DETECTING FAKE VERSUS REAL JOB POSTINGS

Daisy Adhikari, Iram Khan, Huan Nguyen, Madhur Parakh, Kranthi Raj Vellanki
Department of Applied Data Science, San José State University, San Jose, CA, United States

***Abstract*— Identifying fake job postings and real job postings on the web is a big challenge. This is a binary classification problem which can be solved using Machine Learning. In this project, the class in the normal state is the real job postings when compared to the fake ones which fall under the abnormal state. The most popular machine learning algorithms that can be used for binary classification include Logistic Regression, k-Nearest Neighbors, Decision Trees, Support Vector Machine, Naive Bayes, etc. We shall design, develop and implement the models best suited for this classification and find the best model based on a series of machine learning metrics. When solved it can help innocent people who are preyed upon by avoiding scams. With the help of this application, one can be educated and made aware of a possible fraud preying on personal information and resources, making society a safer and secure place.**

***Index Terms*— Fraudulent, NLP, jobs**

I. Introduction

As technology advances, combined with the COVID-19 pandemic period, almost everything has been moved to the virtual world such as meetings, school, shopping, etc. Things can seem more convenient than they may sound, there is one major disadvantage, the rise of online scammers. Online scamming has been a huge problem ever since the internet was born. Now as more people are going online, it will be easier for scammers to deceive people and make money out of them. One popular scam that is common nowadays is fake job postings. Applicants who fall for this online fraud will not only waste their time, but may also waste their money for an application fee, or more dangerously, lose personal information such as Social Numbers. That is why we are proposing an application that utilizes natural language processing and machine learning models to distinguish fake job postings from real job postings on recruiting portals. Since we are focusing on machine learning, there will be no deep learning or neural network involved. Hence, in the project, we will be using four major machine learning algorithms NLP, Naïve Bayes, SGD Classifier, and SVM to rule out deceiving postings. After that, we will compare the accuracy and performance of each model using the F1 score. Unlike most projects on Kaggle, we overcome one main challenge for the unbalanced dataset, which is making it balanced. In the original dataset, 95% of the job postings are real and only 5% of the postings are fake [10]. This could cause over-fitting in most models. Hence, to solve this problem, we are incorporating the Kaggle dataset with a dataset on GitHub that contains only fake job postings. This will make the dataset more balanced. In addition, our distribution of observations will be more even compared to projects that use data straight from Kaggle. The new dataset will have half fake and half real job postings.

II. Significance to the real world

With technological advancement, almost everything has been moved to the virtual world, however, which gave rise to online scammers. Online scamming has been a huge problem ever since the internet was born. One popular scam that is common nowadays is fake job postings. Scammers try to extract money from the job applicant in different online ways. Applicants who fall for this online fraud will not only waste their time, but may also waste their money for an application fee, or more dangerously, lose personal information such as Social Numbers. The solution to this problem is the hour of the need to detect fake job postings on recruiting portals.

III. Literature Review

For online scamming issues, there has been a lot of research already done. Many solutions exist to detect or classify fake against the real online postings. This section is used to understand the existing work done. We have highlighted a few of the research papers below to give the overall picture of online scamming issues.

The authors [1] proposed a paper "Fake Job Recruitment Detection Using Machine Learning Approach" where they used many different classifiers to detect the fraudulent jobs available across the web. The authors approached this scam detection problem by extracting a dataset from Kaggle and applying a different machine learning algorithm to the dataset to predict which algorithm works best to classify the job post as a scam or not a scam. According to the authors, the Random Forest classifier outperformed all other classifiers

Another paper [2] stated that detecting online recruitment fraud has become a topic of research in the job market as many scammers are taking advantage by posting scam jobs on social media for the past few years. The authors used many machine learning algorithms to classify the posts as scams. Out of all the models they used, they concluded that LightGBM and Gradient Boosting gave the highest accuracy.

Authors [3] of "Fake News Detection on Social Media: A Data Mining Perspective" have tried to detect fake news on Social media. According to the authors, the veracity of fake news can be identified by looking at many characteristics including malicious accounts, source of news, and sentiments attached to news. This also depends on the formation of users who think alike and form groups in social media where they tend to follow the similar news of the others in that group,

which results in an echo chamber effect. They concluded that instead of traditional news media, social media has become a more popular and preferred way to consume news.

The authors [4] of the paper "A Smart System for Fake News Detection using Machine Learning" have tried to aggregate the news using different news sites like google news, Feedly, news360, and others. The authors tried to detect fake online news with the help of Machine learning using Naive Bayes, Support Vector Machine, and natural language processing. They compared the results of their models with the existing models. They were able to achieve a great accuracy of 93.5%.

IV. Proposed Approach

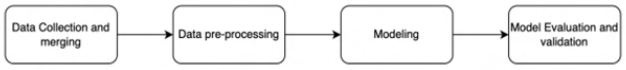

Fig. 1: This figure shows the steps in our experiment design.

The initial step is to collect and merge all the related data for job postings which shall include various attributes like Job ID, Job Description, Salary, Location, Position, etc. In the next step, the data will be processed and made available for feeding into the Machine Learning models. Then we design and develop the models suited for predicting fake jobs and real jobs. In the final step, we conduct model evaluation and validation by using the Machine Learning Evaluation techniques like F1 score and accuracy.

