and the job's requirements. The job fit score is used as the output for prediction in the machine learning model.

4.3 Results Description

The figure 2 presents the graphical user interface (GUI) used for uploading job datasets. The user is prompted to upload a dataset containing key attributes such as Candidate Skills Score, Experience, Education Level, Certifications, Job Requirements, and other relevant fields. Once the dataset is uploaded, the system processes it for analysis and further use in the job matching process. The figure highlights the clean and organized interface, where users can easily navigate through options and review the dataset before starting the preprocessing and training phases.

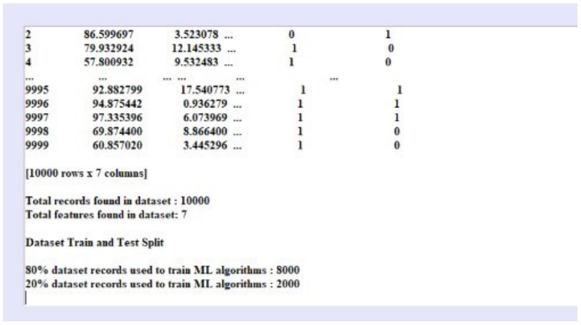

AI-Powered Job Matching System to Classify Job Candidates and Match Them with Suitable Job Openings
ADMIN
USER
Upload Job Match Dataset
Preprocess Dataset
Existing GBR
Existing KNNR
Proposed MLP

```
C:/Users/ramav/Downloads/B17 Job Matching/B17 Job Matching/Datasets/Data.csv loaded

Dataset Values

  Candidate_Skills_Score  Candidate_Experience  ...  Availability_Immediate_Joining  Job_Fit_Score
0            68.727006              7.472816  ...                               1     100.000000
1            97.535715              6.658242  ...                               1     100.000000
2            86.599697              3.523078  ...                               1     100.000000
3            79.932924             12.145333  ...                               0      85.957332
4            57.800932              9.532483  ...                               0      82.470985

[5 rows x 8 columns]
```

Figure 2: Upload Job Dataset

ISSN NO: 9726-001X

Volume 13 Issue 02 2025



```
2           86.599697      3.523078  ...       0              1
3           79.932924     12.145333  ...       1              0
4           57.800932      9.532483  ...       1              0
...            ...        ...  ...       ...             ...
9995          92.882799      17.540773  ...       1               1
9996          94.875442       0.936279  ...       1              1
9997          97.335396       6.073969  ...       1              1
9998          69.874400       8.866400  ...       1              0
9999          60.857020       3.445296  ...       1              0

[10000 rows x 7 columns]

Total records found in dataset : 10000
Total features found in dataset: 7

Dataset Train and Test Split

80% dataset records used to train ML algorithms : 8000
20% dataset records used to train ML algorithms : 2000
```

Figure 3: Data Preprocessing in the GUI

This figure shows the data preprocessing stage in the GUI. After the job dataset is uploaded, the system automatically processes the data to handle missing values, normalize data, and split the dataset into training and testing sets. The user can visually monitor the preprocessing steps, which include handling null values, ensuring the integrity of the dataset, and transforming the data into a format suitable for training machine learning models. This step is crucial for ensuring that the models perform optimally when predicting job matches.

Performance Metrics of the Models

Performance Metrics of Gradient Boosting Regressor (GBR)

- **Mean Absolute Error (MAE)**: 2.788966714252531
- **Mean Squared Error (MSE)**: 11.75114949349804
- **Root Mean Squared Error (RMSE)**: 3.4279949669592633
- **R-squared (R^2)**: 0.9435300216749172

These metrics indicate that the Gradient Boosting Regressor model performed well, with a low MAE and RMSE, suggesting that it made accurate predictions. The high R^2 value of 0.9435 reflects a good fit to the data, meaning that the model explained a significant portion of the variability in job match predictions.

