CSCI 5541 Final Project Report

Aaron Castle, Mark Spengler, Sanketh Kodimala, Gabe Robison
No Name 2.0
castl145@umn.edu, speng026@umn.edu, kodim003@umn.edu,girma039@umn.edu

1 Introduction

1.1 Motivation

As graduating computer science students actively preparing to enter the job market, we (as well as many of our classmates) have first hand experience with the cycle of applying to hundreds of jobs in a hyper-competitive job market. Having spent so much time and effort, we understand how frustrating it is to encounter fake job postings that waste an applicant's time and put their personal data at risk. The prevalence of these fake listings across legitimate platforms highlights the need for automated detection methods that can distinguish real opportunities from scams.

Our project aims to explore how Natural Language Processing (NLP) can be applied to this problem. Specifically, by fine-tuning modern language models to identify linguistic, semantic, or hidden cues of deception in job descriptions, we go beyond simple classification. Through evidence span identification, we aim to build interpretability; we evaluate robustness by testing how small, realistic counterfactual edits (such as changing a job location, visa terms, or remote status) affect model confidence.

By combining technical rigor with a problem that directly impacts students and professionals, this project aligns well with both our academic interests and real-world relevance. Ultimately, we hope to develop a tool that can directly benefit students and other job seekers to safely and efficiently navigate the job search process, reducing the risk of falling victim to fake postings and increasing our chances of getting hired.

1.2 Background

There are lots of previous works regarding fake job posting detection, but they typically focus on supervised classification. Giving a model the text of a job posting or structured metadata and having the model classify it as fake or real. Earlier methods used engineered features and classical classifiers for the predictions. More recent methods use deep learning and transformer based models, which are more accurate than the earlier methods. While these methods work well, they are hard for users to fully trust as they just produce a label without explaining why or which parts of the data triggered its prediction.

There are some challenges that the current practices face. First, there aren't any good datasets out there for fake job detections. Many of them are often imbalanced, with more real job postings than fake ones. So, models may achieve high accuracy during training, but once deployed, it may misclassify fake jobs. Another challenge is that models may use some "shortcuts", like certain words or phrases, to make their prediction. This can lead to some problems, like small and meaningless edits changing the model's prediction. This is a problem that affects the model's robustness.

Our project focuses on three important elements that are often not considered in previous works: interpretability, robustness and model fairness, and usability in real world applications. We do this by fine tuning a LUKE model, which contains architecture for span level representations, making it easy to highlight evidence phrases that contribute to the predictions. This helps with interpretability as it not only gives users a probability score, but it gives the users some phrases that it used to make its predictions. We also perform counterfactual tests by changing some of the attributes (like remote vs onsite, salary, and visa language) to evaluate how the model's confidence changes and if it is reasonable. Lastly, we integrate the model into a Chrome extension that can be used directly on job boards, like LinkedIn.

2 Approach

2.1 Datasets

To train and evaluate our model, we first used the shivamb/real-or-fake-fake-jobposting-prediction dataset from Kaggle. This dataset contains 18,000 job postings with fields like job title, description, benefits, and more. The main challenge we had with this dataset was that it was strongly imbalanced, with only around 500 fake jobs in the dataset. This brings up the potential issue of the model learning to just lean towards a job posting is real every time and still achieve a high accuracy during training. During the actual training of the model on this dataset, it did achieve a high accuracy, 98%, but the problem was that it resulted in a low statistical parity at 4%, indicating that the model has some underlying fairness issues.

To partially mitigate this problem, we combined this data set with the srisaisuhassanisetty/fake-job-postings dataset on Kaggle. This dataset contains exclusively fake job listings, giving the model more fake postings to train and learn upon. When combining the 2 datasets, we did preprocessing to standardize text fields and handle any missing values before training. Training the model using this new dataset improved the performance of the model, achieving a 99% accuracy and a higher statistical parity of 39%.

While combining the datasets resulted in better fairness and accuracy, there are still some limitations. The 2 datasets contain many non-overlapping fields, which resulted in a large number of missing values when they were merged, especially for data in the second dataset. This could affect how the model makes its predictions and the fairness analysis. The second dataset only contains fake job postings, so the presence of missing metadata could be a strong sign that the posting is fake. Because of this, the model could learn that the missing fields are correlated with fake postings, rather than relying on the semantic cues from the job descriptions. This challenge is talked about more in the counterfactual and fairness analysis section.

