Department of Applied Statistics
MAKAUT, WB

Unmasking Real and Fraudulent Job Classifier Model by Machine Learning and NLP

A project report submitted
for partial fulfillment of the requirements for the award of the degree of Master

by

Swadhin Maity
Moumita Roy
Tanmoy Manna

Registration number:223001810293
Registration number: 223001810287
Registration number: 223001810296

M.Sc. in Applied Statistics and Analytics

Under the supervision of

Mr. Chandan Chakraborty

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, WEST BENGAL

Department of Applied Statistics
MAKAUT, WB

CERTIFICATE

This is to certify that the dissertation report entitled 'Unmasking Real and Fraudulent Job Classifier Model by Machine Learning and NLP', submitted by Moumita Roy (Reg. No:223001810287, Roll No:30018022013), Swadhin Maity(Reg. No: 223001810293, Roll No: 30018022019), Tanmoy Manna(Reg. No: 223001810269, Roll No: 30018022022) to MAKAUT, WB, is a record of project work carried out by her under my supervision and guidance, and is worthy of consideration for the award of the degree of Master of Science in Applied Statistics and Analytics of the University.

Name

Co-Supervisor

Dept. of Applied Statistics

Name

Supervisor

Dept. of Applied Statistics

Ms. Anwesha Sengupta

HoD

Dept. Applied Statistics

Department of Applied Statistics
MAKAUT, WB

DECLARATION

I declare that, this project report has been composed by me and no part of this project report has formed the basis for the award of any Degree/Diploma or any other similar title to me.

Date:

Student's name:

Reg. No:

Dept. Applied Statistics

MAKAUT, WB

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to our respected teacher Mr Chandan Chakraborty, Dr Susovon Jana, Department of Applied Statistics and Analytics, for their continuous guidance and support in completing my project work. His continuous supervision, motivation and encouragement helped us a lot towards the explanation of result in understanding the literatures as available with us for the preparation of this project work. I would also like to extend our gratitude to our principle and all teachers of our department for their constant encouragement and inspiration in carrying out our project work. I am thankful to our friends, juniors and seniors for helping us directly or indirectly to complete this project work. Finally, words are not sufficient to express gratitude our family members for supporting us, without their encouragement and support, I would have not reached this stage.

Ms. Anwesha Sengupta

HoD

Dept. Applied Statistics

Student's Signature

ABSTRACT:

In the current job market, where employment opportunities are scarce, individuals seeking jobs often encounter fraudulent job postings and scams. These deceptive job listings not only waste the time and effort of job seekers but also pose significant risks to their personal information and financial security. To address this issue, machine learning models and natural language processing (NLP) techniques are utilized to predict the likelihood of a job posting being genuine or fraudulent.

The objective of this project is to compare five classification models, namely the Logistic Regression Model, Random Forest Model, Multinomial Naïve Bayes Model, SGD Classifier Model, and XGBoost, to identify an effective model that can accurately classify fake and real job postings. Several factors such as job description, location, salary range, and company information are considered in this classification process. The project aims to achieve high precision in distinguishing between legitimate and fraudulent job postings. By doing so, it strives to contribute to the creation of a safer and more trustworthy job market for job seekers worldwide.

1. INTRODUCTION:

According to the Federal Trade Commission, fraudulent business and job opportunities led to a loss of $86 million for Americans in the second quarter of 2022. During this period, approximately 21,600 incidents of scams related to business and job opportunities were reported, and around one-third of these cases resulted in financial losses.

The rise of employment-related scams has been an ongoing issue, particularly in 2020 when individuals who faced job loss due to the Covid pandemic became vulnerable to exploitation by criminals. This has led to a lot of frustration for both job seekers and legitimate employers who are looking to hire qualified candidates.

Fig.1: Some Fraudulent Job ads

In a world where fraudulent job postings pose a significant threat to job seekers, there is hope in leveraging the power of machine learning algorithms and sentiment analysis to detect and prevent such scams. This project aims to utilize machine learning techniques and natural language processing (NLP) to predict the authenticity of job postings, allowing for the identification of genuine opportunities and the prevention of harm caused by fake job listings.

By employing machine learning algorithms and sentiment analysis, this project seeks to analyze the textual data within job postings, including job descriptions, requirements, and company information, to determine the likelihood of a job posting being genuine or fake. The project will utilize the capabilities of NLP to extract meaningful insights and features from the text, enabling the development of a predictive model.

Overall, this project has the potential to make a significant impact on the job search process by providing job seekers with a safe and reliable platform to find legitimate job opportunities while also saving employers time and resources by preventing them from receiving a large volume of fake applications.

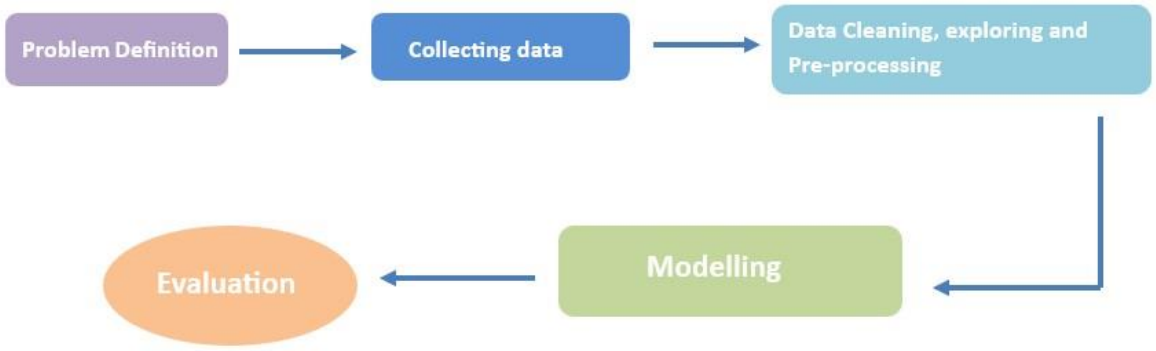
2. OBJECTIVE:

- Identify a classifier to distinguish between fake and genuine job postings.
- Can we find which categories are highly targeted by fraudulent jobs?
- Try to find final result assessed using two distinct models based on text and numeric features.
- Can we train One model on text data, while the other focuses on numeric data?
- Try to combine two models to produce the final output.

3. OVERVIEW OF THE PROJECT:

This project mainly follows five stages. The five steps adopted for this project are –

1. Problem Definition
2. Collecting data
3. Data cleaning, exploring, and pre-processing
4. Modelling
5. Evaluation



4. DATA COLLECTION:

Collecting data on Indian employment scams proved to be a significant challenge at the outset of the project. Limited availability of relevant data hindered our progress. However, our efforts led us to discover the EMSCAD dataset, which was manually collected by The University of the Aegean | Laboratory of Information & Communication Systems Security. This dataset encompasses approximately 18,000 job records.

However, a limitation of the EMSCAD dataset is that it was collected specifically between 2012 and 2014, which raises concerns about its relevance to the current employment scam landscape. Despite this flaw, we will leverage the available data to gain insights and develop a preliminary understanding of Indian employment scams.

This dataset contains 17,880 job descriptions out of which about 800 are fake. The data consists of both textual information and meta-information about the jobs. The dataset can be used to create classification models which can learn the job descriptions which are fraudulent.

The data is combination of integer, binary and textual datatypes. A brief definition of the variables is given below:

	Variable	Datatype	Description
1	job_id	Int	Identification number given to each job posting
2	title	Text	A name that describes the position or job
3	location	Text	Information about where the job is located
4	department	Text	Information about the department this job is offered by
5	salary_range	Text	Expected salary range
6	company_profile	Text	Information about the company
7	description	Text	A brief description about the position offered
8	requirements	Text	Pre-requisites to qualify for the job
9	benefits	Text	Benefits provided by the job
10	telecommuting	Boolean	Is work from home or remote work allowed
11	has_company_logo	Boolean	Does the job posting have a company logo
12	has_questions	Boolean	Does the job posting have any questions
13	employment_type	Text	5 categories – Full-time, part-time, contract, temporary and other
14	required_experience	Text	Can be – Internship, Entry Level, Associate, Mid-senior level, Director, Executive or Not Applicable
15	required_education	Text	Can be – Bachelor's degree, high school degree, unspecified, associate degree, master's degree, certification, some college coursework, professional, some high school coursework, vocational
16	Industry	Text	The industry the job posting is relevant to
17	Function	Text	The umbrella term to determining a job's functionality
18	Fraudulent	Boolean	The target variable à 0: Real, 1: Fake

5. DATA CLEANING AND EXPLORATION:

Our dataset contains four string columns (Title, Location, Department, Salary Range), four HTML Fragment columns (Company Profile, Description, Requirements, Benefits), four Binary columns (Telecommuting, Company logo, Questions, Fraudulent) and five Nominal columns (Employment type, Required experience, Required education, Industry, Function).

