Fake Job Posting Prediction

Milestone: Final Project Report

Group 42

Naveen Kumar Pasala
Shiva Naga Jyothi Cherukuri

pasala.n@northeastern.edu
cherukuri.sh@northeastern.edu

Percentage of Effort Contributed by Student 1:________50%________

Percentage of Effort Contributed by Student 2:________50%________

Signature of Student 1: ___________Naveen Pasala_____________

Signature of Student 2: _______Shiva Naga Jyothi Cherukuri____

Submission Date: __________________04/21/2023_________________

Table of Contents

Project Setting

Nowadays, with the proliferation of social media and online platforms, online frauds have become more prevalent, including the deception of "Fake Job Postings". Fake job postings and portals are causing harm to many individuals seeking employment, as fraudulent individuals profit from them. It is crucial to identify and classify these fraudulent postings to improve the job search experience for both employers and employees.

Problem Definition

"Fake Job Posting Prediction" is a data mining project that aims to uncover and flag fake job listings on online job boards and recruitment websites. This project will employ a blend of natural language processing, machine learning, and web scraping methods to gather and examine data from job postings.
Natural language processing techniques will be used to analyze the job descriptions for signs of fraud, such as unrealistic promises of pay and benefits or inconsistent information. Additionally, machine learning algorithms will be trained to detect patterns in the data that are indicative of fake job postings, such as certain keywords or phrases that are commonly used in scam postings. Finally, the system will be tested and evaluated to ensure that it can accurately identify and flag fake job postings with a high degree of accuracy.
There are several challenges that may arise in this project such as, Data availability and quality, evolving fraud tactics, High class imbalance, False positives, Privacy and security, and keeping the model updated.

Data Source

The first step of the project will involve web scraping job postings from various websites to collect a large dataset of job postings. We found a dataset in Kaggle in the link below.
https://www.kaggle.com/datasets/shivamb/real-or-fake-fake-jobposting-prediction?select=fake_job_postings.csv

Data Description

The above dataset consists of 18 attributes and 17880 rows of data which consists of columns such as job description. The data will be pre-processed to extract relevant features such as job title, company name, location, and job description. Following attributes can be used as predictors,

Sl. No	Attribute	Type
1	Telecommuting	Categorical
2	has_company_logo	Categorical
3	has_questions	Categorical
4	employment_type	Categorical
5	required_experience	Categorical
6	required_education	Categorical
7	Function	Categorical

Fig1. Possible Predictors

Data Mining Tasks

Data Understanding

The selected dataset consists of 18 attributes and 17880 rows of data which consists of columns such as job description, title, company, salary_range, telecommuting, has_compay_logo and one target variable called **'fraudulent.** The dataset mainly consists of text data, categorical variables, and nominal data of values '0' or '1'. The target variable (fraudulent) is also binary in nature where **'0'** is represented as **'non-fraudulent job posting'** and **'1'** is represented by **'fraudulent job posting'.** There are 17014 non-fraudulent and 866 fraudulent records in the dataset.

Data Pre-processing

From the descriptive statistics of dataset, there were 4 columns with binary data as the minimum value is '0' and maximum value '1', all the remaining columns have text data in them. The attribute 'job_id' was dropped from analysis as it is a unique column, which cannot be used for analysis to draw insights. Also, there were missing values in the data, hence assumed a threshold of 50% and dropped all the attributes having more than 50% of null values in each column such as department and salary range attributes, as imputation of such huge data can mislead the analysis. All the remaining columns having missing values were columns with text values hence, replaced all the nulls in those columns with '' (i.e., blank). After preprocessing of data, there are 17880 rows and 15 columns remaining to be processed further.

Data Exploration

In this step, preliminary data analysis to understand the characteristics of the dataset, identification of data quality issues, and exploration of relationships between variables were performed.

Data Imbalance

There is a data imbalance in this dataset as there are 17014 non-fraudulent job postings and 866 fraudulent records are present in this data, which is leading to just 4.84% of total data. Although, there is a data imbalance, but it is the ideal case in the real-world dataset as the fraudulent jobs would be very less compared to non-fraudulent records.

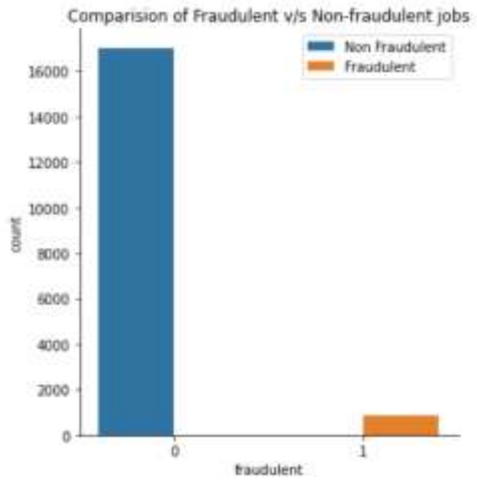


Fig.2. Data Imbalance

Outliers

There are no outliers in this dataset, as all the numeric data is in binary format of either '0' or '1' as shown in the figure below.

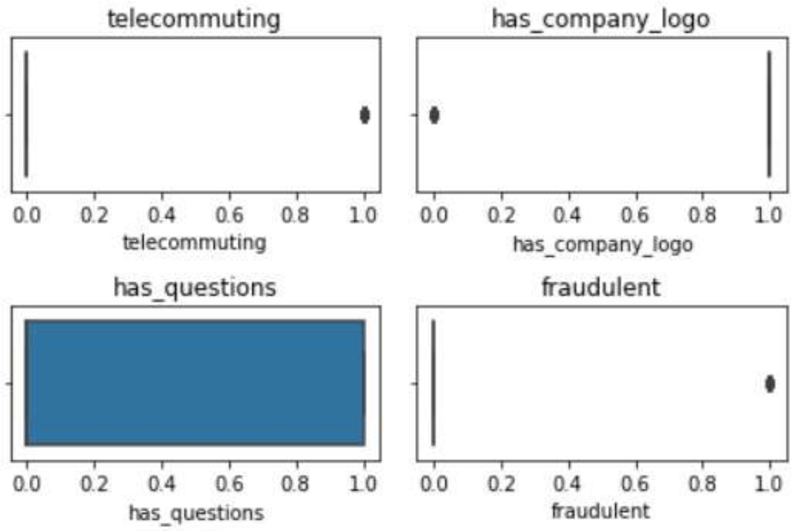


Fig.3. From the above box plot, it is evident that there are no outliers in data.

