

Fake Job Post Prediction

Project Report

Course: Machine Learning Project

Date: October 4, 2025

Team Name: Algo 5

Team Members:

- Tezivindh Kumar - 24bcs10040
- Praneeth Budati - 24bcs10081
- Pranay Reddy - 24bcs10133
- Rishi Harti - 24bcs10239
- Sameer Khan S - 24bcs10245

1. Problem Statement and Initial Approach

1.1 Problem Statement

The proliferation of online job boards has created new avenues for fraudulent activities, posing significant risks to job seekers. These risks include financial scams (e.g., asking for upfront fees for training or equipment), identity theft through the collection of sensitive personal information, and wasted time and effort applying for non-existent positions. Fraudulent job postings often mimic legitimate ones, making them difficult for the average user to detect.

The primary objective of this project is to develop a robust machine learning model capable of automatically and accurately classifying job postings as either legitimate (real) or fraudulent (fake). By analyzing the textual content and associated metadata of a posting, the model aims to serve as a critical first line of defense for job platforms, helping to protect users by flagging suspicious listings for removal or further review.

1.2 Initial Approach

Our initial hypothesis was that fraudulent job postings contain distinct linguistic patterns and characteristics that differentiate them from legitimate ones. We suspected that keywords related to urgency ("immediate hire"), low entry barriers ("no experience needed"), and high rewards ("earn thousands weekly") would be strong indicators of fraud.

Our initial approach was therefore centered on Natural Language Processing (NLP). The plan was as follows:

1. **Consolidate Textual Data:** Combine various text-based columns from the dataset (title, description, requirements, benefits, etc.) into a single, comprehensive text feature for each job post. This would provide the maximum possible context for analysis.
2. **Vectorize Text:** Convert the consolidated text into a numerical format that machine learning algorithms can process. Our initial choice for this was the Term Frequency-Inverse Document Frequency (TF-IDF) method, as it effectively highlights words that are important to a specific document within a larger corpus.
3. **Model Experimentation:** Train and evaluate several standard classification algorithms (like Logistic Regression, Decision Trees, and ensemble methods) on the vectorized data.
4. **Evaluate Performance:** Assess the models not just on accuracy, but more importantly on metrics like Recall and F1-Score, which are critical for an imbalanced dataset where the primary goal is to identify the minority class (fraudulent posts).

2. Data Collection and Challenges

2.1 Data Collection Process

The dataset for this project, titled "Real or Fake Job Posting Prediction," was sourced from the Kaggle platform. This dataset was ideal as it is specifically curated for this classification task. It contains 17,880 rows, each representing a unique job posting, and 18 columns (features) that describe the job.

Key features in the dataset include:

- **Textual Features:** title, department, company_profile, description, requirements, benefits, industry, function.
- **Categorical Features:** employment_type, required_experience, required_education.
- **Binary Features:** telecommuting, has_company_logo, has_questions.
- **Target Variable:** fraudulent (0 for real, 1 for fake).

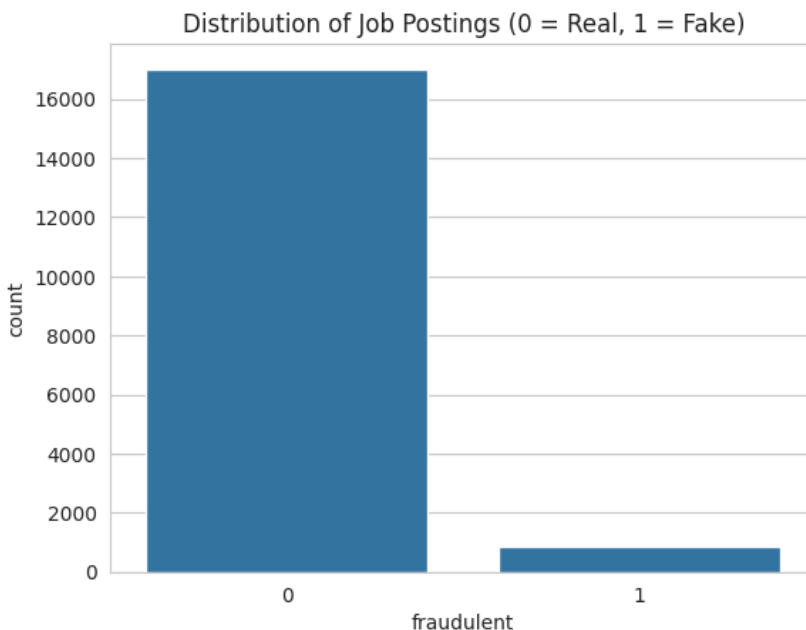
2.2 Data Exploration & Challenges Encountered

