Fake Job Advertisement Detection System Using Machine Learning

INTI NAGA LAKSHMI SUREKHA

PG Scholar. Department of MCA, DNR College, Bhimavaram, Andhra Pradesh

K. Rambabu

(Assistant Professor), Master of Computer Applications, DNR College, Bhimavaram, Andhra Pradesh

ABSTRACT

The rapid growth of online job portals and social media platforms has made it easier for individuals to search for employment opportunities. However, this digital expansion has also facilitated the proliferation of fraudulent job advertisements, which exploit job seekers by promising lucrative employment opportunities that do not exist. Such fake job ads can lead to financial loss, identity theft, and other forms of exploitation, creating a significant need for an automated system to identify and prevent these scams.This project proposes a **Fake Job Advertisement Detection System** leveraging machine learning algorithms to distinguish between legitimate and fraudulent job postings. The system is implemented as a web-based application using **Django**, providing secure user authentication for both regular users and administrators. Users can input job ad content either as text or image files. For image-based ads, the system simulates OCR-like processing to extract textual information. The extracted or provided text is then vectorized using a **TF-IDF vectorizer**, and predictive models, primarily **XGBoost**, classify the ad as either legitimate or fake.The system incorporates robust model evaluation metrics, including accuracy, F1-score, and recall, which are displayed in a comprehensive admin dashboard alongside a comparison of multiple models such as Random Forest, KNN, and SVM. Users receive immediate feedback on their input, with clear indicators highlighting whether the job ad is suspicious. Additionally, the system handles model loading failures gracefully and informs users in case predictions cannot be performed, ensuring a reliable user experience.

This solution provides a dual benefit: it protects individual job seekers from fraud while also giving administrators insights into model performance and user activity. By combining traditional machine learning techniques with modern web technologies, the system offers a scalable, efficient, and user-friendly approach to tackling the growing issue of fraudulent job advertisements. Future improvements may include integration with real OCR tools, natural language processing enhancements, and continual retraining on updated datasets to improve detection accuracy.

Keywords: Fake Job Ads, Machine Learning, XGBoost, Text Classification, Job Fraud Detection, OCR, Django Web Application, User Authentication, Model Evaluation, Dashboard Visualization

I. INTRODUCTION

The modern job market increasingly relies on digital platforms for recruitment, with millions of job advertisements posted daily on websites, social media, and professional networks. While this digitization has made job search faster and more convenient, it has also created fertile ground for fraudulent activity. Fake job advertisements are deceptive postings created to exploit job seekers financially or to collect personal information for malicious purposes.Job seekers affected by these scams often suffer monetary losses, identity theft, or even emotional distress. The primary challenge lies in the subtlety of these fake ads: they often mimic the style and tone of legitimate postings, making manual detection difficult. Traditional approaches, such as manual verification by administrators or reporting mechanisms, are time-consuming and often insufficient given the scale of online postings.Machine learning (ML) offers a promising solution by automating the detection process. ML models can analyze patterns in text content, such as specific keywords, language structures, and unusual claims, to distinguish between legitimate and fraudulent ads. Using supervised learning, models can be trained on labeled datasets containing both fake and genuine job advertisements to achieve high detection accuracy.This project implements a web-based **Fake Job Advertisement Detection System** using Django, integrating machine learning models to classify job ads. Users can input job ads either directly as text or upload images containing ad content. The system extracts textual information, vectorizes it using **TF-IDF**, and uses a trained **XGBoost classifier** to predict the likelihood of the ad being fraudulent. The web interface allows real-time feedback, helping users avoid falling victim to scams.The system also includes an administrative dashboard that provides insights into model performance across various metrics, such as accuracy, F1-score, and recall. This feature ensures that the system remains transparent and allows for continuous improvement. By combining machine learning with an interactive web interface, the system offers both preventive and analytical benefits.The growing reliance on digital platforms for employment highlights the urgent need for automated fake job ad detection. This project aims to contribute to a safer online job market by providing an accessible, reliable, and scalable solution for both job seekers and platform administrators.

II. LITERATURE SURVEY (WITH EXISTING METHODS)

Several studies have explored techniques for detecting fraudulent job advertisements. Traditional approaches rely on manual verification or user reporting, which are not scalable. Recent research focuses on using **machine learning and natural language processing (NLP)** for automated detection.