It is important to note that these datasets focus on fraudulent intent of the job posting rather than the authorship of it. The job postings that are labeled fake are ones that have deceptive intent, like collecting personal information. Our task is not to classify whether a posting is written by a human or AI generated, but rather to detect whether a job posting represents a legitimate job opportunity.

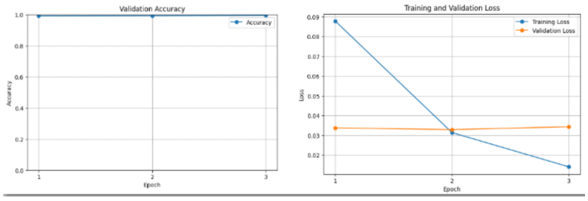

Figure 1: Validation Accuracy and Loss Graphs

2.2 Testing & Training

After merging both datasets, we created a 80/20 training and test split, which is stratified by the fraudulent label. This ensures both splits have enough fraudulent postings to preserve class balance. We then created a validation set from the training set, which was a 70/10 split that was also stratified by the fraudulent label. Next, we used the LukeTokenizer model to tokenize all text inputs. Before training the LUKE model, the base model was evaluated, which resulted in an accuracy of 0.39 and a F1 score of 0.28. We trained the LUKE model for 3 epochs, with a training time of around 11 minutes. After the first epoch, the model achieved an accuracy of 0.991484 and an F1 score of 0.991026. The final 2 epochs improved the accuracy and f1 of the model by an insignificant amount, resulting in a final accuracy of 0.992380 and f1 of 0.991984. This model was saved and used for all the subsequent analyses.

2.3 Span Level Detection

To address the interpretability challenges that plague many black-box fraud detection models, we implement gradient-based evidence span extraction that identifies which phrases in a job posting most strongly contribute to the model's classification decision. Our method leverages the LUKE model's embedding layer to compute token-level attributions via gradient backpropagation. For a given input posting, we perform a forward pass through the model, obtain the fraud probability, and then compute gradients of this probability with respect to the input embeddings. The attribution score for each token is calculated as the L2 (Euclidean) norm of its gradient vector, which captures how much small changes to that token's embedding would affect the prediction.

Raw token attributions require several post-processing steps first in order to produce human-readable evidence spans. First, we merge consecutive subword tokens (for example, "MoneyGram = ["Money", "##Gram"]) by summing their attribution scores to reconstruct complete words. We then remove special tokens, punctuation-only tokens, and common stopwords that provide little semantic value. This ensures the highlighted spans are meaningful to users. Tokens are ranked by their L2 (Euclidean) gradient norms, and a threshold of 35% of the maximum attribution score is applied to select significant tokens. We ensure that at least the top three highest-scoring tokens are selected to ensure every prediction ensures some explanation. Any adjacent high-scoring tokens are grouped into contiguous phrases, and two to five evidence spans are produced per posting. This prevents fragmented outputs and provides helpful context for users.

We chose gradient-based attribution over attention-weight methods because prior work studied during our literature review (Jain & Wallace, 2019) has shown that attention weights do not reliably indicate feature importance. Gradient-based methods directly measure how perturbing inputs affects their output predictions, providing a more accurate explanation. The LUKE architecture's entity-aware design further benefits span extraction by maintaining coherent representations of named entities and phrases. However, while our evidence spans do align with some intuitive scam signals (for example, "MoneyGram", "confidential", "cash"), several limitations remain. Firstly, neural networks still do not think like human brains under the hood, and some words that contribute significantly to fraudulent ratings do not always align with our expectations. Additionally, gradient attributions capture local sensitivity, rather than global feature importance. This means that a span may be highlighted because it is locally influential even if removing it would not change the overall prediction. We recommend that users treat evidence spans as indicative cues that warrant further investigation, which is an idea that we made explicit in our Chrome Extension's AI-generated prediction summaries.

2.4 Counterfactual Analysis

Going beyond the standard performance metrics, we also evaluate the robustness and fairness of our model. We perform counterfactual token analysis focused on attributes that may influence and change the models predictions in unintended ways. We start by computing the statistical parity across unique values of selected metadata attributes. This identifies attributes where the predictions differ the most across groups. Based on these results, we choose 3 attributes with the highest statistical parity to test with counterfactuals. For each of the attributes, we generate counterfactuals of the job postings by changing a specific token or value and keeping the rest of the posting the same. This allows us to isolate and analyze how the individual attribute affects the model's performance.

