CALIFORNIA STATE UNIVERSITY
LONG BEACH
College of Business

IS 670 Machine Learning for Bus Analytics

Group 8: Fraudulent Job posting Detection

Professor: Mostafa Amini

By:

Achyuth Sambaraju – 032240180

Shreya Shree Bukkarayasamudram – 031515066

Antariksh Ramesh - 032175817

Prithviraju Venkataraman - 032163597

Table of Contents

Introduction

In the rapidly evolving landscape of online recruitment, the proliferation of fraudulent job postings has emerged as a significant challenge, undermining the trust and safety of digital job markets. These deceptive listings not only pose severe risks, such as financial losses and privacy breaches, but also compromise the credibility of genuine employers, affecting their ability to attract qualified candidates.

To address this critical issue, our initiative has leveraged the capabilities of machine learning to develop an advanced detection system. This sophisticated model is capable of distinguishing between genuine and fraudulent job postings with a high degree of accuracy. By integrating state-of-the-art data analytics and machine learning techniques, we aim to enhance the integrity and reliability of online job platforms, thus ensuring a secure and reliable environment for job seekers and employers alike.

This report delves into the development and deployment of our machine learning model, outlining the specific methodologies employed, the analysis of the dataset, and the architectural framework of the model. It further discusses the significant outcomes and the practical implications of our efforts. By strengthening the verification processes through technological innovation, we aspire to create a more trustworthy online job marketplace, which is crucial for the healthy functioning of the modern economy.

The sections that follow will provide a detailed account of our approach, from the initial concept to the final implementation, highlighting our model's capabilities in detecting and mitigating the risks associated with fraudulent job postings in the digital era.

Definitions

Classification Models

- **Decision Trees:**
  - Decision trees are non-parametric supervised learning models used for classification and regression tasks. They recursively partition the feature space into regions, with each partition representing a decision rule based on feature values. These models are represented as hierarchical trees, where internal nodes represent decision points based on feature values, and leaf nodes represent the predicted class or value.

- **Random Forest:**
  - Random Forest is an ensemble learning method used for classification, regression, and other tasks. It constructs multiple decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. The underlying idea is to reduce the risk of overfitting by averaging multiple trees, each trained on different parts of the data, thus improving the overall model's performance and accuracy. Random Forest is robust to noise and can handle many input variables.

- **Naive Bayes:**
  - Naive Bayes is a probabilistic supervised learning algorithm based on Bayes' theorem with the "naive" assumption of feature independence. It is commonly used for classification tasks. Despite its simplifying assumptions, Naive Bayes often performs well in practice, particularly on text classification tasks. It calculates the probability of a given instance belonging to each class based on the feature values and selects the class with the highest probability.

- **K-Nearest Neighbors (KNN):**
  - K-Nearest Neighbors is a non-parametric supervised learning algorithm used for classification and regression tasks. It classifies instances based on the majority class among their k nearest neighbours in the feature space. The choice

of k, the number of neighbours, influences the model's bias-variance trade-off: smaller values of k lead to more flexible models, while larger values lead to smoother decision boundaries.

- **Support Vector Machines – Classification (SVC):**
  - Support Vector Machines are supervised learning models used for classification and regression tasks. SVMs aim to find the hyperplane that best separates the data points of different classes while maximizing the margin, the distance between the hyperplane and the nearest data points (support vectors). SVMs are effective in high-dimensional spaces and can handle non-linear decision boundaries through kernel functions, mapping the input space into a higher-dimensional feature space.

- **Multilayer Perceptron (MLP):**
  - A Multilayer Perceptron (MLP) is a type of artificial neural network used for supervised learning tasks like classification and regression. It consists of multiple layers of interconnected neurons, including an input layer, one or more hidden layers, and an output layer. Each neuron applies an activation function to a weighted sum of its inputs, allowing the model to learn complex relationships between input and output variables. Through backpropagation, MLPs adjust the weights of connections between neurons during training to minimize prediction errors, making them effective tools for capturing non-linear patterns in data.

Evaluation Metrics

- **Accuracy**
  - Accuracy is the ratio of correctly predicted instances to the total instances. It is one of the most intuitive performance measures.
  - Accuracy = Number of Correct Predictions / Total Predictions

- **Precision**

- Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. It answers the question: What proportion of positive identifications was correct?
- Precision = True Positives / True Positives + False Positives

- **Recall (Sensitivity)**
  - Recall is the ratio of correctly predicted positive observations to all observations in the actual class. It answers the question: What proportion of actual positives was identified correctly?
  - Recall = True Positives / True Positives + False Negatives

- **F1 Score**
  - The F1 Score is the weighted average of Precision and Recall. Therefore, it considers both false positives and false negatives. It is particularly useful when the class distribution is imbalanced.
  - F1 Score = (Precision x Recall / Precision + Recall) x 2

- **Confusion Matrix**
  - A confusion matrix is a table used to evaluate the performance of a classification model. It provides a summary of prediction results on a classification problem. The matrix shows the number of correct and incorrect predictions made by the model compared to the actual classifications.

Hyperparameters

- **C (Regularization Parameter)**
  - In SVM and other models, the regularization parameter C controls the trade-off between achieving a low error on the training data and minimizing the complexity of the model. A smaller C value creates a simpler model that may have higher training error but lower generalization error. A larger C value creates a more complex model that aims to have lower training error but may overfit the training data.

- **Max Depth (in Decision Trees and Random Forests)**
    - The maximum depth of a tree determines the longest path from the root to a leaf. Limiting the depth of the tree helps prevent overfitting by restricting the model's complexity.

- **n_estimators (in Random Forests)**
    - The number of trees in the forest. Increasing the number of trees improves the model's performance but also increases the computation time.

- **Kernel (in SVM)**
    - The kernel function in SVM transforms the input data into a higher dimensional space to handle non-linear relationships. Common kernels include linear, polynomial, and radial basis function (RBF).

- **Hidden Layers and Neurons (in MLP)**
    - In MLP, hidden layers are the layers between the input and output layers. The number of neurons in each hidden layer and the number of hidden layers together determine the capacity of the neural network to learn complex patterns.

- **Learning Rate (in MLP)**
    - The learning rate is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. It determines the speed at which the model learns.

Problem Statement

The critical issue our project addresses is the pervasive presence of fraudulent job advertisements on online recruitment platforms, which pose serious risks such as monetary loss, identity theft, and harm to legitimate businesses. Studies reveal that approximately one-third of job postings online are deceptive, highlighting the significant scale of this problem within the job market. Given the severity and frequency of such incidents, there is an urgent need to combat this issue effectively.

Our response to this pressing problem involves the development of a predictive model designed to differentiate between genuine and fake job listings. The model's target variable is a binary indicator, with "0" representing genuine postings and "1" indicating fraudulent ones. Leveraging advanced analytics and predictive modelling techniques, our project entails a meticulous data preparation phase to ensure the dataset accurately reflects the distribution of real and fake job advertisements.

Throughout the execution of our project, we intend to explore and deploy various machine learning models to assess their efficacy in identifying fake listings. By leveraging a diverse set of models, we aim to develop a reliable tool that enhances the integrity and trustworthiness of online job markets. The successful outcome of our project holds the promise of contributing to the development of more sophisticated fraud detection mechanisms, thereby reinforcing trust in the job-seeking process for both job seekers and employers.

