1. INTRODUCTION

The rise of online job portals and social media has made it easier for job seekers to find employment opportunities. However, this has also led to an increase in fake job recruitment scams, where scammers pose as legitimate employers to deceive job seekers into revealing sensitive information or paying fees. These scams can cause significant financial and emotional distress to victims.

Traditional methods of detecting fake job recruitment scams rely on manual reporting and verification, which can be time-consuming and ineffective. With the increasing volume of online job postings, there is a need for a more efficient and accurate method of detecting fake job recruitment scams.

This project proposes a machine learning approach to detecting online fake job recruitment scams. By leveraging natural language processing (NLP) and machine learning algorithms, we aim to develop a system that can automatically classify job postings as legitimate or fake. Our approach will utilize a dataset of labeled job postings to train a machine learning model, which will then be evaluated on its accuracy and effectiveness in detecting fake job recruitment scams.

Figure 1.1. Puzzle of REAL and FAKE

With all the improvements and development in technology, systems have now become more online to ease accessibility to the user. But this also increases chances of people getting tricked as they have very less knowledge about these new technologies. Online frauds are becoming quite common these days. People lack knowledge regarding protection of their systems. One category of online frauds is Fake posting of jobs on the internet. Naive people who are desperate to get jobs usually get duped by scammers. People will get e-mail saying that they have been selected for job but then scammers will ask for payment of money to start the process of joining.

People who are innocent will pay hefty amounts of money thinking they will get the job but after payment they stop getting any reply from scammers posing as recruiter. This results a loss of individual's money and time. To stop this trickery we have built a detector system which would find out if the job is fake or legitimate with the help of various machine learning techniques. In recent years, some companies have begun to publish job openings on the internet so that job seekers can access them as soon as feasible. Employment fraud is one of the most serious issues that has arisen in recent years. Fake job postings will also harm the platform's reputation. This could, however, be a hoax by fraudsters who supply work to job seekers in exchange for money. While social networking platforms are an innovation in the computer sector, but then again, these platforms may be manipulated to fake information with ease, such as listing fake employment opportunities.

There is proof that false information spreads more quickly and widely through social media channels because individuals can publish it multiple times with no verification of the information. To begin, supervised machine learning algorithms are being investigated as categorizing methods to address the problem of spotting scammers on job advertising When people connect with misleading content on the internet, they are persuaded because they believe it is true. Most people does not even try to double-check the facts.

To prevent any more fraud cases and loss of huge amount of money to an individual and defamation of big corporations, we definitely need to come out with solutions. Hence, we decided to detect these online bogus job postings using ML algorithms like "K-nearest neighbour (KNN)" and "Random Forest". Different famous job recruitment platforms where corporation posts for job vacancy are Linked in, Glassdoor, Monstor, indeed, naukri.com and many more. A fake job detector for this platform would be beneficial for individuals looking for the employment.

The job market has become increasingly digital, with online job portals and social media platforms providing a vast array of employment opportunities. However, this has also led to a proliferation of fake job recruitment scams, which can cause significant financial and emotional distress to victims. To combat these scams, it is essential to develop effective methods for detecting and preventing them. This project proposes a machine learning approach to detecting online fake job recruitment scams, which can help protect job seekers from these types of scams.

The application is built using Streamlit, a powerful Python framework that allows for rapid development and deployment of machine learning models in a web-based interface. Users can interact with the system in two main ways: by training a model using a dataset of job postings and by making predictions on new job listings. The model, once trained, analyzes various features of job postings, such as the job title, company profile, description, and requirements, to identify suspicious patterns commonly associated with fraudulent listings.

As the job market becomes more competitive and online hiring continues to expand, the need for reliable and efficient tools to detect fake job postings becomes critical. The Fake Job Posting Detector serves as an essential safeguard by offering a user-friendly and highly accurate solution that protects job seekers and enhances trust in the hiring process.

2.LITERATURE SURVEY

2.1 S. Vidros, C. Kolias, G. Kambourakis and L. Akoglu in 2017 were proposed The critical process of hiring has relatively recently been ported to the cloud. Specifically, the automated systems responsible for completing the recruitment of new employees in an online fashion, aim to make the hiring process more immediate, accurate and cost-efficient. However, the online exposure of such traditional business procedures has introduced new points of failure that may lead to privacy loss for applicants and harm the reputation of organizations. So far, the most common case of Online Recruitment Frauds (ORF), is employment scam. Unlike relevant online fraud problems, the tackling of ORF has not yet received the proper attention, remaining largely unexplored until now. Responding to this need, the work at hand defines and describes the characteristics of this severe and timely novel cyber security research topic. **2.2** Devsmit Ranparia and Shaily Kumari in 2018 were proposed With increased number of data and privacy breaches day-by-day it becomes extremely difficult for one to stay safe online. Number of victims of fake job posting is increasing drastically day by day. The companies and fraudsters lure the job-seekers by various methods, majority coming from digital job-providing web sites. We target to minimize the number of such frauds by using Machine Learning to predict the chances of a job being fake so that the candidate can stay alert and take informed decisions, if required. The model will use NLP to analyze the sentiments and pattern in the job posting. The model will be trained as a Sequential Neural Network and using very popular GloVe algorithm. Ibrahim M. Nasser, Amjad H. Alzaanin and Ashraf Yunis Maghari in 2019 were proposed that Online recruitment provides job-seekers an efficient search and reach for jobs. It also helps recruiters searching for qualified candidates which improves the Journal of Engineering Sciences Vol 15 Issue 04,2024 ISSN:0377-9254 jespublication.com Page 2221 recruitment process. However, employment scam has emerged as a critical issue. Some job posts are legitimate, and others are fraud. In this paper, an Artificial Neural Network based model is proposed to detect fraud job posts. The public Employment Scam Aegean Dataset (EMSCAD) is used with proper text preprocessing techniques for training and testing the proposed model. Our model has precision, recall, and f-measure of 91.84%, 96.02%, and 93.88% respectively.

2.3 Tejasva Bhatia and Jasraj Meena in 2020 were proposed that There are a numerous amount of job postings on the internet and sometime these vacancy postings turns out to be

fake. Even on the reputed and trusted job advertising platforms, people fall prey to these fake advertisements . After selection in the job, hiring people start demanding for money and details of bank account. Good number of candidates gets duped and lose loads of money and sometimes even their current jobs. So it would be very helpful to identify if the job listed on the website is real or fake. In this paper, we have used machine learning to detect fraud job vacancy postings.In our proposed ML technique, we have applied two data balancing techniques namely "Adaptive Sympathetic" and "Synthetic Minority Oversampling Technique" in combination with "Term Frequency-Inverse Document Frequency" which if a feature extraction method. 2 Tin Van Huynh, Kiet Van Nguyen, Ngan Luu-Thuy Nguyen and Anh Gia-Tuan Nguyen in 2020 were proposed Determining the job is suitable for a student or a person looking for work based on their job descriptions such as knowledge and skills that are difficult, as well as how employers must find ways to choose the candidates that match the job they require. In this paper, we focus on studying the job prediction using different deep neural network models including TextCNN, Bi-GRU-LSTM-CNN, and Bi-GRU-CNN with various pre-trained word embeddings on the IT job dataset. In addition, we proposed a simple and effective ensemble model combining different deep neural network models. Our experimental results illustrated that our proposed ensemble model achieved the highest result with an F1- score of 72.71%. Moreover, we analyze these experimental results to have insights about this problem to find better solutions in the future.

2.4 P. Santhiya ,S. Kavitha ,T. Aravindh, S. Archana ,Ashwanth V Praveen in 2021 were proposed the The emergence of the World Wide Web and the quick uptake of social media platforms (like Facebook, Instagram and Whatsapp) have made it possible for information to be disseminated in ways that have never previously been seen in the history of humanity. Online hiring has changed the hiring pattern. In particular, putting job adverts on corporate websites and career portals includes looking for a sizable pool of qualified candidates worldwide. Unfortunately, it has become yet another forum for scammers, which threatens applicants' privacy and harms the reputation of companies. The topic of detecting recruiting fraud and scams is addressed in this case study. Three machine learning models are used in the construction of an effective recruitment fraud detection model, which includes a number of significant organizational, job description, and kind of remuneration features.