We first perform a bit flip analysis, where the attributes selected from the parity evaluation are changed (either categorical or binary switches) to measure any changes in the model's predictions. We quantify this using 3 metrics. Flip rate, which is the proportion of samples where the prediction flips/changes due to the counterfactual edit. A flip rate of close to zero means that the model's prediction is not affected by that attribute change and suggests robustness to that attribute. Average probability shift, which is the average absolute change in predicted fraud probability after the edit. It captures the soft sensitivity, even if the prediction doesn't change. Lastly, the max probability shift. This is the largest probability change seen across all samples. This helps rule out hidden sensitivity and any edge case instabilities.

After seeing the results of the bit flip test and its limited sensitivity, we chose to extend our counterfactuals to test textual semantics. We focused on modifying language for things that related to remote and onsite working, exaggerated salary claims, and visa language. Any phrase that mentioned remote working, we modified it to on site working. We adjusted exaggerated salary offers to more reasonable offers. Lastly, we modified visa friendly phrases to more restrictive phrasing. These counterfactual changes were meant to reflect changes scammers or potential employers might make while keeping the overall structure of the posting the same. There is room for future improvements in this area.

2.5 Chrome Extension

To bridge the gap between our research model and real-world usability, we developed a Chrome extension that enables job seekers to analyze job postings directly in their browser with a single click. The extension can scrape data from any job posting website, and the system follows a client-server architecture with three main components: a Chrome extension frontend that handles user interaction and DOM scraping, a FastAPI backend hosted on Hugging Face Spaces, and an ensemble of fine-tuned transformer models (LUKE, RoBERTa, and Distil-

BERT) that perform the actual classification.

When activated via the extension icon, a content script is injected into the activity tab that scrapes the visible job posting text through DOM parsing. The extension is designed to be site-agnostic, working across LinkedIn, Indeed, ZipRecruiter, and any other job board by extracting all visible text content from the page to be sent to our backend API. In the backend, the text is structured into the expected format that our models can then utilize for analysis.

The FastAPI backend processes requests through several stages. As mentioned, the text is first parsed using GPT-4o-mini. Any unstructured text, navigation elements, advertisements, or other noise is removed, and fields matching our training data schema are produced from the text. This preprocessing standardizes inputs regardless of which job site they originate from. Then, the structured posting is fed to our three fine-tuned models: LUKE, RoBERTa, and DistilBERT. Each model independently produces a fraud probability. We implemented ensemble voting to combine these predictions, which improves robustness by reducing the impact of any single model's errors. The final prediction is computed as a weighted average (all weights are equal, but could be changed if needed), with the ensemble confidence reflecting agreement across models. Using the gradient-based method described in section 2.3, we compute evidence spans from the LUKE model's embeddings; these spans are returned alongside the classification to provide interpretability. To make results accessible for non-technical users, we generate a brief explanation summarizing the prediction reason using the evidence spans, risk level, and confirmation that common sources of bias are not influencing our predictions.

Results are displayed on a clean overlay on the current page, which features a color-coded risk indicator, a confidence percentage showing model certainty, 2-5 highlighted phrases that contributed most to the prediction, a human-readable explanation of why the posting was flagged or cleared, and brief information on the ensemble voting outcome.

After our initial poster submission, we extended the system to include the three-model ensemble to improve prediction stability and interpretability. RoBERTa and DistilBERT were fine-tuned using the same 28,000 point dataset and training procedure as LUKE. This ensemble approach mitigates individual model weaknesses (for instance, if LUKE overfits to certain patterns, the other mod-

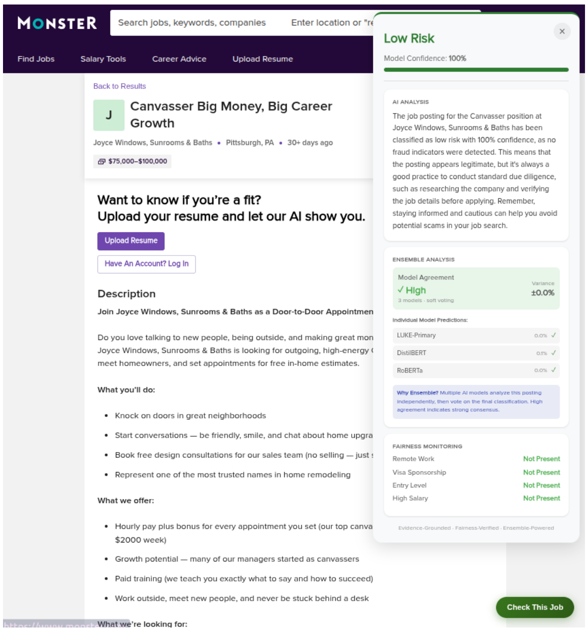

Figure 2: Chrome Extension on monster.com

els may provide a corrective signal). This design choice directly addresses our poster's "Future Plans," where we identified ensemble voting as a potential improvement.