1. **Text Classification Approaches**: Many studies use supervised learning algorithms to classify ads based on textual features. Keywords indicating urgency, financial incentives, or vague job descriptions often correlate with fraudulent postings. Common algorithms include **Random Forest, SVM, and XGBoost**, which show high accuracy in binary classification tasks.

2. **Feature Engineering**: Features extracted from job ad text include term frequency–inverse document frequency (**TF-IDF**), n-grams, and sentiment analysis. Some studies also use lexical patterns, such as capitalization, punctuation, and presence of certain financial terms, as indicators of fraud.
3. **Deep Learning Techniques**: Advanced methods utilize **LSTM and Transformer-based models** for sequence modeling. These models capture contextual relationships in the text and have shown improved performance over traditional classifiers. However, they require larger datasets and more computational resources.
4. **Image-based Detection**: A few studies incorporate optical character recognition (**OCR**) to analyze job ads shared as images on social media. Text extracted from images can then be classified using standard text-based ML models.
5. **Evaluation Metrics**: Studies consistently emphasize **accuracy, precision, recall, and F1-score** for evaluating model performance. These metrics ensure that the system is reliable in identifying fraudulent ads without misclassifying legitimate ones.
6. **Limitations of Existing Systems**: Current systems often lack user-friendly interfaces, are limited to text-only inputs, or require manual OCR preprocessing for images. Many fail to provide real-time feedback, reducing their practicality for everyday job seekers.

This project addresses these gaps by offering a **web-based interface**, handling both text and image inputs, performing real-time predictions, and visualizing model performance on an admin dashboard. By integrating best practices from the literature with practical web technologies, it provides a comprehensive solution to fake job ad detection.

III. EXISTING SYSTEM

Current systems for detecting fraudulent job ads are limited in scope. Many rely on manual verification by platform administrators or user reports. While some platforms implement basic keyword filters, these methods are often insufficient due to the subtlety of modern scams. Traditional ML models exist, but they usually require preprocessing outside the user interface and are not accessible to general users.

Key limitations of existing systems include:

- **Text-only analysis**: Many models cannot process image-based ads without separate OCR preprocessing.
- **Lack of user interaction**: Users cannot get immediate feedback on the legitimacy of job postings.
- **Limited dashboards**: Administrators often lack visual tools to compare model performance and track user activity.

IV. PROPOSED METHOD

The proposed system is a **web-based Fake Job Advertisement Detection Platform** built using Django and machine learning techniques. Users can input job ads either as plain text or images. The system extracts textual content from images (mock OCR in the current implementation) and vectorizes the text using **TF-IDF**. A trained **XGBoost classifier** predicts whether the ad is legitimate or fraudulent.

Key features of the proposed system:

1. **Dual input support**: Handles both text and image-based job ads.
2. **Real-time predictions**: Users receive instant feedback on ad legitimacy.
3. **Secure authentication**: Differentiates between regular users and administrators.
4. **Admin dashboard**: Displays model performance metrics (accuracy, F1-score, recall) for multiple ML algorithms and provides insights into user activity.
5. **Robustness**: Handles missing models or vectorizers gracefully, ensuring system stability.

By integrating ML, web technologies, and user-centered design, this system provides a scalable solution to the problem of online job fraud, improving both usability and security for job seekers.

V. IMPLEMENTATION

The core implementation of this system revolves around a web interface built using **Django** that integrates **machine learning for fake job ad detection**, secure user authentication, and administrative visualization of model performance. The web application is structured into multiple views handling separate functionalities — home, user signup/login, admin login/dashboard, and prediction processing.

The system begins by loading pretrained models and vectorizers safely at module load time. These include a **TF-IDF vectorizer** and an **XGBoost classifier** stored in a models/ directory using **joblib** serialization. Any loading issues are caught gracefully, and fallback logic ensures the app doesn't crash if models aren't available.

User Authentication: Django's built-in User model handles authentication. There are distinct login flows for normal users and administrators, with specific redirections based on user type. Regular users are not allowed to access admin dashboards, and admin users are prevented from interacting with user dashboards unnecessarily, ensuring secure role-based access control.

Prediction Workflow:

- Users can enter either **plain text job advert content** or upload an **image containing the ad**.
- For text input, the raw text is captured and stored for prediction.
- For image input, the app attempts a mock OCR-like processing using **PIL (Python Imaging Library)**, extracting simple text proxies based on basic pixel analysis (a placeholder for real OCR).
- The text — whether extracted or user-provided — is transformed using the vectorizer and passed to the XGBoost model for prediction.