At first, we filled the null location places with “blank” and after that splitted the Location column into three different columns such as country, state and city which was separated by comma (,).

location	country	state	city
US, NY, New York	US	NY	New York
NZ, , Auckland	NZ		Auckland
US, IA, Wever	US	IA	Wever
US, DC, Washington	US	DC	Washington
US, FL, Fort Worth	US	FL	Fort Worth

Fig.2: splitting location into country, states and city

job_id	0
title	0
location	0
department	11547
salary_range	15012
company_profile	3308
description	1
requirements	2695
benefits	7210
telecommuting	0
has_company_logo	0
has_questions	0
employment_type	3471
required_experience	7050
required_education	8105
industry	4903
function	6455
fraudulent	0
country	0
state	440
city	440
dtype: int64	

Fig.3: NULL value list

When we checked the null count of the dataset there were 11547 null department, 15012 null salary_range, 3308 null company_profile, 1 null description, 2695 null requirements, 7210 null benefits, 3471 null employment_type,7050 null required_experience,8105 null required_education,4903 null industry,6455 function, 440 null states and 440 null cities.

Most of them were string so we decided to fill them with “blank” key word which will come in handy for text analysis.

5.1.1 State City Ratio:

A ratio is established to go one level deeper and take state inclusion into account. The fake-to-real job ratio shown below is based on states and cities. To determine how many fake jobs there are for every real job, apply the formula below:

$$\text{ratio} = \frac{state\ \&\ city|fradulent = 1}{state\ \&\ city|fradulent = 0}$$

The graph below only displays ratio values that are higher than or equal to one.

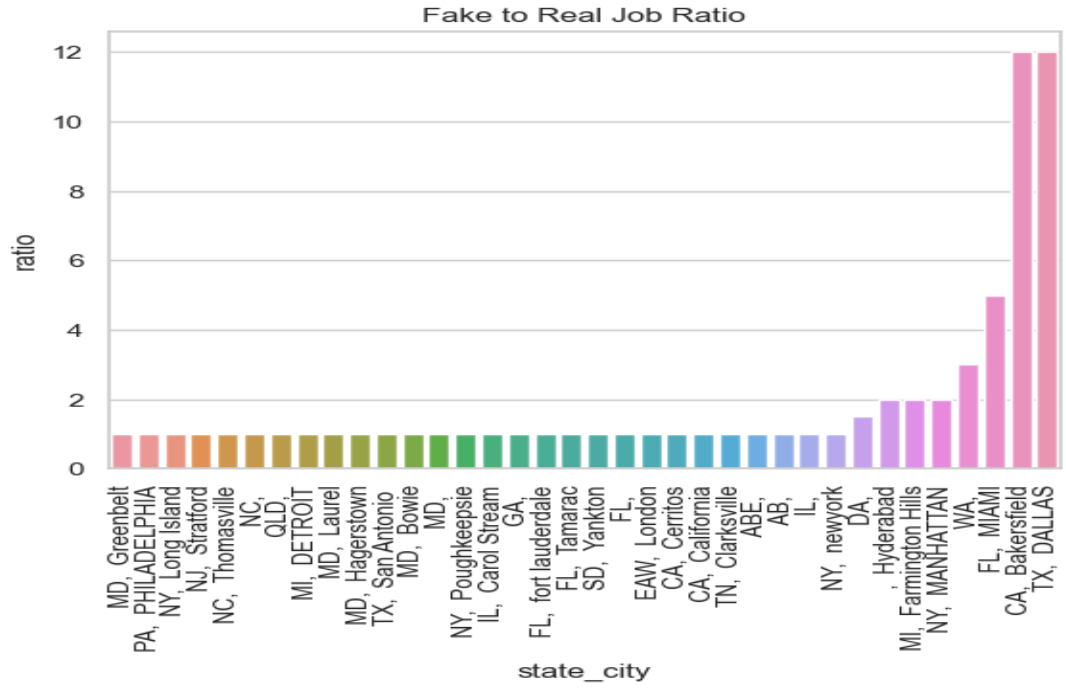


Fig.4 : Fake to real job ratio (ration ≥ 1) with state_city

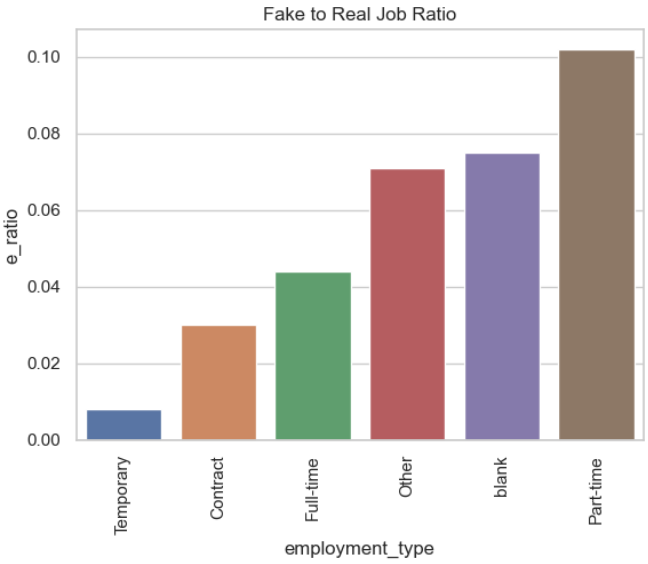
Dallas, Texas and Bakersfield, California has a phoney to actual job ratio of 12:1 approx. There is a strong likelihood that any job listings from these sources are fake.

5.1.2 Employment Type Ratio:

Similarly, an e_ratio is established based on the employment_type column. To determine how many fake jobs there are for every real job, apply the formula below:

$$\text{e\_ratio} = \frac{employment\_type|fradulent = 1}{employment\_type|fradulent = 0}$$

The graph below only displays ratio values:



	employment_type	e_ratio
0	Contract	0.030
1	Full-time	0.044
2	Other	0.071
3	Part-time	0.102
4	Temporary	0.008
5	blank	0.075

Fig.5: Bar graph of employment types ratio with employment types and ratio table

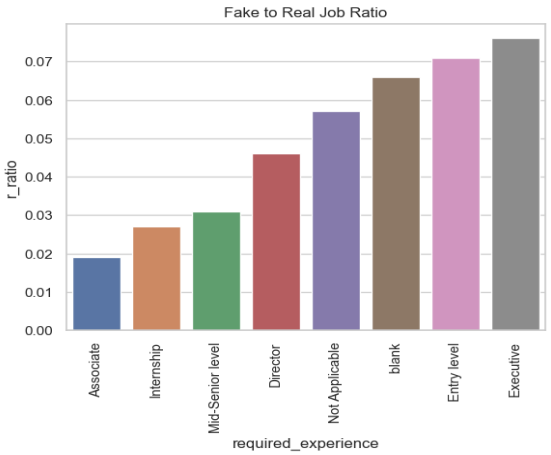
Part-time jobs and which employment_type were left blank has high phoney to actual job ratio. There is a strong likelihood that any job listings for these employment type are fake.

5.1.3 Required Experience Ratio:

Similarly, an r_ratio is established based on the required_experience column. To determine how many fake jobs there are for every real job, apply the formula below:

$$\text{r\_ratio} = \frac{\text{required\_experience}|fradulent = 1}{\text{required\_experience}|fradulent = 0}$$

The graph below only displays ratio values:



	required_experience	r_ratio
0	Associate	0.019
1	Director	0.046
2	Entry level	0.071
3	Executive	0.076
4	Internship	0.027
5	Mid-Senior level	0.031
6	Not Applicable	0.057
7	blank	0.066

Fig.6: Bar graph of required experience ratio with required experience and ratio table

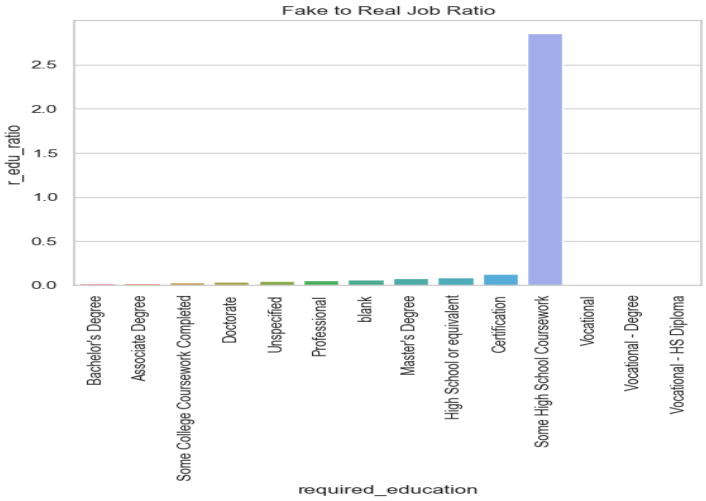
Executive and entry level has high phoney to actual job ratio. There is a strong likelihood that any job listings for this employment type are fake.