The backend is deployed on Hugging Face Spaces. Model weights are also stored in the Hugging Face Hub, enabling efficient loading and version control. The extension is packaged as an unpacked Chrome extension that can be loaded by anyone in developer mode. It is fully functional and, with security and rate limit safety measures, could be published in the Chrome Web Store in the future.

3 Experiments/Results/Error Analysis

3.1 Counterfactual Analysis

From our statistical parity analysis, we selected telecommuting, has_company_logo, and has_questions for the counterfactual test, as these attributes had the highest parity disparities. We also found from this test that postings with unknown or missing values for these attributes were fraudulent 100% of the time. This makes missing values a highly informative signal. This is due to the fact that the merged datasets didn't have many overlapping columns, resulting in many missing values.

The bit flip counter factual testing resulted in a flip rate of 0 across all the selected attributes. Both average probability shift and max probability shift were also 0 for all the attributes. This indicates that changing these attributes didn't affect the model's

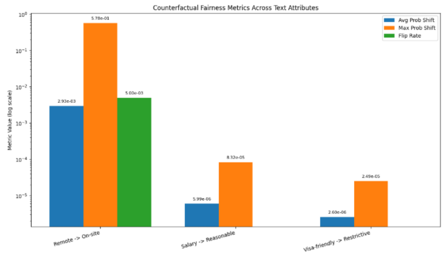

Figure 3: Flip Rate Graph

predictions or its confidence, suggesting that the model doesn't rely heavily on isolated attributes when making predictions, but uses the contextual information in the job posting text.

We get a similar result from the semantic counterfactual tests, but with some differences across attributes. Changing the exaggerated salary claims to more reasonable ones and changing visa friendly phrases to more restrictive ones resulted in flip rates of 0, with very small average probability shifts ($<$ 0.000006) and max probability shifts ($<$0.000083). These results show that the model is not affected much by these changes and doesn't rely heavily on isolated cues, like salary exaggerations or visa phrases, independently from the surrounding context.

Modifying the job postings from remote to onsite produced a small, but almost insignificant sensitivity. This change resulted in a flip rate of 0.005, with an average probability shift of 0.0029 and max probability shift of 0.58. While many predictions stayed the same, having a large max probability shift value suggests that remote work phrases can influence the model's confidence, in some cases. This could be due to the correlations in remote work phrasing and scam-related posts in the training data.

Overall, these results show the model's predictions are mostly stable under bit flip and semantic counterfactuals. The zero flip rates and minimal probability shifts for most attributes show the model's robustness and suggests that it relies more on overall textual cues rather than individual metadata fields. The results of the remote to onsite modifications brings up a potential source of bias in the dataset, which emphasizes caution when interpreting the fairness and robustness results.

3.2 Our Model Compared to ChatGPT

To evaluate the effectiveness of our ensemble predictions, we compared their predictions with those generated by ChatGPT 4.5. In Table 1, there is a comparison between our Chrome extensions' performance and ChatGPT's prediction. Overall, there were many cases in which the prediction of ChatGPT and the prediction of the ensemble performed the same, specifically with trusted sites such as LinkedIn. Verified employers such as Reddit were both recognized as being real and passed with high confidence. As we can see in the ZipRecruiter job posting Earn Money From Home, both ChatGPT and our ensemble predicted a moderate risk and to proceed with caution. There were many red flags that had come up, such as a vague description or an untrusted company. Some of the main discrepancies come from the snagajobs, grabjobs, elitejobs, and dailyremote job postings. In the cases where ChatGPT had given low confidence, our model seemed to do the opposite and provide a lot higher confidence rating, with a difference of 70-90%. One of the main reasons that ChatGPT vs our model was able to decipher from this was because ChatGPT was able to recognize the company and website as not being reputable. A lot of the red flags from ChatGPT had been along the lines of not recognizing the website or company. Along with this, ChatGPT has the ability to search for companies online and determine their patterns in the past, whereas our model does not have this context.