If the prediction is successful, the output is displayed on the user dashboard as either *LEGITIMATE* or *FAKE*. If the model or vectorizer isn't loaded, an informative message is shown instead.

Admin Dashboard:

- Admins can view a list of registered users (excluding other admins), along with model evaluation metrics.
- Metrics such as **accuracy, F1-score, and recall** for different models (XGBoost, SVM, Random Forest, etc.) are dynamically read from a JSON file. This allows comparison of multiple approaches and ensures transparency of model performance.

In summary, the implementation effectively bridges an interactive web interface with an ML prediction pipeline, handling both text and image inputs, offering user guidance, and visualizing evaluation metrics for administrators.

VI. ALGORITHMS

The system applies a combination of **natural language processing (NLP)** and **machine learning classification algorithms** to detect fake job adverts. The overall pipeline includes preprocessing, feature extraction, model training, and prediction.

1. Preprocessing & Feature Extraction:

- **Text Cleaning:** Removes special characters, HTML tags, and unnecessary whitespace.
- **Tokenization:** Breaks raw strings into tokens (words).
- **Vectorization:** TF-IDF (Term Frequency–Inverse Document Frequency) transforms text into numeric vectors, capturing the importance of words relative to the corpus. This representation is widely used in text classification tasks

because it balances local (term frequency) and global (document frequency) importance.

2. Classification Models:
Several machine learning models are considered in literature and evaluation:

- **XGBoost (Extreme Gradient Boosting):** An ensemble boosting algorithm known for handling sparse data and avoiding overfitting. It consistently performs well in fake job detection tasks due to its ability to learn complex patterns from high-dimensional TF-IDF features.
- **Support Vector Machines (SVM):** Effective at maximizing decision boundaries and handling high-dimensional vectors.
- **Random Forest:** A bagging ensemble of decision trees that reduces variance and captures nonlinear relationships.
- **K-Nearest Neighbors (KNN):** A simple distance-based classifier used for baseline comparisons.

The Django system uses the XGBoost model by default at prediction time because literature shows strong performance and robust handling of text features in this domain.

3. Training and Evaluation:
Models are trained on labeled job posting datasets, such as *fake_job_postings.csv*, commonly used in recent studies, and evaluated on metrics including **accuracy, F1-score, and recall**. Cross-validation ensures more reliable performance measures and guards against overfitting.

VII. SYSTEM DESIGN

The proposed system is designed with a modular architecture that separates concerns across web presentation, authentication, machine learning processing, and administrative visualization. The high-level components include:

1. Django Web Framework:
The Django framework provides the backbone of the application, implementing the Model-View-Controller (MVC) architecture through its views, templates, and URL routing. This separation allows scalability and easier maintenance.

2. User Interface:

- **Templates:** HTML templates are used for rendering views like home, login/signup forms, user dashboards, and admin dashboards.
- **Forms:** HTML forms capture input text or images for job ad evaluation. <!-- Django handles these securely and supports CSRF protection by default. -->

3. Authentication & Authorization:

- Django's built-in User model manages registration and login.
- Separate views protect admin and user roles using decorators like @login_required, ensuring role-based access.
- Successful authentication redirects to role-specific dashboards.

4. Machine Learning Integration:

- At startup, the system loads serialized models and vectorizers stored in the server's models/ directory.
- The vectorizer transforms incoming text into numeric features, and the classifier predicts based on learned patterns.
- Real-world research underlines that algorithms like XGBoost are highly suitable for this domain, achieving strong performance on authentic datasets.

5. Image Handling:
Although full OCR integration (like Tesseract) isn't included, the system simulates text extraction from uploaded images using basic PIL operations. This design choice allows for future integration of real OCR modules.