Dataset Overview

The dataset contains 50,447 rows and 24 columns, providing a comprehensive view of various job postings. Each row represents a unique job posting with attributes that describe its characteristics. Below is a detailed overview of the columns included in the dataset:

- **job_id**: A unique identifier for each job posting.
- **title**: The job title.
- **location**: The location of the job, typically in the format "Country, State, City."
- **department**: The department within the company that posted the job.
- **salary**_range: The salary range offered for the position.
- **Salary**: The average salary offered.
- **salary_0_1**: A binary indicator for whether a salary is provided (1) or not (0).
- **company_profile**: A brief description of the company.
- **description**: A detailed description of the job responsibilities.
- **requirements**: The qualifications and skills required for the job.
- **benefits**: The benefits offered with the job.
- **telecommuting**: A binary indicator for whether the job allows telecommuting (1) or not (0).
- **has_company_logo:** A binary indicator for whether the job posting includes the company logo (1) or not (0).
- **has_questions:** A binary indicator for whether the job posting includes questions for the applicant (1) or not (0).
- **Employment_0_1**: A binary indicator for the employment type, where 0 represents "Other" and 1 represents "Full-time."
- **employment_type**: The type of employment (e.g., Full-time, Part-time, Contract, Internship).
- **required_experience**: The level of experience required for the job (e.g., Not Applicable, Internship, Entry-level, Mid-Senior level).
- **intern_0_1**: A binary indicator for whether the job is an internship (1) or not (0).
- **required_education:** The level of education required for the job.
- **Degree_1_0:** A binary indicator for whether a degree is required (1) or not (0).

- **Interview_ol_of:** A binary indicator for whether the job posting mentions interviews (1) or not (0).
- **industry:** The industry sector of the job.
- **function:** The job function or role (e.g., Sales, Marketing, Engineering).
- f**raudulent:** The target variable indicating whether the job posting is fraudulent (1) or not (0).

This dataset is instrumental in identifying and analyzing patterns that differentiate genuine job postings from fraudulent ones, which can enhance the reliability of job search platforms.

Aim & Target Variable

The aim of this project is to leverage data analytics and machine learning techniques to construct a robust framework capable of accurately identifying and flagging fraudulent job postings within online recruitment platforms. By developing sophisticated algorithms trained on extensive datasets, our objective is to create a reliable system that enhances the security and credibility of these digital job markets. Through this initiative, we seek to establish a safer and more trustworthy environment for job seekers and employers alike, thereby fostering meaningful connections and bolstering confidence in the online recruitment process.

Target Variable

The target variable in our dataset is 'fraudulent', which indicates whether a job posting is fraudulent or not. This variable is binary, with the following possible values:

- 0: Genuine job posting
- 1: Fraudulent job posting

This target variable is essential for training our machine learning models, as it allows the algorithms to learn the distinguishing features between genuine and fake job postings. By accurately predicting this target variable, our models aim to reduce the prevalence of fraudulent job advertisements, thereby protecting job seekers from frauds and enhancing the overall trustworthiness of online recruitment platforms.

Methodology

1. Data Acquisition and Understanding

- Data Access
    - Integration and Accessibility: The dataset, critical to our analysis, was accessed and integrated seamlessly via Google Drive with our Python-based analysis environment. This setup ensured efficient data manipulation and transformation directly within our analytical workflow.

- Data Exploration
    - Initial Assessment: Preliminary data exploration was conducted using Python's powerful visualization libraries to understand the distribution, potential biases, and anomalies within the dataset. This step was crucial for planning subsequent data cleaning and analysis phases.

2. Data Cleaning and Pre-processing

- Identifying and Removing Irrelevant Features
    - Streamlining Data: Initial steps involved identifying and removing non-essential identifiers such as 'job_id', which were deemed irrelevant for the predictive modeling process but could lead to model overfitting if included.

- Handling Missing Data
    - Strategic Imputation: Missing values in critical fields were handled by strategic imputation, where 'None' was used as a placeholder to maintain the integrity of the dataset. This approach allowed us to preserve valuable information that could be lost if entire records were removed.

- Optimizing Data for Processing
    - Type Conversion: To enhance computational efficiency, categorical data were transformed into numerical formats suitable for machine learning algorithms, reducing memory usage and processing time.

3. Exploratory Data Analysis (EDA)

- Deep Dive into Data Characteristics
    - Visualization: Using seaborn and matplotlib, detailed visualizations were created to identify trends, outliers, and correlations between variables. This exploratory phase helped to highlight significant predictors and informed the feature selection process.

4. Feature Engineering

- Enhancing Model Input
    - Reducing Dimensionality: Textual data fields were processed using TF-IDF vectorization to transform text into a meaningful numeric format. This method also helped in emphasizing words that are more important for classification and reducing the noise in the dataset.

- Multicollinearity Prevention
    - Correlation Analysis: To ensure that our models were not affected by multicollinearity, we performed a thorough analysis to identify and remove highly correlated variables. This step is crucial to enhance model generalization.

5. Model Development and Evaluation

- Selection of Robust Models
    - Algorithm Selection: A diverse set of algorithms was employed to cater to the different nuances of the dataset. Each model brought a unique approach to handling data imbalances and complex patterns.

- Training and Testing Strategy
    - Data Split Rationale: The choice of a 70 - 30 split for training and testing was aimed at providing a robust amount of data for learning while ensuring enough data remained to adequately test the model's performance in unseen conditions.

- Evaluation Metrics

  - Comprehensive Metrics: Models were evaluated using a suite of metrics including precision, recall, F1-score, and accuracy to provide a holistic view of performance across various aspects of the data.

6. Class Balancing Techniques

- Balancing via Sampling Techniques
  - Undersampling Implementation: To tackle the challenge of class imbalance, Random Under Sampling was utilized. This not only helped in addressing the issues but also ensured that the minority class was adequately represented in the training process.

7. Model Building

- Algorithm Selection and Rationale
  - Decision Trees: We began with Decision Trees because of their simplicity and interpretability. Decision Trees work by making sequential, hierarchical decisions about the data, effectively learning decision rules inferred from the features. They are particularly useful for binary classification tasks like ours.
  - Random Forest: To build on the Decision Trees' foundation, we employed Random Forest, an ensemble technique that builds multiple decision trees and merges their outcomes to improve accuracy and control overfitting. This model is known for its high accuracy, robustness, and ability to handle unbalanced data.
  - Naïve Bayes: Known for its simplicity and efficiency, Naïve Bayes classifiers were applied due to their probabilistic approach, which assumes independence between predictors. Naïve Bayes is particularly effective in classification based on a high-dimensional dataset after the TF-IDF vectorization of text data.
  - Support Vector Machines (SVM): SVM was chosen for its effectiveness in high-dimensional spaces and its versatility in both linear and nonlinear classification through the kernel trick. This model excels in complex classification tasks with clear margin of separation.
  - Multilayer Perceptron (MLP): As a type of neural network, MLPs were utilized to capture complex patterns in the data through layers of neurons with non-

linear activation functions. This model is suited for datasets where patterns are deeply embedded, and non-linear relationships are prevalent.

- K-Nearest Neighbors (KNN): Lastly, KNN was included for its non-parametric nature and ease of implementation. KNN classifies new cases based on a similarity measure (e.g., distance functions). KNN was particularly tested for its effectiveness in identifying fraud cases based on the proximity to known fraudulent samples in the feature space.

- Training and Validation Approach

  Each model underwent a rigorous training and validation process using the split datasets:

  - Training Phase: Models were trained on the balanced dataset, where hyperparameters were employed to ensure that the models do not overfit and can generalize well on unseen data.
  - Testing and Validation: The models were then tested using the 40% hold-out test set, which was untouched during the model training phase to simulate real-world application and test the models' generalizability.
  - Performance Evaluation and Model Tuning
  - Performance metrics such as accuracy, precision, recall and F1-score were calculated to assess each model's efficacy. These metrics helped in understanding the trade-offs between detecting as many frauds as possible and maintaining an elevated level of precision.

8. Conclusion and Recommendations

- Project Insights
  - Achievements: The project successfully developed models capable of detecting fraudulent job postings with high accuracy, thereby potentially reducing the risk of job frauds.

- Future Directions
  - Model Enhancements: Future work could explore more complex ensemble models and deep learning techniques for potentially higher accuracy.