V. Project Development Methodology

A. Data Pre-processing

We have one dataset available for this project, but the data is imbalanced since the ratio of real jobs is more than the fake jobs. Here methods like oversampling can be used to balance our data so that the model does not fit only the legit jobs.The dataset has fields like job description, title, etc. which are long texts, these will have to be processed since the goal is to speed up the data transformation process and enhance the model accuracy. We are going to use the following NLP techniques for text processing:

- Tokenization: In this process, we split the data into smaller units, words.
- Stopword removal: Stopwords refer to common words which do not add any information. In NLP and text mining techniques these need to be eliminated.
- Lemmatization: This is a method that is responsible for grouping different inflected forms of words into their root form which has the same meaning.
- Count vectorization/ One-Hot Encoding: Since all the features are supposed to have numerical values we are going to use this method since it allows us to represent the text data in a numerical format.

The next step is to balance the dataset using the SMOTE technique i.e Synthetic Minority Oversampling Technique. Since the ratio of the fake job postings is less than the real job postings with the help of this technique the data will be oversampled and hence help in avoiding overfitting.

B. Algorithm

The next step is to identify features like Job title, salary, qualification, text, etc., and then divide the data into test and training sets. We will be using the following algorithms for the fake job prediction.

- Naïve Bayes: This is a classification algorithm based on the Bayes Theorem and is easy and fast in predicting the class of the given dataset. Regression: This model produces a continuous variable but we are looking at binary classification, hence we will use a link function that can be used to convert the continuous result into a probability of class assignment.
- Random forest: This classifier uses several decision trees on subsets of the dataset and takes the average to improve the accuracy of the prediction. This algorithm avoids the overfitting problem through its approach.
- SVM: The objective of this algorithm is to find a hyper-plane that classifies the given data points distinctly. This algorithm is widely used due to its significant accuracy and need for less computational effort.
- SGD Classifier: This classification algorithm is considered to have an efficient approach to fit linear classifiers and regressors under the convex loss functions.

We shall apply these models to the text and numerical data separately since we have both numeric and text features, and then combine the results. The models will be compared on the accuracy, F1 score, and ROC curve to arrive at the best model for the problem statement.

C. Evaluation methods

We are going to use the following methods for evaluating and comparing our models:

1) Confusion matrix: It is a matrix representation of the actual vs. predicted values which consist of True Positives, False Positives, True Negatives, and False Negatives (Fig. 2).

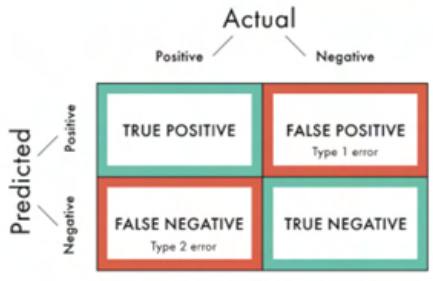

Fig. 2: This figure shows the Confusion matrix.

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Fig. 3: This figure shows F1 Score calculation.

2) F1 Score: Following is the formula to calculate the F1 Score. Here recall is the positive predictions divided by the positive class values i.e the number of true positives and false negatives in the test dataset. Precision is the positive predictions that belong to the positive class.

3) ROC Curve: Receiver Operating Characteristic The curve is a graphical illustration of the classifier's ability. It is a plot of true positive rate v.s false positive rate at various discrimination thresholds. This analysis allows us to select the optimal model and discard the sub-optimal ones. The analysis provides tools to select possibly optimal models and discard sub-optimal ones.

VI. Data Engineering

A. Dataset Description

The dataset that we are using for this project is sourced from Kaggle [5], an online platform to search and publish data. The dataset contains the information of approximately 18000 jobs. Around 800 of the total jobs are fake/fraudulent and the rest are legit. There is one CSV file in the raw dataset that contains the below columns/features:

- Job_id: Unique Job id created by the hiring organization
- Title: Title of the job
- Location: Location where the hired individual will be located.
- Department: Department for which the organization is hiring
- Salary Range: Indicative salary range for the hiring position
- Company Profile: A brief description of the hiring organization
- Requirements: A brief description of the job requirements
- Benefits: Benefits offered by the organization after joining
- Telecommuting: Indicates if the required job opening offers to telecommute or not. The value is 1 for telecommuting jobs and 0 for office jobs.
- Has Company Logo: Indicates whether the hiring organization has a logo available. It has value 1 if the logo is present, else 1.
- Has Questions: Screening Questions for the job. The value is 1 if the screening questions are present and 0 if not.
- Employment type: Whether the organization is offering a full-time, part-time, contract-based job, etc.
- Required Experience: Relevant experience that is required for the hiring position.
- Required Education: Minimum education that is needed for the job.
- Industry: Type of industry of the hiring organization.
- Function: Type of function for which the organization is hiring.
- Fraudulent: This is the target variable. It reflects whether the job is fraudulent or not. Its value is 0 if the job is legit and 1 if the job is fraudulent.

B. Data collection and cleaning

We started data cleaning by checking for the data types of all the columns. All the data types seem to be pretty much aligned, so we don't need any modifications here. Further, we checked for the missing values in the dataset. We had a lot of missing values in the dataset, but we planned to replace the missing value with the keyword 'missing', instead of dropping or replacing it with something else since there might be a connection between missing data and the fraudulent jobs.

