**Fake Job Prediction**

*A project report submitted to ICT Academy of Kerala*

*in partial fulfillment of the requirements*

*for the certification of*

**CERTIFIED SPECIALIST**

**IN**

**DATA SCIENCE & ANALYTICS**

submitted by

BATCH 3 - GROUP 7:

**LABEEBA T**

**AMERDEEP SATISHKUMAR**

**GOWRI MOHAN**

**AKSHARA V**



**ICT ACADEMY OF KERALA**

**THIRUVANANTHAPURAM, KERALA, INDIA**

**Nov 2022**

**List of Figures**

| **FIG. NO** | **DESCRIPTION** | **PAGE NO.** |
| --- | --- | --- |
| 1.1 | Stages Of Development. | 13 |
| 3.1 | Total Count Of Real And Fake Jobs. . | 18 |
| 3.2 | Real And Fake Jobs Based On Telecommunication. | 18 |
| 3.3 | Real And Fake Jobs Based On Company Logo. | 19 |
| 3.4 | Real And Fake Jobs Based On Has Questions. | 19 |
| 3.5 | Real And Fake Jobs Based On Employment Type. | 20 |
| 3.6 | Real And Fake Jobs Based On Type Of Experience Required. | 21 |
| 3.7 | Real And Fake Jobs Based On Required Education. | 21 |
| 3.8 | Real And Fake Jobs Based On Function. | 22 |
| 3.9 | Real And Fake Jobs Based On Industry For Top 50 Industries. | 23 |
| 3.10 | Number Of Real And Fake Jobs Based On The Most Common Title Of Posts. | 24 |
| 3.11 | Number Of Real And Fake Jobs Based On Length Of Text Features. | 24 |
| 3.12 | Heatmap Based On Correlation Between Numerical Features. | 25 |
| 3.13 | Boxplot Of The Numerical Features. | 27 |
| 4.1 | Total Count Of Real And fake Jobs After Undersampling | 32 |
| 4.2 | Total Count Of Real And fake Jobs After Oversampling | 33 |

**List of Tables**

| **Table No** | **Description** | **Page No.** |
| --- | --- | --- |
| 3.1 | Brief Definition Of Variables. | 15-16 |
| 5.1 | Cross Validation Scores Of Different ML Algorithms. | 41 |
| 5.2 | Metrics Scores Of Different ML Algorithms. | 41-42 |
| 5.3 | Metrics Score Of Random Forest Classifier after Undersampling. | 42 |
| 5.4 | Metrics Score Of Random Forest Classifier After Oversampling | 42 |
| 5.5 | Metrics Score Of Random Forest Classifier After SMOTE Oversampling | 43 |
| 5.6 | Cross Validation Scores Of RF Algorithm After Undersampling. | 43 |
| 5.7 | Cross Validation Scores Of RF Algorithm After OVersampling. | 43 |
| 5.8 | Cross Validation Scores Of RF Algorithm After SMOTE Oversampling. | 44 |

**List of Abbreviations**

| 1. | **KNN** | k-nearest neighbors |
| --- | --- | --- |
| 2. | **ORF** | Online Recruitment Frauds |
| 3. | **CNBC** | Consumer News And Business Channel |
| 4. | **EDA** | Exploratory Data Analysis |
| 5. | **NLP** | Natural Language Processing |
| 6. | **SVM** | Support Vector Machines |
| 7. | **EMSCAD** | The Employment Scam Aegean Dataset |
| 8. | **NB** | Naïve Base Classifier |
| 9. | **RF** | Random Forest Classifier |
| 10. | **MLP** | MultiLayer Perceptron |

**Table of Contents**

**Abstract**  8

**Chapter I: Introduction** ……………………………………………………. 9

**Chapter II: Literature Survey** ……………………………………………. 10-11

**Chapter III: Methodology** …………………………………………. 12

1. Problem Definition ……………………………………………… 12

1.1 Overview ……………………………………………………… 12-13

1.2 Problem Statement ……………………………………………. 13-14

1. Data Collection ………………………………………………… 14
2. Data cleaning, exploring, and preprocessing ……………… 14

3.1 Data Understanding …………………………………………………… 14-17

3.2 Exploratory Data Analysis ……………………………………………. 17-26

3.3 Data Pre-processing …………………………………………………… 26

3.3.1 Data Cleaning ………………………………………………… 26-28

3.3.2 Feature Engineering ……………………………………… 28

3.3.3 Feature Reduction ……………………………………… 29

3.3.4 Feature Encoding …………………………………………. 29-30

1. Modeling ……………………………………………………………. 30

4.1 Data Sampling ……………………………………………………… 30-31

4.1.1 Undersampling ……………………………………………. 31-32

4.1.2 Oversampling ……………………………………………… 32-33

4.1.3 SMOTE ……………………………………………………. 33

4.2 Splitting of Preprocessed data ……………………………………… 33-34

4.3 Model Training (Fitting) …………………………………………….. 34

4.3.1 Logistic Regression ………………………………………… 34-35

4.3.2 KNN Classifier …………………………………………… 35

4.3.3 Support Vector Machine ……………………………………… 36

4.3.4 Decision Tree Classifier …………………………………… 36

4.3.5 Random Forest Classifier ……………………………………… 37

1. Evaluating …………………………………………………………….. 37-38

5.1 Evaluation Metrics Used ………………………………………….. 38-40

5.2 Cross Validation ………………………………………………………. 40

5.2.1 K-Fold Cross Validation …………………………………….. 40

5.2.2 Stratified K-Fold Cross Validation ………………………… 40-41

5.3 Results …………………………………………………………… 41-44

**Chapter IV: Implementation of the GUI** ………………………… 45-48

**Chapter V: Conclusion** ……………………………………………. 49

References 50

**Abstract**

In recent years, due to advancement in modern technology and social communication, advertising new job posts has become a very common issue in the present world. There are a lot of job advertisements on the internet, even on the reputed job advertising sites, which never seem fake. But after the selection, the so-called recruiters start asking for the money and the bank details. Many of the candidates fall in their trap and lose a lot of money and the current job sometimes. Identifying it manually is very difficult and almost impossible.

The project proposes an automated solution based on machine learning-based classification approaches to prevent fraudulent job postings on the internet. We can determine which job postings are fraudulent and which are not by conducting an exploratory data analysis on the data and using the insights gained. We will be using different classification algorithms like Logistic Regression, KNN, Decision Tree, Support Vector Machine, Random Forest Classifier,Naïve Bayes classifier etc.And comparing their accuracy and precision score to get the most suitable model. Also we are planning to make our work user friendly through website hosting.

**Chapter I**

**Introduction**

A fake job posting is a (rarely) smartly designed type of scam aimed at job seekers for a variety of unprofessional reasons. Still, these scams can look legit to an unsuspicious person scrolling through the vast pool of jobs. And although most tech talents aren’t actively looking for a new employer, falling for a phantom ad is still realistic.

Employment scam is one of the serious issues in recent times addressed in the domain of Online Recruitment Frauds (ORF).According to CNBC, the number of employment scams doubled in 2018 as compared to 2017.In recent days, many companies prefer to post their vacancies online so that these can be accessed easily and timely by the job-seekers.However, this intention may be one type of scam by the fraud people because they offer employment to job-seekers in terms of taking money from them.Fraudulent job advertisements can be posted against a reputed company for violating their credibility. Scammers will sometimes go the extra mile to draw the attention of their target audience, more often than not, by offering incredibly high salary ranges or another sort of advantage that seems too good to be true.

In recent years, due to advancement in modern technology and social communication, advertising new job posts has become a very common issue in the present world. So, fake job posting prediction tasks are going to be a great concern for all. Like many other classification tasks, fake job posing predictions leave a lot of challenges to face.This project proposed to use different classification algorithms like KNN, decision tree, support vector machine and random forest classifier to predict a job post if it is real or fraudulent.

This project uses data provided by [Kaggle](https://www.kaggle.com/shivamb/real-or-fake-fake-jobposting-prediction). This data contains features that define a job posting. These job postings are categorized as either real or fake.Here We will be splitting the dataset into two category for building the machine learning model for prediction.First without considering the text features and second with considering the text features.We will be comparing the accuracy,precision and f1-score of the different models and select the model having the highest score.And also we are going to create a website application to view whether the job is real or fake according to the prediction by our model.

**Chapter II**

**Literature Survey**

According to various studies, the areas of review spam identification, email spam detection, and fake news detection have received particular attention in the field of online fraud detection.

Review Spam Detection -

On online forums, people frequently discuss the products they buy. This makes it possible for spammers to alter reviews in order to gain financial benefit, necessitating the creation of algorithms to identify spam reviews. This can be accomplished by using Natural Language Processing to extract features from the reviews (NLP).

Email Spam Detection -

Spam emails are regularly found in user inboxes. When handling the issue of email spam detection, various approaches are taken into consideration, including content-based filtering, case-based filtering, heuristic-based filtering, memory or instance-based filtering, and adaptive spam filtering.