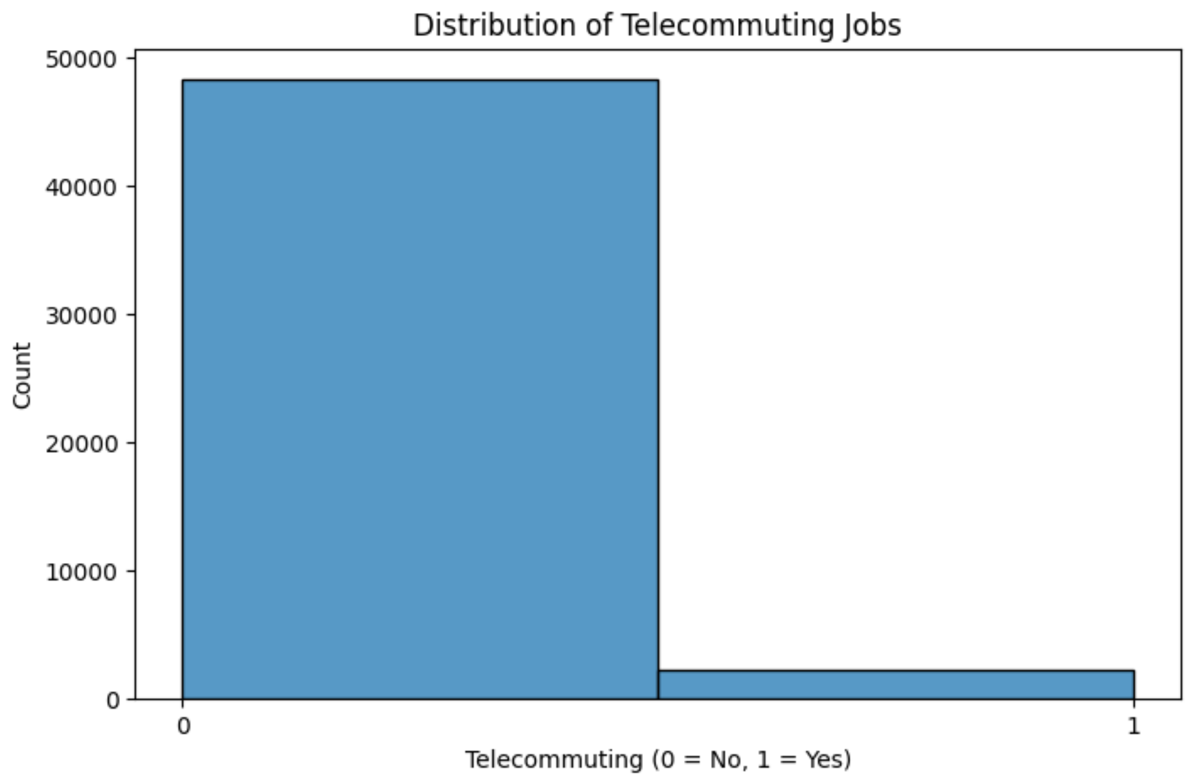
  - Feature Expansion: Incorporating additional features such as geographical data and employer history could provide deeper insights into patterns of fraudulence.

- Visualizations and Decision Support
  - Graphical Illustrations: Throughout the project, extensive use of graphical analyses helped in making informed decisions about model adjustments and in interpreting the models' outcomes effectively.
  - This extensive documentation encapsulates the detailed approach adopted in every phase of the project, ensuring transparency and methodological rigor, which are crucial for the validation and scalability of the machine learning solutions developed.

Visualizations

The visualization phase of the project involved a systematic approach to understanding and interpreting the relationships within the data, which is crucial for effective model building and validation.

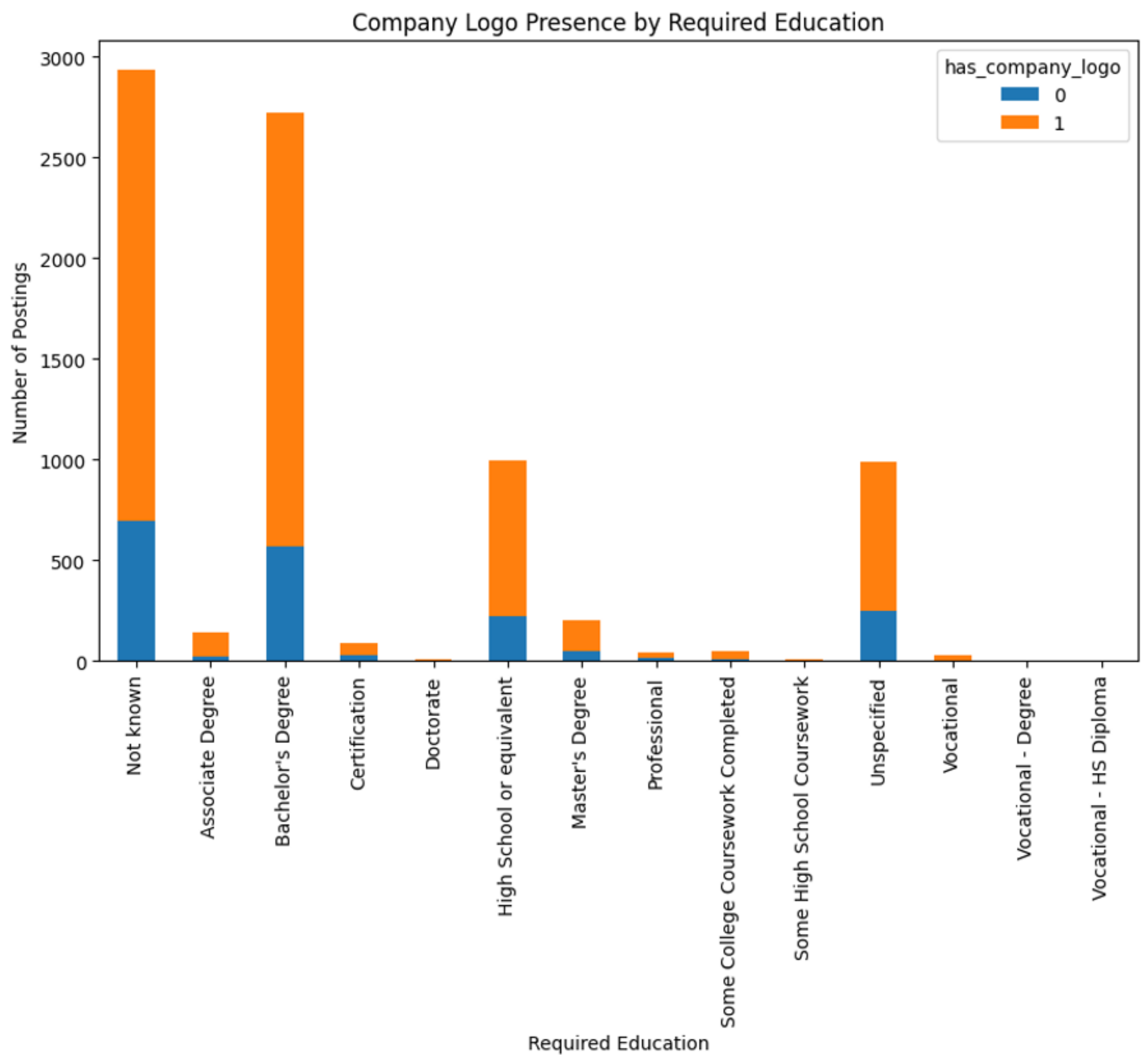
1) Distribution of Telecommuting Jobs



Interpretation:

The "Distribution of Telecommuting Jobs" bar chart unambiguously shows a notable disparity in the quantity of job posts depending on telecommuting opportunities. As seen by the bar for "0" (no) being significantly higher than the bar for "1" (yes), most job positions do not allow telecommuting. This shows that not all the jobs in this dataset allow for telecommuting. There may be fewer options for remote work because employers are offering on-site positions or because these occupations require physical presence.