Data distribution

Distribution of each nominal columns shows, telecommuting is not available for most of the job postings, company logo is present for most of the companies that had job postings. Attribute has_questions had similar distribution for questions asked or not and non-fraudulent jobs are more in dataset.

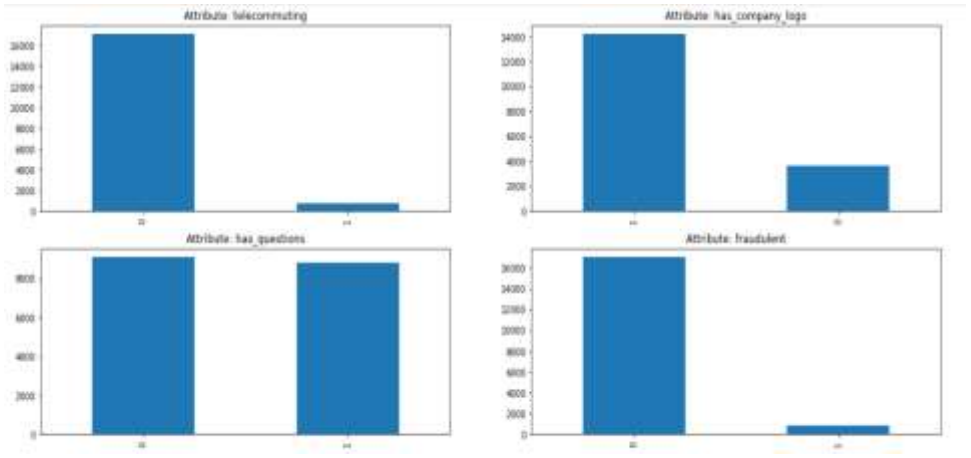


Fig. 4. Data Distribution of nominal columns

Distribution of types of employment types, required experience and required education are as indicated in the below figure Fig.1d. In the dataset, most of the job postings belong to Full-time employment type. Most required experience type is Mid-Senior level and bachelor's degree education type has more job postings compared to other types.

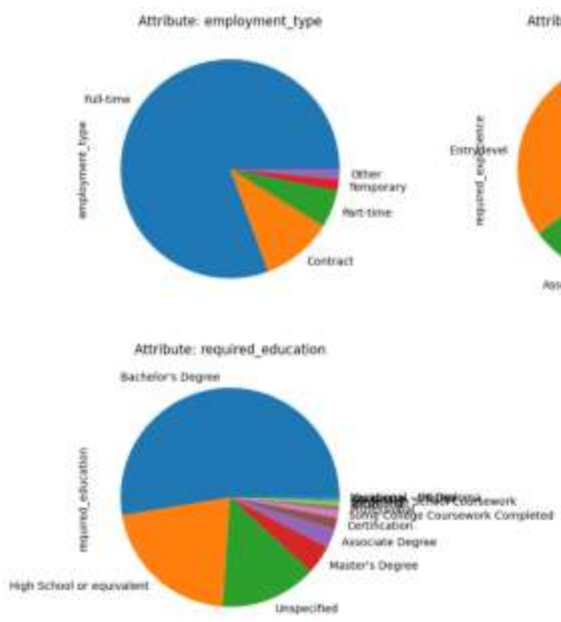


Fig. 5. Distribution of categorical variables

The Information and Technology services industry has the highest number of job postings compared to other industry types as per the below distribution.

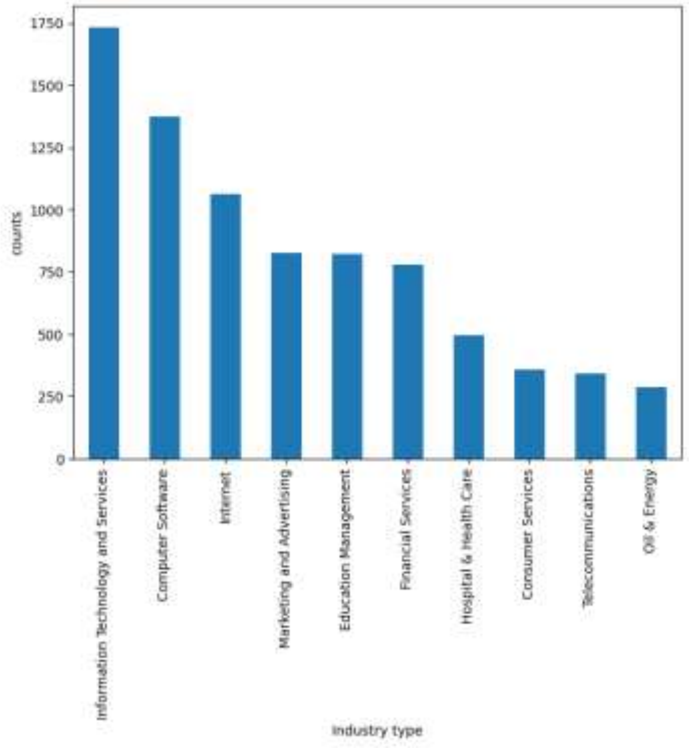


Fig. 6. Distribution of Industry variable

Like the industry type, Information Technology Function type is the most popular job posting in the dataset as shown in the below figure.