Fake News Detection -

In social media, malicious user accounts and echo chamber effects are used to define fake news. Three factors are used to identify fake news: the way it is written, how it spreads, and how a user is related to it. Machine learning algorithms are used in conjunction with features gathered from the news content and social context to identify fake news.

Many studies have been conducted to detect fraudulent job postings online. In a study by Vidros et al., they discovered statistics regarding numerous well-known companies that created false job advertisements with ill motives. On the EMSCAD dataset, they conducted experiments employing a variety of classification techniques, including the naive bayes classifier, random forest classifier, Zero R, One R, and others. The dataset's highest performance was displayed by the Random Forest Classifier, which had a classification accuracy of 89.5%. They found that logistic regression performed very poorly on the dataset.

Alghamdi et al. put forward a model to detect fraud exposure in an online recruitment system. They used the EMSCAD dataset. They worked on this dataset in three steps- data pre-processing, feature selection and fraud detection using classifiers. Random forest classifier showed 97.4% classification accuracy to detect fake job posts.

Shawni Dutta and Samir Bandyopadhyay studied fake job advertisements. They used the dataset "fake job postings". Machine learning models were used to detect fraudulent job posts, and the results of those models were compared to check which models provided better performance. They used various types of classifiers, such as naïve base classifier (NB), multilayer perceptron (MLP), k-nearest neighbor (KNN), and random forest classifier (RF). Random forest classifier achieved accuracy of 98.27%.

Habiba et.al proposed to use different data mining techniques and classification algorithms like KNN, decision tree, support vector machine, naive bayes classifier, random forest classifier, multilayer perceptron and deep neural network on EMSCAd dataset to predict a job post if it is real or fraudulent.

**Chapter III**

**Methodology**

**1. Problem Definition**

This project aims to create a classifier that will have the capability to identify fake and real jobs. Since the data provided has numeric and text features, one model will be used on the text data and another on numeric data. The final model will be selected by comparing the accuracy and precision score for the models which got the highest.

**1.1 Overview**

In modern day, the major challenge faced by any graduate is searching for his or her dream job, the pity however is that they generally fall for fake job postings and end up losing money and time, the proposed project makes use of a machine learning based model to help analyze these fake scams and secure their jobs.

This project follows five stages. The five steps adopted for this project are the following

1. Problem Definition (Project Overview, Problem Statement)
2. Data Collection
3. Data cleaning, exploring, and pre-processing
4. Modeling
5. Evaluating



Fig.1.1: Stages Of Development

**1.2 Problem Statement**

This project aims to create a classifier that will have the capability to identify fake and real jobs.Our solutions for predicting whether the job posting is real or fake include the following questions.

* Whether the recruiter contacts you
* Whether you receive a job offer right away
* Whether the pay is extremely high
* Whether the schedule seems too flexible
* Whether job requirements and description are vague
* Does the company require payment from you
* Does the job promise that you'll get wealthy fast
* Communication appear unprofessional
* Contact information for the employer or company is missing
* Too many spelling mistakes seen in description which may be from an unprofessional source.
* A company requests confidential information before hiring

**2. Data Collection**

This project uses data provided from Kaggle, which is originally from Laboratory of Information & Communication Systems Security of University of the Aegean (Greece). This is a public dataset of 17,880 real-life job ads where there are 17,014 legitimate and 866 fraudulent job ads published between 2012 and 2014. The dataset consists of 17,880 observations and 21 features. The data is a combination of integer, binary and textual data types.

**3. Data cleaning, exploring, and pre-processing**

**3.1 Data Understanding**

The dataset consists of 17,880 observations and 21 features. The data is a combination of integer, binary and textual data types. 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 | country | text | Information about which country the job is located |
| 5 | state | text | Information about which state the job is located |
| 6 | city | text | Information about which city the job is located |
| 7 | department | text | Information about the department this job is offered by |
| 8 | salary\_range | text | Expected salary range |
| 9 | company\_profile | text | Information about the company |
| 10 | description | text | A brief description about the position offered |
| 11 | requirements | text | Pre-requisites to qualify for the job |
| 12 | benefits | text | Benefits provided by the job |
| 13 | telecommuting | boolean | Is work from home or remote work allowed |
| 14 | has\_company\_logo | boolean | Does the job posting have a company logo |
| 15 | has\_questions | boolean | Does the job posting have any questions |
| 16 | employment\_type | text | 5 categories – Full-time, part-time, contract, temporary and other |
| 17 | required\_experience | text | Can be – Internship, Entry Level, Associate, Mid-senior level, Director, Executive or Not Applicable |
| 18 | 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 |
| 19 | Industry | text | The industry the job posting is relevant to |
| 20 | Function | text | The umbrella term to determining a job’s functionality |
| 21 | Fraudulent | boolean | The target variable if 0: Real, 1: Fake |

Table No 3.1: Brief definition Of Variables.

Since most of the datatypes are either Booleans or text a summary statistic is not needed here.

Categorical Features in the dataset are : ['title', 'location', 'country', 'state', 'city', 'department', 'salary\_range', 'company\_profile', 'description', 'requirements', 'benefits', 'employment\_type', 'required\_experience', 'required\_education', 'industry', 'function']

Numerical Features n the dataset are : ['telecommuting', 'has\_company\_logo', 'has\_questions', 'fraudulent']

The only integer is job\_id which is not relevant for this analysis. The dataset is further explored to identify null values. Variables such as department and salary\_range have a lot of missing values. These columns are dropped from further analysis. For categorical values we have filled null values with (data\_unavailable) using .fillna() and for text features we filled with a blank space. The dataset is not having any outliers.After that we have dropped the duplicate values from the data set using .duplicated() function. Next we have combined text features into a single feature column named ’text’. After doing this all steps our dataset count has dropped from 17880 rows and 21 cols to 17542 rows and 13 columns.

**3.2 Exploratory Data Analysis**

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns,to spot anomalies,to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.It is a good practice to understand the data first and try to gather as many insights from it. EDA is all about making sense of data in hand,before getting them dirty with it.

We have plotted some plots using seaborn and matplotlib so that we get some insights about how the target variable is dependent with independent variables.The dataset is highly unbalanced with 16702 (95% of the jobs) being real and only 840 or 5% of the jobs being fraudulent. A countplot of the same can show the disparity very clearly.

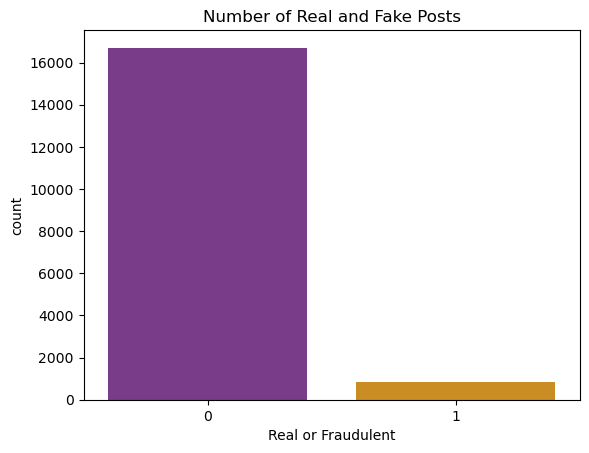


Fig 3.1: Total count of real & fake jobs

When there is no telecommunication chance of fake job postings is greater. If the company logo is present often it is a real job and chances for the postings to be fake is higher if the logo is absent.

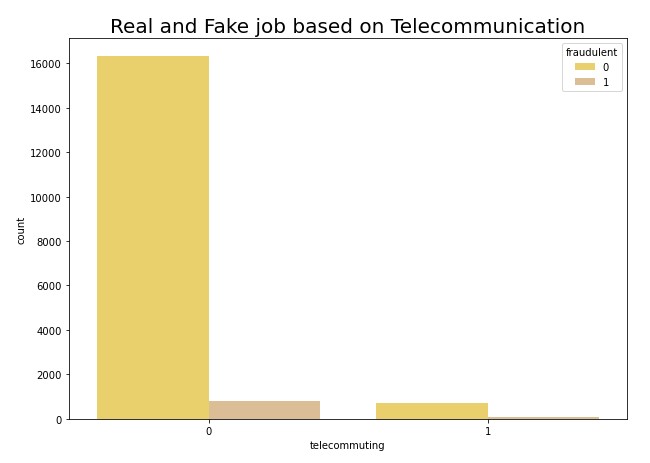
.

Fig 3.2: Real And Fake Jobs Based On Telecommunication.