Upon initial exploration, two major challenges became immediately apparent.

1. Severe Class Imbalance:

The most critical challenge was the severe imbalance between the two classes. Out of 17,880 job postings, only 866 (4.8%) were labeled as fraudulent, while the remaining 17,014 (95.2%) were legitimate.

Figure 1: Distribution of fraudulent (1) vs. real (0) job postings, highlighting the class imbalance.



This imbalance poses a significant problem because a naive model could achieve over 95% accuracy by simply classifying every post as "real." This would render the model useless, as its purpose is to detect the rare fraudulent cases. This challenge necessitated the use of specialized techniques like stratified data splitting and class weighting during the modeling phase.

2. Missing Data:

The dataset contained a substantial amount of missing data across several columns, as shown below:

- salary_range: 84% missing
- department: 65% missing
- company_profile: 18% missing
- requirements: 15% missing
- benefits: 40% missing

This required a thoughtful preprocessing strategy. Since our approach was heavily reliant on text, we decided to handle the missing values in textual columns by replacing them with empty strings. This ensures that no data is lost during the text consolidation step and prevents errors during vectorization.

3. Methodology and Solution Approach

To address the challenges identified, we adopted a systematic methodology focusing on robust feature engineering, careful model selection, and techniques specifically designed for imbalanced classification.

3.1 Data Preprocessing and Feature Engineering

Our solution approach was centered on creating a single, powerful feature from the available text data.

1. **Text Consolidation:** We created a new feature called `combined_text` by concatenating the content of eleven textual and categorical columns: `title`, `department`, `company_profile`, `description`, `requirements`, `benefits`, `employment_type`, `required_experience`, `required_education`, `industry`, and `function`. Before concatenation, any missing (NaN) values in these columns were filled with an empty string (''). This approach creates a holistic textual representation of each job posting, ensuring that all available descriptive information is used by the model.
2. **Text Vectorization using TF-IDF:** The `combined_text` feature was then converted into a numerical matrix using the `TfidfVectorizer`. This was a crucial step because it effectively converts unstructured text into a structured feature space, automatically gives higher weight to words that are frequent within a posting but rare across all postings, and filters out common, non-informative words. We limited our vocabulary to the top 1,500 most frequent terms to reduce dimensionality and prevent overfitting.

3.2 Model Selection and Training

We chose to train and compare six different classification algorithms:

1. Logistic Regression
2. K-Nearest Neighbors (KNN)
3. Decision Tree
4. Random Forest
5. Support Vector Machine (SVM)
6. XGBoost

3.3 Addressing Class Imbalance

We implemented two key strategies to combat the class imbalance issue:

1. **Stratified Data Splitting:** The data was split into an 80% training set and a 20% testing set, ensuring that the proportion of fraudulent postings was maintained in both sets.
2. **Class Weighting:** We adjusted the class weights in our models to force them to pay more attention to the minority (fraudulent) class, either by using the `class_weight='balanced'` parameter or by manually calculating `scale_pos_weight` for XGBoost.