2.5 Banu Priya Prathaban, Subash Rajendran, G Lakshmi ,D. Menaka in 2021 were proposed In the recent times it is found to that there is a growing interest in Internet of

Things (IoT) and its respective sophisticated cloud architectures. However, such development does not ensure the confidentiality, integrity due to its security vulnerabilities in its improvements. Therefore data breaching occurs rapidly in all sectors. Nowadays, several job-seekers are becoming the prime victim for such digital loophole fraudsters, when there is an advertisement for bulk requirement. In order to reduce frauds and scams that are designed to take advantage of people who seek jobs, a model to predict the genuineness of posting digital vacancies for job is real or fake. This paper presents the implementation of Prediction of Employment Scam Model (POESM), intending for the classification of fraudulent and non-fraudulent digital job posting advertisements. In our work, we used eight techniques such as Logistic Regression, Naive Bayes, Multiple Layer Perceptron, KNearest Neighbor, Decision Tree, Random Forests, Adaboost, Gradient Boosting classifiers. To analyze the dataset, supervised machine learning techniques is used to capture several essential information.

2.6 B. Alghamdi and F. Alharby in 2022were proposed the research attempts to prohibit privacy and loss of money for individuals and organization by creating a reliable model which can detect the fraud exposure in the online recruitment environments. This research presents a major contribution represented in a reliable detection model using ensemble approach based on Random forest classifier to detect Online Recruitment Fraud (ORF). The detection of Online Recruitment Fraud is characterized by other types of electronic fraud detection by its modern and the scarcity Journal of Engineering Sciences Vol 15 Issue 04,2024 ISSN:0377-9254 jespublication.com Page 2222 of studies on this concept. The researcher proposed the detection model to achieve the objectives of this study. For feature selection, support vector machine method is used and for classification and detection, ensemble classifier using Random Forest is employed. Shawni Dutta and Samir Kumar Bandyopadhyay in 2022 were proposed To avoid fraudulent post for job in the internet, an automated tool using machine learning based classification techniques is proposed in the paper. Different classifiers are used for checking fraudulent post in the web and the results of those classifiers are compared for identifying the best employment scam detection model. It helps in detecting fake job posts from an enormous number of posts. Two major types of classifiers, such as single classifier and ensemble classifiers are considered for fraudulent job posts detection. However, experimental results indicate that ensemble classifiers are the best classification to detect scams over the single classifiers.

3.SYSTEM ANALYSIS

3.1 Problem Statement:

To analyze the detection of fake Online Recruitment Using Machine Learning Techniques

3.2 Existing System:

Information quality in social media is an increasingly important issue, but webscale data hinders experts' ability to assess and correct much of the inaccurate content, or "fake news," present in these platforms. This paper develops a method for automating fake news detection on Twitter by learning to predict accuracy assessments in two credibility-focused Twitter datasets: CREDBANK, a crowdsourced dataset of accuracy assessments for events in Twitter, and PHEME, a dataset of potential rumors in Twitter and journalistic assessments of their accuracies. We apply this method to Twitter content sourced from BuzzFeed's fake news dataset and show models trained against crowdsourced workers outperform models based on journalists' assessment and models trained on a pooled dataset of both crowdsourced workers and journalists. All three datasets, aligned into a uniform format, are also publicly available. A feature analysis then identifies features that are most predictive for crowdsourced and journalistic accuracy assessments, results of which are consistent with prior work. We close with a discussion contrasting accuracy and credibility and why models of non-experts outperform models of journalists for fake news detection in Twitter.

3.2.1 Disadvantages:

Because of Privacy Issues the Facebook dataset is very limited and a lot of details are not made public.

- Having less accuracy
- More complex.

3.3 Proposed System:

The proposed framework, the sequence of processes that need to be followed for continues detection of fake job post with active learning from the feedback of the result given by the classification algorithm. This framework can easily be implemented by social networking companies. 1. The detection process starts with 1 the selection of the post that needs to be tested. 2. After the selection of the post, the suitable attributes (i.e. features) are

selected on which the classification algorithm is implemented. 3. The attributes extracted is passed to the trained classifier. The classifier gets trained regularly as new training data is feed into the classifier. 4. The classifier determines whether the post is fake or genuine. 5. The classifier may not be 100% accurate in classifying the post so; the feedback of the result is given back to the classifier. 6. This process repeats and as the time proceeds, the no. of training data increases and the classifier becomes more and more accurate in predicting the fake job post.

3.3.1. Objectives:

The main objective of fake online recruitment detection using machine learning is to develop an intelligent system that can accurately identify fraudulent job postings and recruitment scams. By leveraging machine learning algorithms, the system analyzes various features such as job descriptions, recruiter details, salary information, and posting patterns to distinguish between legitimate and fake listings. The goal is to enhance online job security, protect job seekers from fraud, and minimize financial and personal data exploitation. This system aims to improve trust in online job portals by providing an automated and efficient detection mechanism.

- Develop an intelligent system to detect fake online job postings.
- Use machine learning algorithms to analyze job-related features.
- Identify fraudulent recruitment patterns based on textual and metadata attributes.
- Protect job seekers from scams and financial/data exploitation.
- Enhance trust and security in online job portals.
- Provide an automated and efficient detection mechanism

3.3.2. Applications:

The main applications of our project are:

- ✓ Online Job Portals
- ✓ Recruitment Agencies
- ✓ Cybersecurity
- ✓ Government and Law Enforcement
- ✓ Banks and Financial Institutions
- ✓ Educational Institutions

3.3.3. Advantages:

- Cost-effective.
- Easy analysis using topic modeling.
- Algorithm is very simple to implement**.**

3.3.4. Algorithm:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

The first algorithm for random decision forests was created in 1995 by Tin Kam Ho[1] using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered "Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.).The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho[1] and later independently by Amit and Geman[13] in order to construct a collection of decision trees with controlled variance.

Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

Limitations:

- **Slow Prediction Time** – Can be slower in real-time detection due to multiple decision trees
- **High Storage Requirement** – Needs significant storage space for large models with many trees.

3.4. Software Envinornment:

For implementing fake online recruitment detection using machine learning, the software environment consists of:

1. Operating System

- Windows 10 or Windows 11 – Provides a stable and user-friendly environment for running machine learning models and development tools.

2. Programming Language

- Python – Chosen for its extensive libraries, ease of use, and strong support for machine learning and data science.

3. Development Tools

- Jupyter Notebook – For interactive coding and model development.
- VS Code / PyCharm – For efficient script writing and debugging.
- Google Colab – For cloud-based execution of machine learning models.

4. Key Python Libraries

- Machine Learning: Scikit-Learn, TensorFlow, Keras
- Data Processing: Pandas, NumPy
- Natural Language Processing (NLP): NLTK, SpaCy
- Feature Extraction: TF-IDF, CountVectorizer
- Visualization: Matplotlib, Seaborn
- Model Evaluation: Scikit-Learn metrics (accuracy, precision, recall, F1-score)

This software environment ensures smooth development, execution, and deployment of the machine learning model for detecting fake job postings efficiently.

3.5. System Requirements:

3.5.1. Hardware Requirements:

- System : Pentium i3 Processor
- Hard Disk : 500 GB.
- Monitor : 15'' LED

- Input Devices : Keyboard, Mouse
- Ram : 2 GB

3.5.2. Software Requirements:

- Operating system : Windows 10
- Coding Language : Python

3.5.3 Languages Used:

Python:

Python is an object-oriented, high-level programming language. It is built with a highlevel data structure assimilated with dynamic typing and also dynamic binding makes the language easy and used for the Application Development. Moreover, it also acts as a glue language to link the other components and codes for high-level applications. It is a simple and uncomplicated language that is easy to learn and understand syntax, emphasizes readability, and hence reduces the cost of program maintenance. Python assists modules and packages, which helps program modularity and code reuse and reduces prolixity. The Python interpreter along with the extensive standard libraries accessible in binary form without charge for all utmost platforms, as well as can be freely distributed to all.

3.5.4 Libraries:

Pandas:

Pandas is an open-source python library providing virtuous production, high-level data structures as well as data analytic tools for the python programming language. Panda library provides data structures and multiple operations for manipulating numerical values. 4.5.2. OS. The OS library in python dispenses interactive functions that run with the operating system and this OS module in Python's standard utility module. OS library provides a portable method of utilizing operating system-dependent functionality.

Matplotlib

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib Library is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack in python language.

Seaborn

Seaborn is an amazing visualization library for statistical graphics plotting in Python language. It provides beautiful default styles and colour palettes to make statistical plots more attractive. Seaborn library is built on the top of Matplotlib library and is also closely integrated with the data structures from pandas.

NumPy

NumPy is the fundamental package for scientific computing in Python Language. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of the routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much Matplotlib library.

4.DESIGN PROCESS

4.1. System Architecture:

The process of detecting fake online recruitment using machine learning begins with importing essential libraries like Pandas, NumPy, and Scikit-Learn for data processing and model training. The dataset is then loaded and preprocessed, handling missing values and encoding categorical data. In the feature engineering stage, text-based attributes such as job descriptions are transformed using TF-IDF or CountVectorizer. Next, the dataset is split into training and testing sets to evaluate model performance. A machine learning model, such as Random Forest or Logistic Regression, is then trained to classify job postings as real or fake. The model is evaluated using accuracy, precision, recall, and F1-score to measure its effectiveness. Finally, the trained model is used to predict fraudulent job listings, enhancing job portal security and protecting job seekers from scams.