Performance Metrics of KNN Regressor

- **Mean Absolute Error (MAE)**: 5.980638206411
- **Mean Squared Error (MSE)**: 63.14398777028975
- **Root Mean Squared Error (RMSE)**: 7.946319133428367
- **R-squared (R^2)**: 0.6965624832940389

The KNN Regressor model shows relatively higher errors compared to GBR, with a higher MAE, MSE, and RMSE. The R^2 value of 0.6966 indicates that the model has a lower fit to the data than the GBR model. While the model still provides useful predictions, the performance metrics suggest it does not capture as much of the variation in job matches.

Performance Metrics of MLP Regressor

- **Mean Absolute Error (MAE)**: 2.104308359447359
- **Mean Squared Error (MSE)**: 7.167948658257607
- **Root Mean Squared Error (RMSE)**: 2.677302496591972
- **R-squared (R^2)**: 0.96555452676429

The MLP Regressor performed exceptionally well, with the lowest MAE and RMSE, indicating that it made accurate predictions. Its R^2 value of 0.9656 reflects an excellent fit to the data, showing that the model can reliably predict job matches with high accuracy.

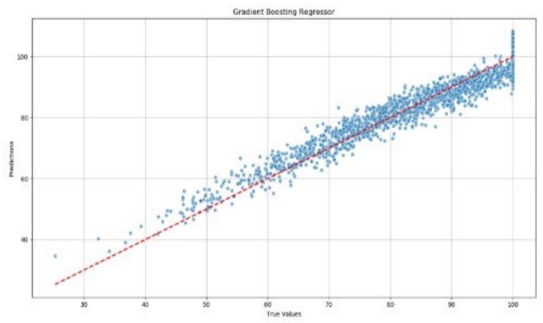


Figure 4: Performance Metrics and Regression Scatter Plot of GBR Regressor Model

This figure displays the performance metrics of the GBR Regressor along with its corresponding regression scatter plot. The scatter plot shows the predicted job matches versus the actual values, and the closeness of the points to the ideal diagonal line indicates the accuracy of the model. The performance metrics (MAE, MSE, RMSE, and R^2) are displayed on the side to give a quantitative view of the model's effectiveness in predicting job matches.

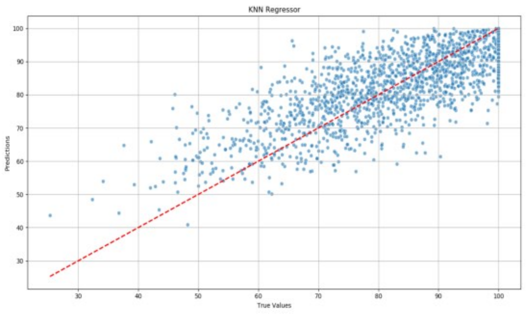


Figure 5: Performance Metrics and Regression Scatter Plot of KNN Regressor Model

This figure shows the KNN Regressor model's performance metrics and the regression scatter plot. While the MAE, MSE, and RMSE are higher than those for the GBR and MLP models, the scatter plot illustrates how the KNN model's predictions deviate from the actual job matches. The model's predictions are not as close to the ideal line, which corresponds to the lower R² value, showing less accuracy in its predictions.

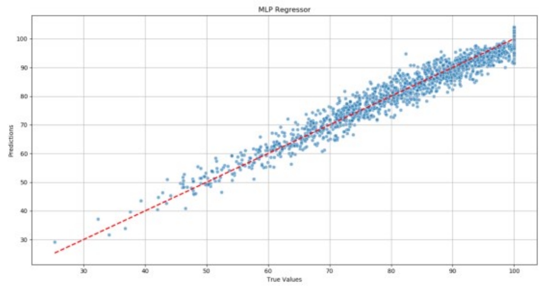
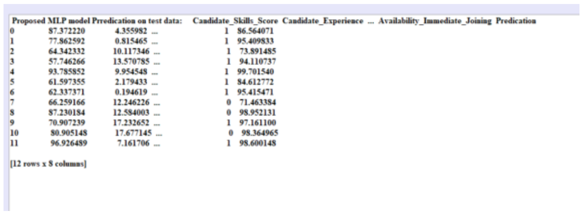