5.1.4 Required Education Ratio:

Similarly, an r_edu_ratio is established based on the required_education column. To determine how many fake jobs there are for every real job, apply the formula below:

$$\text{r\_edu\_ratio} = \frac{\text{required\_education}|fradulent = 1}{\text{required\_education}|fradulent = 0}$$

The graph below only displays ratio values:



	required_education	r_edu_ratio
0	Associate Degree	0.022
1	Bachelor's Degree	0.020
2	Certification	0.126
3	Doctorate	0.040
4	High School or equivalent	0.089
5	Master's Degree	0.081
6	Professional	0.057
7	Some College Coursework Completed	0.030
8	Some High School Coursework	2.857
9	Unspecified	0.046
10	Vocational	NaN
11	Vocational - Degree	NaN
12	Vocational - HS Diploma	NaN
13	blank	0.059

Fig.7: Bar graph of required education ratio with required education and ratio table

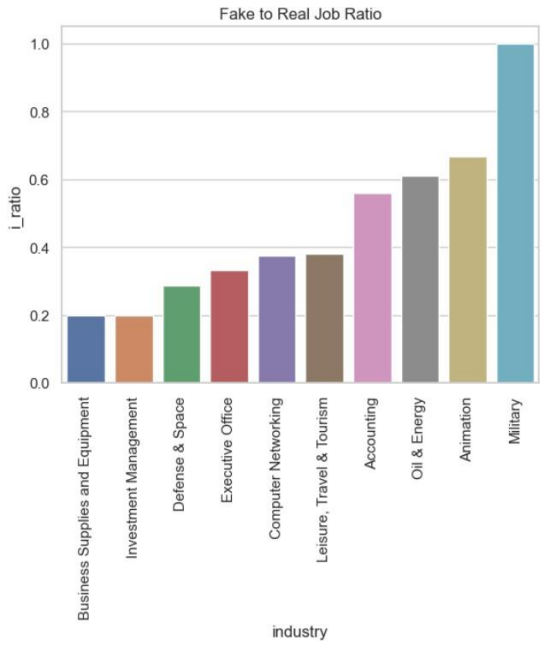
Some High School Coursework and Certification has high phoney to actual job ratio. There is a strong likelihood that any job listings for this employment type are fake. On the other side Vocational, Vocational-Degree and Vocational- HS Diploma has a ratio of 0. So they are slightly risk free.

5.1.5 Industry Ratio:

Similarly, an i_ratio is established based on the industry column. To determine how many fake jobs there are for every real job, apply the formula below:

$$\text{I\_ratio} = \frac{\text{industry}|fradulent = 1}{\text{industry}|fradulent = 0}$$

The graph below only displays ratio values:



	industry	i_ratio
0	Accounting	0.559
1	Airlines/Aviation	0.016
2	Alternative Dispute Resolution	NaN
3	Animation	0.667
4	Apparel & Fashion	0.021
...	...	...
127	Wholesale	0.100
128	Wine and Spirits	NaN
129	Wireless	NaN
130	Writing and Editing	NaN
131	blank	0.059

132 rows × 2 columns

Fig.8: Bar graph of top 10 industry ratio with industry and ratio table

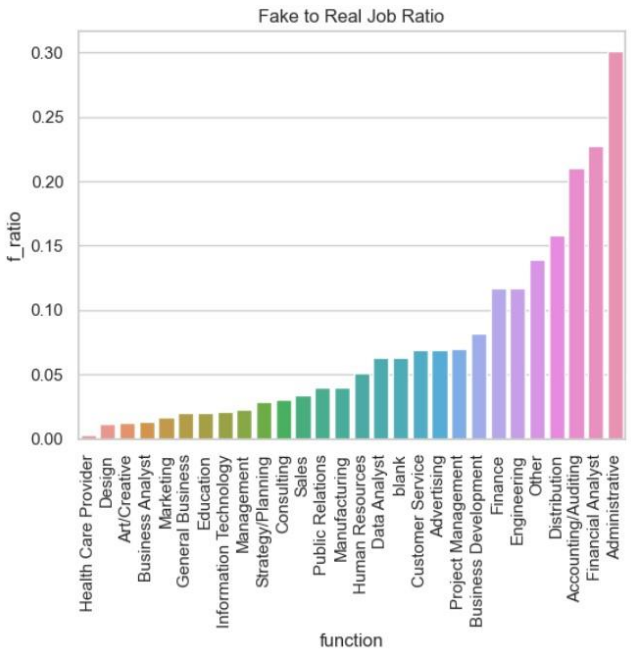
Military, Animation, oil and Energy have high phoney to actual job ratio. There is a strong likelihood that any job listings for this industry type are fake.

5.1.6 Function Ratio:

Similarly, an f_ratio is established based on the function column. To determine how many fake jobs there are for every real job, apply the formula below:

$$\text{f\_ratio} = \frac{\text{function}|fradulent = 1}{\text{function}|fradulent = 0}$$

The graph below only displays ratio values:



	function	f_ratio
0	Accounting/Auditing	0.210
1	Administrative	0.301
2	Advertising	0.069
3	Art/Creative	0.013
4	Business Analyst	0.014
5	Business Development	0.082
6	Consulting	0.031
7	Customer Service	0.069
8	Data Analyst	0.063
9	Design	0.012

Fig.9: Bar graph of function ratio with function and ratio table

5.2 TEXT CLEANING:

Now we are left with four string columns and four HTML fragments columns. We are creating a new column “text” by combining “Title”, “location”, “department” ,”company profile”, “description”, “requirements” and “benefits”.

Then we applied a bunch of text cleaning techniques to prepare the text for the next step, which is features extraction. Text cleaning techniques used are:

- ✓ ***Lowercasing:***

After we uploaded our data and save it into pandas’ data frame, we lowercased all the job posts using the function *str.lower().*

- ✓ ***Drop null:***

Removing the empty text samples.

- ✓ ***Tokenization:***

Using the natural language toolkit (NLTK), which is one the best-known and most-used natural language processing (NLP) libraries, we tokenized the job posts by the function *word_tokenize(),* which returns a tokenized copy of the text.

- ✓ ***Remove punctuation:***

Every token with non-alphabetical characters was removed.

- ✓ ***Remove stopping words:***

A stop word is a commonly used word, such as the, a, an, in, etc. which any search engine ignores when indexing entries or retrieving them. In text mining, those words are not wanted, because, they reserve space in the dataset, and take valuable processing time. So, we removed them from our text data.

- ✓ ***Remove HTML tags:***

Every html tags, single letters, URL, multiple spaces are removed from the "text" column.

After doing all of this we created a clean data frame containing four binary columns (Telecommuting, Company logo, Questions and Fraudulent), five ratio columns (Employment type ratio, Required experience ratio, Required education ratio, Industry ratio, Function ratio), text column and character count column.

	telecommuting	has_company_logo	has_questions	fraudulent	ratio	e_ratio	r_ratio	r_edu_ratio	i_ratio	f_ratio	text	character_count
0	0	1	0	0	0.03	0.071	0.027	0.059	0.059	0.017	market intern us ny new york market food creat...	1727
1	0	1	0	0	0.03	0.071	0.027	0.059	0.059	0.017	audienc develop intern us ny new york market f...	1736
2	0	1	1	0	0.00	0.075	0.027	0.059	0.059	0.017	market traine russian market gr athen market u...	1050
3	0	1	1	0	0.00	0.075	0.027	0.059	0.059	0.017	oud stage market nl ut amersfoort summaview ee...	1924
4	0	1	0	0	0.03	0.044	0.027	0.059	0.059	0.017	market intern us ny new york fusemachin combin...	2129
...	...	...	...	...	...	...	...	...	...	...	...	...
14756	0	1	1	0	0.52	0.075	0.019	0.059	0.612	0.000	purchas specialist us tx houston valor servic ...	2688
14757	0	0	0	0	0.05	0.044	0.031	0.020	0.114	0.000	materi manag hospit experi requir near casper ...	465
14758	0	0	0	0	0.00	0.044	0.057	0.046	0.134	0.000	execut assist purchas depart us nj lakewood pu...	704
14759	0	1	1	0	0.05	0.044	0.046	0.020	0.019	0.000	purchas director us human capit usual biggest ...	3414
14760	0	0	0	0	0.00	0.044	0.019	0.022	0.121	0.000	purchas agent us wi franksvill account financ ...	1579

14761 rows × 12 columns

Fig.10: Cleaned Data Frame

6. TEXT ANALYSIS:

6.1 Word Cloud:

A word cloud is a visual representation of textual data where the size of each word is determined by its frequency or importance in the given text. It provides a quick and intuitive way to analyze and visualize the most prominent words in a document, webpage, or a larger collection of text.

We split the clean data set into two parts, one is fraudulent and the other is legitimate. Python libraries such as **wordcloud** and **matplotlib** tools are used for generating word clouds.