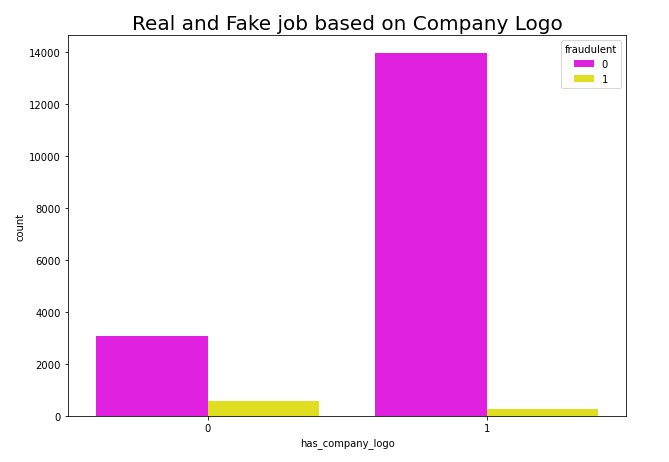


Fig 3.3: Real And Fake Jobs Based On Company Logo.

If job postings have no questionnaire present, it's more likely to be fake. And comparatively full-time type of employment has higher no. of fake jobs.

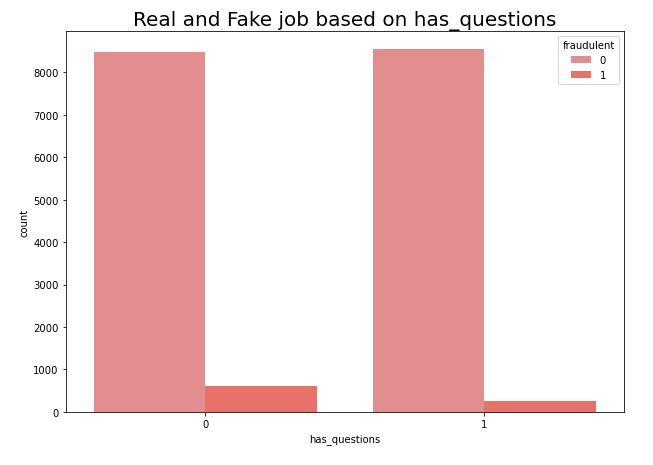


Fig 3.4: Real And Fake Jobs Based On Has\_Questions.

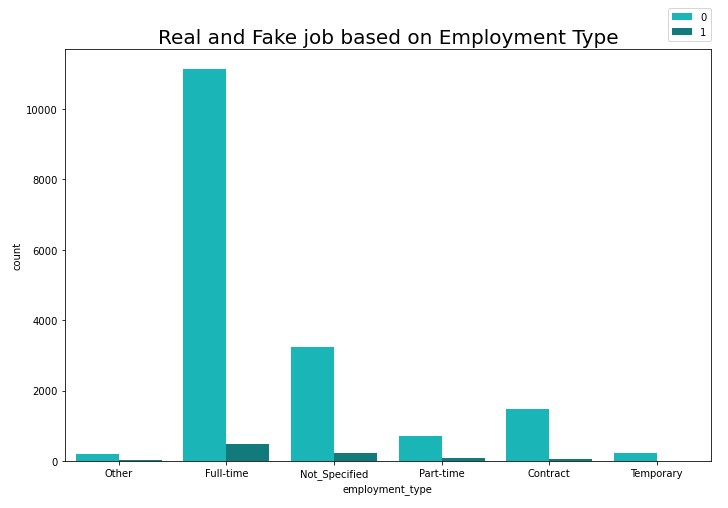


Fig 3.5: Real And Fake Jobs Based On Employment Type.

The graphs below show that most fraudulent jobs belong to if experience required or education required is not specified.

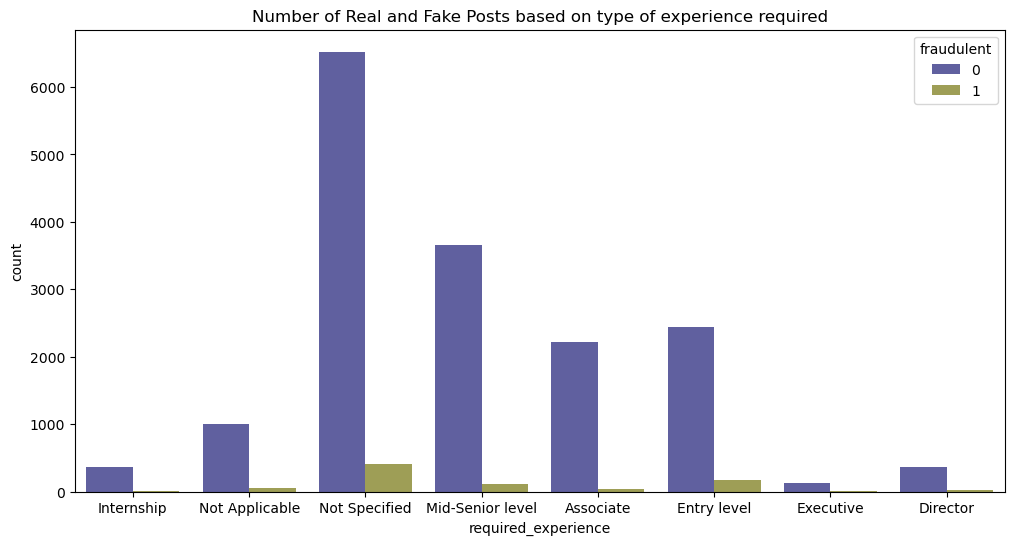


Fig 3.6: Real And Fake Jobs Based On Type Of Experience Required.

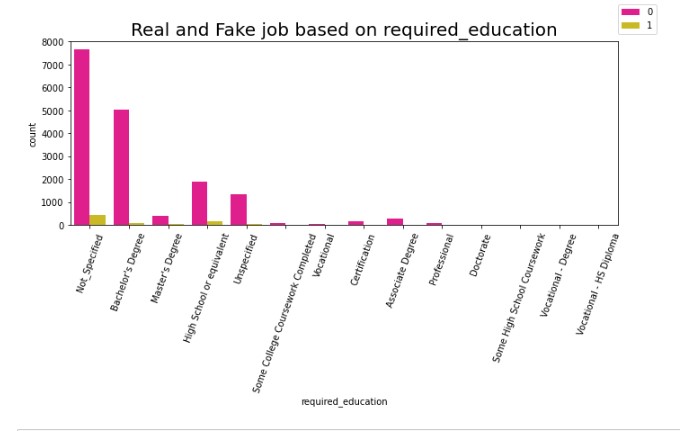


Fig 3.7: Real And Fake Jobs Based On Required\_Education.

The chance for job posting to be fake is greater when type of function or type of industry is not specified.

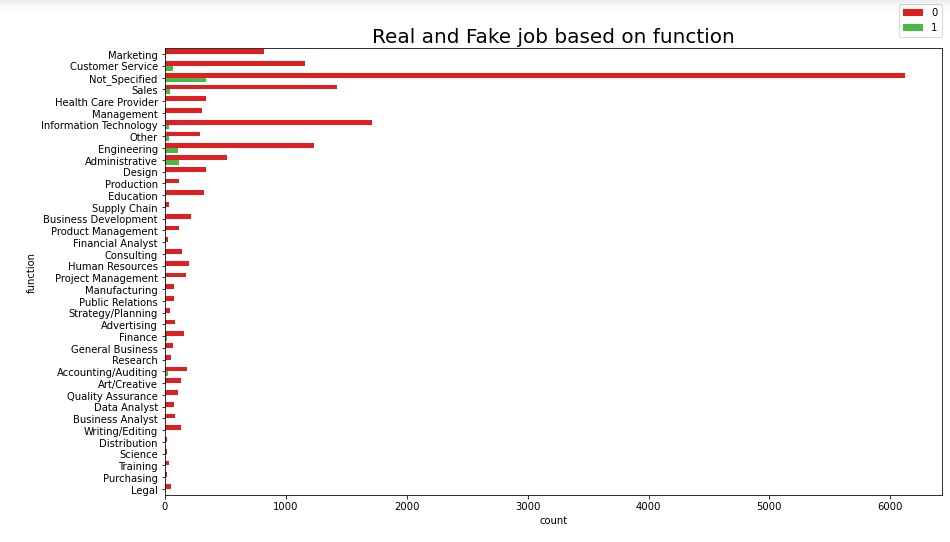


Fig 3.8: Real And Fake Jobs Based On Function.

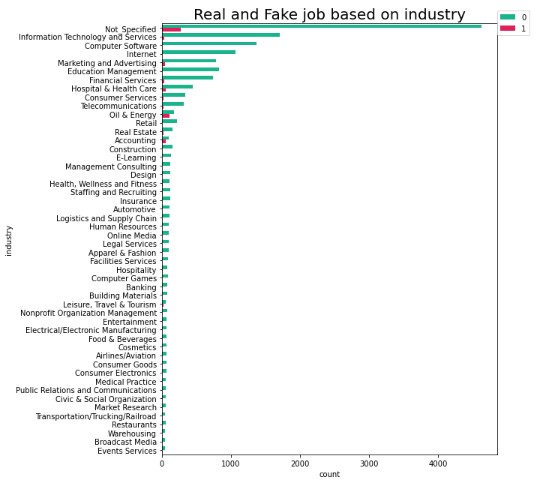


Fig 3.9: Real And Fake Jobs Based On Industry For Top 50 Industries.