We also noticed that the location column included country, state, and city separated by commas. So, we created two new columns 'country' and 'city' from the location column.

We then created two new columns to separate minimum and maximum salary from the salary range column. The new columns are minimum salary and maximum salary.

VII. Exploratory Data Analysis

In this part, we will be talking about exploratory data analysis. This section will cover the various visualizations of the dataset which will show the characteristics of the dataset. As previously mentioned one of the challenges we had with our dataset is that the dataset was not balanced at all.

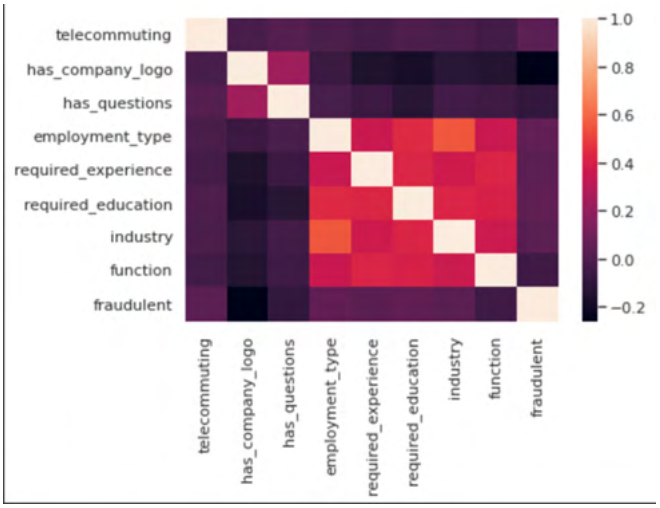

Fig. 4: This figure shows the heatmap of raw dataset.

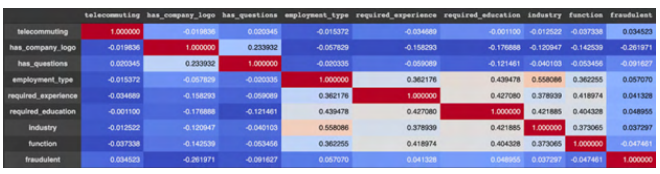

Fig. 5: This figure shows the correlation between various features.

The above images show the correlation between the different features in our dataset. Specifically, we used a heatmap

to show that the scale is from zero to one, with zero representing no correlation between the features, and one representing a direct correlation. Looking at the figure above it is very clear that there is a strong correlation between the features of employment type, required education, required experience, industry, and experience. This goes to show that these features are the ones that are most important when creating our model.

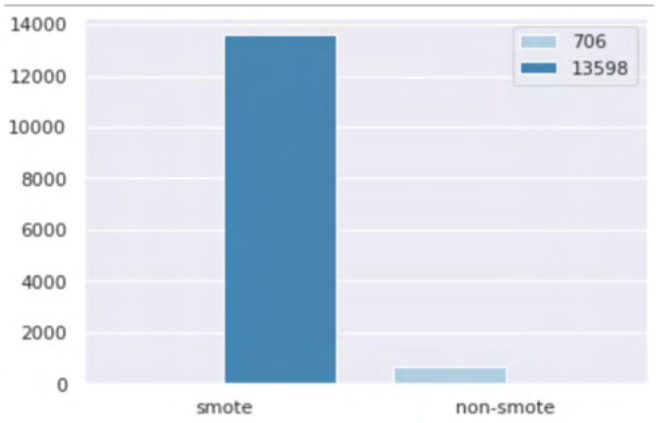

Fig. 6: This figure shows fake jobs after and before SMOTE.

The image above shows how much data the dataset had for fraudulent data before smote was applied and after smote was applied. As shown in the figure above, before conducting smote on the dataset we only had 706 fake job postings to analyze and train our model with. To have more accurate results for our model we have decided to use the SMOTE method to get more data to train the model with. Using the SMOTE method we were able to increase the number of fraudulent data points from 706 to 13,598.

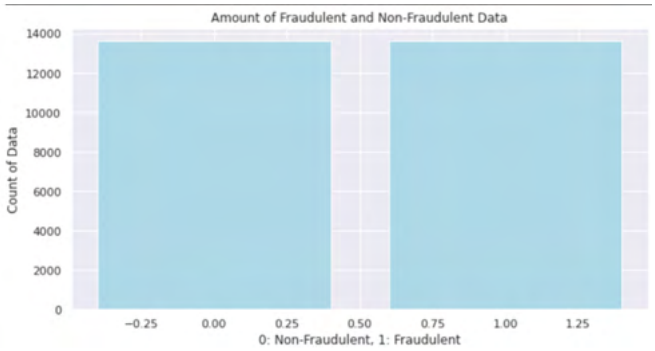

Fig. 7: This figure shows the fraudulent and non-fraudulent data after applying SMOTE.

The figure above shows the amount of fraudulent data, and non-fraudulent data that exists within the dataset. As you can see from the visualization above you can see that the dataset is balanced equally, with the same amount of data for both fraudulent and non-fraudulent data. This is what we wanted to accomplish using SMOTE.

From the figures above you can see that there is no change in the data size for non-fraudulent data. This is exactly what we want as we do not want this data to increase as well. This would not make sense as the dataset would still be unbalanced. This is why the SMOTE method was only applied to the fraudulent data from the dataset.