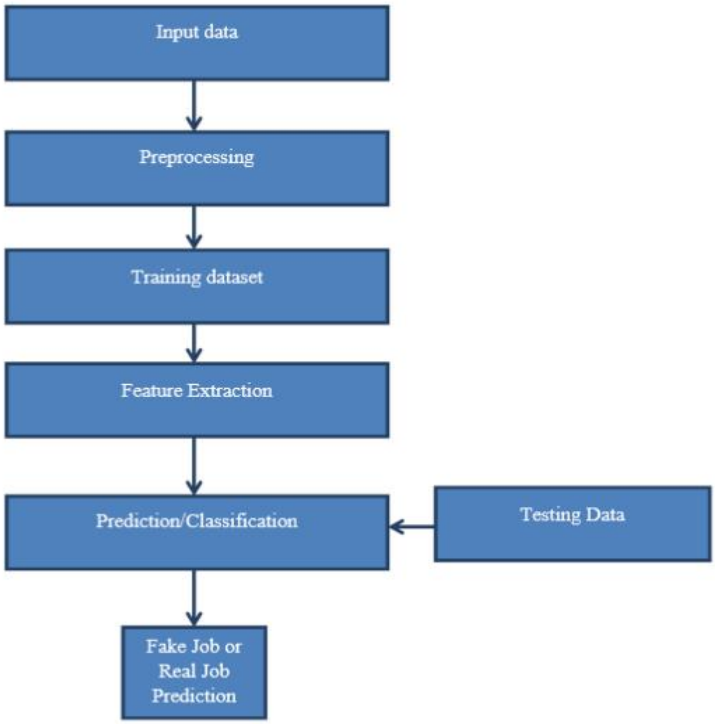


Figure 4.1.1 Basic Architecture of Fake job detection using ML.

Steps:

[1] Import Libraries (Pandas, NumPy, Sklearn, etc.)

[2] Load and Preprocess the Dataset (Handling missing values, encoding categorical data, etc.)

[3] Feature Engineering (TF-IDF, CountVectorizer for text data)

[4] Train-Test Split (Splitting dataset for training and testing)

[5] Build and Train the Model (Using Logistic Regression, Random Forest, etc.)

[6] Evaluate the Model (Using accuracy, precision, recall, F1-score)

[7] Make Predictions (Classifying job postings)

Data Wrangling

In this section of the report will load in the data, check for cleanliness, and then trim and clean given dataset for analysis. Make sure that the document steps carefully and justify for cleaning decisions.

Data collection

The data set collected for predicting given data is split into Training set and Test set. Generally, 7:3 ratios are applied to split the Training set and Test set. The Data Model which was created using Random Forest, logistic, Decision tree algorithms and Support vector classifier (SVC) are applied on the Training set and based on the test result accuracy, Test set prediction is done.

Preprocessing

The data which was collected might contain missing values that may lead to inconsistency. To gain better results data need to be preprocessed so as to improve the efficiency of the algorithm. The outliers have to be removed and also variable conversion need to be done.

Building the classification model

The prediction of wine quality, A high accuracy prediction model is effective because of the following reasons: It provides better results in classification problem.

- It is strong in preprocessing outliers, irrelevant variables, and a mix of continuous, categorical and discrete variables.

- It produces out of bag estimate error which has proven to be unbiased in many tests and it is relatively easy to tune with.

Construction of a Predictive Model Machine learning needs data gathering have lot of past data‘s. Data gathering have sufficient historical data and raw data. Before data pre-processing, raw data can‘t be used directly. It‘s used to preprocess then, what kind of algorithm with model. Training and testing this model working and predicting correctly with minimum errors. Tuned model involved by tuned time to time with improving the accuracy.

4.2. UML Diagrams:

The Unified Modelling Language permits the technologist to specific AN analysis model Mistreatment of the modelling notation that's ruled by a group of grammar linguistics and pragmatic rules. A UML system is diagrammatical mistreatment 5 completely different views that describe the system from a clearly different perspective. Every read is outlined by a group of diagrams that is as follows.

It represents the dynamic of behaviour as elements of the system and portrays the interactions of assortment between varied structural components delineated within the user model and structural model read. Use case Diagrams represent the practicality of the system from a user’s purpose of readings cases are used throughout needs induction and analysis to represent the practicality of the system. Use cases specialize in the behaviour of the system from the external purpose of reading. Actors are external entities that move with the system. Samples of actors embody users like administrators, bank client …etc., or another system like central info.

4.2.1 Use Case Diagram:

Use case diagrams to model the practicality of system treatment actors and use cases. Use cases are services or functions provided by the system to its users.

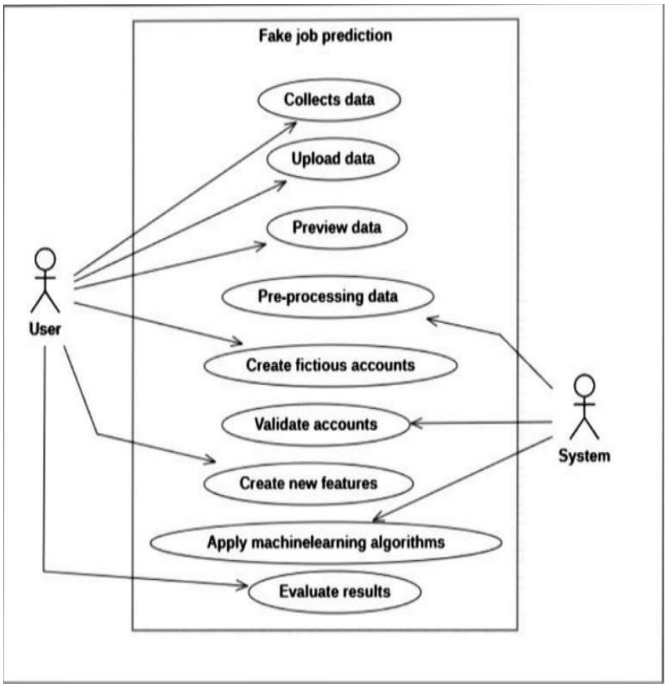


Figure 4.2.1. Use case diagram for detection of fake job detection using machine learning

4.2.2 Class Diagram:

Class diagrams are the backbone of virtually each object-oriented methodology as well as UML. They describe the static structure of a system. Categories represent associate degree abstraction of entities with common characteristics. Associations represent the relationships between categories.

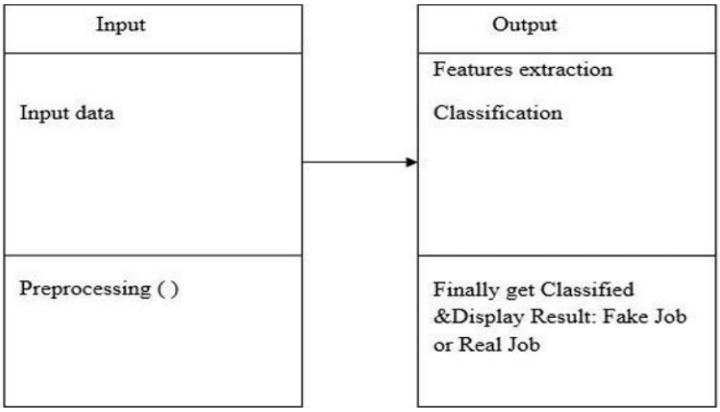
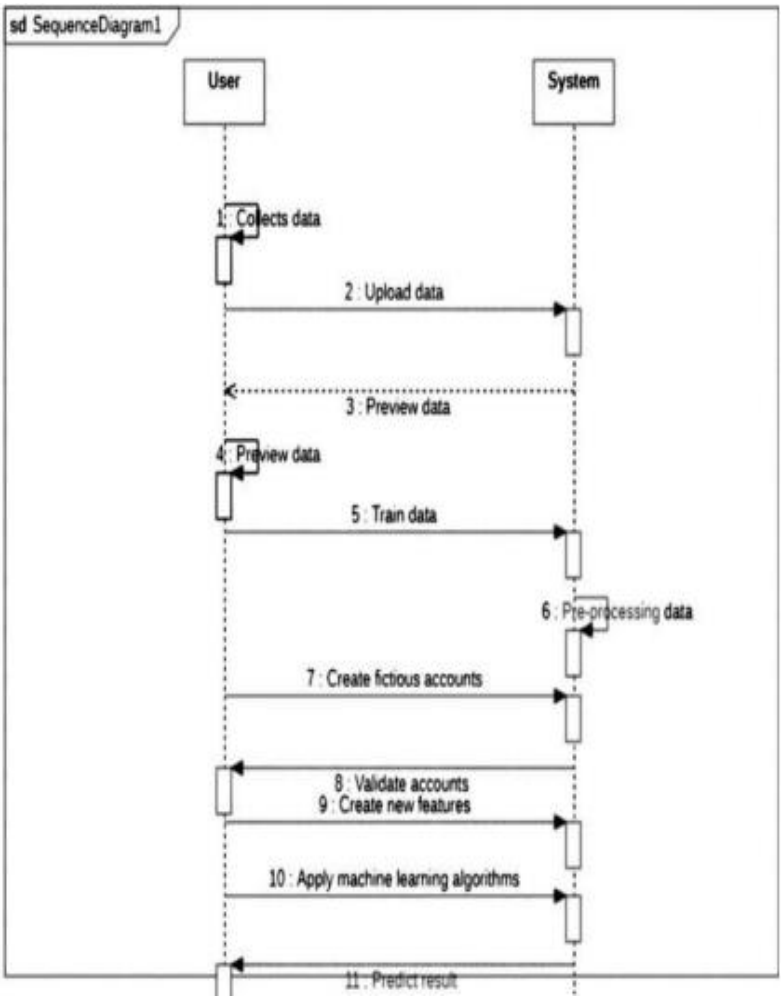