Most common title of posts found in job postings is English Teacher Abroad.

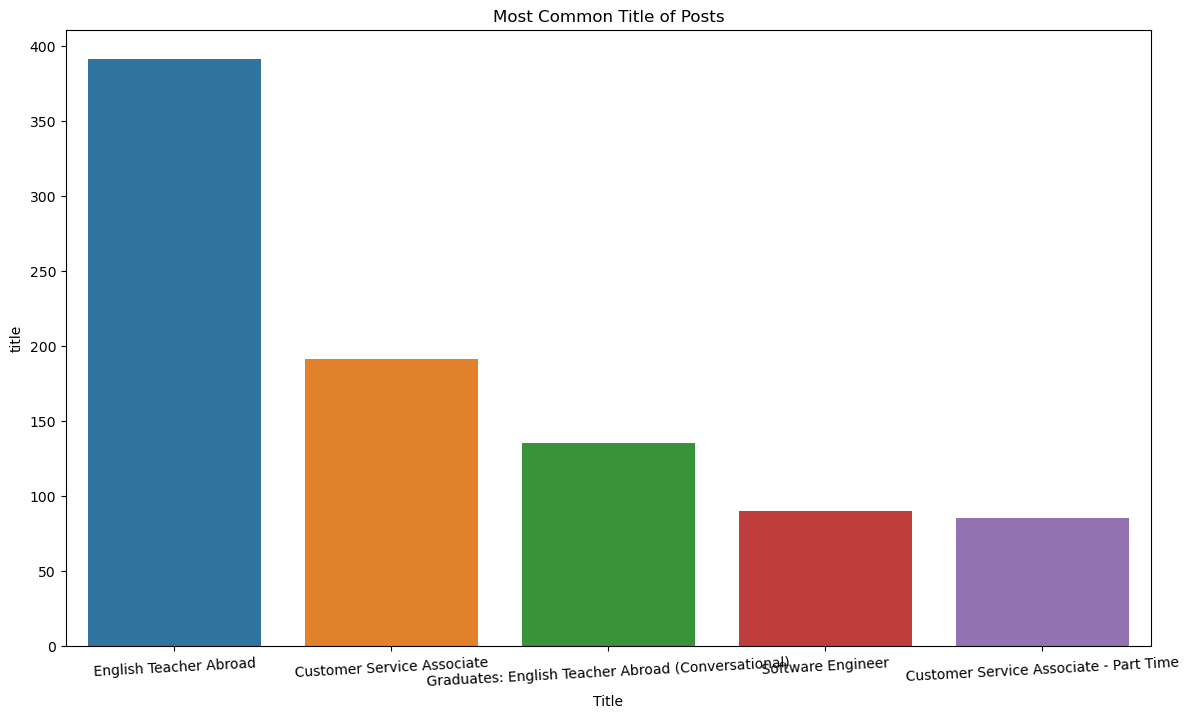


Fig.3.10: Number Of Real And Fake Jobs Based On The Most Common Title Of Posts.

To further extend the analysis on text related fields, the text-based categories are combined into one field called text. The fields that are combined are - title, company\_profile, description, requirements, benefits. A histogram describing a character count is explored to visualize the difference between real and fake jobs. What can be seen is that even though the character count is fairly similar for both real and fake jobs, real jobs have a higher frequency. Or we could say that fake job postings have less information than real job postings.

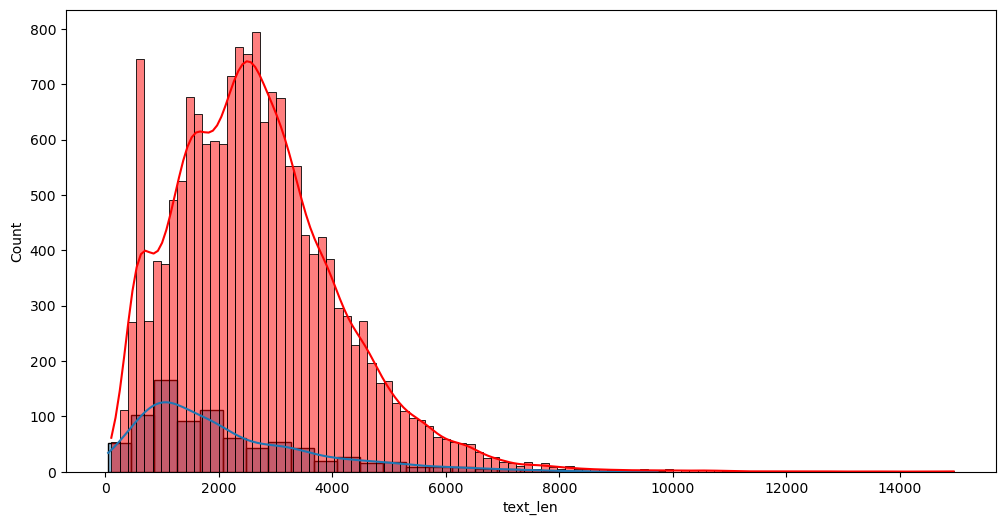
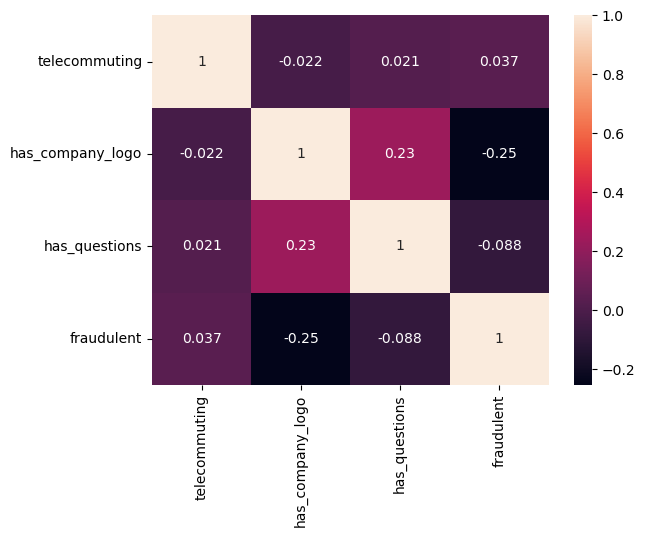


Fig 3.11: Number Of Real And Fake Jobs Based On Length Of Text Features.

After the textual features the numeric features of this dataset is explored. We do this by plotting correlation matrices. The correlation matrix does not exhibit any strong positive or negative correlations between the numeric data. However, has\_company\_logo is negatively correlated with dependent features. i.e, if the company logo is present often it is a real job post.

****

## Fig 3.12: Heatmap Based On Correlation Between Numerical Features.

## Key findings from EDA:

* Dataset is highly unbalanced.
* When there is no telecommunication chance of fake job postings is greater.
* has\_company\_logo is negatively correlated with dependent features. i.e, if the company logo is present often it is a real job post.
* If job postings have no questionnaire present, it's more likely to be fake.
* Comparatively full-time type of employment has higher no. of fake jobs.
* If experience required or education required and type of function or type of industry is not specified, the chance for job posting to be fake is greater.
* Most common title of posts found in job postings is English Teacher Abroad.
* Fake job postings have less information than real job postings.

**3.3 Data Pre-processing:**

Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In this step we clean our dataset by dealing null values, treating outliers, encode the object columns, drop unwanted columns, data splitting etc.

**3.3.1 Data Cleaning**

Techniques for cleaning up messy data include the following:

**Handling missing data:** The next step of data preprocessing is to handle missing data in the datasets.There are a variety of reasons a data set might be missing individual fields of data. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

* Missing values in the categorical column ‘required\_experience’ is replaced with ‘Not Applicable’ as it already had a category named same.
* Missing values in the categorical column ‘required\_education’ is replaced with ‘Unspecified’ as it already had a category named same.
* Missing values in all other categorical columns are replaced with ‘Not Specified’.

**Handling Outliers:** A data point that varies greatly from other results is referred to as an outlier. An outlier may also be described as an observation in our data that is incorrect or abnormal as compared to other observations. Outliers can be caused by errors in data collection or processing, or may represent legitimate but unusual observations.

* From the Box plot given below it is evident that our dataset has no outliers that need to be treated, as all columns in the plot have binary values to begin with.

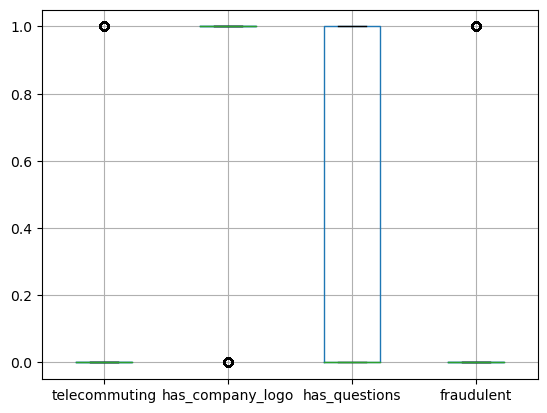


Fig 3.13: Boxplot Of The Numerical Features.