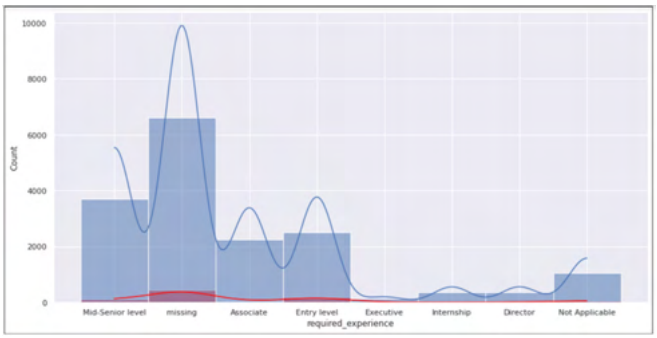

Fig. 8: This figure shows the fake vs real job categories on the raw dataset.

The figure above shows a visualization based on the original dataset before smote was applied. Red represents the fraudulent data, and blue represents the non-fraudulent data. The graph shows the count or the density of the relationship between the required experience and how many of those jobs are real or fake. Based on the visualization above, we can see that most of the job postings seem to lack information on the amount of experience required. One thing we did notice though was that entry-level jobs seemed to have the most fraudulent data although it is not the second-highest amount of data collected for that category.

VIII. Model Development

The main aim of the project is to classify online job postings as fraudulent or non-fraudulent. This is a binary classification problem where we have two discrete values, one for fake jobs and the other for non-fake jobs. There are many machine learning models for solving classification problems. We are using Random Forest, Naive Bayes, kNN, SVM, and Stochastic Gradient Descent algorithms to detect fake jobs vs. real jobs.

We have divided our dataset into train and test datasets. The train set will be used to train the required Models and the test set will be used to test if the models are able to classify the job postings accurately.

A. Naive Bayes

Naive Bayes is an algorithm which comes under supervised machine learning. The basis of this model is Bayes theorem. It calculates the probability given the prior probabilities for the features.

In our project, we are using Natural Language Processing, so we needed to implement the Multinomial method of Naive Bayes algorithm because this technique works great when we have text in the dataset.

Multinomial Naive Bayes: For this method, we used the sklearn package called 'MultinomialNB'. This method works on feature vectors where each value on this vector is represented by the number of occurrences of a term. We have created the features vector of words using TF-IDF technique.

Below figure has k different values with pk probability for n number of elements. The formula is shown below in the figure.[16]

$$P(X_1 = x_1 \cap X_2 = x_2 \cap \ldots \cap X_k = x_k) = \frac{n!}{\prod_i x_i!} \prod_i {p_i}^{x_i}$$

Fig. 9: This figure shows the formula used in Naive Bayes.

There could be a zero probability problem with this algorithm. So, to solve it, there is a technique called Laplace smoothing. In this technique, we uses a smoothing parameter that is called alpha. We have implemented alpha=1 for our project. The value of 1 is preferred to handle the problem of probability of zero in Naive Bayes model.

B. Random Forest

Random Forest, or we can call it random decision forests, is one of the most popular supervised machine learning algorithms applicable in both classification and regression problems. It is said to perform better in classification problems.[9] We can kind of get the gist of Random Forest in simple terms which is the collection of many decision trees. However, it is important to distinguish the differences in their behaviors. First of all, overfitting is a common problem for decision trees. However, Random Forest was able to overcome this problem by outputting the majority ranking or average, Second, since Random Forest consists of many different Decision Trees, it is expected to operate much slower than one Decision Tree alone.[9] In this project, we used the Scikit-learn library's Random Forest implementation. A couple of things to understand when using this version of Random Forest. It uses Gini Importance to calculate each node's importance in each Decision Tree. Feature importance is an important concept to understand for Random Forest. "Feature importance is calculated as the decrease in node impurity weighted by the probability of reaching that node." [11] The mathematics behind the two concepts. First, assuming there are two children nodes, Random Forest then calculates the node's importance.

$$ni_j = w_j C_j - w_{left(j)} C_{left(j)} - w_{right(j)} C_{right(j)}$$

Fig. 10: This figure calculates the Random Forest Node's importance.[11]

$$fi_i = \frac{\sum_{j:node\ j\ splits\ on\ feature\ i} ni_j}{\sum_{k \in all\ nodes} ni_k}$$

Fig. 11: This figure calculates the feature importance of each tree.[11]

We performed hyper-parameter tuning that would increase the predictive power:

- n_estimators: number of trees the algorithm builds before averaging the predictions.
- Individual entries are indicated with a black dot, a so-called bullet.
- mini_sample_leaf – determines the minimum number of leaves required to split an internal node. [11]

We utilized the RandomSearchCV function to find the optimal hyperparameters for each run.

```
dation =  0.9878165453878981
{'n_estimators': 17, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': 50}
```

Fig. 12: This figure Random Forest hyperparameters for each run.