3.3 Model Performance

No matter whether our model succeeds or fails, the reason for its performance can always be traced to the evidence it relies on and the data set it was trained on. Our model generally performs well when there are grammatical errors, poor phrasings, or misleading benefits/contradictory statements, which can clearly identify fake job postings. Since these job postings come more from unknown sources, they tend to get flagged more often than not. Based on this, our model can detect fake jobs purely on the text it is given. On the contrary, our model can fail when a job description contains minimal red flags, or it would rely on contextual knowledge such as a company's history/reputation or anything outside of the posting itself, where other LLMs such as ChatGPT are able to leverage it. The mismatched cases between ChatGPT can show our model's strengths in which our model can identify subtle cues and changes in the linguistics, where it mainly struggles with contextual cues, which can define the truth.

3.4 Error Analysis

Based on the incorrect predictions of our base model, which can be seen in Table 2, there are clear failure cases in which our current model struggles to predict correctly. Right now, our model is showing a bias towards predicting a model as non-fraudulent (label 0), which results in three false negatives and one false positive, labeling fraudulent jobs as non-fraudulent and vice versa. This 75% error rate in this sample indicates limitations on the model's ability to choose between the two classes. Based on this, we can determine that the model was trained well on non-fraudulent jobs but is still struggling to predict fraudulent jobs correctly.

One reason our current approach fails could be its inconsistent understanding of what determines a fake job from a real job. It could simply be looking at surface-level text statements rather than bringing together contextual information within the whole paragraph. One similarity that I can notice between all of the false negatives would be the usage of capital letters. Within the 3 input statements, the usage of capital letters triggers an incorrect response, as this typically can be present within fake job postings. In the case of these error samples, the usage of 'JAVA' or 'HTML5' could look suspicious to our model, even though it is just listing common frameworks and technologies. Along with this, the false positive could be set off due to the technical language that is being used throughout the posting. Usage of technical vocabulary such as 'geophysical' or 'algorithms' could lead to an incorrect response if the model was trained on fake jobs that are more technical. The model could be failing to recognize that fake jobs can span a wide range of areas.

Several possible solutions could be implemented to help address failure cases. One way that the model could benefit is from additional contextual embeddings to provide the model with more contextual knowledge, allowing it to uncover hidden phrases within the text. This would help differentiate the all-capital sentences of 'APPLY TODAY, MAKE FREE MONEY!' versus frameworks that are simply in all capitals. Along with this, it could help with vague descriptions, regardless of whether they are technical or not. Finally, feature engineering could focus on specific patterns, such as more technical postings being flagged incorrectly, to help further enhance the robustness of the model to catch all cases.

4 Discussion

4.1 Replicability

Our work prioritizes reproducibility through complete documentation and public resource sharing. We provide full access to our fine-tuned model weights on Hugging Face (gaberobison7/luke-job-fraud-detector, tmarkspengler/output-distilbert, tmarkspengler/output-roberta), enabling researchers to directly replicate or extend our classification pipeline without retraining. Our GitHub repository contains all training scripts, ensemble implementation code, gradient-based attribution methods, and the complete Chrome extension codebase with FastAPI backend architecture.

The datasets we used are publicly available on Kaggle (shivamb/real-or-fake-fake-jobposting-prediction and srisaisuhassanisetty/fake-job-postings), ensuring anyone can recreate our exact training corpus. We documented all hyperparameters explicitly: batch_size=32, learning_rate=2e-5, num_epochs=3, weight_decay=0.01, max_sequence_length=512. Our data preprocessing pipeline, including the 80/20 stratified train-test split and 70/10 validation split, is fully specified in our methods section.

Overall, with access to our public code, models, and datasets, independent researchers should be able to reproduce our core findings with minimal effort.

4.2 Future work

Several directions could extend our work: collecting diverse fake postings with matching field structures to eliminate misleading patterns from missing data; incorporating web search or knowledge graphs to verify company reputation and detect impersonation fraud; implementing continuous learning to adapt to evolving fraud tactics; expanding fairness tests to cover job category, location, and company size; refining span attribution with validation data and exploring alternative explainability methods (LIME, SHAP); distilling the ensemble into a smaller model for faster inference without external API calls; and conducting user studies to validate whether our explanations genuinely help decision-making. These improvements would strengthen interpretability, fairness, and real-world applicability.