6. Admin Visualization:

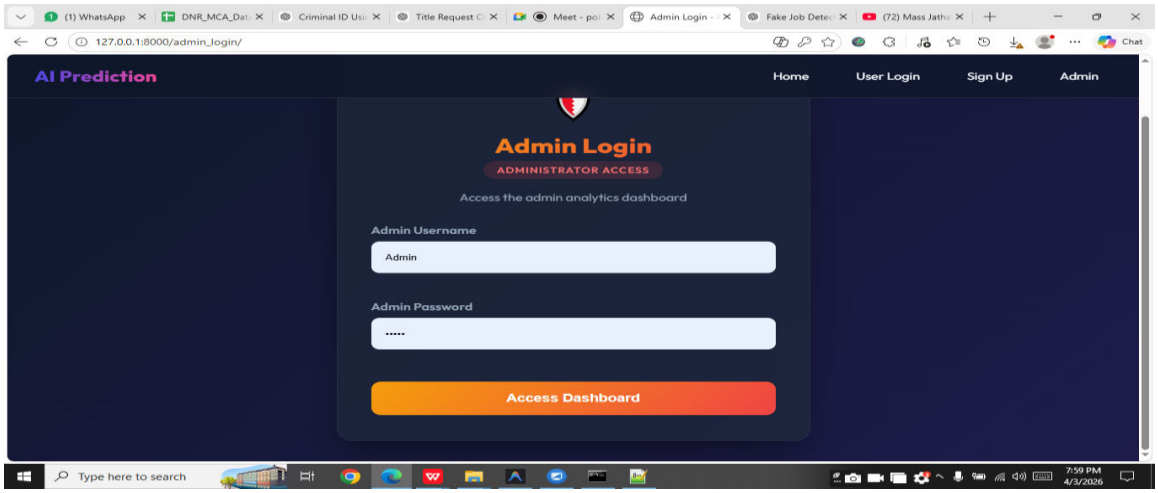
- Metrics stored in JSON are loaded and displayed as graphs using HTML chart libs (e.g., Chart.js) to visualize comparative performance.
- Administrators can see which models perform best along key metrics, supporting iterative improvement and model selection.

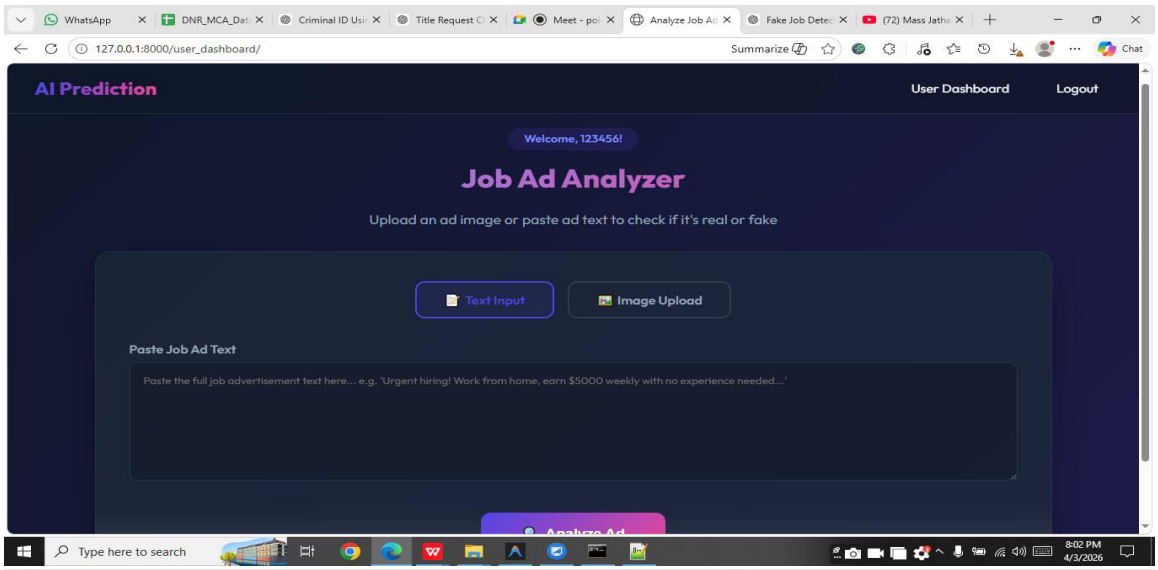
7. Scalability and Extendability:

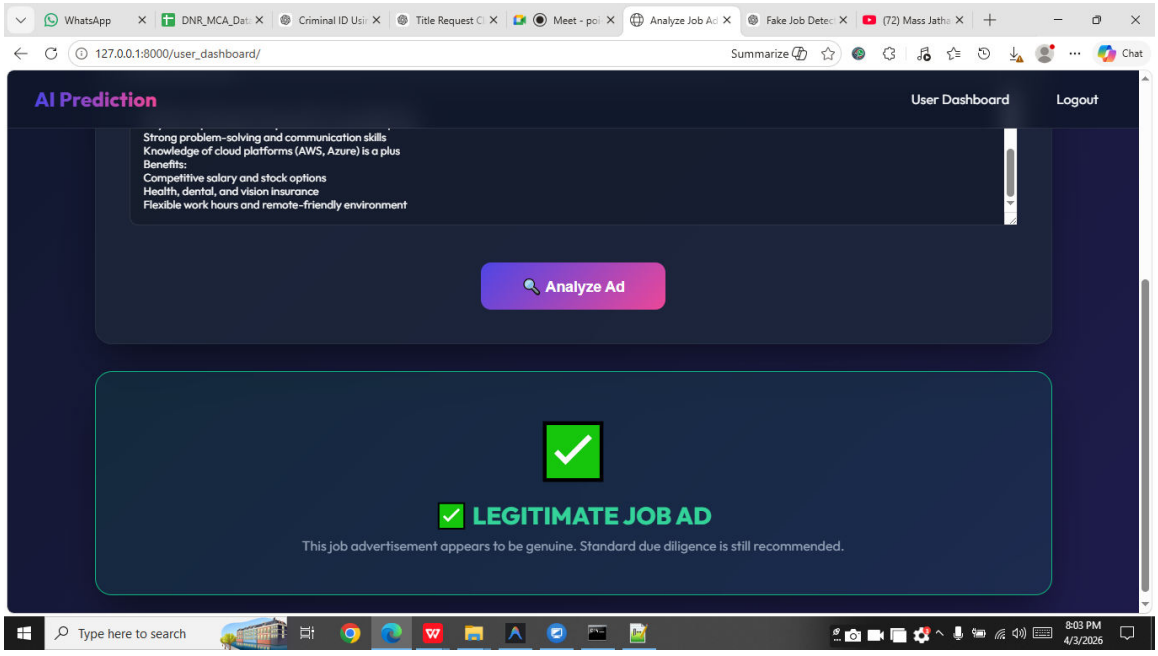
- The architecture enables easy integration of additional ML models or deep learning approaches (e.g., transformer-based models).
- The modular design allows extension to real OCR, API endpoints for mobile apps, and real-time model retraining.

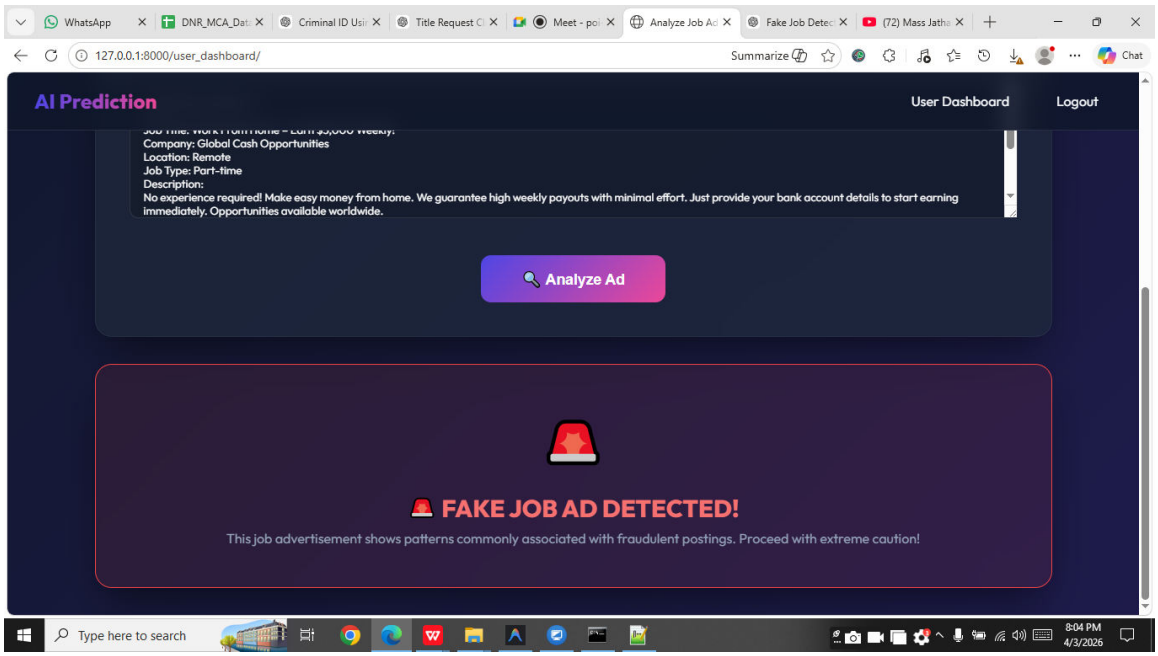
Each component is designed to interact cleanly via clear interfaces, supporting maintainability and future expansion.