Word cloud for fraudulent and real jobs:

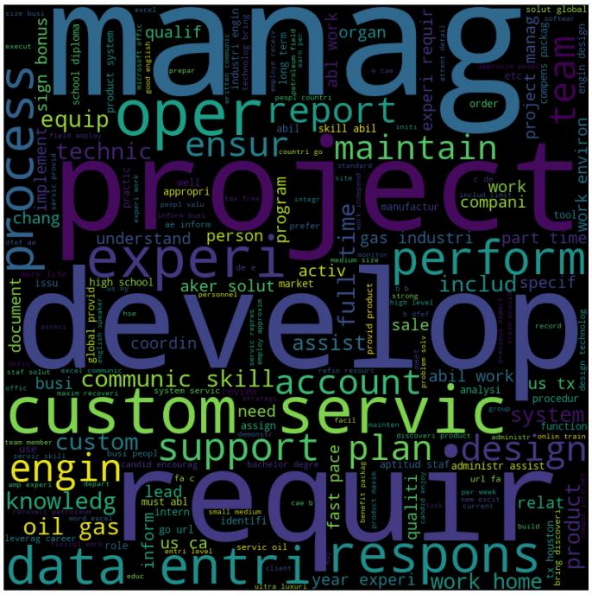


Fig.11: For fraudulent Jobs

Fig.12: For Real Jobs

From above we can see that words like “develop”, “custom service”, “manage”, “project”, ”require” etc are most frequently used for fraudulent jobs. There is a strong likelihood that any job listings these words more are fake.

Now word like “food”, ” market”, ”manage”, ”purchase”, ”company” etc are most frequently used for fraudulent jobs. There is a strong likelihood that any job listings these words more are legitimate.

Now let's take a close look at the bar diagram of word frequency for both fraudulent and legitimate jobs.

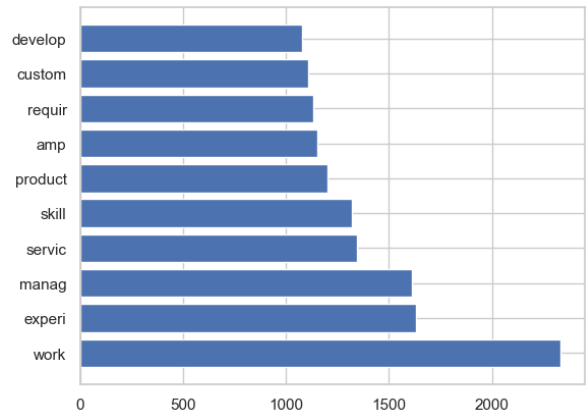


Fig.13: For fraudulent Jobs

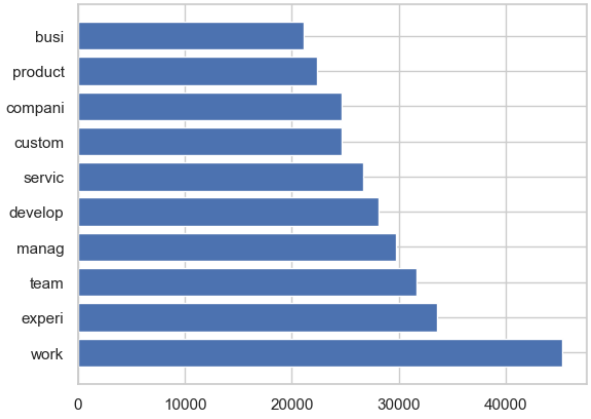


Fig.14: For Real Jobs

6.2 Feature Extraction Using TF-IDF:

Pre-processing and feature extraction constitute a barrier in text classification problems because text input must be translated into a representation that is appropriate for the learning method [36]. The text data must be transformed into term vectors. The term vector provides us with numerical values for each phrase found in a text. The most practical and well-liked method for converting words into vectors is TF-IDF.

TF-IDF stands for Term Frequency-Inverse Document Frequency. It is a numerical statistic that reflects the importance of a term in a document or a collection of documents. TF-IDF is commonly used in information retrieval, text mining, and natural language processing tasks.

Term Frequency (TF) = $\frac{Number\ of\ occurrences\ of\ a\ term\ in\ a\ document}{Total\ number\ of\ words\ in\ the\ document}$

Inverse Document Frequency (IDF) = $log\ \frac{Total\ number\ of\ documents}{Number\ of\ documents\ containing\ the\ term}$

TF-IDF : It combines the TF and IDF measures to determine the overall importance of a term within a document or a collection of documents. TF-IDF is calculated by multiplying the TF value of a term in a document by its IDF value.

TF-IDF = TF * IDF

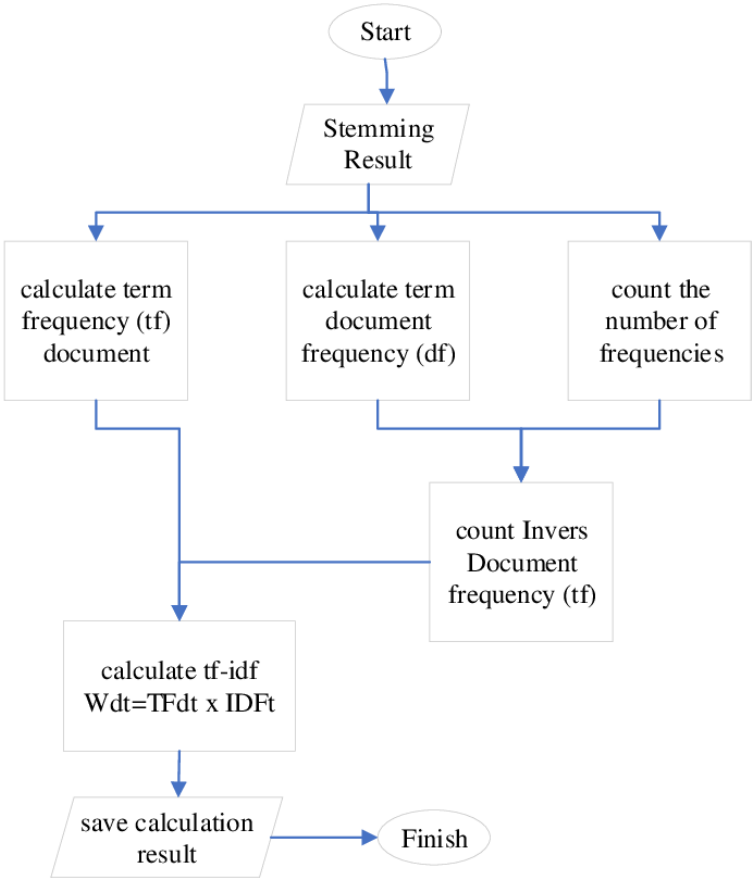


Fig.15: TF-IDF Flow Chart

7. Exploratory Visualization:

The final dataset has 17880 observations and 20 features. The dataset is highly unbalanced with 17014(95.15% of the jobs) being real and only 866 or 4.8% of the jobs being fraudulent. A count plot of the same can show the disparity very clearly.

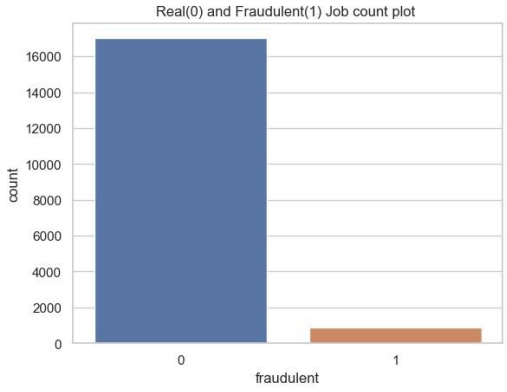


Fig.16: Real and fake job count bar graph

The first step to visualize the dataset in this project is to create a correlation matrix to study the relationship between the numeric data.

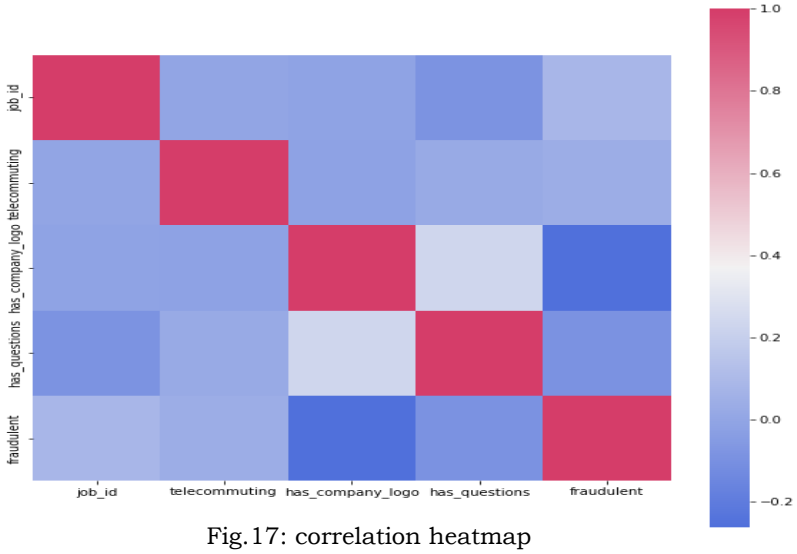


Fig.17: correlation heatmap

There are no significant positive or negative correlations between the numerical data in the correlation matrix.

For the Boolean variable telecommuting, a fascinating trend was discovered. There is a 92% possibility that the job will be fraudulent in cases where both of these variables have values of zero.

The linguistic aspects of this dataset are examined after the numeric features. We begin our study from location.