4.3 Limitations

Our system has several important limitations. The model operates solely on posting text and cannot verify company reputation or cross-reference with official sources, explaining why in some cases ChatGPT outperformed our model on posts from questionable sites. Our gradient-based evidence spans measure local sensitivity rather than global importance, so users should treat them as helpful hints rather than definitive proof. Counterfactual tests covered only specific attributes and did not exhaustively test all bias dimensions. The Chrome extension can only process one posting at a time and has many seconds of latency for each fraud detection request. We did not test adversarial robustness against scammers deliberately crafting evasive postings.

4.4 Ethics

Our fraud detection system carries significant ethical responsibilities. False negatives could lead vulnerable job seekers to share sensitive information with scammers, causing identity theft and financial loss. False positives could discourage applicants from pursuing legitimate opportunities, especially from smaller companies with informal communication styles. To mitigate these risks, we provide confidence levels rather than binary classifications, show evidence spans so users can assess model reasoning, and explicitly check for bias against neutral attributes. Our ensemble architecture reduces individual model failures, and we include messaging encouraging independent verification even for low-risk postings.

However, our system could create false security, and we cannot account for sophisticated scammers who deliberately evade detection. Privacy concerns exist since job content is sent to OpenAI's API for parsing. While we do not log this data, users should be aware their browsing activity generates external API calls.

4.5 Datasets

Our work demonstrates effective strategies for handling imbalanced datasets by combining two complementary Kaggle sources, improving statistic parity from 4% to 39%. However, this fusion revealed important challenges: the two datasets had difference column structures, which created patterns of missing values that have unknown effects on fraud detection. This experience provides a cautionary lesson for researchers working with merged fraud detection datasets. Future work should prioritize implementing strategies for handling this missing or misaligned data. While we do not contribute a new benchmark, our transparent documentation of these challenges provides valuable guidance for research constructing more robust fraud detection corpora that better generalize to real-world deployment scenarios.

5 Acknowledgments

We acknowledge the use of Claude Sonnet 4.5 and ChatGPT 5 to assist with writing refinement and clarity improvements, organizing report structure and flow, converting the document to LaTeX format, checking citation formatting consistency, and proofreading for grammatical and conceptual errors. All research understanding, project design decisions, methodological choices, experimental planning, dataset selection, and all interpretations are our own work.

6 Appendix

Job	ChatGPT	Ensemble
https://www.linkedin.com/jobs/collections/recommended/?currentJobId=4304830926	Low Risk, 95%	Low Risk, 100%
https://www.linkedin.com/jobs/collections/recommended/?currentJobId=4342656532	Low Risk, 85%-90%	Low Risk, 100%
https://www.careeronestop.org/Toolkit/Jobs/find-jobs-details.aspx?keyword=43-4051&location=New%20Jersey&radius=0&lang=en&fc=0&jvid=219abe55d14e4a488431345fba2cf3ae8003&detailpage=true	Low Risk, 80%	Low Risk, 99.9%
https://www.careeronestop.org/Toolkit/Jobs/find-jobs-details.aspx?keyword=43-4051&location=New%20Jersey&radius=0&lang=en&curpage=533&fc=1&jvid=61719e110d1e4705830c2d00f229f2d98003&detailpage=true	Low Risk, 80%-90%	Low Risk, 97%
https://www.localjobs.com/job/springfield-tn-community-resource-coordinator	Low Risk, 95%	Low Risk, 99.9%
https://www.ziprecruiter.com/Jobs/Earn-Money-From-Home/-in-Minneapolis,MN?lvk=uotwqy6CqUvPefdrb30s-w.--045rZXSaJ	Moderate Risk, 70%-75%	Moderate Risk, 71.3%
https://www.snagajob.com/jobs/1111569945?utm_campaign=google_jobs_apply&utm_source=google_jobs_apply&utm_medium=organic	High Risk, 5%-10%	Low Risk, 88.1%
https://grabjobs.co/us/job/full-time/others/remote-reward-economy-lead-all-genders-133115039?utm_campaign=google_jobs_apply&utm_source=google_jobs_apply&utm_medium=organic	High Risk, 30%-40%	Low Risk, 100%
https://theelitejob.com/job/13109/earn-money-from-home-%E2%80%93-free-work-from-home-jobs?utm_campaign=google_jobs_apply&utm_source=google_jobs_apply&utm_medium=organic	High Risk, 5%-10%	Moderate Risk, 77.6%
https://theelitejob.com/job/1457/free-time-extra-earn-money-without-investment-work-form-home-job-for-freshers	High Risk, 5%-10%	Low Risk, 95%
https://dailyremote.com/remote-job/sales-assistant-for-a-junk-removal-and-hauling-company-in-the-us-home-based-part-time-4338721	Low Risk, 85%-90%	Low Risk, 96.7%
https://dailyremote.com/remote-job/sales-assistant-for-a-junk-removal-and-hauling-company-in-the-us-home-based-part-time-4338721	Moderate Risk, 70%-75%	Low Risk, 99.9%
https://wfhcentral.22web.org/job/experienced-chat-support-advisor-up-to-35-hourly-flexible-remote-position/	High Risk, 10%-15%	Low Risk, 100%
https://www.monster.com/job-openings/canvasser-big-money-big-career-growth-pittsburgh-pa--7081c20f-a9e0-42d0-ba5c-c837d8009181	Low Risk, 90%-95%	Low Risk, 100%
https://www.monster.com/job-openings/make-money-making-a-difference-roseburg-or--cb8039e9-653f-459b-8efc-73bef5296d2b	Low Risk, 95%	Low Risk, 94.3%