**Identify and remove duplicates.** When two records seem to repeat, an algorithm needs to determine if the same measurement was recorded twice, or the records represent different events. Removing duplicate values from your data set plays an important role in the cleansing process. Duplicate data takes up unnecessary storage space and slows down calculations at a minimum. At worst, duplicate data can skew analysis results and threaten the integrity of the data set.

* In our dataset we found that we have 339 duplicate values that need to be dropped.
* We dropped those duplicated rows using the drop\_duplicates function in pandas.

**3.3.2 Feature Engineering**

* For this project we splitted the dataset in two ways :

1. Dataset which takes up only the numerical columns or binary features and categorical columns.
2. Dataset that would consider textual features along with other features.

* A new feature is added to the dataset named ‘text’.
* The fields that are combined in newly created feature text are the textual columns - ‘title’, ‘company\_profile’, ‘description’, ‘requirements’, ‘benefits’.
* Among these 21 attributes, we have used only 8 attributes which are later converted into numerical values. Telecommuting, has\_company\_logo, has\_questions, employment\_type, required experience, required\_education , industry and function are changed into numerical value.
* The main goal to convert these attributes into two forms is to classify fraudulent job advertisements without doing any text processing and natural language processing. In this work, we have used only those categorical attributes dropping newly added feature ‘text’.

**3.3.3 Feature Reduction**

* All features that have a high number of missing values are dropped. For eg., "salary\_range" and ”department” which have missing values up to 11547 and 15012 are dropped.
* Columns that are irrelevant for data modeling purposes are dropped. Those dropped columns include : ‘job\_id’, ‘location’, ‘city’, ‘country’, and ‘state’.
* Since we are not planning on doing NLP we also dropped the newly created feature ‘text’ along with columns ‘title’, ‘company\_profile’, ‘description’, ‘requirements’, ‘benefits’.
* After exploratory data analysis and preprocessing we got the dataset with 17542 rows and 9 columns.

**3.3.4 Feature Encoding**

Most Machine Learning algorithms cannot work with categorical data and needs to be converted into numerical data. Sometimes in datasets, we encounter columns that contain categorical features (string values) for example parameter *Gender* will have categorical parameters like *Male*, *Female*. These labels have no specific order of preference and also since the data is string labels, machine learning models misinterpreted that there is some sort of hierarchy in them.

One approach to solve this problem can be label encoding where we will assign a numerical value to these labels for example *Male* and *Female* mapped to *0* and *1*. But this can add bias in our model as it will start giving higher preference to the *Female* parameter as 1>0 and ideally both labels are equally important in the dataset. To deal with this issue we will use **One Hot Encoding** technique.

In this technique, the categorical parameters will prepare separate columns for both Male and Female labels. So, wherever there is Male, the value will be 1 in Male column and 0 in Female column, and vice-versa.

* We converted non numerical values to numerical by one hot encoding , for that we used(pd.get\_dummies).
* We have also done with label encoder, but one hot encoding has given better results, so we chose one hot encoding.
* One challenge we faced in doing one hot encoding is that some features have a high number of unique values i.e., feature ‘industry’ had 132 unique values and the feature ‘function’ had 38 unique values.
* So, we set a threshold value for 100 and took only the top most values for industry and function that repeated more than 100 times.

We grouped other values in corresponding columns into a new category 'Others'. Thus, unique values in ‘industry’ and ‘function’ got reduced to 26 and 23 respectively. Then we did one hot encoding.

**4. Modeling**

**4.1 Data Sampling:**

Machine learning techniques often fail or give misleadingly optimistic performance on classification datasets with an imbalanced class distribution.

The reason is that many machine learning algorithms are designed to operate on classification data with an equal number of observations for each class. When this is not the case, algorithms can learn that very few examples are not important and can be ignored in order to achieve good performance.

Data sampling provides a collection of techniques that transform a training dataset in order to balance or better balance the class distribution. Once balanced, standard machine learning algorithms can be trained directly on the transformed dataset without any modification. This allows the challenge of imbalanced classification, even with severely imbalanced class distributions, to be addressed with a data preparation method.

There are many different types of data sampling methods that can be used, and there is no single best method to use on all classification problems and with all classification models. Like choosing a predictive model, careful experimentation is required to discover what works best for your project.

Our Dataset is highly unbalanced. So to resolve this problem, we have to over-sample it. Oversampling is the opposite of undersampling since in oversampling we balance out the class by just inserting new points into the minority class instead of removing them from the majority class. There are different types of oversampling techniques like Random oversampling and SMOTE to name a few. We preferred pandas inbuilt function (.sample) to balance the dataset.

**4.1.1 Undersampling**

As the name suggests this is the technique in which we balance uneven datasets by keeping all of the data in the minority class and decreasing the size of the majority class.It is one of several techniques data scientists can use to extract more accurate information from originally imbalanced datasets.Here we done undersampling first to check whether the machine model is giving better results.So initially class 0(Real) had 16702 values and class 1(Fake) had 840 values. So we did under sampling to make class 0 to be equal to class 1.After under sampling the size of the data changed to 1680 rows.

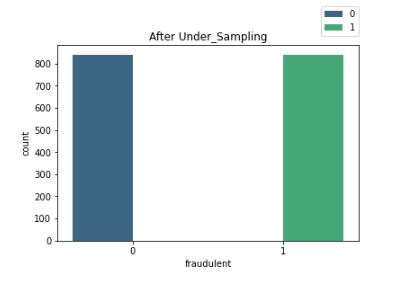


Fig 4.1: Total Count Of Real And Fake Jobs After Undersampling.

**4.1.2** **Oversampling**

As the name suggests this is the technique in which we balance uneven datasets by keeping all of the data in the majority class and increasing the size of the minority class. Here we increased the size of the minority class 1 to equal class 0. After oversampling our data was increased to 33404 rows.

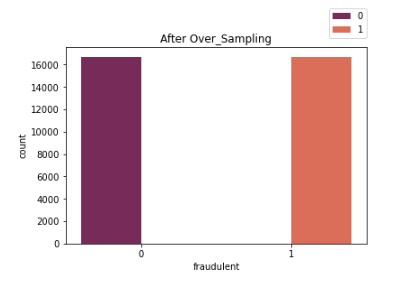


Fig 4.2: Total Count Of Real And Fake Jobs After Oversampling.

**4.1.3** **SMOTE**

SMOTE -Synthetic Minority Oversampling Technique is an oversampling technique where the synthetic samples are generated for the minority class. This algorithm helps to overcome the overfitting problem posed by random oversampling. It focuses on the feature space to generate new instances with the help of interpolation between the positive instances that lie together.SMOTE is in the library imblearn and it imported by (from imblearn.over\_sampling import SMOTE).We also done over sampling with SMOTE but we got precision and recall less.

**4.2 Splitting of preprocessed data**

In machine learning before modeling, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model. The train-test split is a technique for evaluating the performance of a machine learning algorithm.The procedure involves taking a dataset and dividing it into two subsets.

**Training Set:** A subset of dataset to train the machine learning model, and we already know the output.

**Test set:** A subset of dataset to test the machine learning model, and by using the test set, the model predicts the output.

The majority of the dataset is put aside for training, and a subset of testing data is also prepared. After the model has been developed using training data, it can be tested using testing data to determine its performance. The testing data serves as fresh, previously undiscovered data, enabling evaluation of the model's precision and degree of generalization.

The train-test procedure is appropriate when there is a sufficiently large dataset available. In this project, the function train\_test\_split( ) of the library sci-kit-learn of Python was used to split the data into the training set and testing set by a ratio of 75:25.

**4.3 Model Training (Model Fitting):**

Training is the most important step in machine learning. In training, you pass the prepared data to your machine learning model to find patterns and make predictions. It results in the model learning from the data so that it can accomplish the task set. Over time, with training, the model gets better at predicting. We have used hold out cross validation. 75% of the total data was used for training and 25% was used for testing and checking the model performance

**4.3.1 Logistic Regression**

Logistic regression is a Machine Learning classification algorithm that is used to predict the probability of certain classes based on some dependent variables. In short, the logistic regression model computes a sum of the input features (in most cases, there is a bias term), and calculates the logistic of the result.

The output of logistic regression is always between (0, and 1), which is suitable for a binary classification task. The higher the value, the higher the probability that the current sample is classified as class=1, and vice versa.

Logistic regression is also known as Binomial logistics regression. It is based on the sigmoid function where output is probability and input can be from -infinity to +infinity.

**4.3.2 KNN Classifier**

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

How does K-NN work?

The K-NN working can be explained on the basis of the below algorithm:

* Step-1: Select the number K of the neighbors
* Step-2: Calculate the Euclidean distance of K number of neighbors
* Step-3: Take the K nearest neighbors as per the calculated Euclidean distance
* Step-4: Among these k neighbors, count the number of the data points in each category.
* Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
* Step-6: Our model is ready

# 4.3.3 Support Vector Machine Algorithm

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors, and hence the algorithm is termed as Support Vector Machine.