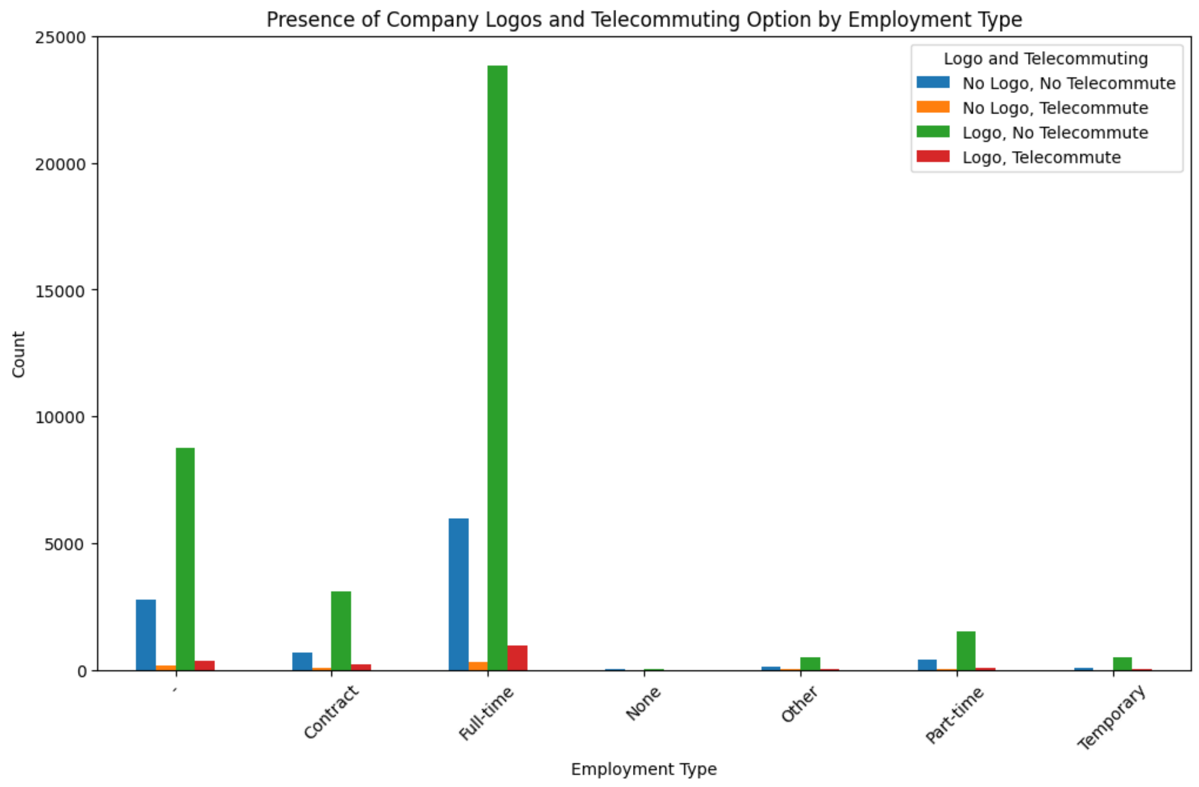
2) Company logo presence by required education



Interpretation:

The "Company Logo Presence by Required Education" bar chart shows the quantity of job ads arranged according to the educational qualifications, further subdivided by the posting's presence or absence of the company logo. The statistics clearly show that job ads needing a "Bachelor's Degree" or "Associate Degree" predominate. Additionally, a sizable fraction of these postings has corporate logos, indicating a strong branding presence. It is interesting to note that jobs with more educational requirements, like "Master's Degree" and "Doctorate," have fewer listings overall but still have a comparable amount of logo presence. There are a lot of entries in the "Not known" category, both with and without logos, which suggests different approaches to defining educational qualifications.

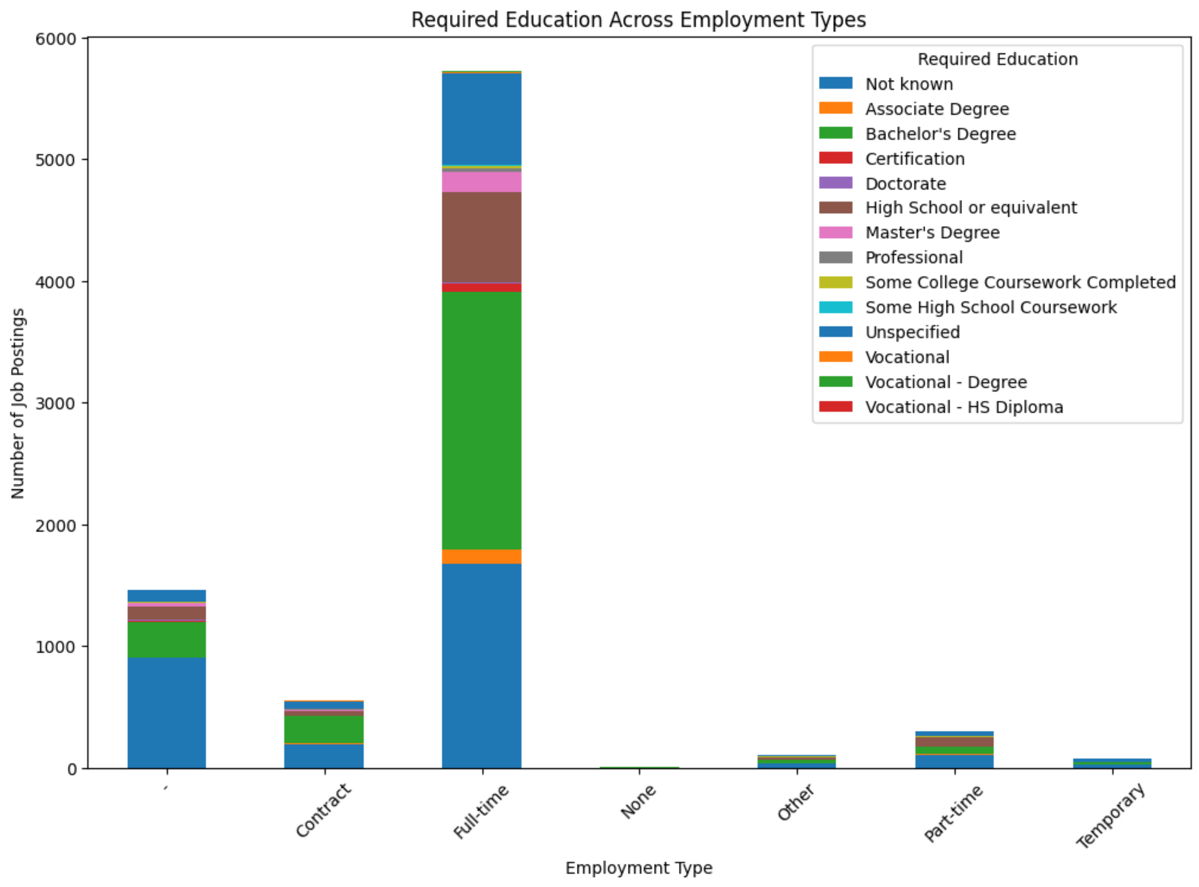
3) Presence of Company Logos and Telecommuting Option by Employment Type



Interpretation:

The bar chart titled "Presence of Company Logos and Telecommuting Option by Employment Type" illustrates a detailed breakdown of job postings across different employment types, categorized by the presence of a company logo and the option to telecommute. Full-time positions dominate the dataset, with a significant majority featuring company logos but not offering telecommuting. This suggests that while full-time roles are likely to project corporate branding via logos, they tend to require physical presence at the workplace. In contrast, telecommuting options, though generally less available, appear slightly more prevalent in part-time and contract roles, indicating a possible flexibility associated with these employment types. Notably, there are very few job postings that offer both a company logo and telecommuting, underscoring the rarity of such fully branded, flexible job opportunities.