Figure 6: Performance Metrics and Regression Scatter Plot of MLP Regressor Model

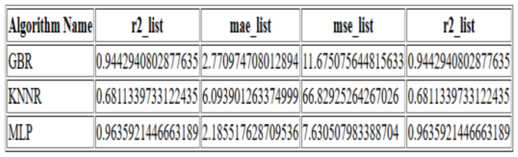
The MLP Regressor model's performance metrics and regression scatter plot are shown in this figure. The scatter plot demonstrates that the MLP model's predictions closely follow the ideal diagonal line, indicating highly accurate predictions of job matches. The low MAE, MSE, and RMSE values, along with the high R² value, further confirm the effectiveness of the MLP model for this task.



```
Proposed MLP model Prredication on test data:    Candidate_Skills_Score  Candidate_Experience  ...  Availability_Immediate_Joining  Predication
0          87.372220         4.355982  ...                1   86.564071
1          77.862592         0.815465  ...                1   95.409833
2          64.342332        10.117346  ...                1   73.891485
3          57.746266        13.570785  ...                1   94.110737
4          93.785852         9.954548  ...                1   99.701540
5          61.597355         2.179433  ...                1   84.612772
6          62.337371         0.194619  ...                1   95.415471
7          66.259166        12.246226  ...                0   71.463384
8          87.230184        12.584003  ...                0   98.952131
9          70.907239        17.232652  ...                1   97.161100
10          80.905148       17.677145  ...                0   98.364965
11          96.926489        7.161706  ...                1   98.600148

[12 rows x 8 columns]
```

Figure 7: Model Prediction on the Test Data

This figure illustrates how the trained models (GBR, KNN, MLP) make predictions on the test data. The system processes the input features, and each model generates its own set of predicted job matches. The comparison of predictions across all models is shown, highlighting the difference in accuracy and performance between the models.



Algorithm Name	r2_list	mae_list	mse_list	r2_list
GBR	0.9442940802877635	2.770974708012894	11.675075644815633	0.9442940802877635
KNNR	0.6811339733122435	6.093901263374999	66.82925264267026	0.6811339733122435
MLP	0.9635921446663189	2.185517628709536	7.630507983388704	0.9635921446663189

Figure 8: Performance Comparison Graph of All Models

The final figure presents a performance comparison graph of all the models—GBR, KNN, and MLP. This graph visually compares the performance metrics (MAE, MSE, RMSE, and R²) for each model. It is clear from the graph that the MLP model outperforms the GBR and KNN models, providing more accurate predictions of job matches. The comparison graph allows users to easily see which model is the best for the job matching task, based on the performance metrics.

V.CONCLUSION

The AI-powered job matching system developed in this project efficiently classifies job candidates and matches them with suitable job openings using machine learning models such as Gradient Boosting Regressor (GBR), K-Nearest Neighbors (KNN), and Multi-Layer Perceptron (MLP). The system processes and analyzes candidate profiles and job descriptions to deliver accurate, personalized job recommendations, optimizing the recruitment process for both employers and job seekers. By employing advanced algorithms, the system enhances the quality of hires, reduces manual effort, and ensures that candidates are matched to jobs that align with their skills, experience, and preferences. The performance of the models has been evaluated using various regression metrics, with the proposed MLP model providing the best predictions for job fit scores.

REFERENCES

[1] T. G. Sougandh, S. S. K, N. S. Reddy, and M. Belwal, "Automated resume parsing: A natural language processing approach," in 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS), 2023.

[2] J. Singh and S. Singh, "A natural language processing approach to extracting information from resumes," in 5th International Conference on Industrial Engineering and Applications (ICIEA), 2020.