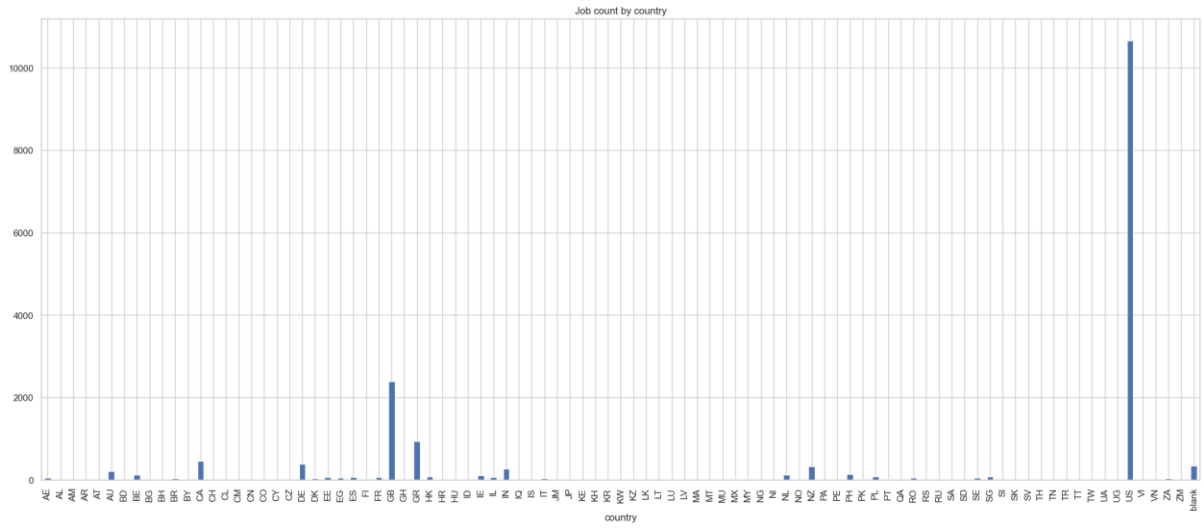


Fig.18: Bar graph of Job count with respect to countries

The previous chart illustrates which countries generate the most jobs. United State, GB (United Kingdom of Great Britain and Northern Ireland), Greece, Canada have the most job posts overall. Another bar plot is made to investigate this further. The distribution of fake and actual jobs in the top 10 cities is depicted in this bar plot.

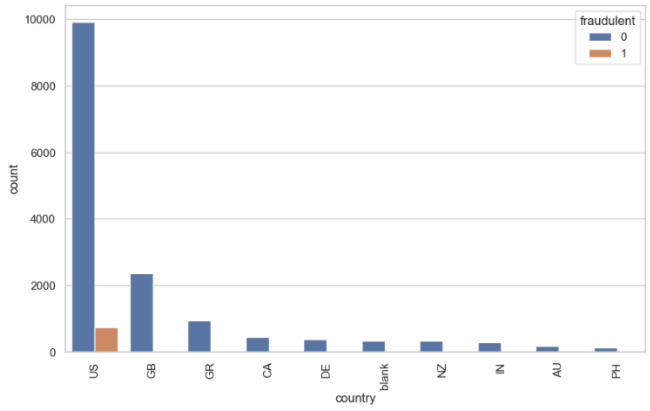


Fig.19: fake and actual jobs in the top 10 cities

Now let's see for the distribution of fake and actual jobs in the top four state_cities is depicted in this bar plot.

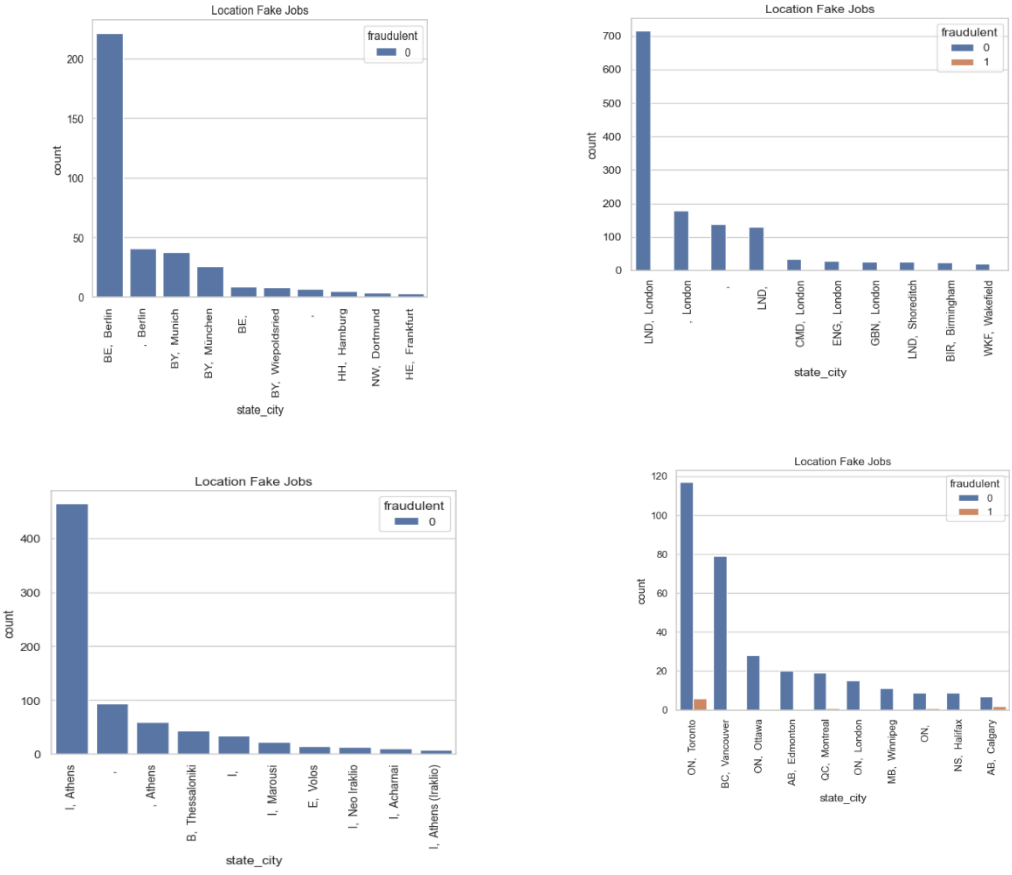


Fig.20: top left(US) top right(GB) bottom left(GR) bottom Right(CA)

8. MODEL ANALYSIS:

8.1 METHODOLOGY:

8.1.1 Logistic regression:

- One of the most well-known Machine Learning algorithms, under the Supervised Learning method, is logistic regression. Using a predetermined set of independent factors, it is used to predict the categorical dependent variable.
- Logistic regression forecasts the results of a dependent variable with a categorical component. As a result, the conclusion must be a discrete or categorical value. Rather of providing the exact values of 0 and 1, it provides the probabilistic values that fall between 0 and 1. It can be either Yes or No, 0 or 1, true or false, etc.
- The main difference between linear regression and logistic regression is how they are used. While logistic regression is used to solve classification difficulties, linear regression is used to solve regression problems.
- In logistic regression, we fit a "S" shaped logistic function, which predicts two maximum values (0 or 1), rather than a regression line.
- The logistic function's curve shows the possibility of several things, including whether or not the cells are malignant, whether or not a mouse is obese depending on its weight, etc.
- Because it can classify fresh data using both continuous and discrete datasets, logistic regression is a key machine learning algorithm.
- Logistic regression may be used to categorise observations using a variety of data types and can quickly identify the variables that will work best for the classification.

The below image is showing the logistic function:

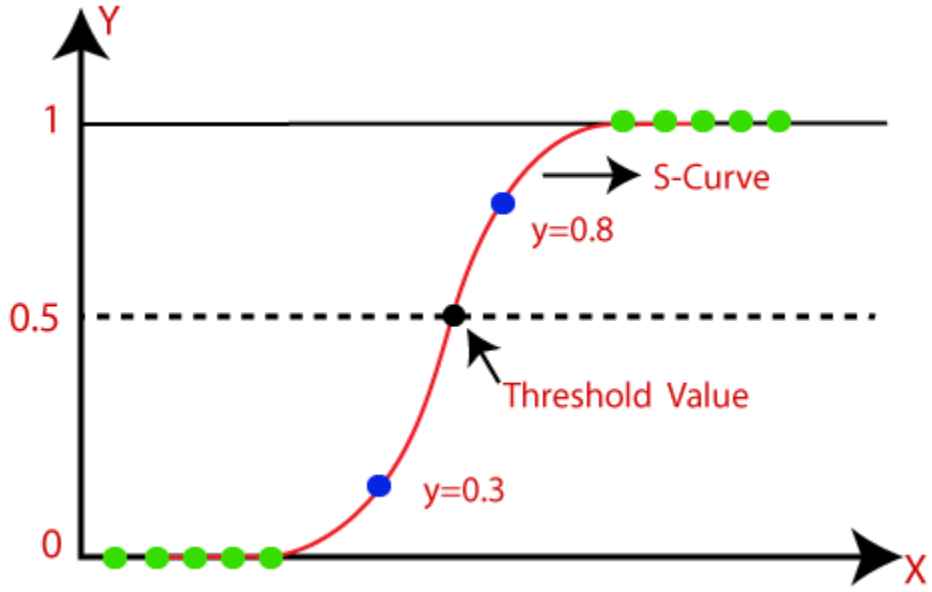


Fig.21: logistic function curve

Assumptions for Logistic Regression:

- The dependent variable must be categorical in nature.
- The independent variable should not have multi-collinearity.