C. kNN- k Nearest Neighbor

K Nearest Neighbor is one of the most widely used Supervised algorithms. kNN finds the similarity score of all the points using distance functions such as Euclidean distance, Hamming distance, Minkowski distance, etc. kNN algorithm starts by finding the suitable number of neighbors i.e k. It then calculates the distance between the new neighbor and the 'k' closest neighbor. The new neighbors are categorized based on the maximum number of neighbors. We have implemented kNN using the scikit-learn package 'KNeighborsClassifier' [6]. We trained the model after doing the initial data preprocessing and oversampling. We tried to change a few parameters such as nearest neighbor and distance metric. Minkowski distance and 3 nearest neighbors using kNN provide an accuracy of 94The Minkowski distance between two points x and y is calculated as below:

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

Fig. 13: This figure shows the formula to calculate Minkowski Distance.

Here, i is the ith element for each feature vector, and p is an integer. In the cases when p is equal to 1, the distance becomes Manhattan distance, and when p=2, the distance becomes Euclidean distance. We also tried the kNN model with Manhattan and Euclidean distance but got lesser accuracy.

D. SVM

We started off researching the SVM model to see if it would be a good fit for our project. After reading about it we felt confident that this would be one of the best models to create as it is good for classification problems but can also detect outliers. After starting to build the SVM model to predict fake versus real job postings using the test data that was split. Initially, we started by creating the SVM model using an RBF which stands for radial basis function kernel,

which is one of the types of kernels used to create the SVM model. We got an accuracy rate of 86%, and although this is somewhat good we were not satisfied with the accuracy. So we started playing around with the different kernels that could be used within the SVM model, which include the Gaussian, Laplace, Sigmoid, and Poly kernels. After trying all these different kernels, we have realized that the kernel that performs the best is the poly kernel. Once we have settled on using this kernel, the next thing that we wanted to mess with was the degree that affects the accuracy directly as well. We were able to land on the best-performing degree by using the trial and error method, which happens to be eight point seven. This degree in combination with the poly kernel yields an accuracy rate of 93%. Compared to some of the other models we have tried and what was found online, we were able to yield a better accuracy using the SVM model alone in conjunction with our cleaned dataset.

E. Stochastic Gradient Descent

SGD or Stochastic Gradient Descent has been successfully used in machine learning problems that are faced when natural language processing and text classification are involved. Ease of implementation and efficiency are some of the major advantages of the algorithm which led us to choose this model. We had a lot of opportunities for code tuning using this technique.

The maximum number of iterations was chosen to be 1000. First, we used the hinge loss with the l2 penalty and then with the l1 penalty which resulted in the same accuracy. Then we used the log loss with an l2 penalty which resulted in less accuracy i.e 82% when compared to the former. Hence, the former selection of hinge loss with an l2 penalty was considered for better accuracy of 87%.

IX. Evaluation

As our problem is a binary classification problem, so, for evaluating and comparing all the models we used, we have used the Confusion matrix, F1 score, Recall, and AUC - ROC curve.

- Confusion matrix : It is used to check how well the classification model is performing in terms of the matrix representation of True Positives, False Positives, True Negatives, and False Negatives.
- Recall: It tells how many were the correct positive predictions out of all positive predictions.
  Recall = TP/(TP + FN)
  F1 = 2*((precision*recall)/(precision+recall))
- AUC - ROC curve: This metric is used to measure the performance at different threshold settings for the classification problems. ROC is the probability curve.
- ROC: This curve represents the probability and shows the relationship between sensitivity and specificity graphically. The curves covering the top-left area are the best performance.

Model Name	F1 Score	AUC (Proba)	Recall	Run Time
KNN	0.53	0.9	0.78	2.85 seconds
Random Forest	0.7	0.96	0.63	8.58 seconds
Naive Bayes	0.31	0.87	0.54	0.31 seconds
SVM	0.42	0.92	0.81	399.4 seconds
SGD Classifier	0.34	0.91	0.81	1.05 seconds

Fig. 14: Figure shows F1 score, AUC, Recall and Run Time of Models

A. Multinomial Naive Bayes

The Confusion matrix (Fig. 15) classified 3094 True Negatives, which means 3094 jobs that were classified as non-fraudulent are accurately predicted, and 87 True Positives, which means 87 jobs that were fraudulent were classified correctly by kNN. There are 322 False Positives and 73 False Negatives.

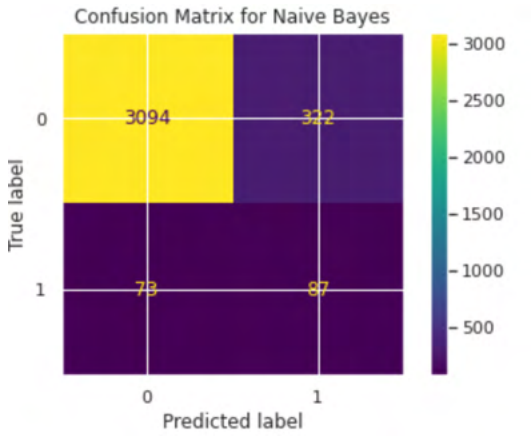

Fig. 15: Confusion Matrix for Multinomial Naive Bayes

B. Random Forest

The Confusion matrix (Fig. 16) classified 3388 True Negatives, which means 3388 jobs that were not classified as fraudulent are accurately predicted. There are 101 True Positives, which means 101 fraudulent jobs were classified correctly by Random Forest. These are the two most desired cases out of the four.

C. KNN

The Confusion matrix (Fig. 17) classified 3231 True Negatives, which means 3231 jobs that were classified as non-fraudulent are accurately predicted, and 125 True Positives, which means 125 fraudulent jobs were classified correctly by kNN. There are 185 False Positives and 35 False Negatives.