**4.3.4 Decision Tree Algorithm**

The Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too. It splits the data based on the entropy and information gain. It looks like a flow chart. It creates the various categories within the categories. It splits the data of the high information gain first. They visually flow like trees, hence the name. The goal of using a Decision Tree is to create a training model that can be used to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).

**4.3.5 Random Forest Algorithm**

A Random Forest Algorithm is a supervised machine learning algorithm which is extremely popular and is used for Classification and Regression problems in Machine Learning. We know that a forest comprises numerous trees, and the more trees the more robust they will be. Similarly, the greater the number of trees in a Random Forest Algorithm, the higher its accuracy and problem-solving ability. Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. It is based on the concept of ensemble learning which is a process of combining multiple classifiers to solve a complex problem and improve the performance of the model.

The following steps explain the working Random Forest Algorithm:

* Step 1: Select random samples from a given data or training set.
* Step 2: This algorithm will construct a decision tree for every training data.
* Step 3: Voting will take place by averaging the decision tree.
* Step 4: Finally, select the most voted prediction result as the final prediction result.

**5. Evaluating**

After training your model, you have to check to see how it’s performing. This is done by testing the performance of the model on previously unseen data. The unseen data used is the testing set that you split our data into earlier. If testing was done on the same data which is used for training, you will not get an accurate measure, as the model is already used to the data, and finds the same patterns in it, as it previously did. This will give you disproportionately high accuracy.

When used on testing data, you get an accurate measure of how your model will perform and its speed.

**5.1 Evaluation Metrics Used**

Machine learning models cannot have 100 per cent efficiency otherwise the model is known as a biased model, which further includes the concept of overfitting and underfitting. It is necessary to obtain the accuracy on training data, but it is also important to get a genuine and approximate result on unseen data otherwise the Model is of no use. So, to build and deploy a generalized model we require to Evaluate the model on different metrics which helps us to better optimize the performance, fine-tune it, and obtain a better result.

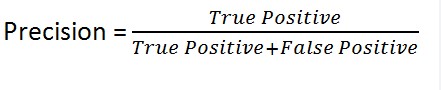
The models are evaluated based on these five metrics:

**Accuracy**:This formula defines this metric



Accuracy is a metric that generally describes how the model performs across all classes. It is useful when all classes are of equal importance. It is calculated as the ratio between the number of correct predictions to the total number of predictions**.**we use accuracy because it is useful since we are trying to identify both real and fake jobs, unlike a scenario where only one category is essential.

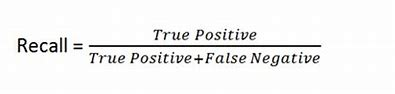
**Precision:** This formula defines this metric

****

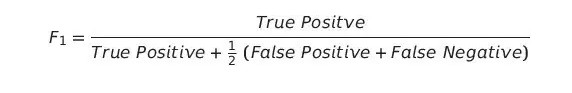
The precision is calculated as the ratio between the number of *Positive* samples correctly classified to the total number of samples classified as *Positive* (either correctly or incorrectly). The precision measures the model's accuracy in classifying a sample as positive.We use precision because we want our model to be as correct as possible when it says 1 and don’t care too much when it predicts 0.

**Recall:**

Recall is defined as the correctly predicted classifications over all the classifications of mem-bers of a certain class. The model gives the information as to how many objects that actually belong to the class in question get non-classified or get classified outside that class.



**F1-Score:** F1 score is a measure of a model’s accuracy on a dataset. The formula for this metric is



The F1-score combines the precision and recall of a classifier into a single metric by taking their harmonic mean.F1-score is used because, in this scenario, both false negatives and false positives are crucial. This model needs to identify both categories with the highest possible score since both have high costs.

**5.2 Cross Validation:**

Cross-validation is a technique in which we train our model using the subset of the data-set and then evaluate using the complementary subset of the data-set.

The three steps involved in cross-validation are as follows :

1. Reserve some portion of sample data-set.
2. Using the rest of the dataset we train the model.
3. Test the model using the reserve portion of the dataset.

**5.2.1 K-Fold Cross Validation**

In this method, we split the dataset into k numbers of subsets(known as folds) then we perform training on all the subsets but leave one (k-1) subset for the evaluation of the trained model. In this method, we iterate k times with a different subset reserved for testing purposes each time.We took 15 folds.

**5.2.2 Stratified K-Fold Validation**

Stratified k-fold cross-validation is just the same as k-fold cross-validation, but in Stratified k-fold cross-validation, it does stratified sampling instead of random sampling.

Stratified sampling is a process of rearranging the data to ensure that each fold or group is a good representative of the complete dataset. To deal with the bias and variance, it is one of the best approaches.We gave the number of folds to be 15.

| **Cross**  **Validation** | **Logistic Regression** | **KNN** | **Decision Tree** | **Random Forest** | **SVC** |
| --- | --- | --- | --- | --- | --- |
| **K-Fold** | **95.496** | **96.545** | **96.767** | **97.126** | **96.471** |
| **Stratified K-Fold** | **95.838** | **96.539** | **96.881** | **97.257** | **96.938** |

Table No 5.1 Cross Validation Scores Of Different ML Algorithms

(Before Resampling)

**5.3 Results**

We found that the Random Forest Algorithm shows better scores than other models.

**Without Resampling:**

| **Models** | **Accuracy** | **precision** | **recall** | **F1 Score** |
| --- | --- | --- | --- | --- |
| **Logistic** | **96.03** | **75.55** | **17.25** | **28.09** |
| **KNN** | **96.73** | **80.68** | **36.04** | **49.82** |
| **DT** | **96.92** | **71** | **52.79** | **60.64** |
| **RF** | **97.40** | **88.78** | **48.22** | **62.5** |
| **SVC** | **95.62** | **85.71** | **3.04** | **5.88** |

Table No 5.2: Metrics Scores Of Different ML Algorithms.

**With Resampling:**

Since Random Forest Classifier Given best scores, we are only going to find the metrics of Random Forest Classifier.

1.

| **UnderSample** | **Accuracy** | **precision** | **recall** | **F1 Score** |
| --- | --- | --- | --- | --- |
| **RF** | **85.476** | **84.018** | **87.619** | **85.780** |

Table No 5.3: Metrics Score of Random Forest Classifier After Undersampling.

2.

| **OverSample** | **Accuracy** | **precision** | **recall** | **F1 Score** |
| --- | --- | --- | --- | --- |
| **RF** | **93.509** | **93.157** | **93.916** | **93.535** |

Table No 5.4: Metrics Score of Random Forest Classifier After Oversampling.

3.

| **SMOTE** | **Accuracy** | **precision** | **recall** | **F1 Score** |
| --- | --- | --- | --- | --- |
| **RF** | **91.76** | **30.98** | **71.42** | **44.17** |

Table No 5.5: Metrics Score of Random Forest Classifier After SMOTE Oversampling.

**Cross Validation After Resampling**

1.Undersampling

| **Cross Validation** | **RF** |
| --- | --- |
| **K-Fold** | **85.773** |
| **Stratified K-Fold** | **85.892** |

Table No 5.6: Cross validation Score Of RF Algorithm after Undersampling.

2.Oversampling

| **Cross Validation** | **RF** |
| --- | --- |
| **K-Fold** | **93.826** |
| **Stratified K-Fold** | **93.839** |

Table No 5.7: Cross validation Score Of RF Algorithm after Oversampling.

3.SMOTE

| **Cross Validation** | **RF** |
| --- | --- |
| **K-Fold** | **97.10** |
| **Stratified K-Fold** | **97.18** |

Table No 5.8: Cross validation Score Of RF Algorithm after SMOTE.

Since the Random Forest Model of oversampled data gave better accuracy, precision, recall and f1\_score we use the oversampled model for prediction.

**Chapter IV**

**Implementation of the GUI**

Model deployment is the process of putting machine learning models into production. This makes the model’s predictions available to users, developers or systems, so they can make business decisions based on data, interact with their application and so on. The simplest way to deploy a machine learning model is to create a web service for prediction. Once you finish model training you have numerous choices for deployment of your project over the internet which are Django, Python Flask, Streamlit etc. In our project, the web application framework was executed using **Streamlit** and the website was hosted using **Streamlit Cloud.**

Streamlit is an open source app framework in python language. It helps us create beautiful web-apps for data science and machine learning in a little time. It is compatible with major python libraries such as scikit-learn, keras, pytorch, latex, numpy, pandas, matplotlib, etc. So, after building the web application using streamlit in pycharm we deployed it in streamlit cloud. Once the deployment process has finished, the application becomes publicly accessible on the base URL.