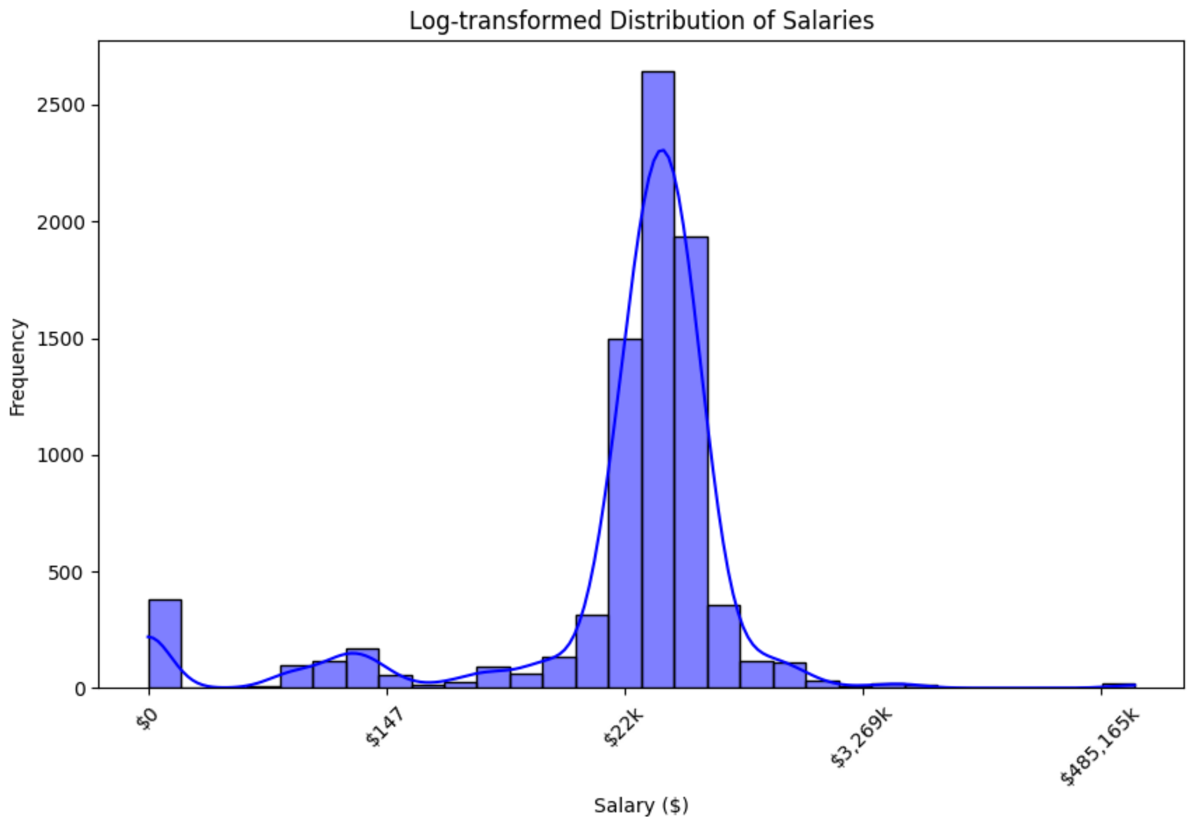
4) Required Education Across Employment Types



Interpretation:

The graphic "Required Education Across Employment Types" illustrates the disparities in educational requirements between job categories and shows the requirements for various kinds of work. Most full-time employment in the data requires a bachelor's degree, with high school diplomas or comparable training coming in second. This indicates a wide range of options, from entry-level to higher education levels. Although they are rarer, contract jobs have a comparable educational profile to full-time work, with a focus on bachelor's and associate's degrees. It is interesting to note that, although being far less common than full-time jobs, part-time and temporary jobs also emphasize lower educational degrees, which may indicate that they require fewer specialized work duties. This implies that obtaining full-time and contract jobs, which come with more responsibility and longer-term career prospects, requires more education.

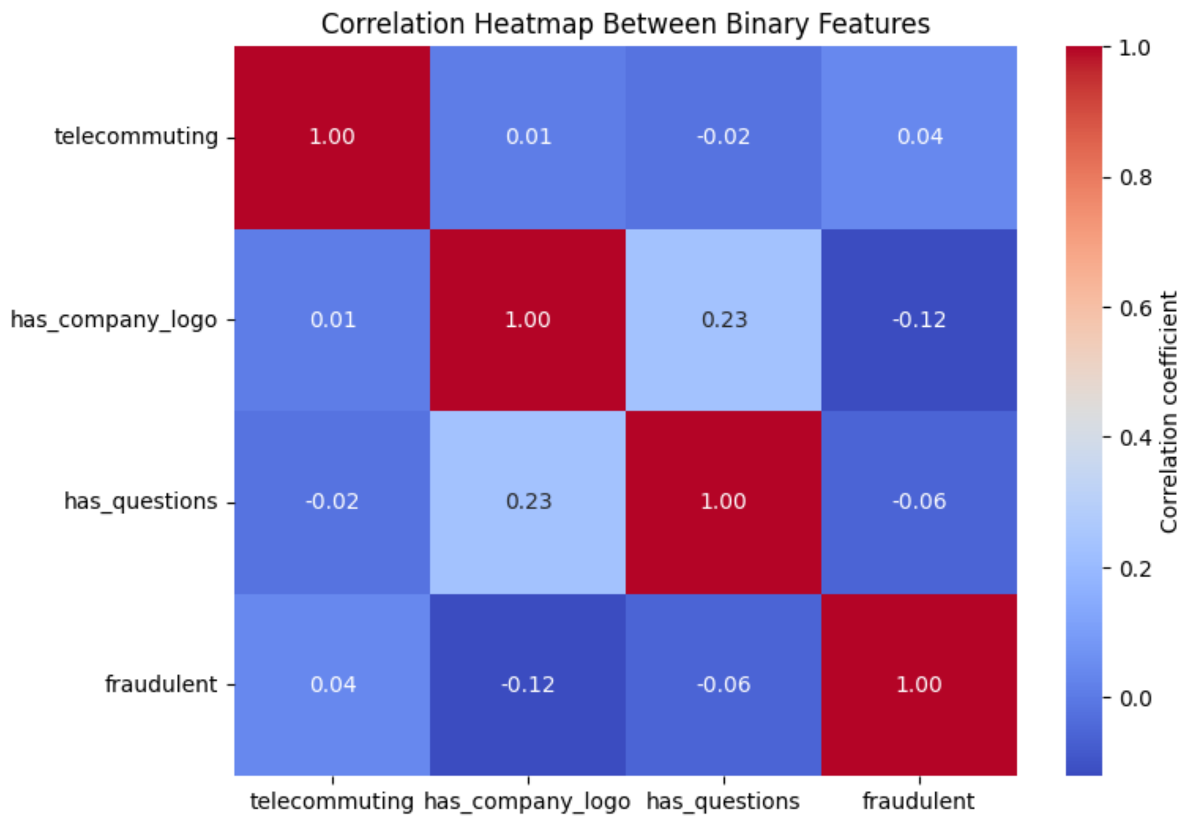
5) Log-transformed Distribution of Salaries



Interpretation:

After a logarithmic transformation is performed to equalize the scale of the data, the salary distribution of job posts is displayed in the histogram titled "Log-Transformed Distribution of Salaries." The graph exhibits a prominent peak at $32K, indicating that a considerable proportion of job advertisements provide salaries in the vicinity of this sum. Because of the right-skewed distribution, most incomes are concentrated around the median; nonetheless, a considerable proportion of positions pay much more, up to over $65K. The distribution drops off as salaries rise, showing fewer high-paying jobs. Lower salary ranges, below $15K, are less common. A smoothed depiction of the salary distribution over the sample is provided by the kernel density estimate superimposed on the histogram, which validates the skewness towards higher incomes.