Figure 4.2.2. Class Diagram for detection of fake online recruitment using machine learning

4.2.3. Sequence Diagram:

A sequence diagram or system sequence diagram (SSD) shows process interactions arranged in time sequence in the field of software engineering. It depicts the processes

involved and the sequence of messages exchanged between the processes needed to carry out the functionality. Sequence diagrams are typically associated with use case realizations in the 4+1 architectural view model of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.



4.2.3. Sequence Diagram for detection of fake online recruitment using machine learning

5. METHODOLOGY

The **methodology** for detecting fake online job postings using machine learning involves several key steps to ensure accurate classification and fraud detection.

1) **Data Collection** – Job postings are gathered from various sources, including online job portals, company websites, and recruitment platforms. This dataset includes details such as job descriptions, company names, recruiter emails, and salary information.
2) **Data Preprocessing** – The collected data is cleaned by handling missing values, removing duplicates, and encoding categorical variables. Textual data, such as job descriptions, is processed using Natural Language Processing (NLP) techniques to extract relevant features.
3) **Feature Engineering** – Important features are extracted using TF-IDF (Term Frequency-Inverse Document Frequency) or CountVectorizer to convert text data into numerical form. Other relevant attributes, such as recruiter email domains and salary ranges, are also considered.
4) **Train-Test Split** – The dataset is divided into training and testing sets, typically in an 80-20 or 70-30 ratio, to train the model and evaluate its performance on unseen data.
5) **Model Selection and Training** – Machine learning algorithms like Random Forest, Logistic Regression, or Support Vector Machines (SVM) are used for classification. The model learns patterns from training data and distinguishes between fake and genuine job postings.
6) **Model Evaluation** – The trained model is evaluated using performance metrics such as accuracy, precision, recall, and F1-score to assess its effectiveness in detecting fraudulent job listings.
7) **Prediction and Deployment** – The final model is used to predict new job postings as real or fake. The system can be deployed on job portals or integrated into recruitment platforms to provide real-time fraud detection.

This methodology ensures an efficient and automated approach to identifying fake online job postings, enhancing trust and security in the recruitment process.

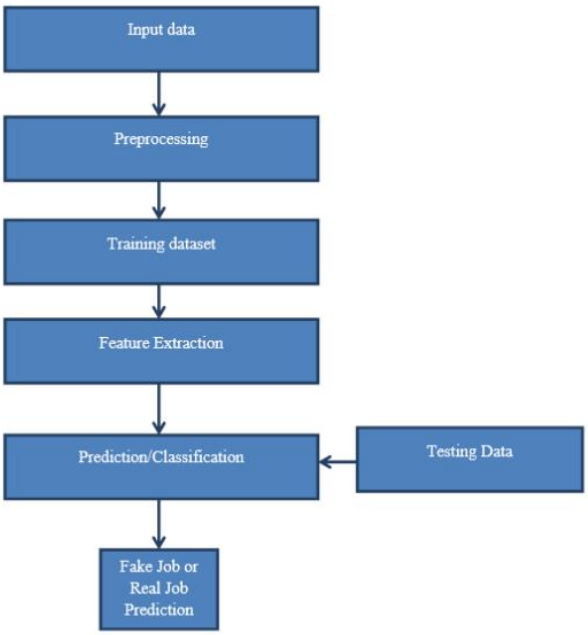


Figure 5.a Basic workflow of system

5.1. Data Gathering:

The data collection process is crucial for building an accurate machine learning model to detect fake job postings. The dataset is gathered from various sources, including online job portals (e.g., LinkedIn, Indeed, Glassdoor), company websites, and recruitment platforms. Web scraping techniques using BeautifulSoup and Selenium can be employed to extract job details automatically. Additionally, publicly available datasets containing labeled real and fake job postings can be used for training the model.

The collected dataset typically includes job descriptions, company names, recruiter emails, job titles, salary information, posting dates, and job locations. Fraudulent job listings often have suspicious recruiter details, unrealistic salary offers, or vague job descriptions. Ensuring data diversity is important to make the model robust against different types of scams.

After gathering data, preprocessing steps such as removing duplicates, handling missing values, and encoding categorical variables are performed. This ensures that the dataset is clean and structured for effective machine learning model training.

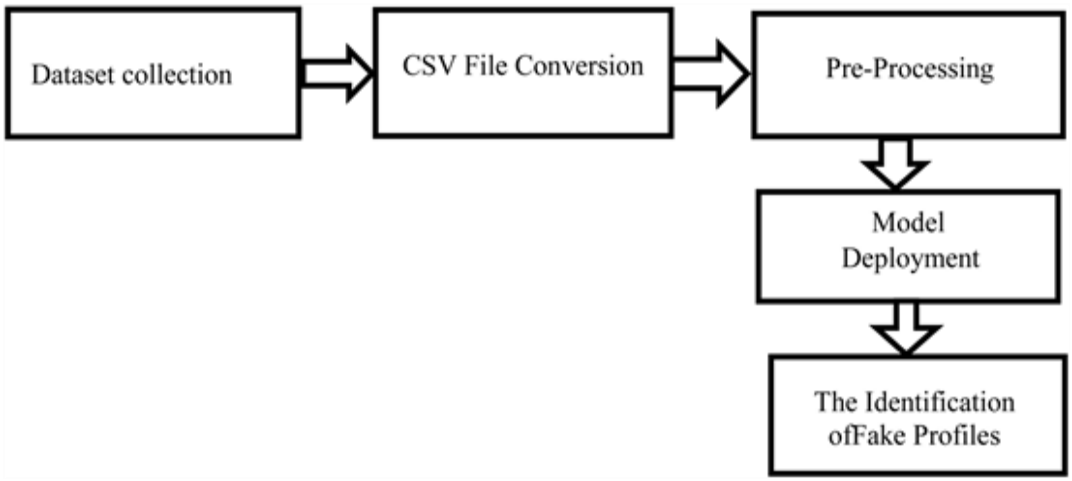


Figure 5.1.1 Data collection process

To get started, we'll need to do the following things:

Step 1: Create a Kaggle Account

- ✓ Go to http://Kaggle.com/ and sign up for an account if you don't already have one.
- ✓ Fill out the registration form with your email address, username, and password.