8.1.2 Random forest:

- Popular machine learning algorithm Random Forest is a part of the supervised learning methodology. It can be applied to ML issues involving both classification and regression. It is built on the idea of ensemble learning, which is a method of integrating various classifiers to address difficult issues and enhance model performance.
- According to what its name implies, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead than depending on a single decision tree, the random forest uses forecasts from each tree and predicts the result based on the votes of the majority of predictions.

- The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:

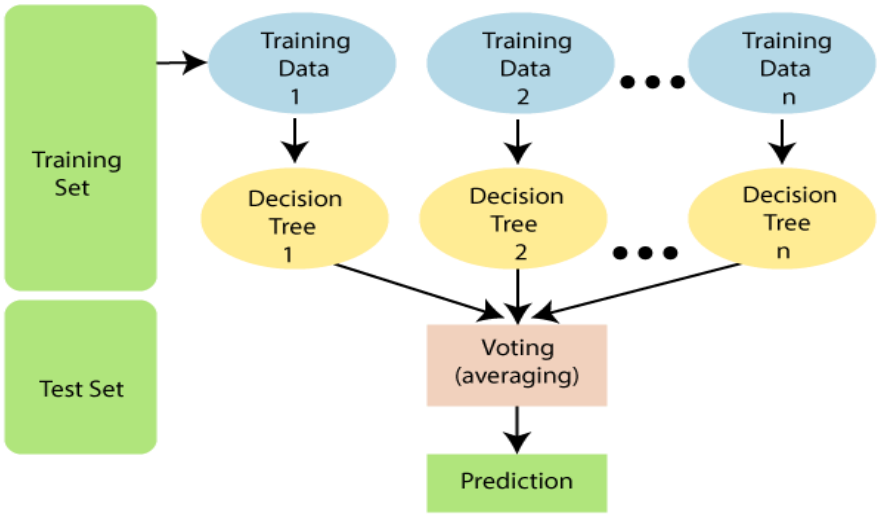


Fig.22: Random Forest algorithm

Assumptions for Random Forest:

Some decision trees may predict the correct output, while others may not, because the random forest combines numerous trees to forecast the class of the dataset. But when all the trees are combined, they forecast the right result. Consequently, the following two presumptions for an improved Random Forest classifier:

1. For the dataset's feature variable to predict true outcomes rather than a speculated result, there should be some actual values in the dataset.

2. Each tree's predictions must have extremely low correlations.

How does Random Forest algorithm work?

First, N decision trees are combined to generate the random forest, and then predictions are made for each tree that was produced in the first phase.

The stages and graphic below can be used to demonstrate the working process:

Step 1: Pick K data points at random from the training set.

Step 2: Create the decision trees linked to the subsets of data that have been chosen.

Step 3: Select N for the size of the decision trees you wish to construct.

Repeat steps 1 and 2 in step 4.

Step 5: Assign new data points to the category that receives the majority of votes by looking up each decision tree's predictions for the new data points.

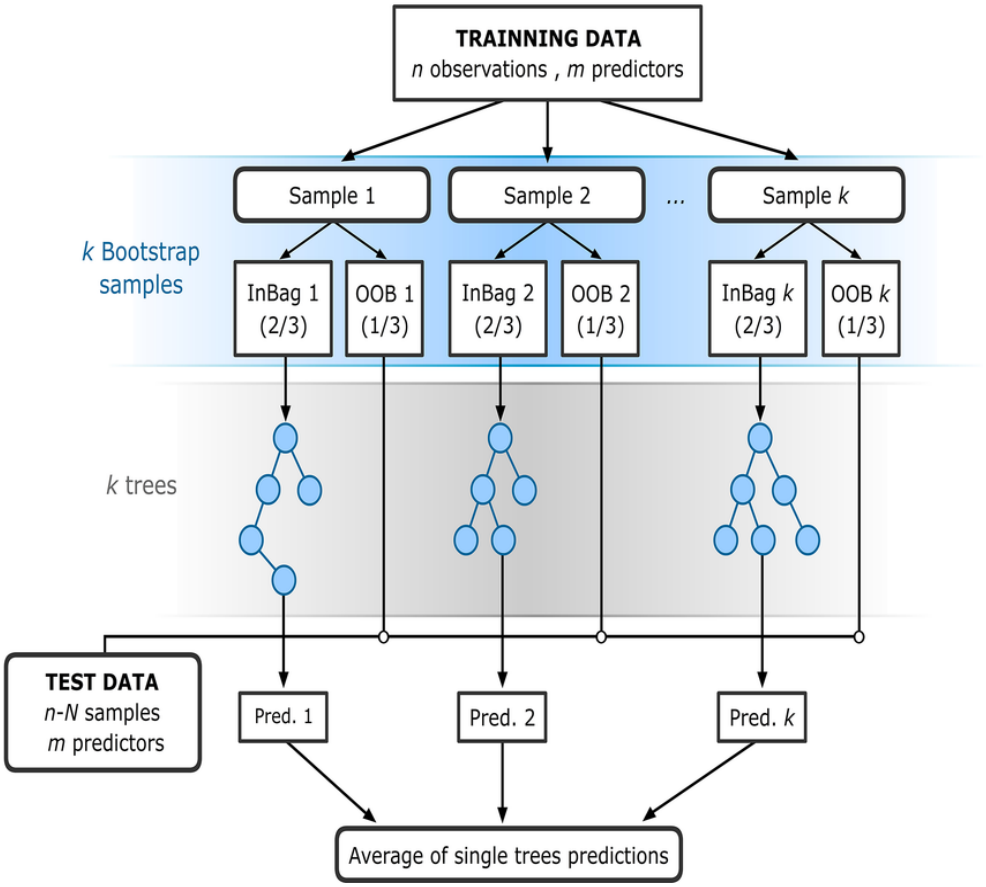


Fig.23: Flow chart of Random Forest

8.1.3 Multinomial Naïve Bayes :

A probabilistic learning technique called the Multinomial Naive Bayes algorithm is mostly employed in natural language processing (NLP). The method, which guesses the tag of a text such as an email or newspaper article, is based on the Bayes theorem. For a given sample, it determines the probabilities of each tag, and then outputs the tag with the highest probability.

The Naive Bayes classifier is a collection of many methods, all of which are based on the idea that each feature being classified is independent of every other feature. The existence or absence of one feature has no bearing on the other feature's existence or absence.

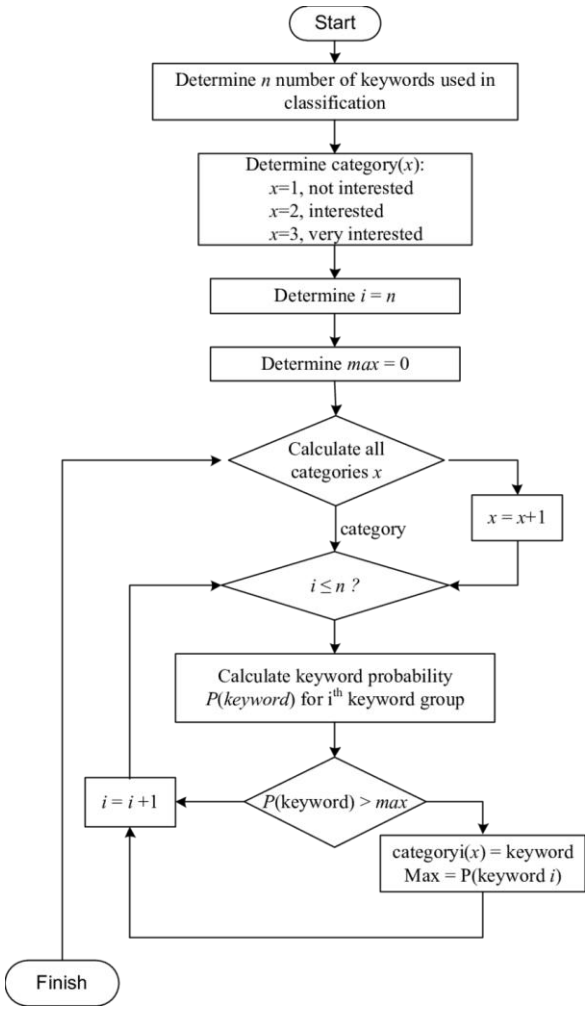


Fig.24: Flow chart of Multinomial Naïve Bayes

8.1.4 Stochastic Gradient Decent (SGD):

A straightforward yet highly effective method for fitting linear classifiers and regressors under convex loss functions, such as (linear) Support Vector Machines and Logistic Regression, is stochastic gradient descent (SGD). SGD has been present in the machine learning field for a while, but in the context of large-scale learning, it has just recently attracted a lot of attention.Large-scale and sparse machine learning issues that arise frequently in text

classification and natural language processing have been successfully tackled with SGD. The classifiers in this module are easily scalable to situations with more than 105 training examples and more than 10^5 features because the data is sparse.