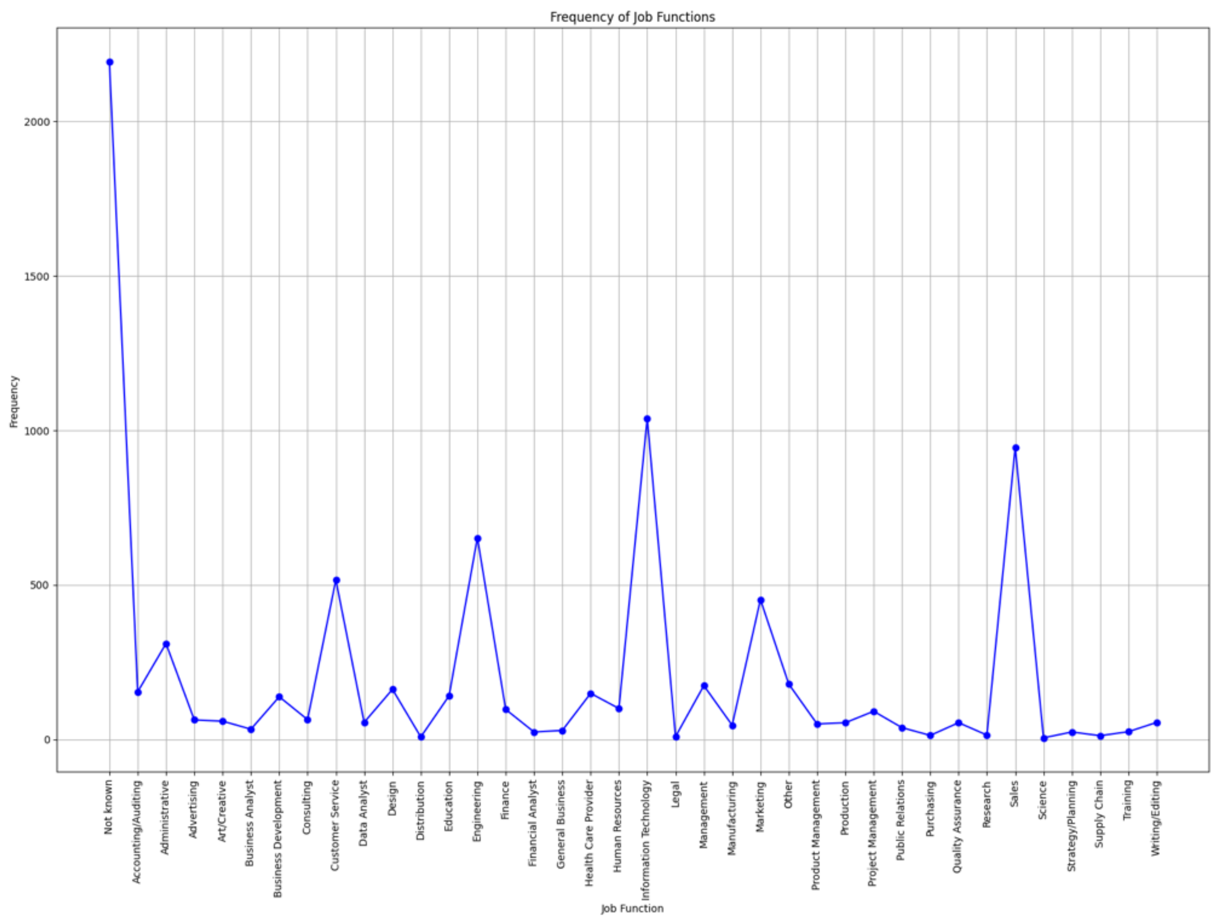
6) Correlation Heatmap Between Binary Features



Interpretation:

The correlation heatmap illustrates the weak overall connections between binary features of job ads. Application questions and firm logos have the largest association (0.23), indicating that job posts with logos typically have more formal application procedures. Jobs with logos are less likely to be fraudulent, according to a small negative association (-0.12) between firm logos and fraudulent posts. The fact that telecommuting has little correlation with other characteristics highlights how independent it is as a job asset. This analysis demonstrates that whereas job advertisements with extensive branding are more rigorous and reliable, other characteristics such as fraud and telecommuting are irrelevant.

7) Frequency of Job Functions



Interpretation:

The "Frequency of Job Functions" line graph shows how different job functions are distributed throughout the dataset. The graph is noteworthy because it shows multiple peaks that correspond to posting concentrations in particular functions. High spikes in the categories "Administrative," "Not Known," and "Information Technology" indicate that there are the most job openings in these fields, suggesting either a prominent level of market demand or a consistent categorization across the dataset. Roles with a lower frequency of postings, such as "Legal," "Science," and "Writing/Editing," indicate specialty markets. Job searchers may find this distribution useful in determining the relative availability of employment in various disciplines, and employers may find it useful in understanding the competitive nature of job functions.

Data Understanding and Preprocessing

Upload Data:

- We began by mounting Google Drive to access the dataset stored in Google Drive and imported the necessary libraries for data manipulation and visualization. The dataset used for this project is comprehensive, consisting of 50,447 rows and 24 columns, each representing various attributes of job postings.

Data Cleaning:

- Column Selection and Renaming: We started by examining the column names and selecting the desired columns for our analysis. Columns such as 'job_id', 'salary_range', 'salary_0_1', 'Employment_0_1', 'intern_0_1', 'Degree_1_0', and 'Interview_ol_of' were dropped as they were not directly relevant to our analysis.
- Handling Missing Values: We checked for missing values across the dataset. Columns with missing values included 'location', 'description', 'requirements', 'benefits', and 'employment_type'. Missing values were filled with 'None' to ensure that the dataset was complete and ready for analysis.
- Data Type Conversion: We examined the variable types and converted categorical variables to the 'category' data type. This included columns such as 'title', 'location', 'department', 'Salary', 'company_profile', 'description', 'requirements', 'benefits', 'employment_type', 'required_experience', 'required_education', 'industry', and 'function'.

Descriptive Statistics:

- We generated descriptive statistics for the entire dataset to understand the distribution and central tendencies of the numeric variables. This step helped in identifying patterns and outliers in the data.

Correlation Analysis:

- We conducted a correlation analysis to identify any highly correlated variables. This was done by calculating the correlation matrix for the numeric columns and visualizing it using a heatmap. However, no highly correlated variables were found, indicating that multicollinearity was not a concern for this dataset.

Dummy Variables:

We converted all the categorical variables into dummy variables so that the classification models can use that data while training.

Text Vectorization:

To effectively use textual data in our machine learning models, we applied text vectorization techniques. This process involved converting text data into numerical format using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. Here are the steps we followed:

- TF-IDF Vectorization: We used the TfidfVectorizer from the sklearn library to transform textual columns into numerical features. The text columns vectorized included 'title', 'company_profile', 'location', 'benefits', 'description', and 'requirements'.
- Feature Extraction: For each text column, we specified parameters such as max_features, min_df, and max_df to control the number of features generated and to filter out rare or simple words. This helped in creating a meaningful representation of the text data.
- Integration with Original Dataset: The resulting TF-IDF matrices were converted to DataFrames and concatenated with the original dataset, replacing the original text columns. This ensured that the text data was now in a numerical format, ready for model training.

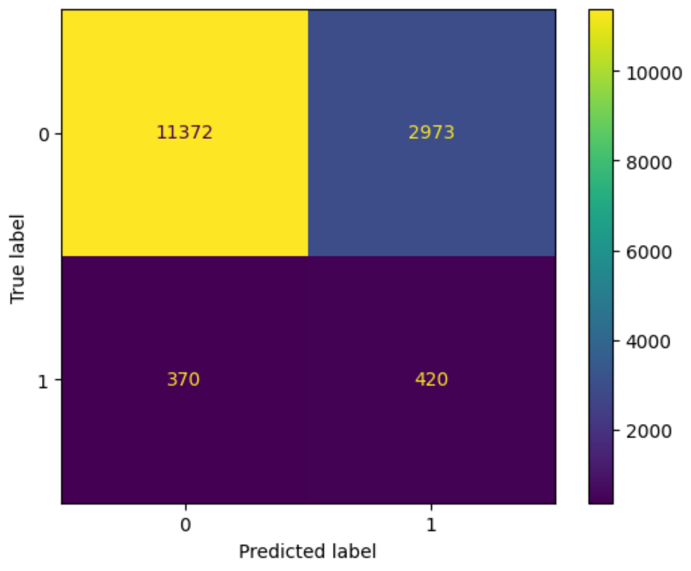
By vectorizing the text data, we were able to incorporate valuable information from job descriptions, company profiles, and other textual fields into our models, enhancing their ability to distinguish between genuine and fraudulent job postings.