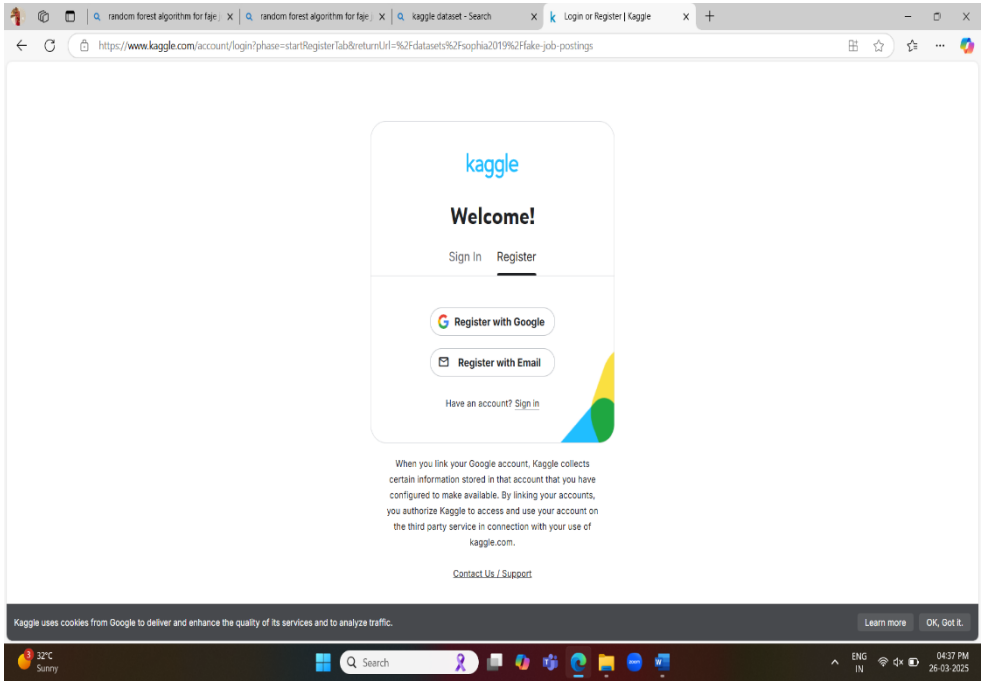


Figure 5.1.2 Login to Kaggle.com

Step 2: Find the Dataset

- ✓ Log in to your Kaggle account.

- ✓ Click on the "Datasets" tab at the top of the page.
- ✓ Search for the dataset you want to download using the search bar.
- ✓ Click on the dataset title to go to its page.

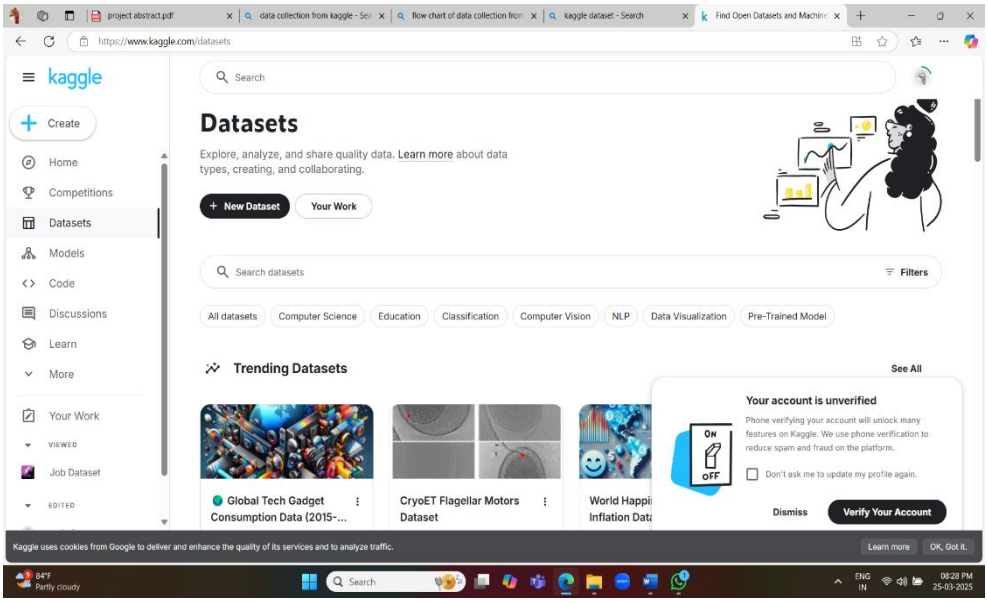


Figure 5.1.3 Finding the Dataset

Step 3: Download the Dataset

- ✓ On the dataset page, click on the "Download" button.
- ✓ Select the file format you want to download the dataset in (e.g., CSV, JSON, etc.).
- ✓ Choose the specific files you want to download (e.g., train.csv, test.csv, etc.).
- ✓ Click on the "Download" button to start the download process.

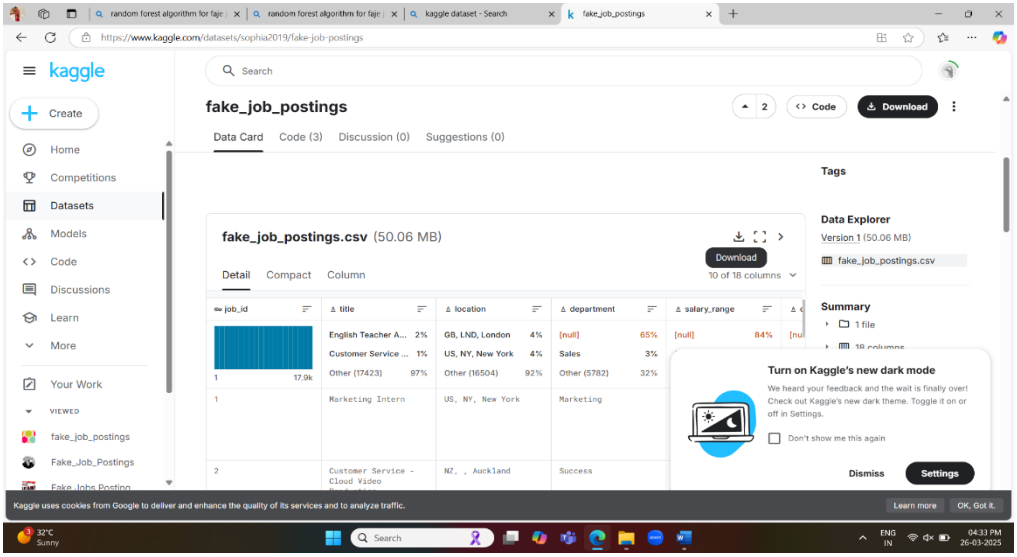


Figure 5.1.4 Downloading the DataSet

Step 4: Download Using Kaggle API (Optional)

- ✓ If you want to download the dataset using the Kaggle API, you'll need to install the Kaggle API library using pip: pip install Kaggle
- ✓ Create a new API token on the Kaggle website by going to your account settings and clicking on "API".
- ✓ Save the API token to a file named kaggle.json in the .kaggle directory.
- ✓ Use the Kaggle API to download the dataset using the following command: kaggle datasets download <dataset-owner>/<dataset-name>

Step 5: Unzip the Dataset (If Necessary)

- ✓ If the dataset is compressed in a ZIP file, you'll need to unzip it before you can use
- ✓ Use a tool like 7-Zip or WinRAR to unzip the file.

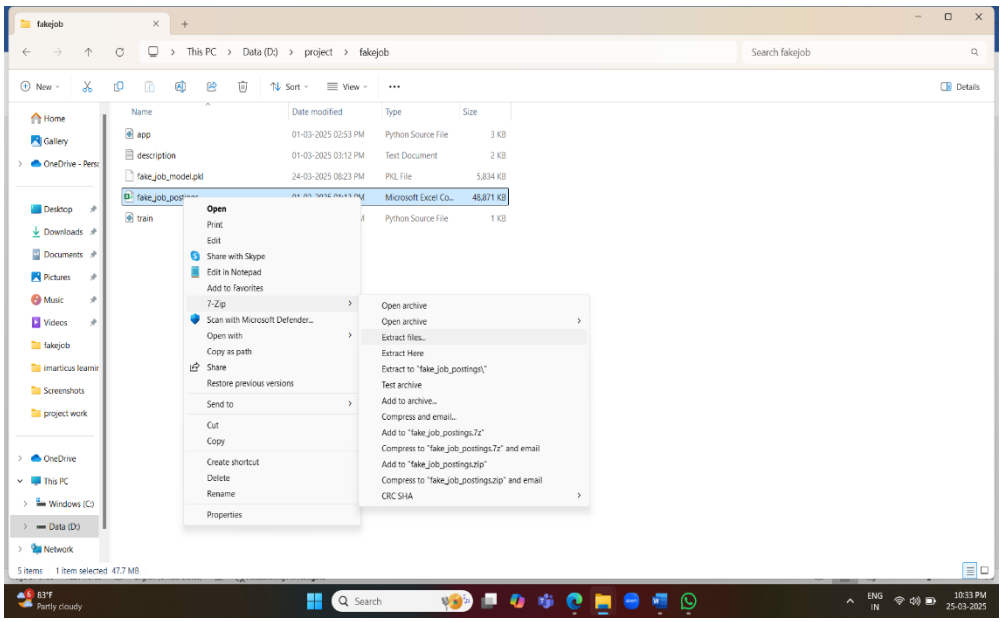


Figure 5.1.5 Extracting the Dataset

That's it! You should now have the dataset downloaded and ready to use.