4. Model Performance Analysis

4.1 Evaluation Metrics

Due to the severe class imbalance, accuracy is a poor indicator of model performance. Instead, we focused on:

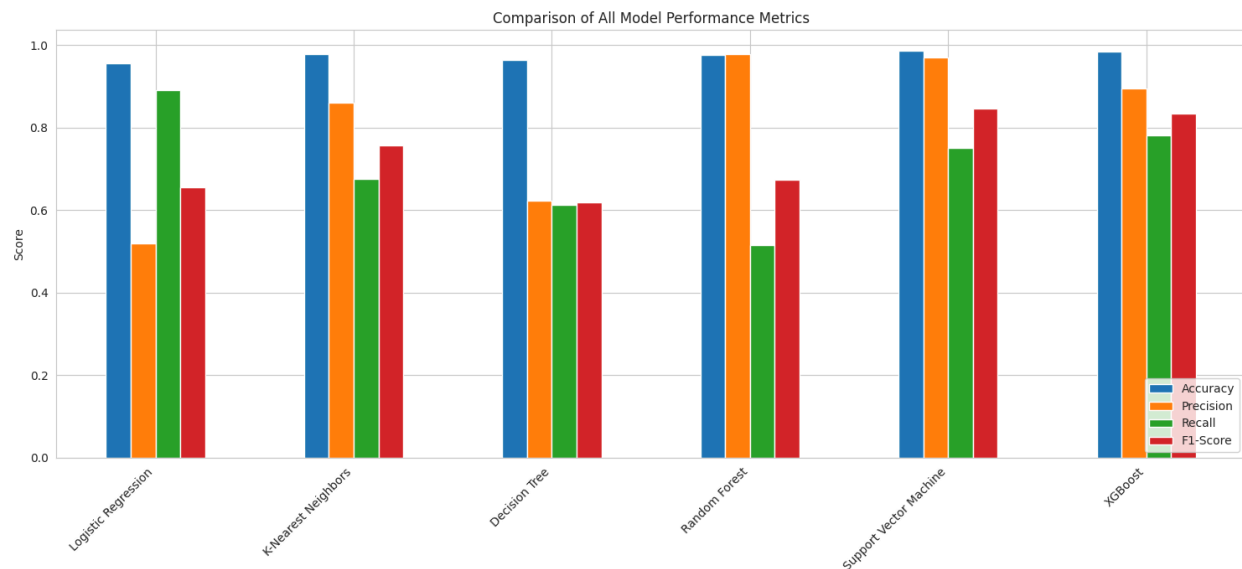
- **Precision:** Measures the reliability of the positive (fraudulent) predictions.
 - *Formula: $\text{True Positives} / (\text{True Positives} + \text{False Positives})$*
- **Recall (Sensitivity):** Measures the model's ability to find all the actual fraudulent postings.
 - *Formula: $\text{True Positives} / (\text{True Positives} + \text{False Negatives})$*
- **F1-Score:** The harmonic mean of Precision and Recall, providing a single, balanced score.
 - *Formula: $2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$*

4.2 Performance Results

The performance of the six trained models on the test set is summarized below, sorted by F1-Score.

Model	Accuracy	Precision	Recall	F1-Score
Support Vector Machine	0.9869	0.9701	0.7514	0.8469
XGBoost	0.9849	0.8940	0.7803	0.8333
K-Nearest Neighbors	0.9790	0.8603	0.6763	0.7573
Random Forest	0.9760	0.9780	0.5145	0.6742
Logistic Regression	0.9547	0.5185	0.8902	0.6553
Decision Tree	0.9634	0.6235	0.6127	0.6181

Figure 2: A comparison of performance metrics across all trained models.



4.3 Comments on Performance

- **Support Vector Machine (SVM):** The clear winner. It achieved the highest F1-Score (0.8469), demonstrating an outstanding balance between Precision (97%) and Recall (75%).
- **XGBoost:** A very strong runner-up with an F1-Score of 0.8333.
- **Logistic Regression:** Achieved the highest Recall (89%) but its extremely low Precision (52%) makes it impractical.
- **Random Forest:** Extremely precise (98%) but its low Recall (51%) was a major weakness.
- **KNN and Decision Tree:** Delivered mediocre performance compared to the top contenders.