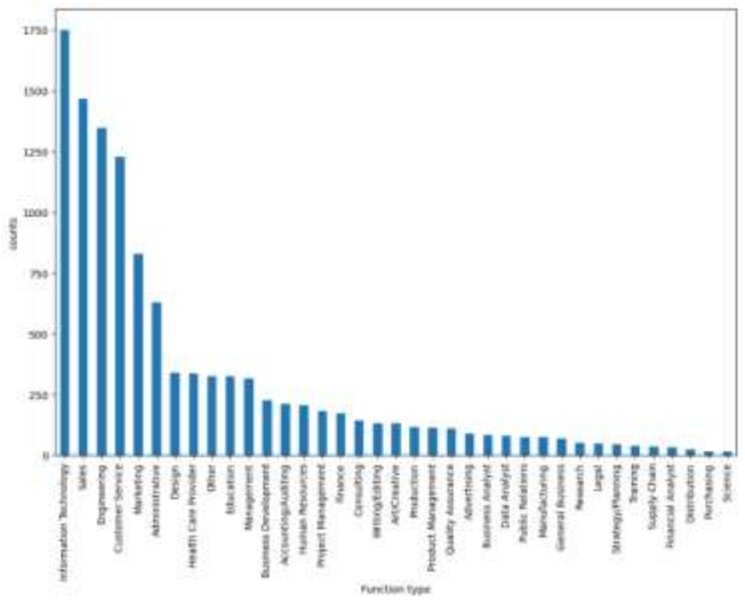


Fig. 7. Distribution of Function variable

One Hot encoding

Converted categorical attributes, employment type and required experience into dummy variables, which can be used as predictors during model building.

Correlation between Attributes:

Understanding the correlation between each attribute with the target variable (fraudulent column) is very important in the process of selecting the predictor variables for model building. Hence, plotted a correlation matrix to understand the relation of each attribute with the target variable (i.e., fraudulent). From the below image, we could clearly observe that Entry-level jobs are leading to 4% of fraudulent data and it's similar in case of part time jobs. Also, we can see that if a company has a logo, then it's 26% less likely to be fraudulent, which is a very good insight.

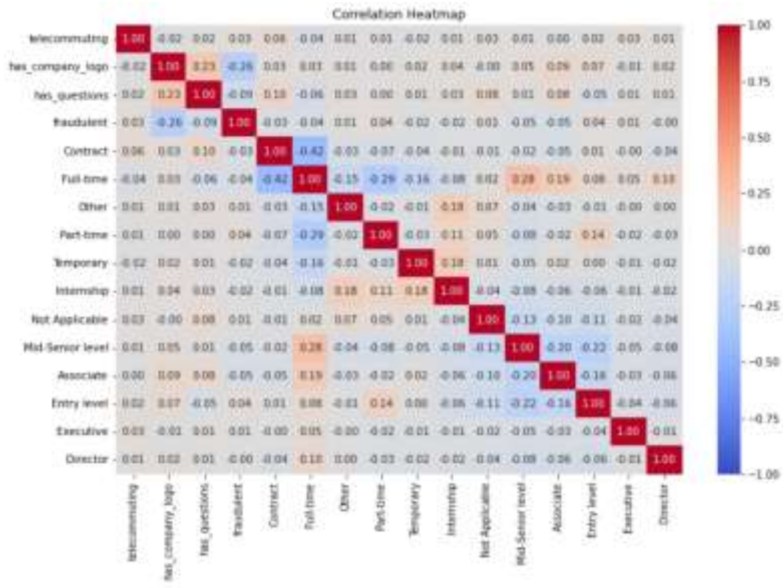


Fig. 8. Correlation Heatmap

Word Analysis

Merged all the columns having text data into Job details column and performed the text analysis to identify the most frequently used words in fraudulent jobs and non-fraudulent job postings and below are the images showing the most frequently used words in both the job posting types.

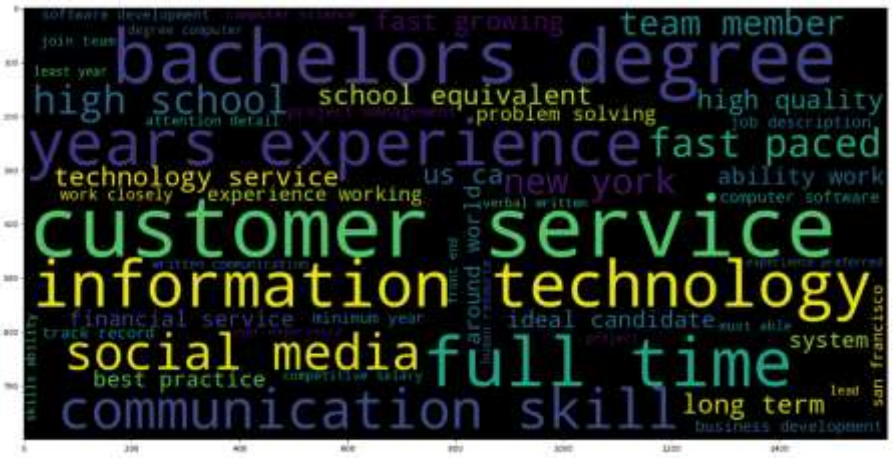


Fig.9. Word frequency of Non-Fraudulent Jobs

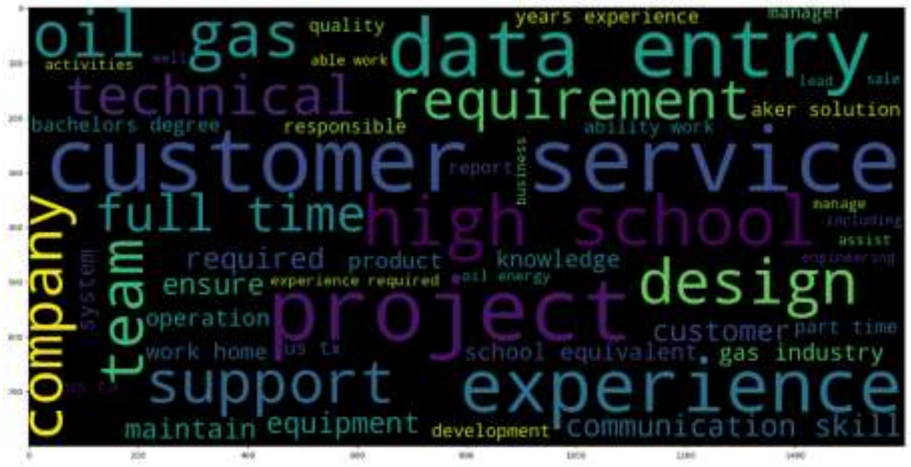


Fig. 10. Word frequency of Fraudulent jobs

Computed the count of existence of each word in fraudulent and non-fraudulent jobs and got the words which are only used in fraudulent jobs and not in non-fraudulent jobs. Assuming a threshold of words occurred more than 50 times, considers words such as ‘aker’, ‘subsea’, ‘accion’, ‘novation’ and ‘overviewaker’ and created a dummy column for each of these words.