SGD does not belong to any particular family of algorithms and is only an optimisation tool.

Classification:

The class SGDClassifier implements a straightforward stochastic gradient descent learning procedure that supports various classification loss functions and penalties. The decision boundary of an SGDClassifier that was trained using the hinge loss and is comparable to a linear SVM is shown below.

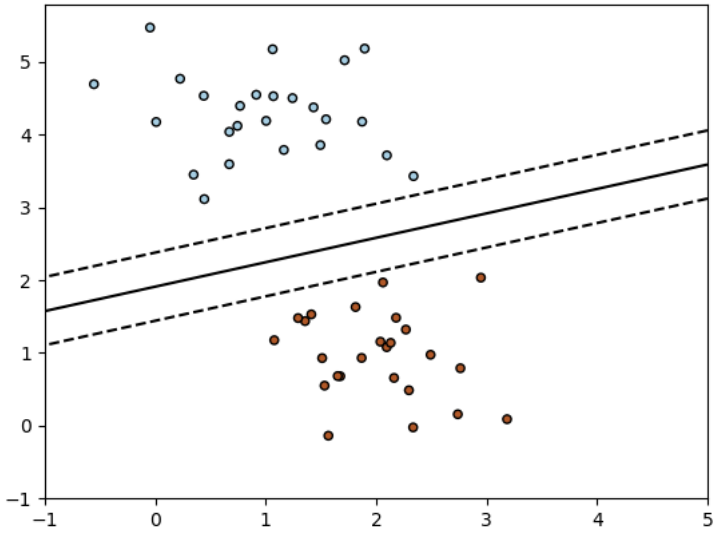
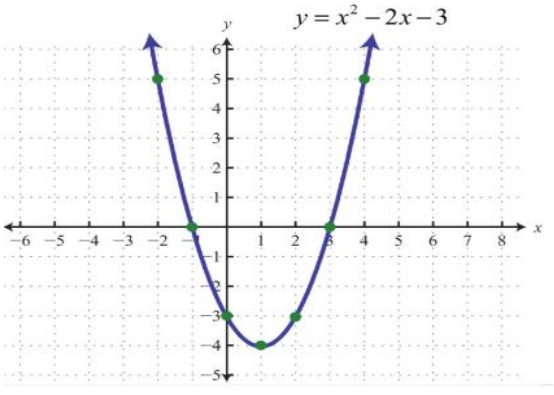


Fig.25: decision boundary of SGD classifier

What is the objective of Gradient Decent?

Gradient simply refers to a surface's slope or tilt. In order to get to the lowest point on the surface, one must literally descend a slope. Imagine a two-dimensional graph, like the parabola shown in the illustration below.



The parabola's lowest point in the graph above is at x = 1. To determine the value of "x" such that "y" is least is the goal of the gradient descent process. The objective function "y" in this context refers to the goal of the gradient descent method, which is to fall to the lowest point.

8.1.5 XGBoost:

XGBoost, which stands for "eXtreme Gradient Boosting," is a powerful machine learning algorithm that has gained significant popularity and success in various data science competitions and real-world applications. It is an implementation of the gradient boosting framework, which combines multiple weak prediction models (typically decision trees) into an ensemble model that can make accurate predictions.

Here are some key characteristics and features of the XGBoost algorithm:

1. Gradient boosting: XGBoost employs the gradient boosting technique, which builds an ensemble of weak models in a sequential manner. Each subsequent model is trained to correct the mistakes made by the previous models, resulting in a strong final model.

2. Decision trees as base learners: XGBoost uses decision trees as its base learners. Decision trees are constructed by recursively partitioning the input data based on specific feature values to create a tree-like model for prediction.

3. Regularization: XGBoost incorporates regularization techniques to prevent overfitting and improve generalization. It applies both L1 (Lasso) and L2 (Ridge) regularization terms on the model weights and includes them in the objective function that is optimized during training.

4. Objective function: The objective function in XGBoost combines a loss function and a regularization term. The loss function measures the model's performance and is chosen based on the specific problem (e.g., regression, classification). The regularization term penalizes overly complex models.

5. Feature importance: XGBoost provides a measure of feature importance, indicating which features are most relevant for making predictions. This information can be valuable for feature selection and understanding the underlying patterns in the data.

6. Handling missing values: XGBoost can handle missing values by learning how to treat them during training. It uses a technique called "sparsity-aware learning" to automatically learn the best direction to handle missing values in each node of the decision trees.

7. Parallel processing: XGBoost supports parallel processing, making it efficient and scalable for large datasets. It can take advantage of multi-core CPUs to train models faster.

8. Flexibility: XGBoost can be used for both regression and classification tasks. It supports various loss functions and evaluation metrics, allowing users to customize the algorithm according to their specific needs.

Overall, XGBoost is known for its high predictive accuracy, speed, and flexibility. It has become a popular choice in various domains, including finance, healthcare, natural language processing, and computer vision, among others.

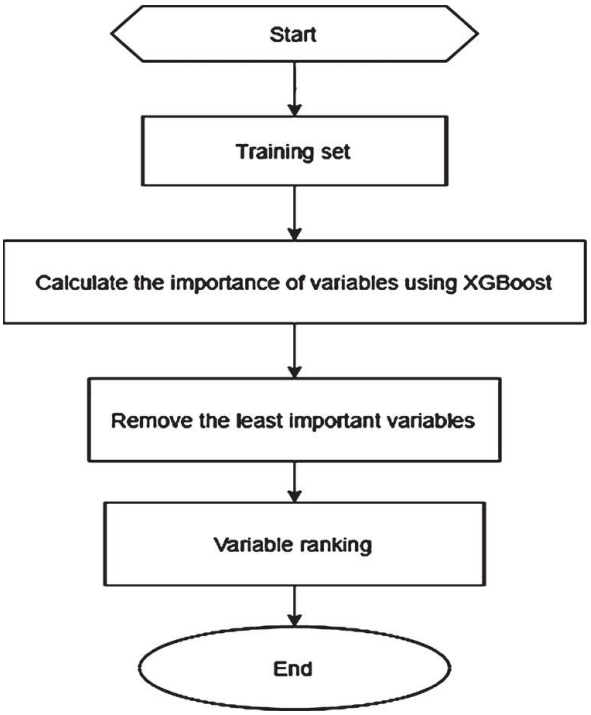


Fig.26: Flow chart of XGBoost

8.2 MODEL IMPLEMENTATION:

Below is a visual representation of how this project is implemented. The dataset is divided into three categories: text, numeric, and the y-variable. The text dataset is transformed into a term-frequency matrix to facilitate further analysis. Subsequently, using the sci-kit learn library, the datasets are split into test and train datasets. The initial models, Naïve Bayes and SGD, are trained using the train set, which accounts for 70% of the dataset.

To determine if a job posting is fraudulent, the outcomes of both models are combined based on two test sets: numeric and text. If both models agree that a particular data point is not fraudulent, it is classified as fraudulent. This approach helps alleviate the bias that Machine

Learning algorithms tend to exhibit towards majority classes. The trained model is then applied to the test set to assess its performance.

The Accuracy and F1-score of both Naïve Bayes and SGD models are compared, and based on this evaluation, the final model for our analysis is selected.

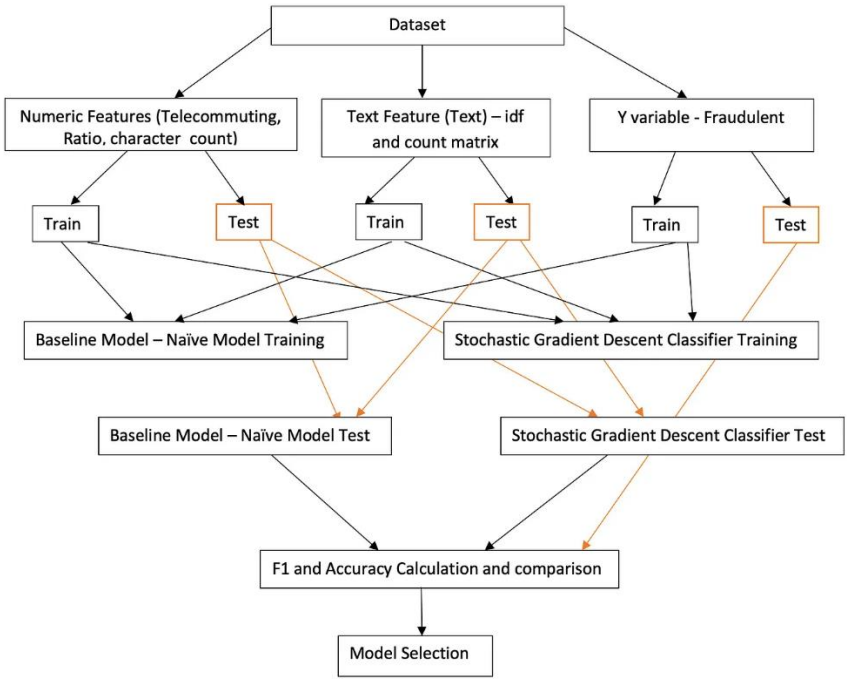


Fig.27: Flow chart of model selection

8.3 BENCHMARK:

The accuracy and F1-scores of Naïve Bayes and OTHER Classifier models are compared, and a final model is selected based on the evaluation. Naïve Bayes serves as the baseline model in this comparison. Its suitability lies in its ability to calculate conditional probabilities for two events by utilizing the probabilities of each event's occurrence. This characteristic proves to be highly valuable when encoding such probabilities.