5.2. Data Preparation:

Step 1: Data Collection

- ✓ Gather data: Collect a large dataset of job postings from various sources, including job boards, company websites, and social media platforms.
- ✓ Label the data: Label each job posting as either "fake" or "legitimate" based on certain criteria, such as:

- Job posting contains suspicious keywords or phrases.
- Job posting requires payment or personal information.
- Job posting has poor grammar or spelling.
- Job posting is duplicated or similar to other postings.

5.2.1. Data Preprocessing:

Data Preprocessing Steps:

Here are the common data preprocessing steps:

1. Handling missing values: Decide how to handle missing values, such as imputing them with mean, median, or mode values.

2. Data normalization: Scale numeric data to a common range, usually between 0 and 1, to prevent feature dominance.

3. Data transformation: Transform data into a suitable format, such as converting categorical variables into numerical variables.

4. Handling outliers: Identify and remove outliers or noisy data points that can negatively impact model performance.

5. Feature scaling: Scale features to have similar magnitudes to prevent feature dominance.

6. Encoding categorical variables: Convert categorical variables into numerical variables using techniques such as one-hot encoding or label encoding.

Data Preprocessing Techniques:

Here are some common data preprocessing techniques:

1. Tokenization: Split text data into individual words or tokens.

2. Stopword removal: Remove common words like "the", "and", "a", etc. that do not add much value to the text.

3. Stemming or Lemmatization: Reduce words to their base form so that words like "running", "runs", and "runner" are treated as the same word.

4. Vectorization: Convert text data into numerical vectors using techniques such as bag-of-words or TF-IDF.

5.2.2 Data Cleaning:

Data cleaning, also known as data cleansing or data scrubbing, is the process of identifying and correcting errors, inconsistencies, and inaccuracies in a dataset.

Data cleaning is important because:

- ✓ **Improves data quality**: Data cleaning helps to ensure that the data is accurate, complete, and consistent.
- ✓ **Reduces errors**: Data cleaning helps to identify and correct errors, which can lead to incorrect insights and decisions.
- ✓ **Increases efficiency**: Data cleaning helps to reduce the time and effort required to analyze and process data.
- ✓ **Enhances decision-making**: Data cleaning helps to ensure that decisions are based on accurate and reliable data.

```
# Load dataset

def load_data(uploaded_file):

    if uploaded_file is not None:

        df = pd.read_csv(uploaded_file)

        df = df[['title', 'company_profile', 'description', 'requirements', 'fraudulent']].fillna('')

        return df

    return None
```

5.3. Creation of Fake Job Detection:

5.3.1. Feature Extraction:

Feature extraction is the process of selecting and transforming raw data into features that are more suitable for modeling and analysis.

Here are some common types of features used in fake job detection:

- ➢ Text features: Extracted from job posting text, such as:
  - Bag-of-words (BoW)
  - Term Frequency-Inverse Document Frequency (TF-IDF)

- Named Entity Recognition (NER)

- Metadata features: Extracted from job posting metadata, such as:
  - Job title
  - Company name
  - Location
  - Job type (full-time, part-time, etc.)
- Behavioral features: Extracted from user behavior, such as:
  - User engagement (likes, shares, etc.)
  - User feedback (ratings, reviews, etc.)
- Network features: Extracted from network analysis, such as:
  - Social network analysis (SNA)
  - Graph-based features

Feature selection:

```
def train_model(df):
  X_train, X_test, y_train, y_test = train_test_split(
      df[['title', 'company_profile', 'description', 'requirements']],
      df['fraudulent'],
      test_size=0.2,
      random_state=42
  )
```

5.3.2. Algorithm:

Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but

their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

The first algorithm for random decision forests was created in 1995 by Tin Kam Ho[1] using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered "Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.).The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho[1] and later independently by Amit and Geman[13] in order to construct a collection of decision trees with controlled variance.

Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

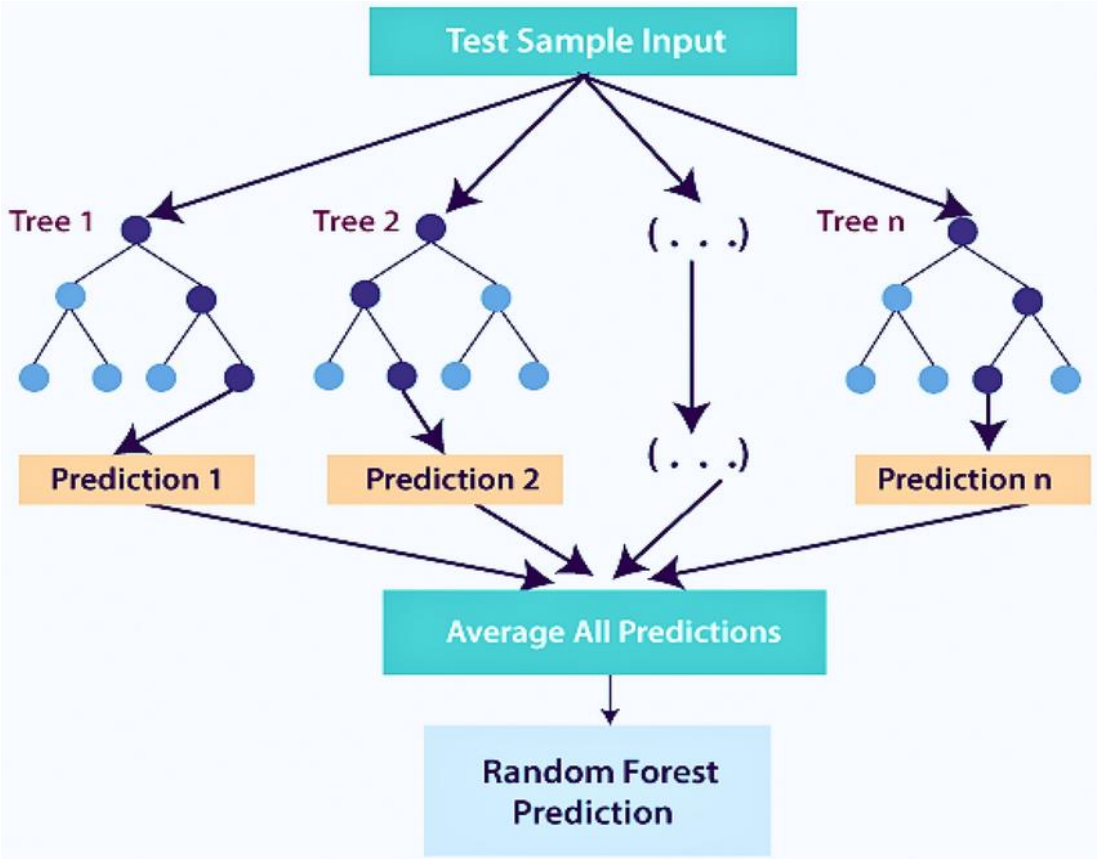


Figure 5.1.6 Random Forest Algorithm

5.4. Fake Job Detection based on features:

Overview of fake job detection based on features of title, company_profile, description, and requirements:

1. Title:

- Length: Number of characters in the title.

- Keywords: Presence of keywords like "urgent", "immediate", "hiring now", etc.

- Punctuation: Presence of excessive punctuation marks.

2. Company Profile:

- Length: Number of characters in the company profile.

- Keywords: Presence of keywords like "leading", "top", "best", etc.

- Grammar: Presence of grammatical errors.

3. Description:

- Length: Number of characters in the description.

- Keywords: Presence of keywords like "competitive salary", "benefits", "opportunity", etc.

- Tone: Tone of the description (e.g., formal, informal, enthusiastic, etc.).

4. Requirements:

- Length: Number of characters in the requirements.

- Keywords: Presence of keywords like "experience", "skills", "education", etc.

- Specificity: Level of specificity in the requirements (e.g., specific skills, qualifications, etc.).

5.4.1. About Streamlit:

Streamlit is an open-source Python library that allows you to create and share beautiful, custom web apps for data science and machine learning in minutes.

Key Features:

1. Easy to use: Streamlit's API is simple and intuitive, making it easy to create web apps without extensive web development knowledge.

2. Fast development: Streamlit's caching mechanism and automatic widget generation enable fast and iterative development.

3. Customizable: Streamlit allows you to customize the layout, appearance, and behavior of your app using various themes, widgets, and configuration options.

4. Interactive: Streamlit supports interactive visualizations, allowing users to explore and manipulate data in real-time.

5. Sharing: Streamlit apps can be shared via URLs, making it easy to collaborate with others or deploy apps to production.