D. SGD Classifier

The Confusion matrix (Fig. 18) classified 2945 True Negatives, which means 2945 jobs that were classified as non-fraudulent are accurately predicted, and 130 True Positives, which means 130 fraudulent jobs were classified correctly by SGD. There are 471 False Positives and 30 False Negatives.

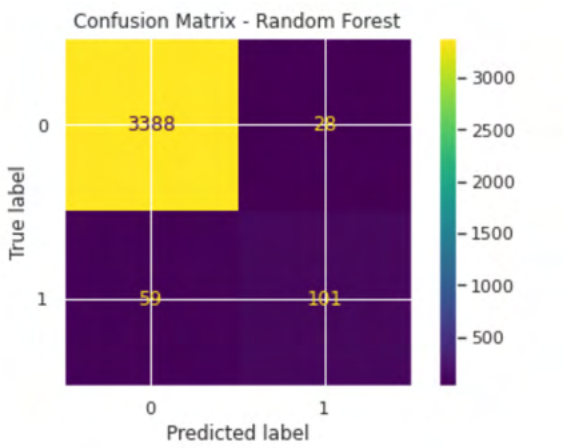

Fig. 16: Confusion Matrix for Random Forest

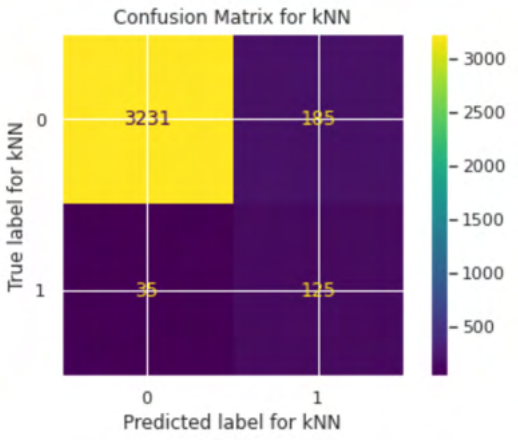

Fig. 17: Confusion Matrix for KNN

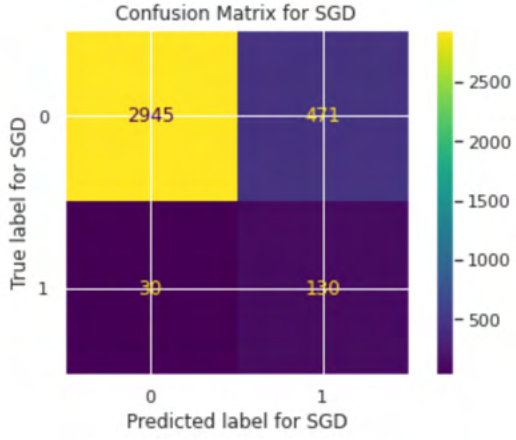

Fig. 18: Confusion Matrix for SGD Classifier

E. SVM

The Confusion matrix (Fig. 19) classified 3062 True Negatives, which means 3062 jobs that were classified as non-fraudulent are accurately predicted, and 130 True Positives, which means 130 fraudulent jobs were classified correctly by SGD. There are 354 False Positives and 30 False Negatives.

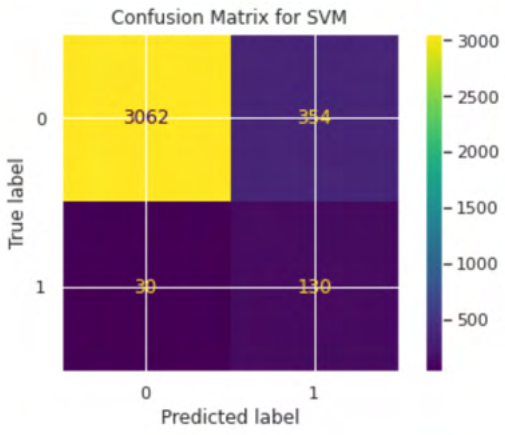

Fig. 19: Confusion Matrix for SVM

X. Result

We evaluated the five models used in the project based on Accuracy, F1 score, AUC, and Recall and did a comparison to see which model performs better than the other models. Below is the comparison chart of the models:

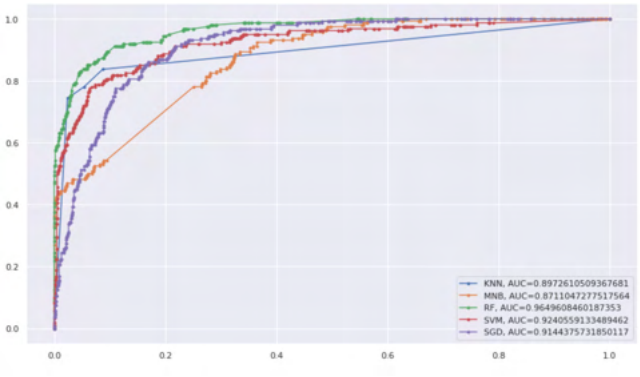

Fig. 20: Figure shows the ROC curve for all the Models

Model Name	Accuracy
KNN	94%
Random Forest	98%
Naive Bayes	89%
SVM	90%
SGD Classifier	86%

Fig. 21: Comparison of Models

XI. Future Work

We have been wondering how we could make this project better. One of the biggest challenges we faced during this project is that we did not have enough data on fake job listings. Something that would be really cool to try in the future is to collect data on fake job postings to better the quality of our dataset. This would be great as we will be able to compare our results with the real data as we used the SMOTE and GLOVE methods to balance the dataset.