Feature Selection

In the above cleaned dataset, there are totally 21 attributes, hence performed the feature selection to select the most important features from dataset using chi square test.

Data Preparation

As all the data is in binary values of ‘0’ and ‘1’, we had not performed any data standardization or scaling. But as our data is imbalanced, we had to oversample the minority class using Random Over Sampler method.

Data Partitioning

As our data is label, we are building supervised machine learning models hence we had partitioned the data into 70% training and 30% testing.

Data Mining Models:

The model selection depends on the type of predictors in the dataset. It is also very necessary to try multiple models and compare their performance to determine which one is the best fit for predicting fake job postings. Below is the list of models we are considering exploring with our data.

Logistic Regression:

It is a statistical method used for binary classification problems, where the outcome variable is taking on two possible values. It is commonly used for modeling the relationship between a binary dependent variable and one or more independent variables. This model can be used to predict the probability of a binary outcome whethera job posting is fake or not.

<u>Advantages:</u>

- Easy to implement and interpret.
- Computationally efficient and can handle large datasets with multiple predictors.

<u>Disadvantages:</u>

- Assumes that the relationship between independent and dependent variable is linear, which can be a limitation if the relationship is more complex.
- Sensitive to outliers and the presence of multicollinearity.

<u>Implementation:</u>

The base logistic regression model was executed with accuracy of 80.35%, Precision of 17.67%, recall as 78.96% and F1 score of 28.87%.

Decision Trees:

Decision Trees are a popular machine learning algorithm used for both classification and regression tasks. They work by recursively splitting the dataset into subsets based on the values of input variables and creating a tree-like structure of decisions to classify or predict the target variable.

The input variables are used to make a series of binary decisions that result in final decision about target variable. Each internal node of the tree represents a decision based on the value of an input variable, and each leaf node represents a classification or prediction of result.

<u>Advantages:</u>

- It can handle both categorical and continuous input variables and handle missing data without requiring imputation.
- They can handle non-linear relationships between input and output variables.

<u>Disadvantages:</u>

- Decision Trees are prone to overfitting the training data, especially when the tree is deep or complex. This can result in poor generalization performance on new data.
- They can be sensitive to small variations in the training data, which can result in different

trees being generated for similar datasets.

Implementation:

Base decision tree was executed with accuracy of 83.66%, Precision of 20.01%, recall as 74.53% and F1 score of 31.56.

Random Forest:

It is an ensemble learning method that combines multiple Decision Trees to improve the accuracy and generalization performance of the model. In a Random Forest,a set of decision trees is trained on different subsets of the data, using randomly selected subsets of the input variables for each tree. By the combination of predictions of all the trees, the final prediction is made.

Advantages:

- Random Forest can handle both regression and classification problems and can handle both categorical and continuous input variables.
- It is less prone to overfitting than a single decision tree as it combines multiple decision trees, reducing variance of the model and improving its generalization performance.

Disadvantages:

- Random Forest can be computationally expensive, for large datasets and complex models with many decision trees.
- It can be difficult to interpret the results of a Random Forest model, especially when the number of decision trees is large.

Implementation:

The base model was executed with accuracy of 83.94%, Precision of 20.20%, recall as 73.80% and F1 score of 31.72%.

Naive Bayes:

It is a probabilistic machine learning algorithm that is based on Bayes' Theorem. It is commonly used for classification tasks and works by calculating the conditional probabilityof each class given the input variables and selecting the class with the highest probabilityas the prediction. Naive Bayes assumes that the input variables are independent of eachother, which is known as the "naive" assumption. This simplifies the calculation of the conditional probabilities and makes the algorithm computationally efficient.

Advantages:

- Naive Bayes is computationally efficient and can handle large datasets with many input variables with easy implementation and interpretation.

Disadvantages:

- Naive Bayes assumes that the input variables are independent of each other, which may not be true in practice. This can lead to suboptimal performance when the input variables are correlated.
- Naive Bayes is sensitive to the quality and relevance of the input data and may not perform well with noisy or incomplete data.

Implementation:

The base model was executed with accuracy of 80.79%, Precision of 17.53%, recall as 75.64% and F1 score of 28.47%

KNN:

KNN (k-Nearest Neighbors) is a non-parametric machine learning algorithm used for both classification and regression tasks. KNN works by finding the k nearest neighbors to a givendata point based on the distance metric and making predictions based on the class or value of most of the k neighbors.

Advantages:

- KNN is easy to understand and implement which can handle both regression and classification problems along with categorical and continuous input variables.

Disadvantages:

- KNN can be computationally expensive, especially for large datasets and high- dimensional input spaces.
- It is sensitive to the choice of distance metric and the value of k, which can affect the accuracy and generalization performance of the model.

Implementation:

The base model was executed with accuracy of 91.01%, Precision of 20.77%, recall as 27.67% and F1 score of 23.78%.

XGBoost:

XGBoost (Extreme Gradient Boosting) is a powerful ensemble learning method that combines multiple weak learners to create a strong learner. XGBoost uses gradient boosting to train a series of decision trees and combines their predictions to make the final prediction.

Advantages:

- It has a built-in regularization parameter to prevent overfitting, and it can handle missing data without requiring imputation.
- It is computationally efficient and can handle large datasets and high-dimensional input spaces.

Disadvantages:

- It may not perform well on datasets with imbalanced class distributions, where the minority class is significantly smaller than the majority class.
- It may require tuning of many hyperparameters, which can be time-consuming and challenging for beginners.

Implementation:

The base model was executed with accuracy of 84.13%, Precision of 20.34%, recall as 73.43% and F1 score of 31.86%.