The benchmark model for this project is Naïve Bayes. The overall accuracy of this model for textual data is 0.966, and the F1-score is 0.633. And for numerical data accuracy is 0.950 and the F1-score is 0.191. Any other model's capabilities will be compared to the results of Naïve Bayes.

8.4 MODEL TESTING:

8.4.1 For textual data:

The first experiment consists of the bag of words (bow) modeling of the job description, benefits, requirements and company profile HTML fields shown in Table 2. Before feeding our data to five classifiers, namely Logistic Regression Model, Random Forest Model, Multinomial Naïve Bayes Model, SGD Classifier Model, and XGBoost, we applied stopword filtering excluding most common English parts of speech such as articles and propositions. For each run, the corpus was split into training and cross-validation subsets using the k-fold cross-validation strategy (k = 10). The results are displayed in Tables 3 and 4.

Algorithm	Accuracy	Precision	recall	F1 Score
Logistic Regression	0.980	0.863	0. 0.760	0.633
Random Forest	0.977	0.993	0.595	0.633
Multinomial Naïve Bayes	0.966	0.798	0.524	0.633
SGD Classifier	0.977	0.798	0.785	0.633
XGBoost	0.982	0.950	0.719	0.693

Comparing the results, it turned out that XGBoost, followed by Random Forest achieved the highest recall, while the highest precision is achieved equally by Random Forest, and XGBoost. Moreover, we have noticed that XGBoost and Logistic Regression achieved the best f-measure, and accuracy. So, we are concluding that the most effective ML method dealing with our text classification problem is XGBoost .

Although, Multinomial Naïve Bayes(MNB) got the lowest accuracy. We think the reason behind the low performance in MNB lies in the nature of features. Our feature selection method ended up with a large vocabulary size, MNB achieves better performance when smaller vocabulary size is used. We already know that using the full vocabulary limits the model applicability on memory constrained, and its unnecessary in a way that many words may contribute little to the TC task and could have been removed safely from the vocabular. Moreover, we think the large dataset size made the Random Forest and the XGBoost the most efficient method, which is similar to a previous research results that showed in XGBoost and Random Forest achieves high accuracy in the case of large number of instances.

So, finally, according to our Text Classification problem, and taking into consideration the text pre-processing we did, and feature extraction we have applied, the XGBOOST is the best Machine Learning algorithm to solve the problem at hand and Random Forest classifier also give us the high accuracy .

8.4.2 For Numeric data:

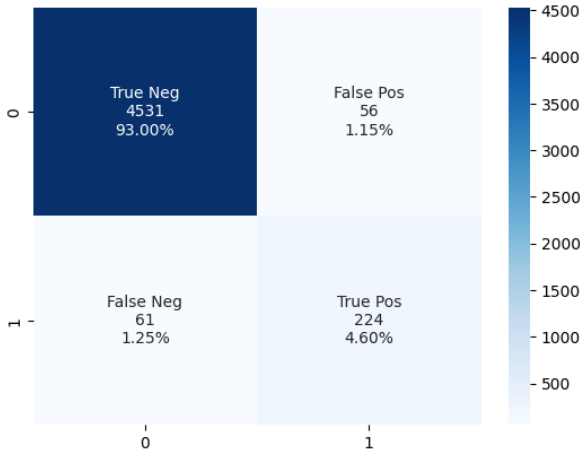
Algorithm	Accuracy	Precision	recall	F1 Score
Logistic Regression	0.956	0.760	0.289	0.419
Random Forest	0.970	0.815	0.603	0.693
Multinomial Naïve Bayes	0.950	0.896	0.107	0.191
SGD Classifier	0.945	0.473	0.0371	0.068
XGBoost	0.971	0.809	0.615	0.699

Comparing the results, it turned out that XGBoost, followed by Random Forest achieved the highest recall, while the highest precision is achieved equally by Random Forest and XGBoost. Moreover, we have noticed that XGBoost and Logistic Regression achieved the best f-measure, and accuracy. So, we are concluding that the most effective ML method dealing with our numeric classification problem is XGBoost .

After getting XGBoost as a best model for both textual and numeric data we make a prediction array such as, if we get for fraudulent coumn i = 0 and j = 0 for both models and its add 0 to the prediction array and for any other case it add 1 to the array. After doing this we get a accuracy of 0.975 and f1-score 0.792.
As mentioned above, the final model performs better than the established benchmark of the baseline model. The model will be able to identify real jobs with very high accuracy. However, its identification of fake jobs can still be improved upon.

8.4.3 Confusion Matrix:

A confusion matrix can be used to evaluate the quality of the project. The project aims to identify real and fake jobs.



The confusion matrix above displays the following values — categorized label, number of data points categorized under the label, and percentage of data represented in each category. The test set has a total of 4587 real jobs and 285 fake jobs. Based on the confusion matrix, it is evident that the model identifies real jobs 98.77% of the time. However, fraudulent jobs are identified only 78.59% of the time. Only 2.4% of the time has the model not identified the class correctly. This shortcoming has been discussed earlier as well as Machine Learning algorithms tending to prefer the dominant classes.

9. Conclusion:

Addressing the issue of fake job postings is a critical real-world challenge that demands proactive solutions. The primary objective of this project is to offer a potential remedy for this problem. To achieve optimal outcomes, the textual data undergoes preprocessing, while relevant numerical fields are carefully selected. Multiple models are utilized and their outputs are combined to generate the most favorable results. This approach helps mitigate the bias that machine learning models often exhibit towards the dominant class.

One fascinating aspect of this project is the identification of specific locations that serve as hotspots for fraudulent jobs. For instance, Bakersfield, California, demonstrates a fake to real job ratio of 12:1, highlighting the need for heightened monitoring in such areas. Another intriguing observation is that a significant proportion of fraudulent postings target entry-level positions. It appears that scammers tend to focus on younger individuals with a bachelor's degree or a high school diploma who are seeking full-time employment. The most challenging aspect of the project revolved around the pre-processing of the text data, which was in a complex format. Cleaning and organizing it required considerable effort.

10. Future Aspect:

1. We will try to find a recent EMSCAD dataset on INDIA.
2. We shall try to build a website where we can implement our model.
3. Try to analyze the textual data more effectively.

11. REFERENCE:

1) Dataset :- http://emscad.samos.aegean.gr/

2) Blanzieri, E.; Bryl, A. A survey of learning-based techniques of email spam filtering. Artif. Intell. Rev. **2008**,29, 63–92.

3) Guzella, T.S.; Caminhas, W.M. A review of machine learning approaches to spam filtering. Expert Syst. Appl.**2009**, 36, 10206–10222

4) Saadat, N. Survey on spam filtering techniques. Commun. Netw. **2011**, 3, 153–160.

5) Dinakar, K.; Reichart, R.; Lieberman, H. Modeling the detection of Textual Cyberbullying. In Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media, Barcelona, Spain, 17–21 July 2011.

6) Laboratory of Information and Communication Systems, University of the Aegean, Samos, Greece. EMSCAD Employment Scam Aegean Dataset, 2016. Available online: http://icsdweb.aegean.gr/emscad (accessed on 22 February 2017)

7) Vidros, S.; Kolias, C.; Kambourakis, G. Online recruitment services: Another playground for fraudsters. Compute. Fraud Secur. **2016**, 2016, 8–13.

8) Androutsopoulos, I.; Koutsias, J.; Chandrinos, K.V.; Spyropoulos, C.D. An Experimental Comparison of Naive Bayesian and Keyword-based Anti-spam Filtering with Personal e-Mail Messages. In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, Greece, 24–28 July 2000; ACM: New York, NY, USA, 2000; pp. 160–167.

9) Pantel, P.; Lin, D.; others. Spamcop: A spam classification & organization program. In Proceedings of the AAAI-98Workshop on Learning for Text Categorization, Madison, WI, USA, 26–27 July 1998; pp. 95–98.

10) Yeh, C.Y.;Wu, C.H.; Doong, S.H. Effective spam classification based on meta-heuristics. In Proceedings of the 2005 IEEE International Conference on Systems, Man and Cybernetics,Waikoloa, HI, USA, 10–12 October 2005; Volume 4, pp. 3872–3877.