Based on this analysis, the **Support Vector Machine (SVM)** was selected as the final model.

5. Assessment of Model Success




5.1 Addressing the Problem Statement

The project successfully addressed the problem statement by developing a high-performing machine learning model capable of identifying fraudulent job postings. The final SVM model, with an F1-Score of 84.7%, provides a powerful and reliable tool for enhancing the safety of online job platforms.

5.2 Solution Implementation and Analysis

To demonstrate the model's practical utility, a prediction function (`predict_job_posting`) was created. This function encapsulates the entire preprocessing pipeline (text consolidation and TF-IDF transformation) and uses the trained SVM model to make predictions on new, unseen job posting data.

We tested this function on three sample job postings:

1. **A Clearly Legitimate Job:** A standard "Senior Software Engineer" role with detailed, professional language.
 - **Prediction:**  REAL (Confidence: 98.19%)
 - **Analysis:** The model correctly identified the post as legitimate with very high confidence, showing it understands the features of a genuine job description.
2. **A Suspicious Job:** A "Work From Home Data Entry" role with vague language and red flags like "IMMEDIATE HIRE" and "No interview required."
 - **Prediction:**  REAL (Confidence: 43.26%)
 - **Analysis:** This was an interesting case. While the model predicted "REAL," its confidence was extremely low (below 50%). This indicates a high level of uncertainty and suggests the post is ambiguous. In a real-world system, such a low-confidence prediction would be an ideal candidate for manual review by a human moderator.
3. **A Clearly Fake Job:** A "Get Rich Quick" scheme with classic scam language.
 - **Prediction:**  FAKE (Confidence: 98.42%)
 - **Analysis:** The model identified this post as fraudulent with extremely high confidence. This confirms that the model has effectively learned the key textual indicators and keywords associated with fraudulent listings

These tests confirm that the solution is not only accurate based on aggregate metrics but also performs logically on individual, real-world examples. It has successfully learned the patterns required to solve the problem.

6. Recommendations and Next Steps

6.1 Recommendations

1. **Production Deployment:** The SVM model is robust and ready for integration into a job platform's backend to scan new submissions.
2. **Implement a "Review and Quarantine" System:** Use prediction confidence to trigger actions like auto-quarantine (>80% fake), flag for priority review (50-80% fake), or flag for review due to ambiguity (<60% real).
3. **User-Facing Indicator:** Consider adding a "Verified" badge to postings that the model confidently identifies as real.

6.2 Future Improvements and Modifications

1. **Advanced NLP Models:** Leverage models like Word2Vec, GloVe, or BERT to understand semantic context and improve detection of novel scam techniques.
2. **Incorporate Non-Textual Features:** Improve performance by including features like salary_range, has_company_logo, company_profile, and recruiter account metadata.
3. **Continuous Retraining and Active Learning:** Periodically retrain the model with new data and use feedback from human moderators to continuously fine-tune it.
4. **Ensemble Modeling:** Experiment with an ensemble of the top two models (SVM and XGBoost) to potentially create a new model that outperforms both.

7. Conclusion

This project successfully demonstrated the power of machine learning in tackling fraudulent job postings. By transforming text into features and applying classification algorithms with strategies to handle class imbalance, we developed an effective model. The final SVM model emerged as the top performer with an F1-Score of 84.7%, proving its ability to balance high precision and high recall. The practical application of this model can significantly enhance the safety and integrity of online job platforms.

The final Support Vector Machine (SVM) model emerged as the top performer, achieving an F1-Score of 84.7%, proving its ability to balance the critical needs of high precision and high recall. The practical application of this model can significantly enhance the safety and integrity of online job platforms, protecting users from financial loss and data theft. The recommendations and future work outlined in this report provide a clear roadmap for further refining and deploying this valuable tool.