Performance Evaluation:

Logistic Regression

It is used to predict the likelihood of a binary outcome, such as determining whether a job posting is fraudulent or not. The accuracy, precision, recall, and F1 score of the model are measures used to assess the performance of the model in making correct predictions. The accuracy of 80.35%, which is lower and the precision score of 17.67% indicates that the model has a high number of false positives. However, the recall score of 78.96% suggests that the model is capturing most of the relevant instances of the target class. The F1 score of 28.87% is also lower, indicating that the logistic regression model's overall performance is worse than the base model.

```
Model 1 - Logistic
RegressionAccuracy:
0.8035048471290082
Precision: 0.17671345995045418
Recall: 0.7896678966789668
F1-score:
0.28879892037786775
```

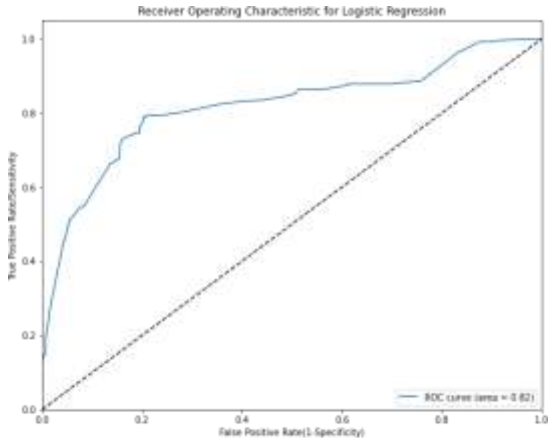


Fig11. ROC Curve for Logistic Regression

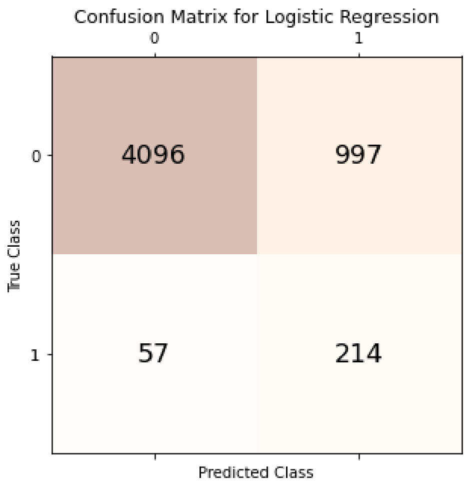


Fig12. Confusion Matrix for Logistic Regression

Decision Trees:

The decision tree model has an accuracy of 83.66%. The precision score of 20.01% is also comparable to the base model's precision score. The recall score of 74.53% suggests that the decision tree model is capturing most of the relevant instances of the target class. However, the F1 score of 31.56% is like the base model's F1 score, indicating that the decision tree model's overall performance is not very good.

```
Model 2 - Decision Tree
Accuracy: 0.8366890380313199
Precision: 0.2001982160555005
Recall: 0.7453874538745388
F1-score: 0.31562500000000004
```

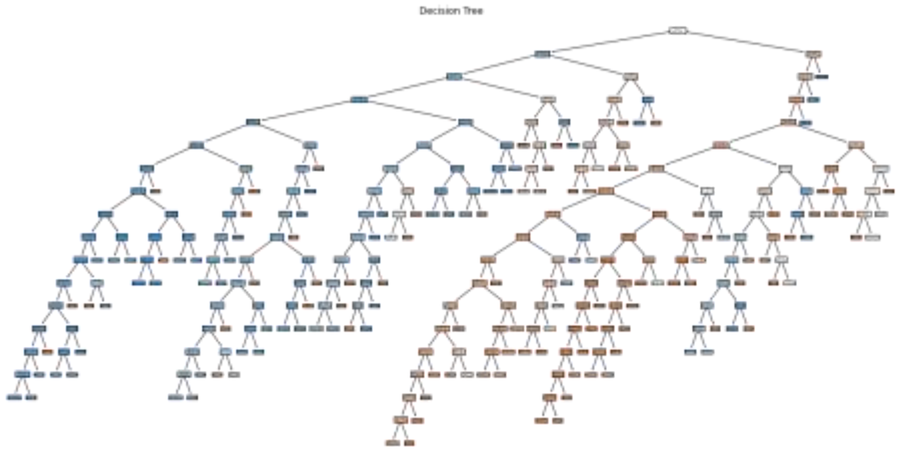


Fig13. Decision Tree

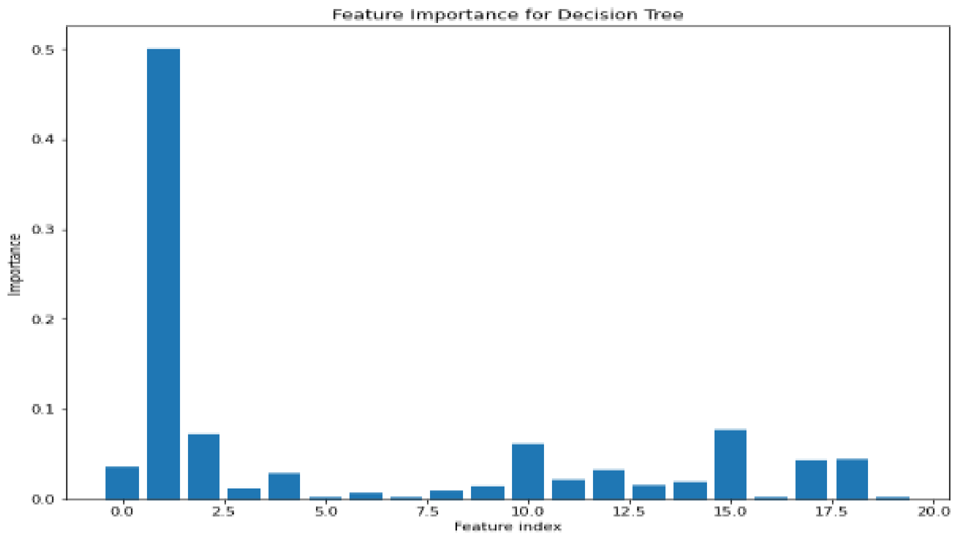


Fig14. Feature Importance for Decision Tree

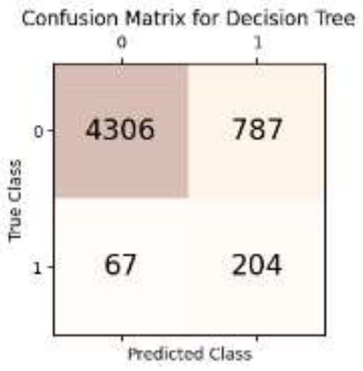


Fig 15. Confusion Matrix

Random Forest:

The model has a high accuracy of 83.94%. However, the low precision of 20.20% indicates that there may be a high number of false positives. The recall score of 73.80% suggests that the model may not be capturing all the relevant instances of the target class. The F1 score of 31.72% is a harmonic mean of precision and recall, indicating that the model's overall performance is good compared to other models but not the ideal model.

```
Model 3 - Random Forest
Accuracy: 0.8394854586129754
Precision: 0.20202020202020202
Recall: 0.7380073800738007
F1-score: 0.317208564631245
```

Fig16. Random Forest

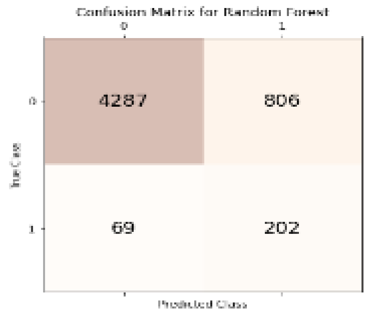


Fig 17. Confusion Matrix for Random Forest Model

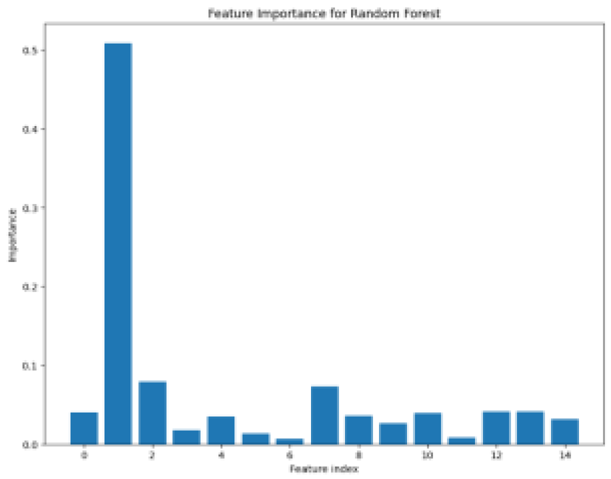


Fig18. Feature Importance for Random Forest

Naïve Bayes:

The model has an accuracy of 80.79%, which is comparable to the other models' accuracy. However, the precision score of 17.53% indicates that the model has a high number of false positives. The recall score of 75.64% suggests that the model is capturing most of the relevant instances of the target class. The F1 score of 28.47% is lower than the other models' F1 scores, indicating that the model's overall performance is not very good.

```
Model 4 - Naive Bayes
Accuracy: 0.807979120059657
Precision: 0.17536355859709152
Recall: 0.7564575645756457
F1-score: 0.2847222222222222
```

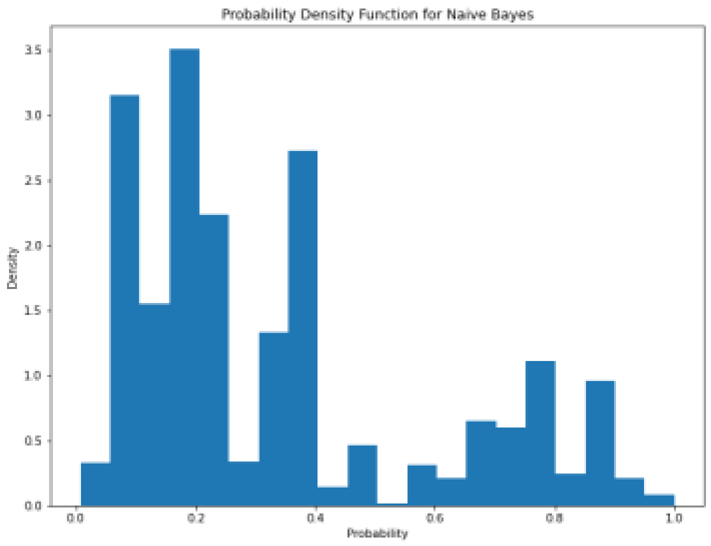


Fig 19. Probability density function for Naïve Bayes

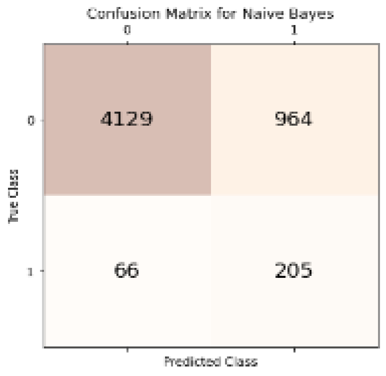


Fig 20. Confusion Matrix

KNN:

The model has a high accuracy of 91.01%, but a low precision score of 20.77%, indicating a high number of false positives. The recall score of 27.67% suggests that the model is capturing only a small fraction of the relevant instances of the target class. The F1 score of 23.78% is low, indicating that the model's overall performance is not very good.

```
Model 5 - K-Nearest Neighbors Model
```

```
Accuracy: 0.9101416853094706
Precision: 0.2077562326869806
Recall: 0.2767527675276753
F1-score: 0.23734177215189872
```

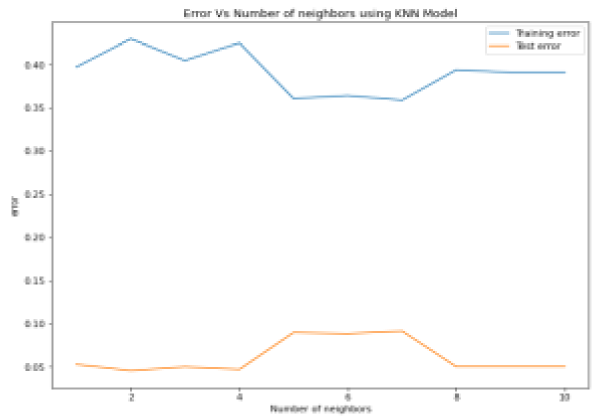


Fig 21. Testing and Training errors of a KNN model at different n's

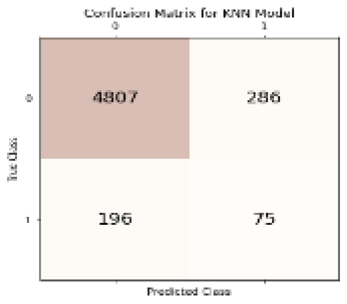


Fig 22. Confusion Matrix for KNN Model

XGBClassifier:

The model has a high accuracy of 84.13%. However, the precision score of 20.34% indicates that the model has a high number of false positives. The recall score of 73.43% suggests that the model is capturing most of the relevant instances of the target class. The F1 score of 31.86% is similar to other models' F1 scores, indicating that the model's overall performance is average.

```
Model 6 - XGBoost Model
Accuracy: 0.8413497390007457
Precision: 0.2034764826175869
Recall: 0.7343173431734318
F1-score: 0.31865492393915135
```

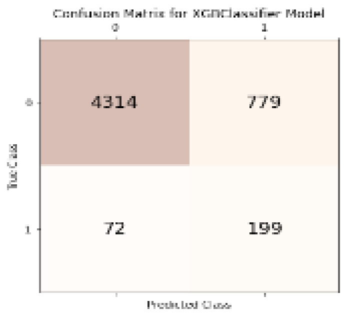


Fig 23. Confusion Matrix for XGBClassifier

Comparing the Evaluation metrics for all the above models:

a. *Comparing the accuracies of all the models*

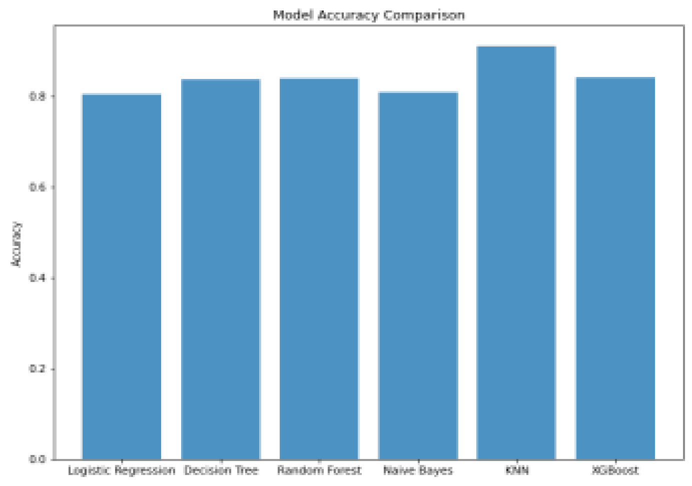


Fig24. Accuracy Comparison.

KNN Model has the highest accuracy of 95.5% compared to all the other models.

b. *Precision comparison of all models*

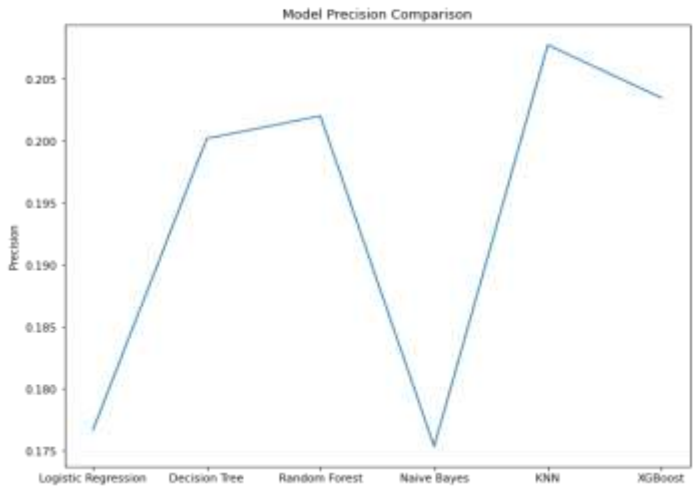


Fig25. Precision Comparison

The KNN model has the highest precision of 0.207 and the Naive Bayes model has lowest precision value of 0.175 compared to other models.

c. Comparing Recall of all models

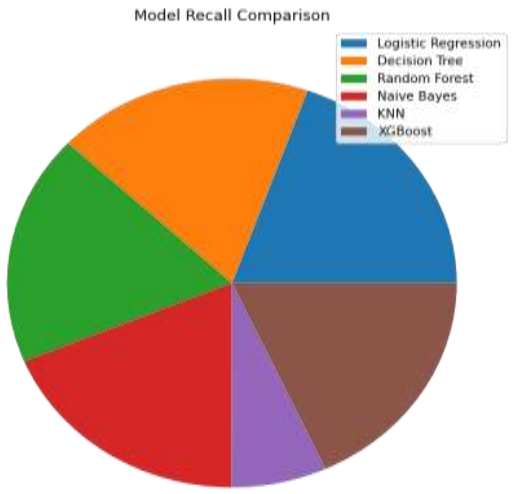


Fig 26. Recall Comparison.

The Logistic Regression model has the highest recall value of 0.789 and KNN model has lowest recall value of 0.276 compared to all the models.

d. F1 Scores comparison of models

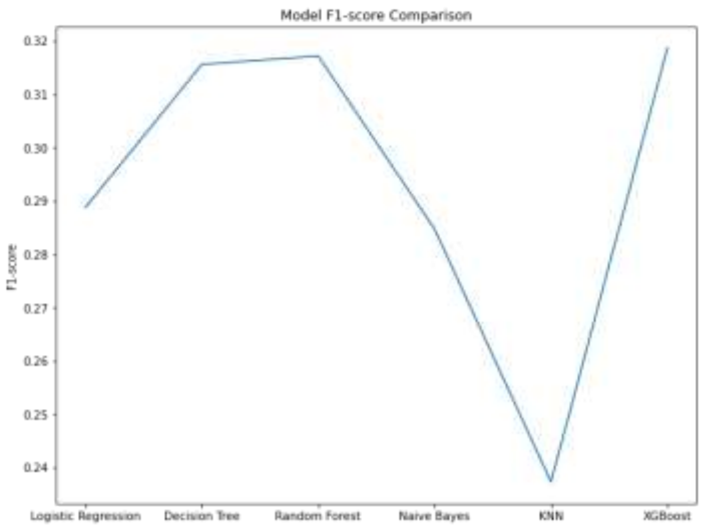


Fig 27. F1 Score Comparison

XGBoost model has the highest value of F1 Score of 0.318 and KNN model has the lowest value of F1 Score of 0.237 compared to all models.