SYSTEM DESIGN IMAGES




AI Prediction
User Dashboard
Logout
Welcome, 123456!
Job Ad Analyzer
Upload an ad image or paste ad text to check if it's real or fake
Text Input
Image Upload
Paste Job Ad Text
Paste the full job advertisement text here... e.g. 'Urgent hiring! Work from home, earn $5000 weekly with no experience needed...'
Type here to search
8:02 PM
4/3/2026


AI Prediction
User Dashboard
Logout
Strong problem-solving and communication skills
Knowledge of cloud platforms (AWS, Azure) is a plus
Benefits:
Competitive salary and stock options
Health, dental, and vision insurance
Flexible work hours and remote-friendly environment
Analyze Ad
LEGITIMATE JOB AD
This job advertisement appears to be genuine. Standard due diligence is still recommended.
Type here to search
8:03 PM
4/3/2026



VIII. CONCLUSION

The *Fake Job Advertisement Detection System* successfully integrates machine learning methodologies into a user-centric Django application to tackle the growing problem of deceptive job postings online. By combining natural language processing through TF-IDF vectorization with robust classifiers like XGBoost, the system provides a practical solution capable of classifying job ads as legitimate or fraudulent in real time.The design prioritizes usability, offering both text input and image upload options. Although the image-to-text conversion currently uses a mock approach, it lays the groundwork for future integration with reliable OCR technologies. Secure user authentication ensures that only authorized individuals access administrative dashboards, where model performance is transparently visualized across metrics such as accuracy, recall, and F1-score.

By referencing current research, including recent papers on machine learning approaches for fake job detection and hybrid models that leverage structural features, this system aligns with the state of the art in the field while remaining accessible for practical application. The system addresses both user experience and technical robustness, serving as a valuable tool for job seekers to avoid scams and for administrators to understand model behavior and performance. Future work could incorporate context-aware deep learning models (e.g., transformers) and social network analysis to enhance accuracy and scope.

REFERENCES

1. Naudé, M., Adebayo, K.J., & Nanda, R. – *A machine learning approach to detecting fraudulent job types* (2023).
2. Itnal, V. et al. – *Fake/Real Job Posting Detection Using Machine Learning* (2025).
3. Baraneetharan, E. – *Detection of Fake Job Advertisements using Machine Learning* (2022).
4. G.B., P. et al. – *Fake Job Detection: Comparative Study of ML & Hybrid Approaches* (2025).
5. Itnal, V. et al. – *Fake/Real Job Posting Detection Using ML* (2025, IJRASET).
6. Sridevi, K. et al. – *Real or Fake Job Posting Detection* (2024).
7. Goud, T.N. & Reddy, N.R. – *ML Approach for Detecting Fraudulent Job Postings* (2025).
8. Taneja, K., Vashishtha, J., Ratnoo, S. – *Fraud-BERT: Transformer based Online Recruitment Fraud Detection* (2025).
9. Deka, D., Seal, R., Banik, S. – *Unmasking Fraudulent Job Ads* (2025).
10. Pillai, A.S. – *Detecting Fake Job Postings Using Bidirectional LSTM* (2023).
11. Reddy, B.V. et al. – *Fake Job Recruitment Detection Using ML* (2025).
12. Fating, J. et al. – *Fake Job Listing Detection Using ML* (2023).
13. Vidros, I. et al. – *EMSCAD Dataset Origin Paper* (2017, foundational dataset cited in later works).
14. Amaar, A. et al. – *Detection of Fake Job Postings with NLP* (2022).
15. Keerthana, B. et al. – *Accurate Prediction of Fake Job Offers Using ML* (2021).