**The website can be accessed through below link:**

[**https://batch3team7-mainproject-app-bivbp4.streamlit.app/**](https://batch3team7-mainproject-app-bivbp4.streamlit.app/)

**or**

[**bit.ly/Main\_project**](http://bit.ly/Main_project)

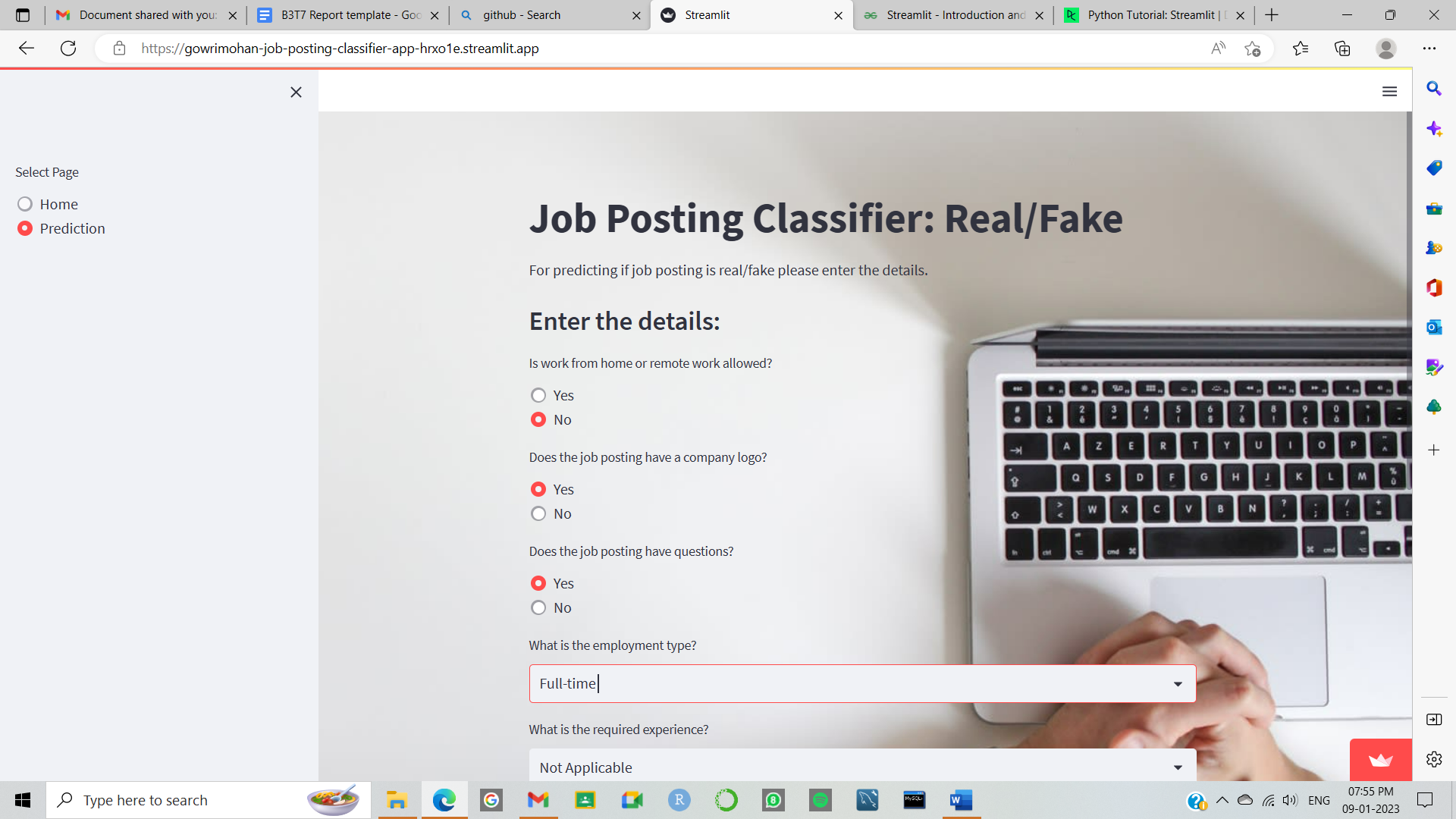
**The screenshots of the website is given below:**