We would be able to answer questions such as whether the accuracy rate will be going higher or lower, and how much of a change it makes. Hence refine our models using extensive data collection and provide all job search sites with the capability to filter fraudulent job posting in turn helping the society at large.

REFERENCES

[1] "1.5. stochastic gradient descent," scikit. [Online]. Available: https://scikit-learn.org/stable/modules/sgd.html. [Accessed: 09-May-2022].

[2] "1.6. nearest neighbors," scikit. [Online]. Available: https://scikit-learn.org/stable/modules/neighbors.html#classification. [Accessed: 09-May-2022].

[3] C. Prashanth, D. Chandrasekaran, B. Pandian, K. Duraipandian, T. Chen, and M. Sathiyanarayanan, "Reveal: Online fake job advert detection application using machine learning," 2022 IEEE Delhi Section Conference (DELCON), 2022.

[4] H. Tabassum, G. Ghosh, A. Atika, and A. Chakrabarty, "Detecting online recruitment fraud using machine learning," 2021 9th International Conference on Information and Communication Technology (ICoICT), 2021.

[5] I. M. Nasser, A. H. Alzaanin, and A. Y. Maghari, "Online recruitment fraud detection using ann," 2021 Palestinian International Conference on Information and Communication Technology (PICICT), 2021.

[6] J. Brownlee, "Smote for imbalanced classification with python," Machine Learning Mastery, 16-Mar-2021. [Online]. Available: https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/. [Accessed: 09-May-2022].

[7] J. Pennington, "GloVe: Global Vectors for Word Representation," Glove: Global vectors for word representation. [Online]. Available: https://nlp.stanford.edu/projects/glove/#:~:textGloVe%20is%20an%20unsupervised%20learning,Getting%20started%20(Code%20download). [Accessed: 09-May-2022].

[8] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media," ACM SIGKDD Explorations Newsletter, vol. 19, no. 1, pp. 22–36, 2017.

[9] N. A. Patel and R. Patel, "A survey on fake review detection using Machine Learning Techniques," 2018 4th International Conference on Computing Communication and Automation (ICCCA), 2018.

[10] O. Nindyati and I. G. Bagus Baskara Nugraha, "Detecting scam in online job vacancy using behavioral features extraction," 2019 International Conference on ICT for Smart Society (ICISS), 2019.

[11] P. E. S. -, By, -, and P. E. Staff, "Implementing 3 naive Bayes classifiers in scikit-learn," Packt Hub, 07-May-2018. [Online]. Available: https://hub.packtpub.com/implementing-3-naive-bayes-classifiers-in-scikit-learn/. [Accessed: 09-May-2022].

[12] "Random Forest: Introduction to random forest algorithm," Analytics Vidhya, 24-Jun-2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/. [Accessed: 09-May-2022].

[13] S. Bansal, "Real / fake job posting prediction," Kaggle, 29-Feb-2020. [Online]. Available: https://www.kaggle.com/datasets/shivamb/real-or-fake-fake-jobposting-prediction. [Accessed: 09-May-2022].

[14] S. Dutta and S. K. Bandyopadhyay, "Fake job recruitment detection using machine learning approach," International Journal of Engineering Trends and Technology, vol. 68, no. 4, pp. 48–53, 2020.

[15] S. Ronaghan, "The mathematics of decision trees, random forest and feature importance in Scikit-learn and Spark," Medium, 01-Nov-2019. [Online]. Available: https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3. [Accessed: 09-May-2022].

[16] S. T. S. Sesha, "'job-posting prediction using naive Bayes classifier,'" Medium, 01-May-2022. [Online]. Available: https://medium.com/@craxtarun/job-posting-prediction-using-naive-bayes-classifier-9101af534624. [Accessed: 09-May-2022].

XII. APPENDIX

A. Visualization

We started with Exploratory Data Analysis in our dataset to perform some visualizations to get a better understanding of the data. In the EDA, we plotted data to visualize fraudulent vs non-fraudulent jobs, balanced vs unbalanced data, etc. We also plotted the heat-map to check for any correlations in the data. The other visualizations that we have are from the results of various models used in this project such as confusion matrix, ROC Curve, etc.

B. Saving the Model for Demo

We used Python's pickle package to save our model. This will be shared with the code.

C. Report

All the team members of the group contributed equally to the project. All the project work including code, slides, and documentation is our original work.

D. Version Control

The main problem in projects like this would be updating the changes from the slightest to bigger ones for the entire team. If everyone is working on the same file at the same time, it would be very difficult to know who is changing what. That is why we are using google colaborar as the main tool. It allows us to work on the project within one place, instead of having to download and re-upload the file and share it with everyone each time we make any changes. Moreover, we utilize Github for version control of the project. We would occasionally commit and push changes with concise descriptions for team members to see.