These steps in data understanding, preprocessing, and text vectorization laid a solid foundation for building accurate and reliable machine learning models for detecting fraudulent job postings. If you need more specific details or additional data, please let me know.

Classification Models

1. Decision Trees

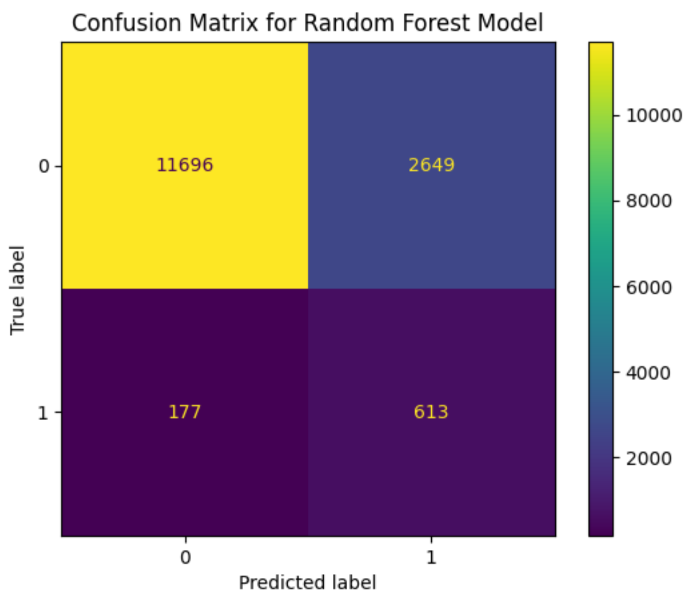
- Training Performance: The Decision Tree model achieved an accuracy of 68% during the training phase, suggesting that it could be underfitting given its simplistic approach with a maximum depth of 2.
- Testing Performance: On the test data, accuracy improved slightly to 78%. This indicates a lack of complexity in the model to capture the nuances of the dataset fully, reflecting a need for more depth to handle variability in the data.
- Insight: Increasing tree depth might capture more complex patterns, potentially increasing both accuracy and the model's ability to generalize better.



The above confusion matrix is based on testing data. According to the confusion matrix, the model classified 2,973 cases as false positives and 370 as false negatives, while correctly predicting 11,372 true negatives and 420 true positives. This suggests that while the model is good at detecting the negative class, it frequently predicts the positive class incorrectly or has a high false-positive rate. The model's moderate recall is indicated by the comparatively small number of true positives in comparison to true negatives. As a result, there is a need for improvement in the model's ability to distinguish between the two classes to decrease false positives and increase true positives. In summary, the model is effective at identifying negative instances but struggles to accurately predict positive instances.

2. Random Forest

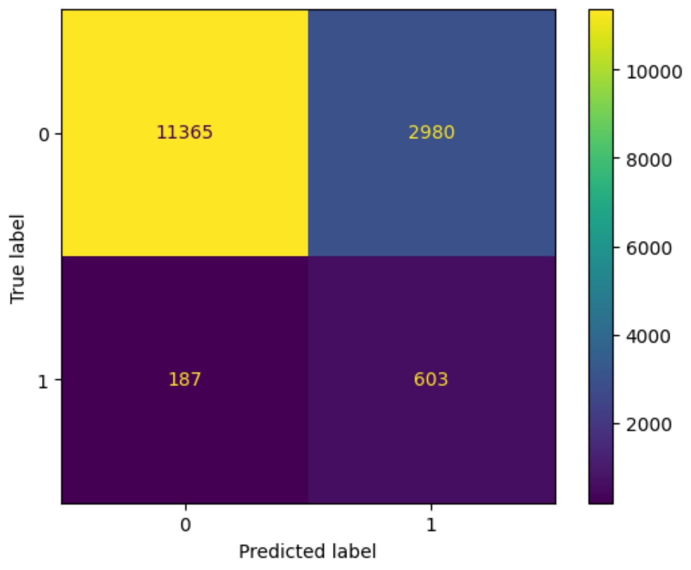
- Training Performance: Achieved a training accuracy of 73%, suggesting better generalization than Decision Trees due to the ensemble effect of multiple trees reducing variance.
- Testing Performance: Improved to 81% on the test set, which underscores Random Forest's capability to maintain performance across unknown data better than a single Decision Tree.
- Insight: While Random Forest shows promise, tuning the number of trees and max depth could optimize performance, balancing between overfitting and underfitting.



The above confusion matrix is based on testing data. The Random Forest model's confusion matrix reveals that while it correctly predicted 613 true positives and 11,696 true negatives, it incorrectly identified 2,649 instances as false positives and 177 as false negatives. With a comparatively lower rate of false positives compared to true negatives, this suggests improved precision and shows that the model is quite effective at identifying the negative class. In comparison to the previous model, the true positive count has increased, suggesting improved recall. Given the circumstances, the model does a decent job of differentiating between the two classes, decreasing false positives, and raising true positives, which enhances overall accuracy and strikes a balance between precision and recall.

3. Naive Bayes

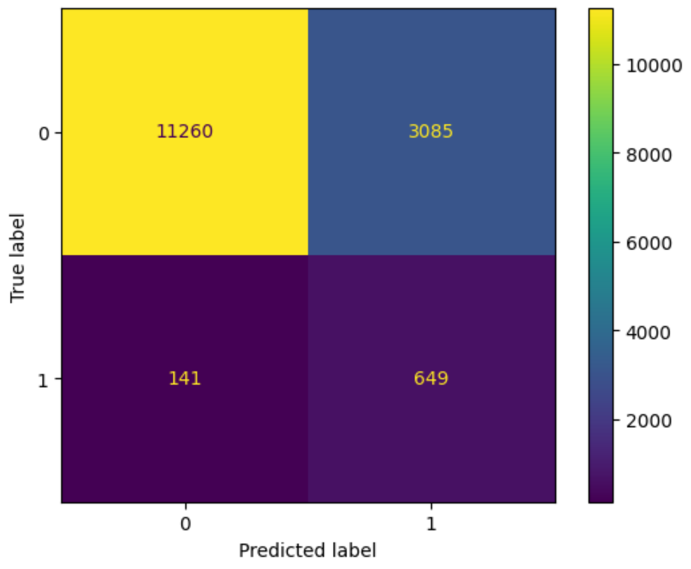
- Training Performance: Consistently showed an accuracy of 82% during the training phase, indicating stable performance under the assumption of feature independence.
- Testing Performance: Maintained similar accuracy levels on the test set, demonstrating good generalization capabilities of the model.
- Insight: Considering its stable performance, exploring feature selection, and reducing dimensionality could further refine its efficacy and reduce potential bias from irrelevant features.



The above confusion matrix is based on testing data. A binary classification model's performance is indicated by the confusion matrix. The matrix demonstrates that 603 true positives and 11,365 true negatives were correctly predicted by the model. 2,980 cases were incorrectly identified as false positives and 187 as false negatives. The substantial number of true negatives indicates that the model performs well in predicting the negative class. It still produces a sizable number of false positives, or predictions of the positive class that are not accurate. An enhanced recall is suggested by the comparatively higher number of true positives and the lower number of false negatives. Given the circumstances, the model successfully distinguishes both classes and, in comparison to previous models, has a better capacity for precisely predicting positives, improving overall precision and recall.