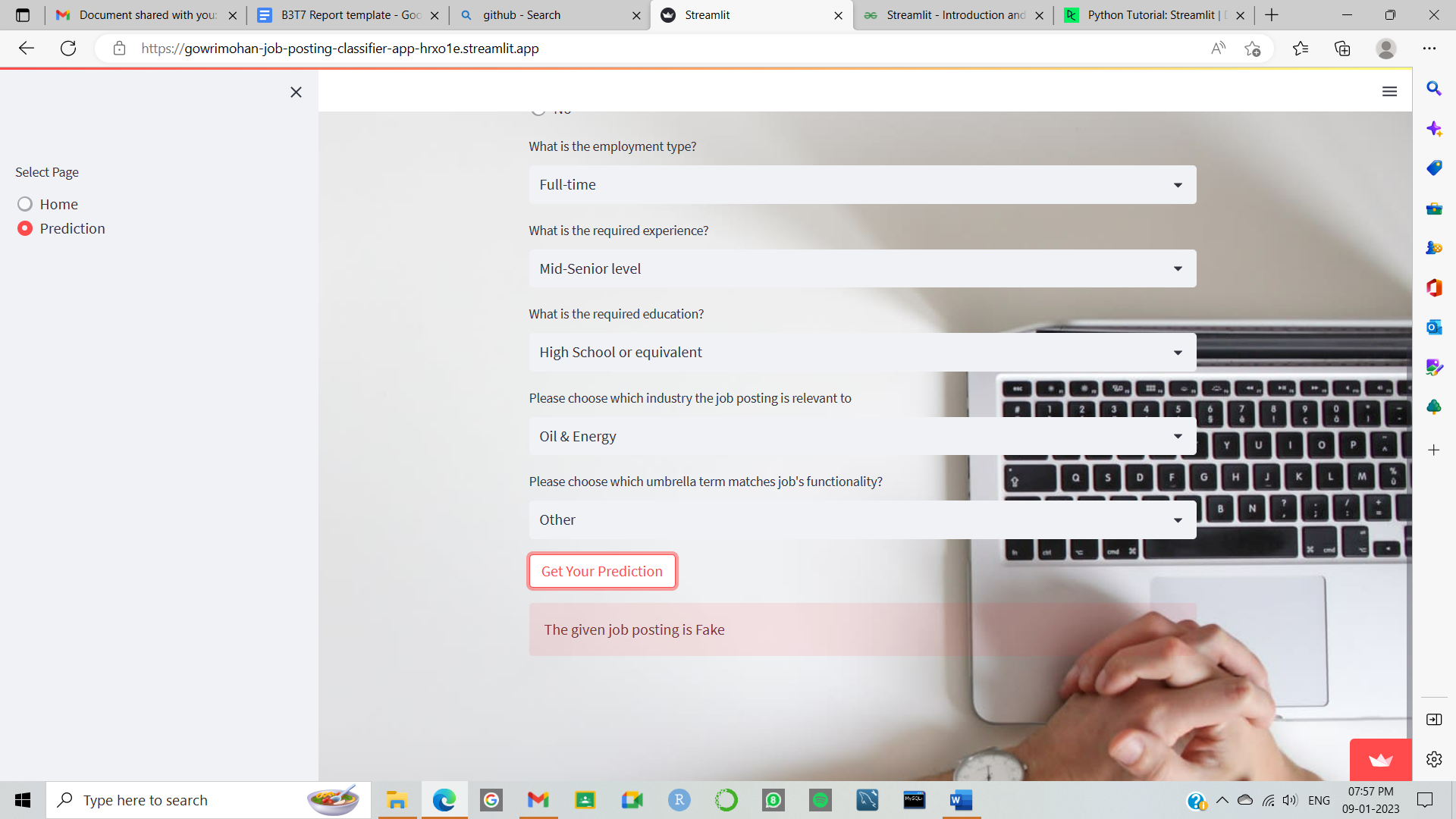
1. **Home Page**

****

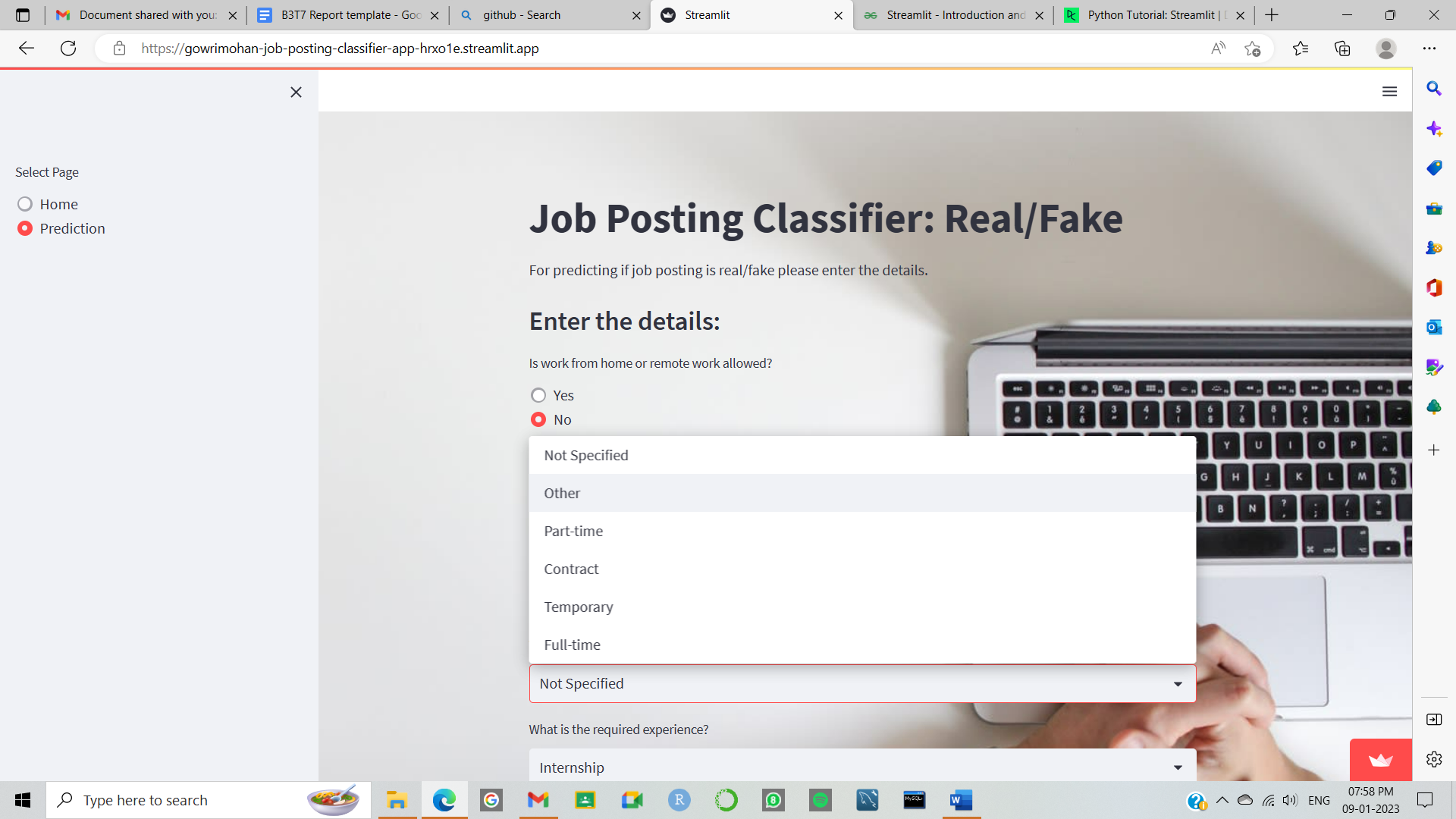
1. **Prediction page**

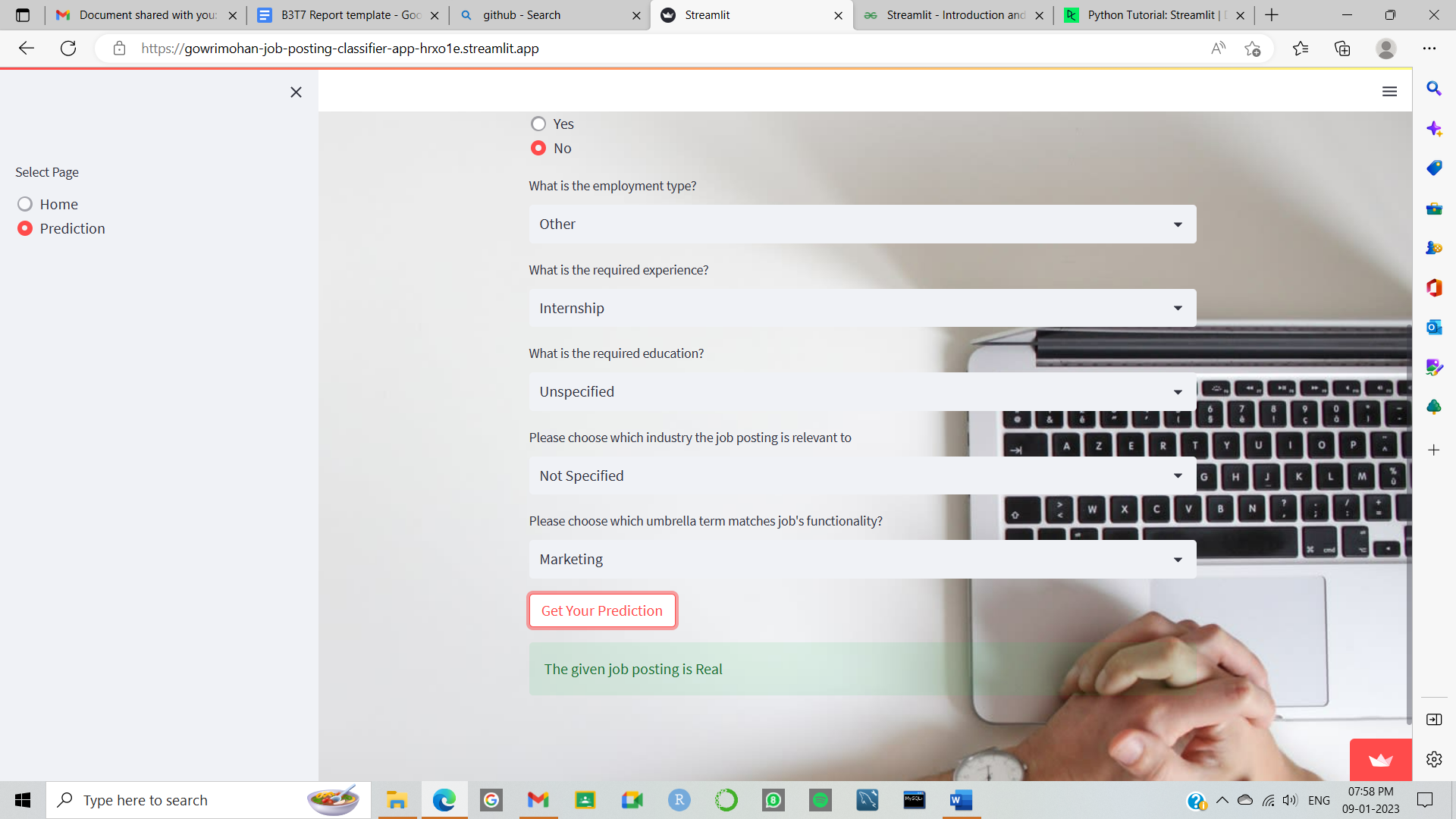
i) When Job Posting is fake





ii) When Job Posting is real





**Chapter V**

**Conclusion**

Job scam detection has become a great concern all over the world at present. In this paper, we have analyzed the impacts of job scams which can be very rare in the research field creating a lot of challenges to detect fraudulent job posts.

Job Posting Classifier: Real/Fake will help job searchers receive only real employment offers from firms. In this project, we have applied machine learning algorithms to classify and detect fake jobs from real jobs in a large dataset of job posts. Machine learning algorithms such as logistic regression, KNN classifier, decision tree classifier, random forest classifier and support vector machine are used for classification purposes.

Additionally, it can be inferred that data cleaning is a major step in any machine learning algorithm. The accuracy of the algorithms can be fine-tuned by cleaning and preprocessing the data in a proper way. By applying the classifiers we come to know that the random forest classifier gives the most accurate result about 93.83 in detecting the fake jobs compared to other classification algorithms.

**References**

1. S. Vidros, C. Kolias , G. Kambourakis ,and L. Akoglu, “Automatic Detection of Online Recruitment Frauds: Characteristics, Methods, and a Public Dataset”, Future Internet 2017, 9, 6; doi:10.3390/fi9010006
2. Alghamdi, B. and Alharby, F. (2019) An Intelligent Model for Online Recruitment Fraud Detection. Journal of Information Security, 10, 155-176. https://doi.org/10.4236/s.2019 103009
3. Bandyopadhyay, Samir & Dutta, Shawni. (2020). Fake Job Recruitment Detection Using Machine Learning Approach. International Journal of Engineering Trends and Technology. 68. 10.14445/22315381/IJETT-V68I4P209S
4. Habiba, Sultana & Islam, Md & Tasnim, Farzana. (2021). A Comparative Study on Fake Job Post Prediction Using Different Data Mining Techniques. 10.1109/ICREST51555.2021.9331230.
5. Allcott, H., & Gentzkow, M. (2017). Social media and fake news in the 2016 election. Journal of economic perspectives, 31(2), 211-36.
6. <https://imbalanced-learn.org/stable/introduction.html>
7. <https://www.w3schools.com/>
8. [https://www.analyticsvidhya.com/blog/2021/10/machine-learning-model-de](https://www.analyticsvidhya.com/blog/2021/10/machine-learning-model-deployment-using-streamlit/)

[ployment-using-streamlit/](https://www.analyticsvidhya.com/blog/2021/10/machine-learning-model-deployment-using-streamlit/)

1. <https://streamlit.io/>