Table 1: Job Risk Analysis Comparison

Text Excerpt	True Label	Predicted Label
Customer Service Representative: Burrell Behavioral Health is seeking qualified candidates for a Customer Service Representative positions at three clinics: Transitions, Main Center for Adults, and Medical Towers. These positions require a strong verbal and communication background, assessing caller needs and fielding calls appropriately, as well as computer literacy to provide a high level of service. Experience with multi-line phone systems is preferred. Must be able to multi-task, problem solve quickly, work independently, and have excellent computer skills. The ideal candidate will have advanced skills in Excel, Word and Outlook. MAKE A DIFFERENCE, APPLY TODAY! This position offers a comprehensive benefit package, including: Health and Dental Insurance, Paid Vacations, Life and Disability Insurance, Professional Liability Insurance, Retirement Plan Benefits...	1	0
Processing Geophysicist: Seismic Exchange, Inc. (SEI) is a prime source of premium 2D and 3D seismic data for the upstream oil and gas industry. We have an extensive proprietary seismic data library... We are looking for candidates to process seismic data acquired by geophysical field crews. Our client companies use the final processed product we generate in their search for oil and gas reserves. The processing of this data is very computer intensive, and involves the use of complex algorithms. A candidate with a strong math background, good attention to detail, logical thought processes, and a strong work ethic will do well in this position. A candidate must also be eager to learn and willing to accept challenging assignments. No previous experience is required as we provide in-house training. Education: Bachelor's Degree in the Geosciences, Math, Physics or Engineering	0	1
Front-End Developer: RideAmigos needs a great Front-End Developer to work with us on our "cloud-based" software as a service offering. We are a rapidly growing company working on innovation in the transportation space. In short, we do cool things. This is an on-site position, full-time with flexible hours. You will be working on our platform, applying your skills and experience to enhance our product. This opening is for a qualified developer focused in front-end web technologies. Basically, you'll provide high-quality user-focused markup, while working to build and maintain codebases for the visual aspects of our projects. Experience: extensive knowledge of web design standards & practices, HTML5, CSS3 & LESS, JavaScript, jQuery, Twitter Bootstrap, git, Photoshop, UI/UX sensibilities, responsive web development, mobile web development, cross-browser & cross-device compatibility...	1	0
Android Developer: inFullMobile Sp. z o.o. is a mobile software development house, specializing in the client side of mobile software for clients all around the globe. We are a young company, where you can make the difference, the company structure is horizontal so everyone has a say. We are looking for the best of the best JAVA developers, some out of the box thinking, the courage to get involved with the bleeding edge technologies and projects. A proven track record in Android / JAVA projects (C, C++, Obj-C is a plus), Participation in specification and planning of new product modules and features, A good knowledge of the English language (both written and spoken), Out-of-the-box thinking and creativity... Hands on experience in development for any Mobile OS platform or technology - iOS, Android, Symbian, WebOS, MeeGo, Maemo, Windows Phone, BlackBerry OS, Series40 or related technologies...	1	0

Table 2: Job Posting Classification Results