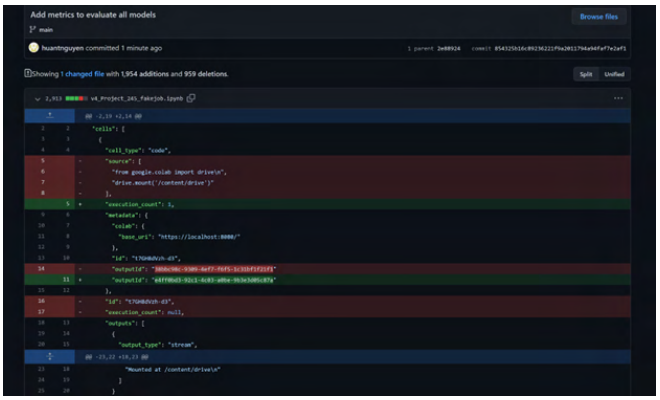

Fig. 22: This figure shows the GitHub snapshot of version control.

E. Lesson Learned

After doing this project we have learned the importance of conducting various models and tweaking the models' features to perfect them. The first model that is made is not always going to be the best model. Furthermore, more than the model, the first and most important thing to do is to make sure that the data is clean and ready to be put into the models. Without good data that is balanced, it is hard to have models that are correct and accurate.

It is also important to perform oversampling or under sampling to have a balanced dataset to avoid over-fitting. For text data, the NLP techniques are the most crucial. The GloVe or other features vector creation techniques are required when dealing with text classification.

F. Team Work

With the help of Agile methodology and project management plan, we were able to achieve the desired results with everyone contributing equally. With the help of practicing pair programming, the process of coding and discovering bugs was much smoother. Moreover, with version control from Github, every member of the team was able to stay up to date with the latest version and commit to the project.

G. Used Laxtex

We used Latex to prepare and write up our project in the IEEE format. We also used the overleaf online Latex editor to put everything together.

H. Used Grammarly

We used Grammarly to check for all grammar errors in our project content. We put together all the writing in a Google Doc, then run Grammarly to fix all the suggested errors. We had a few premium suggestions that were not needed to change. Finally, we moved all the content from Google Docs to the overleaf editor.

I. Practice Pair Programming

We have been practicing pair programming throughout this project. Since we had two people working on data cleaning and data transformation, applying the process of pair programming was much easier. We often have two pair programming at a time. The person doing data cleaning with another member of the team, and the person doing data transformation with another member. What we would do is that because each of us had to build our chosen machine learning model, we would pair above like above and let the person who is building the model be the driver (who writes the code) with the other person reviewing the code. After that, the two would switch roles. We continued doing it with different pairs until we successfully ran all the code.

J. Evaluation of Performance

Summarizing the evaluation of performance, since our machine learning problem is a binary classification one, the metrics we used include F1 score, Recall, Confusion matrix, AUC, and ROC curves. All of the above metrics are calculated for every model combined with the accuracy for comparison.

K. Innovation

Most of the work done on real job prediction is using different machine learning algorithms and then just comparing them to find the best-outperformed model. In our project, we tried to do something different apart from the work which has been already done on this dataset.Below are some of the key highlights.

- As a part of innovation, we have used 'Max Voting' of Ensemble Technique to improve the results of machine learning algorithms and give the overall best performance model. It combines the predictive power of all the base classifier algorithms for each data point. The final prediction is the maximum score or the predictions of the majority of models after comparing all of them is considered the final prediction.
- We have used SMOTE i.e. Synthetic Minority Oversampling Technique that was taught in our course work to balance the unbalanced dataset.
- We have used GloVe for Vector representation for words in all our algorithms except for Naive Bayes. For Multinomial Naive Bayes, we used the Count vectorizer technique for the same purpose as it cannot work with negative values generated by GloVe.

L. Velocity

We have collected the dataset from Kaggle to analyze the fake vs real job detection. The data we are using were gathered at a time and is not live data that is being streamed. but in the future, it can be streamed from different job websites to extract the live job postings for huge data volumes.

M. Technical Difficulties

There are many difficulties we faced while working on this Project.

- One of the biggest problems was to deal with data that was very unbalanced. It had almost 16,000 data points for jobs that are real, and then about a thousand job postings that are fake. We used the SMOTE technique to solve this.
- For text-based data we used NLP techniques like Tokenization, Stop Word Removal and Lemmatization to deal with text data.
- For creating the vector of words, we tried using GloVe technique for all algorithms but it did not work for MultinomialNB as GloVe uses negative values. As this does not work with Multinomial Naive Bayes, we used Countvectorizer to solve this issue.

N. Practiced Agile Methodology

We practiced the Agile methodology and used ClickUp for our project involving constant collaboration and continuous improvement and had regular scrum meetings to monitor the progress and discuss any issues or backlogs.

We broke down the project activities into Business Understanding, Data Engineering, Model Design and development, Evaluation and validation, Deployment, and documentation and used a Gantt Chart which depicts the work plan and helps in logging and monitoring progress.

O. Used Creative presentation techniques

We have used Prezi as our presentation tool since it uses motion, zoom, and spatial relationships and helps in bringing ideas to life offering a better presentation.

P. Literature Survey

We did a lot of Literature surveys to understand the overall work done related to online scamming. We referred to around 10 papers as a part of the literature review and tried to understand how the different authors have come up with machine learning solutions to this major problem of detecting the fake vs real online postings in all the areas. We then tried to implement our approach to depict the fake job postings online.