e. Comparing ROC Curves of all the models

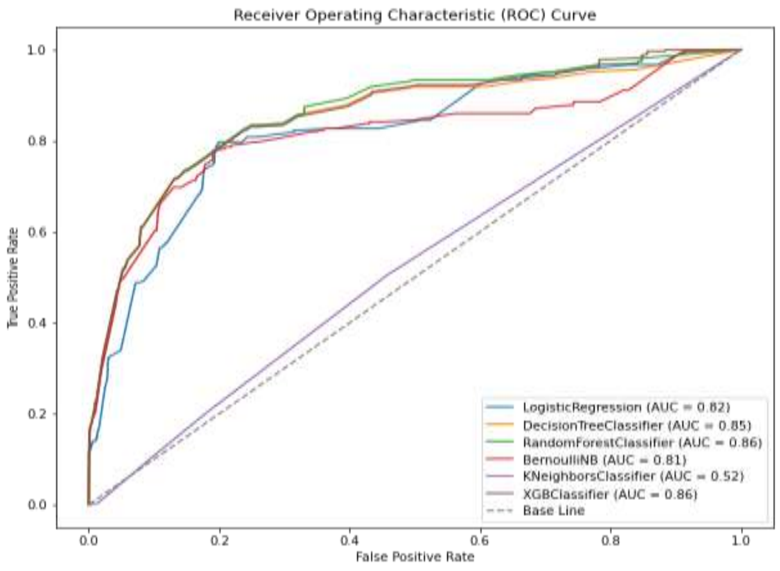
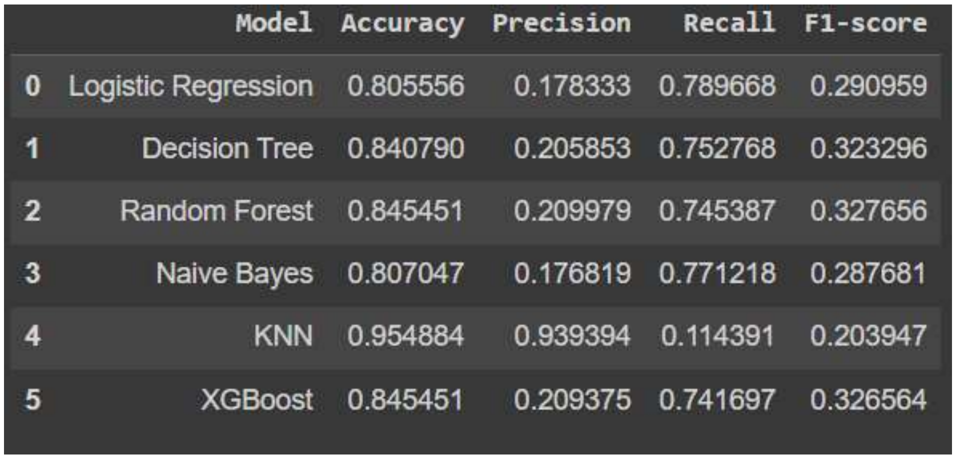


Fig 28. ROC Curve comparisons.

ROC curve indicating the Area Under the Curve (AUC) is highest for Random Forest model and XGBClassifier models.

f. Summary Table to show all the models and their performance metrics.



	Model	Accuracy	Precision	Recall	F1-score
0	Logistic Regression	0.805556	0.178333	0.789668	0.290959
1	Decision Tree	0.840790	0.205853	0.752768	0.323296
2	Random Forest	0.845451	0.209979	0.745387	0.327656
3	Naive Bayes	0.807047	0.176819	0.771218	0.287681
4	KNN	0.954884	0.939394	0.114391	0.203947
5	XGBoost	0.845451	0.209375	0.741697	0.326564

Fig 29. Summary metrics table.

Hyperparameter Tuning:

Implemented hyperparameter tuning on the best model from all the above models. The best model seems to be the Random Forest model with an accuracy of 0.845 and a recall of 0.745. Used Grid Search CV function from sklearn.model_selection library for tuning the parameters, in the first round the accuracy of the model has dropped to 84.1% and later tried to optimize the parameters to improve the accuracy of the model more but it remained the same. So, after hyperparameter tuning the model was still performing the same, hence there was no further effect on the model. So, the model is performing better with default parameters.

Project Result:

Looking at all the above statistics, the best model for the given dataset seems to be the Random Forest model with an accuracy of 0.839 and a recall of 0.738. Although the precision values of the KNN and XGBClassifier models are higher than that of the Random Forest model, their recall values are considerably lower, which means that they have a higher chance of missing out on positive instances.

The Naive Bayes model also has a high recall value, but it has the lowest precision value among all the models, which implies that it may not be suitable for this dataset. The K-Nearest Neighbors model has a lowest recall and F1 Score than all other models, making it unsuitable for this dataset. On the other hand, the Decision Tree and Random Forest models have nearly similar metrics. But the Random Forest model has slightly higher accuracy, F1 Score and precision than compared to Decision tree which makes it the better model. Additionally, Random Forest models can handle categorical features well and can handle missing values in the dataset, making it a good choice for this nominal dataset. Therefore, the Random Forest model seems to be the best model for this dataset.

Impact of the project outcomes/Learnings:

In this project, the goal was to be able to identify the fraud job postings based on the text analysis performed on the fraudulent jobs posts and analyze them accurately. The data imbalance was high as we had 96% of real postings and only 4% of fraud postings, which would be the ideal case. The analysis helped us to find the fraud job posts based on the top 5 frequent words repeated in the Fraudulent posts. The purpose of this project was to accurately classify fraudulent job postings by analyzing their text and converting them into binary variables, which provided valuable experience in working with text data. The project was a great opportunity to learn how to use the chi-square test to select the most important features from various attributes. We gained practical knowledge in implementing various machine learning models, evaluating their performance, and selecting the optimal model. We had also been involved tuning the hyperparameters of the model to improve its performance, which was a novel concept we learned during this project.

References:

1. "A Comparative Study on Fake Job Post Prediction Using Different Data Mining Techniques": https://www.researchgate.net/publication/349884280_A_Comparative_Study_on_Fake_Job_Post_Prediction_Using_Different_Data_mining_Techniques
2. Matplotlib: https://matplotlib.org/
3. Seaborn: https://seaborn.pydata.org/
4. scikit-learn: https://scikit-learn.org/stable/
5. XGBoost: https://xgboost.readthedocs.io/en/stable/