[3] Rodriguez, L. G., & Chavez, E. P. (2019). Feature selection for job matching application using profile matching model. 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS). https://doi.org/10.1109/ccoms.2019.8821682

[4] Almalis, N. D., Tsihrintzis, G. A., Karagiannis, N., & Strati, A. D. (2015). FoDRA — A new contentbased job recommendation algorithm for job seeking and recruiting. 2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA). https://doi.org/10.1109/iisa.2015.7388018.

[5] Harris, C. G. (2017). Finding the best job applicants for a job posting: A comparison of human resources search strategies. 2017 IEEE International Conference on Data Mining Workshops (ICDMW). https://doi.org/10.1109/icdmw.2017.31.

[6] Chalidabhongse, J., Jirapokakul, N., & Chutivisarn, R. (2006). Facilitating job recruitment process through job application support system. 2006 IEEE International Conference on Management of Innovation and Technology. https://doi.org/10.1109/icmit.2006.262244.

[7] Mishra, R., Rodriguez, R., & Potillo, V. (2020). An AI Based Talent Acquisition and Benchmarking for Job.

[8] Koh, M. F., & Chew, Y. C. (2015). Intelligent job matching with self-learning recommendation engine. Procedia Manufacturing, 3, 1959-1965. https://doi.org/10.1016/j.promfg.2015.07.241.

[9] Lee, D., Kim, M., & Na, I. (2018). Artificial Intelligence based career matching. Journal of Intelligent & Fuzzy Systems, 35(6), 6061-6070. https://doi.org/10.3233/jifs-169846.

[10] What is natural language processing? Machine Learning Mastery. https://machinelearningmastery.com/natural-language-processing/ International Journal on Soft Computing, Artificial Intelligence and Applications (IJSCAI), Vol.11, No.3, August 2022.

[11] What is natural language processing? (2020, July 2). IBM - United States. https://www.ibm.com/cloud/learn/natural-language-processing.

[12] ML | Stochastic gradient descent (SGD). (2021, September 13). GeeksforGeeks. Doi:- https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/.

[13] International standard classification of occupations (ISCO). (n.d.). ILOSTAT. Doi:- https://ilostat.ilo.org/resources/concepts-and-definitions/classification-occupation.

[14] Scikit-learn. Retrieved December 7, 2021, from doi:- https://scikitlearn.org/stable/modules/classes.html#module-sklearn.linear_model.

[15] KP Sanal Kumar, S Anu H Nair, Deepsubhra Guha Roy, B Rajalingam, R Santhosh Kumar, 'Security and privacy preservation using Edge intelligence in beyond 5G networks harnessing Federated Machine Learning and Artificial Immune Intrusion Detection System" *Computers and Electrical Engineering*, Pergamon, Elsevier.

[16] R. Santhoshkumar, M. Kalaiselvi Geetha, 'Deep Learning Approach for Emotion Recognition from Human Body Movements with Feed forward Deep Convolution Neural Networks', *Procedia Computer Science,* Vol.152, pp.158-165, May 2019, Elsevier, ISSN 1877-0509.

[17] Rajalingam, B., Al-Turjman, F., Santhoshkumar, R. et al. Intelligent multimodal medical image fusion with deep guided filtering. *Multimedia Systems* 28, 1449–1463 (2022)

[18] R. Santhoshkumar, M. Kalaiselvi Geetha, 'Vision based Human Emotion Recognition using HOG-KLT feature' *Advances in Intelligent System and Computing, Lecture Notes in Networks and Systems*, Vol.121, pp.261-272, ISSN: 2194-5357.

[19] R. Santhoshkumar, M. Kalaiselvi Geetha, 'Emotion Recognition on Multi View Static Action Videos using Multi Blocks Maximum Intensity Code (MBMIC)', *Lecture Notes in Computational Science and Engineering* ,Springer, ISSN 1439-7358.