4. K-Nearest Neighbors (KNN)

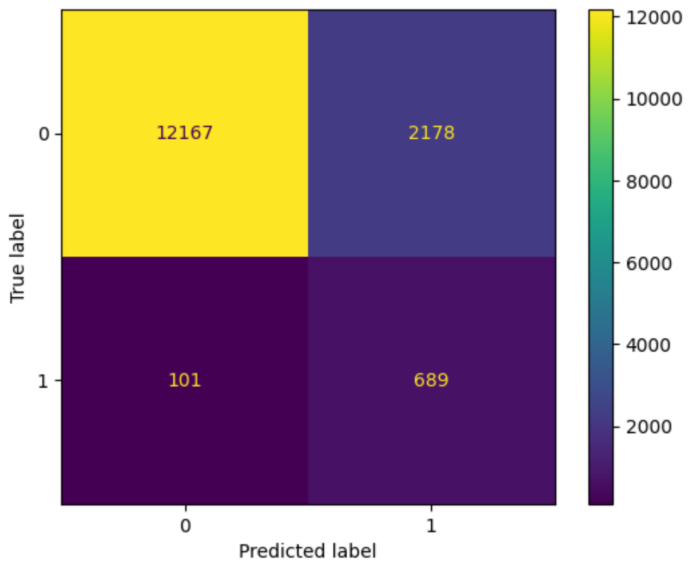
- Training Performance: KNN presented an accuracy of 86% in training, reflecting its effectiveness in a controlled environment where the feature distances significantly impact classification.
- Testing Performance: This dropped to 79% in testing, suggesting issues with overfitting or the influence of noise and scale in real-world data.
- Insight: Adjusting the number of neighbors and applying feature scaling techniques can enhance KNN's ability to perform more robustly on unseen data.



The above confusion matrix is based on testing data. The current model's confusion matrix displays 3,085 false positives and 141 false negatives, in addition to 11,260 true negatives and 649 true positives. This shows that while a considerable number of false positives are still produced, the model correctly detects most negative cases. The model is more adept at identifying the positive class, as evidenced by the rise in true positives and the decline in false negatives, which show an improvement in recall. Yet, the false positive rate is high, indicating that the model occasionally mispredicts negative occurrences as positive. The model has become more effective overall at identifying true positives, but more work may be required to lower the number of false positives to achieve more balanced accuracy.

5. Support Vector Machine (SVM)

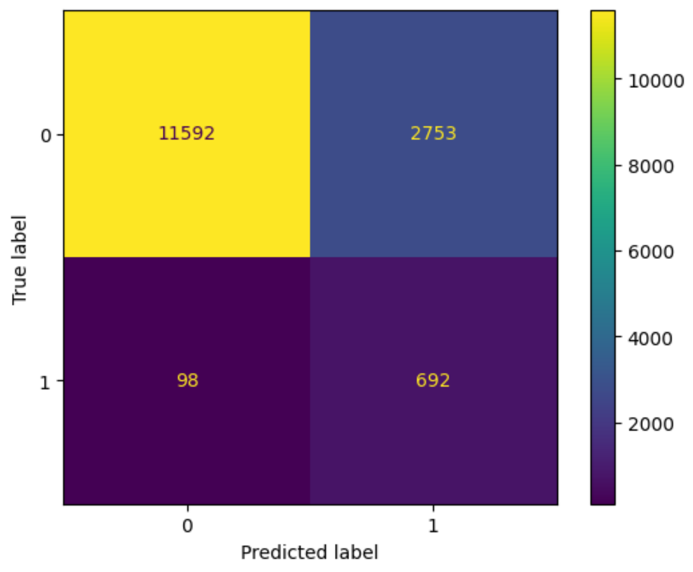
- Training Performance: SVM achieved an accuracy of 79% during training. Its high dimensionality handling and effective margin optimization strategy potentially make it suitable for this dataset's complexity.
- Testing Performance: Demonstrated the highest accuracy of all models at 85% on the test set, reinforcing its robustness and capability to handle complex patterns effectively.
- Insight: Exploring different kernels and regularization methods may help in achieving even better performance, particularly in handling non-linear separations more effectively.



The above confusion matrix is based on testing data. The current model's confusion matrix displays 12,167 true negatives, 689 true positives, 2,178 false positives, and 101 false negatives. This suggests that the model detects most negative cases correctly, but it still generates a considerable number of false positives. Recall has improved, as seen by the rise in true positives and the fall in false negatives, suggesting the model is more adept at identifying the positive class. Although there has been progress, the false-positive rate remains elevated, indicating that the model may occasionally mispredict negative occurrences as positive. The model has become more effective overall at identifying true positives, but more work may be required to lower the number of false positives to achieve more balanced accuracy.

6. Multilayer Perceptron (MLP)

- Training Performance: MLP showed a robust performance with a training accuracy of 90%, indicative of its ability to capture complex, non-linear relationships in the data.
- Testing Performance: However, the performance slightly decreased to 81% on the test dataset, which might suggest a beginning of overfitting to the training data.
- Insight: Tuning the architecture (number of layers, neurons per layer) and introducing dropout or other regularization techniques could help in preventing overfitting and improve its generalization to new data.



The above confusion matrix is based on testing data. The current model's confusion matrix displays 2,753 false positives and 98 false negatives, in addition to 11,592 true negatives and 692 true positives. This suggests the model generates a considerable number of false positives even though it correctly detects most negative cases. The model is more adept at identifying the positive class, as evidenced by the rise in true positives and the decline in false negatives, which show an improvement in recall. The model may occasionally mistakenly predict negative occurrences as positive, as evidenced by the comparatively high false positive rate despite this improvement. The model has become more effective overall at identifying true positives, but more work may be required to lower the number of false positives to achieve more balanced accuracy.

Conclusion

MLP models, with their capability to learn non-linear and complex patterns through multiple layers and neuron configurations, have proven to be highly adaptable for various predictive modeling tasks, including classification problems like fraudulent job posting detection. Each layer in an MLP can learn various aspects of the data, making it possible to capture a wide array of characteristics ranging from simple to complex. The choice of layer and neuron configuration is crucial, as it directly impacts the model's ability to generalize unseen data without fitting too closely to the training data (overfitting).

The comparison across different machine learning models highlights the strengths and weaknesses of each approach. Decision Trees and Random Forest models offer ease of interpretation and robustness against outliers. In contrast, SVM and Naive Bayes are beneficial for their efficiency and effectiveness in higher-dimensional spaces, respectively. KNN provides simplicity and effectiveness, especially in cases where the relationship between features is spatially oriented.

This comprehensive analysis allows stakeholders to make informed decisions about the most appropriate models to deploy based on specific needs and constraints, such as computational resources, required accuracy, and the nature of the data being analysed.

Future Scope

The future scope of this project involves several key areas for enhancement and exploration:

- Model Optimization: Further tuning of the SVM model by exploring different kernel functions, regularization techniques, and hyperparameters to improve performance.
- Feature Engineering: Developing new features and improving existing ones through techniques like natural language processing (NLP) to capture more intricate patterns in the job postings.
- Real-time Detection: Implementing the model in a real-time environment to detect fraudulent job postings as they are submitted, enhancing the security of online job platforms.
- Scalability: Ensuring the model's scalability to handle larger datasets efficiently, which is crucial for real-world applications with vast amounts of job postings.
- User Feedback Integration: Incorporating user feedback and incorporating mechanisms for continuous learning and improvement of the model based on new data and evolving patterns of fraudulent activities.
- Cross-platform Implementation: Extending the model's application across multiple job platforms to create a unified system for detecting fraudulent postings, thereby improving the overall trustworthiness of online recruitment.

By focusing on these future directions, the project can significantly contribute to creating safer and more reliable digital job markets, ultimately benefiting job seekers and employers worldwide.

References

1) "Real / Fake Job Posting Prediction | Kaggle." https://www.kaggle.com/datasets/shivamb/real-or-fake-fake-jobposting-prediction.

2) "Real/Fake Job Listings Dataset | Kaggle." 11 Mar. 2024, https://www.kaggle.com/datasets/prxshetty/fake-real-job-listings-dataset.

3) "Job Scams | Consumer Advice - Federal Trade Commission." https://consumer.ftc.gov/articles/job-scams.

4) "Scammers impersonate well-known companies, recruit for fake jobs on ...." 08 Aug. 2023, https://consumer.ftc.gov/consumer-alerts/2023/08/scammers-impersonate-well-known-companies-recruit-fake-jobs-linkedin-and-other-job-platforms.

5) "Job Scams: What to do | FTC Bulkorder Publications." https://www.bulkorder.ftc.gov/publications/job-scams-what-do.

6) "Imposter Scams | consumer.gov." https://consumer.gov/scams-identity-theft/imposter-scams.