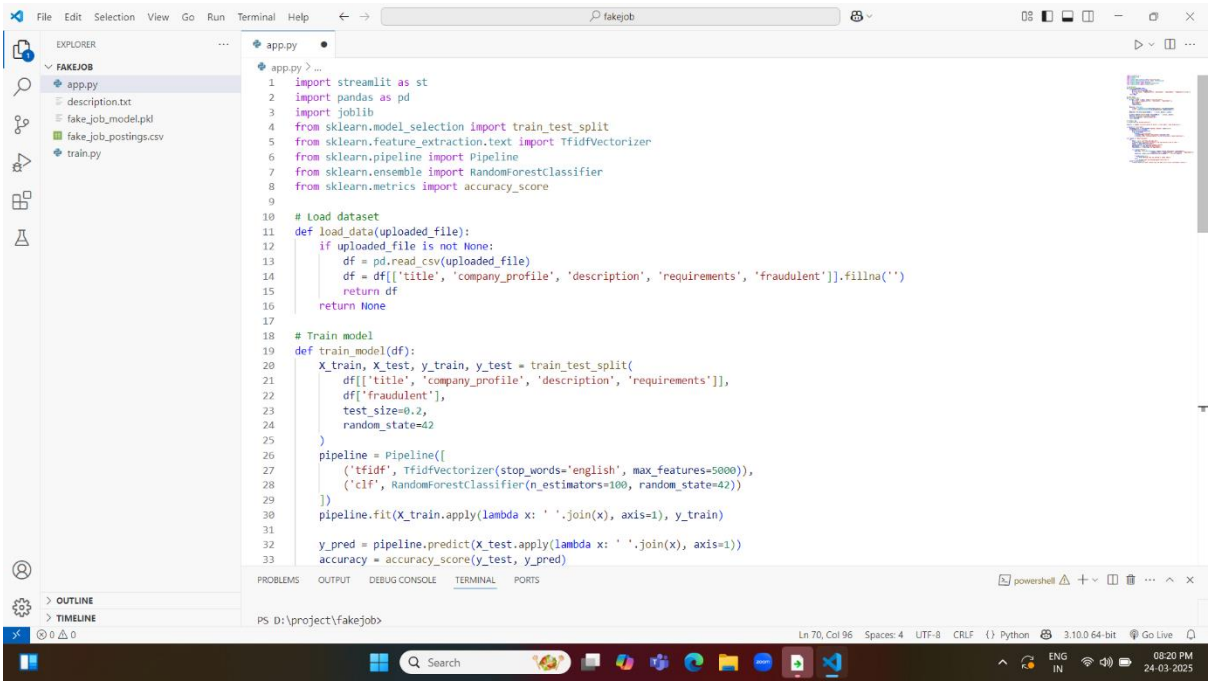


Figure 5.1.7 Visual Studio Code

PYTHON CODE:

import streamlit as st

import pandas as pd

import joblib

from sklearn.model_selection import train_test_split

from sklearn.feature_extraction.text import TfidfVectorizer

```
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
# Load dataset
def load_data(uploaded_file):
    if uploaded_file is not None:
        df = pd.read_csv(uploaded_file)
        df = df[['title', 'company_profile', 'description', 'requirements', 'fraudulent']].fillna('')
        return df
    return None
# Train model
def train_model(df):
    X_train, X_test, y_train, y_test = train_test_split(
        df[['title', 'company_profile', 'description', 'requirements']],
        df['fraudulent'],
        test_size=0.2,
        random_state=42
    )
    pipeline = Pipeline([
        ('tfidf', TfidfVectorizer(stop_words='english', max_features=5000)),
        ('clf', RandomForestClassifier(n_estimators=100, random_state=42))
    ])
    pipeline.fit(X_train.apply(lambda x: ' '.join(x), axis=1), y_train)
    y_pred = pipeline.predict(X_test.apply(lambda x: ' '.join(x), axis=1))
```

```
    accuracy = accuracy_score(y_test, y_pred)
    joblib.dump(pipeline, "fake_job_model.pkl")
    return accuracy
# Streamlit App
st.title("Fake Job Posting Detector")
option = st.sidebar.selectbox("Choose an option", ["Train Model", "Make Prediction"])
if option == "Train Model":
    uploaded_file = st.file_uploader("Upload a dataset", type=["csv"])
    if uploaded_file is not None:
        df = load_data(uploaded_file)
        if df is not None:
            accuracy = train_model(df)
            st.write(f"Model trained with accuracy: {accuracy:.2f}")
            st.success("Model training complete. You can now proceed to make predictions.")
elif option == "Make Prediction":
    try:
        model = joblib.load("fake_job_model.pkl")
        st.write("Enter job details to check if the job posting is real or fake.")
        title = st.text_input("Job Title")
        company_profile = st.text_area("Company Profile")
        description = st.text_area("Job Description")
        requirements = st.text_area("Job Requirements")
            if st.button("Predict"):
            input_data = pd.DataFrame([[title, company_profile, description, requirements]],
```

```
                columns=['title', 'company_profile', 'description', 'requirements'])
        prediction = model.predict(input_data.apply(lambda x: ' '.join(x), axis=1))
        if prediction[0] == 1:
            st.error("Warning: This job posting is likely fake!")
        else:
            st.success("This job posting appears to be real.")
except FileNotFoundError:
    st.error("Model not found. Please train the model first in the 'Train Model' section.")
```

5.4.2. Code Implementation:

- Run the above code in VS code and enter below command:
  "Streamlit run.py" as shown in below image

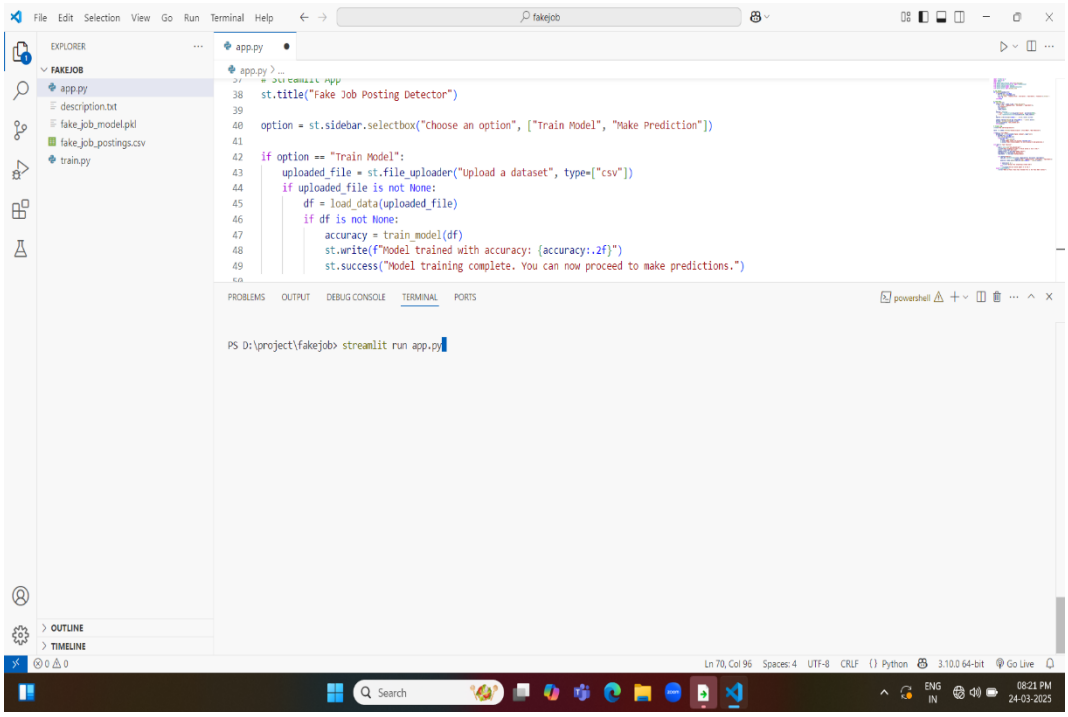


Figure 5.1.8 Code Implementation

- After running the above command the page will display as:

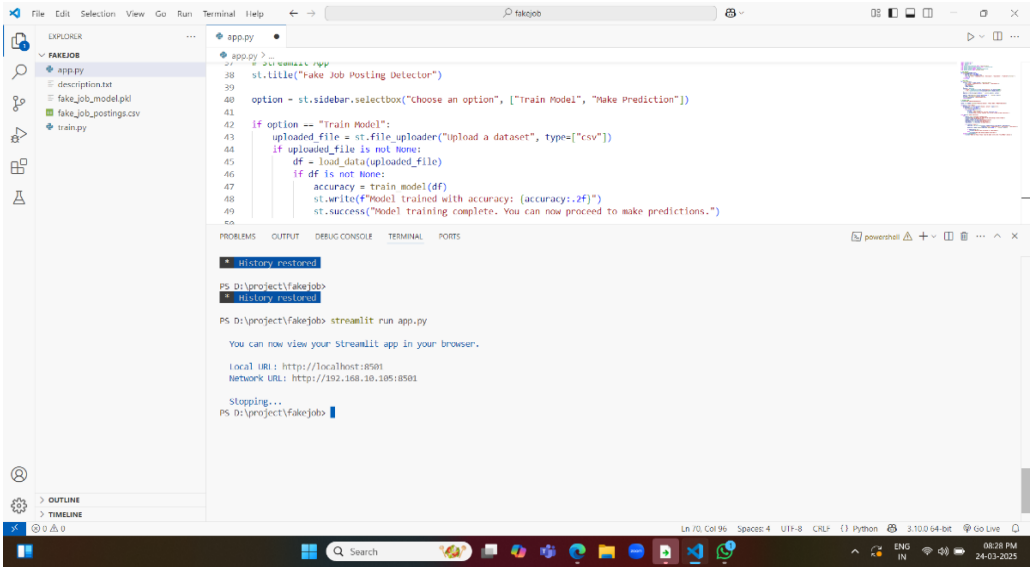


Figure 5.1.9 Executing the code

Now, the code is ready to predict the ouputs. Lets us know whether the job is “REAL” or “FAKE”.

6.RESULTS AND DISCUSSIONS

Results:

The interface is divided into two sections. On the left sidebar, there is a dropdown menu labeled "Choose an option," offering two choices: "Train Model" (which is currently selected) and "Make Prediction." Below the dropdown, there is a button to execute the selected action. The main content area features a file upload section where users can either drag and drop a file or browse files manually. The upload is limited to CSV files with a maximum size of 200MB. The application seems designed to allow users to train a model or make predictions to detect fake job postings. There is also a "Deploy" button in the top-right corner, suggesting deployment or sharing capabilities. The interface is clean and user-friendly, facilitating the process of uploading datasets and running the desired tasks.

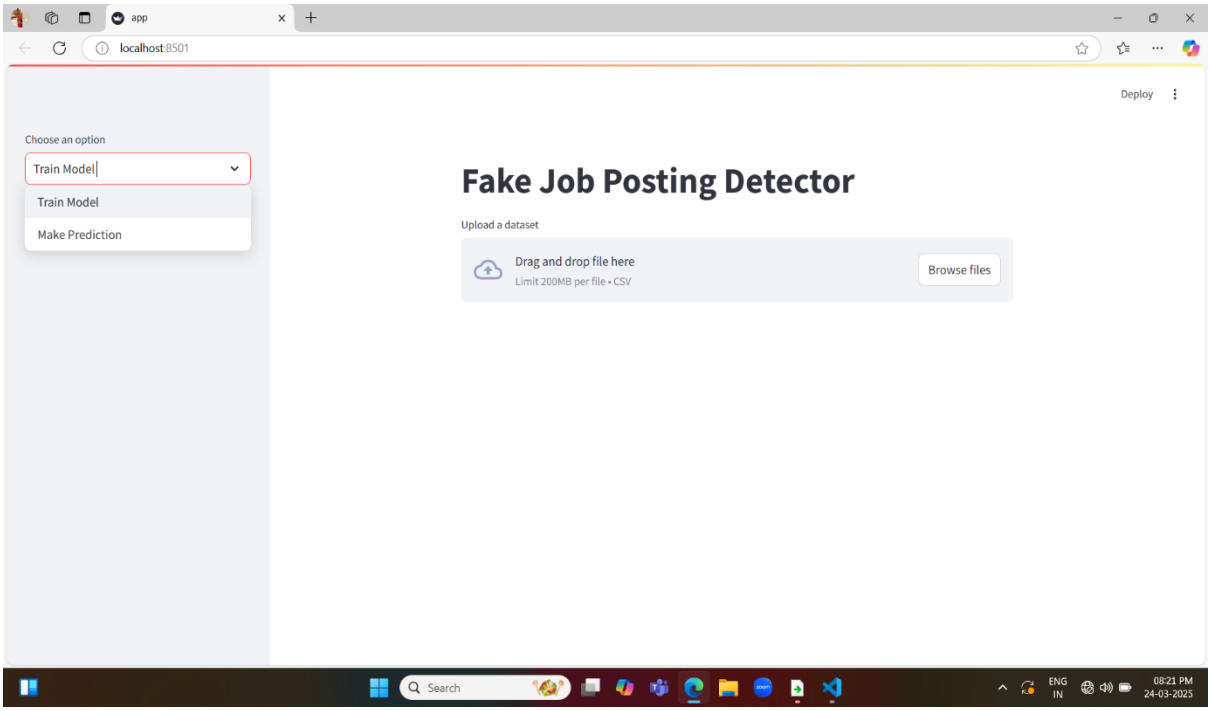


Figure 6. 1: Interface of Web Browser

- When the "Train Model" option is selected from the dropdown menu in the sidebar, the interface prompts the user to upload a dataset in CSV format, with a file size limit of 200MB. The upload section provides two options: dragging and dropping the file or using the "Browse files" button to select a file manually. Once a dataset is uploaded, the application is expected to process the data and initiate the training of a machine learning model to detect fake job postings. This option suggests that

the uploaded data will be used to train and fine-tune the model for improved accuracy in identifying fraudulent job listings.

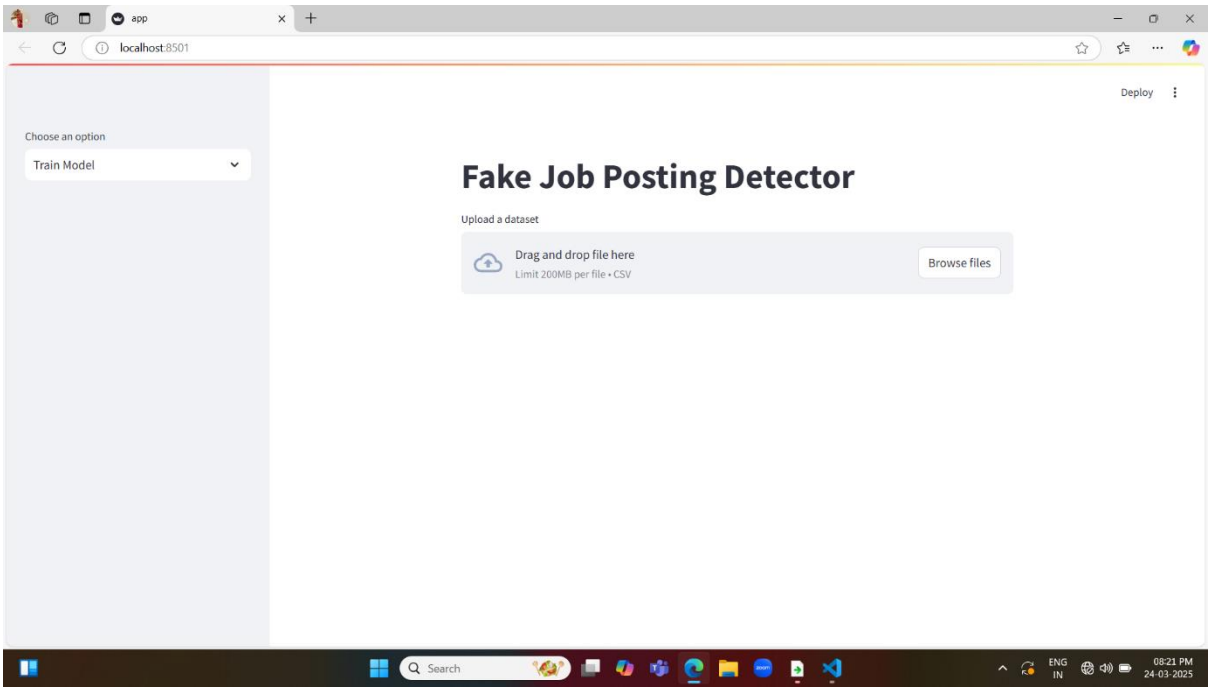


Figure 6. 2: Opening of streamlit browser

- The model achieved an accuracy of 0.98, meaning it correctly identified 98% of the job postings as either genuine or fake during the training or validation process. This high accuracy suggests that the model is well-trained and effective in distinguishing between legitimate and fraudulent job postings. However, while a 0.98 accuracy score indicates strong performance, it is essential to be cautious, as extremely high accuracy can sometimes be a sign of overfitting, where the model performs exceptionally well on training data but may not generalize as effectively to unseen or real-world data. Further evaluation, such as testing on new data or using metrics like precision, recall, and F1-score, can help ensure that the model maintains consistent performance in practical applications.

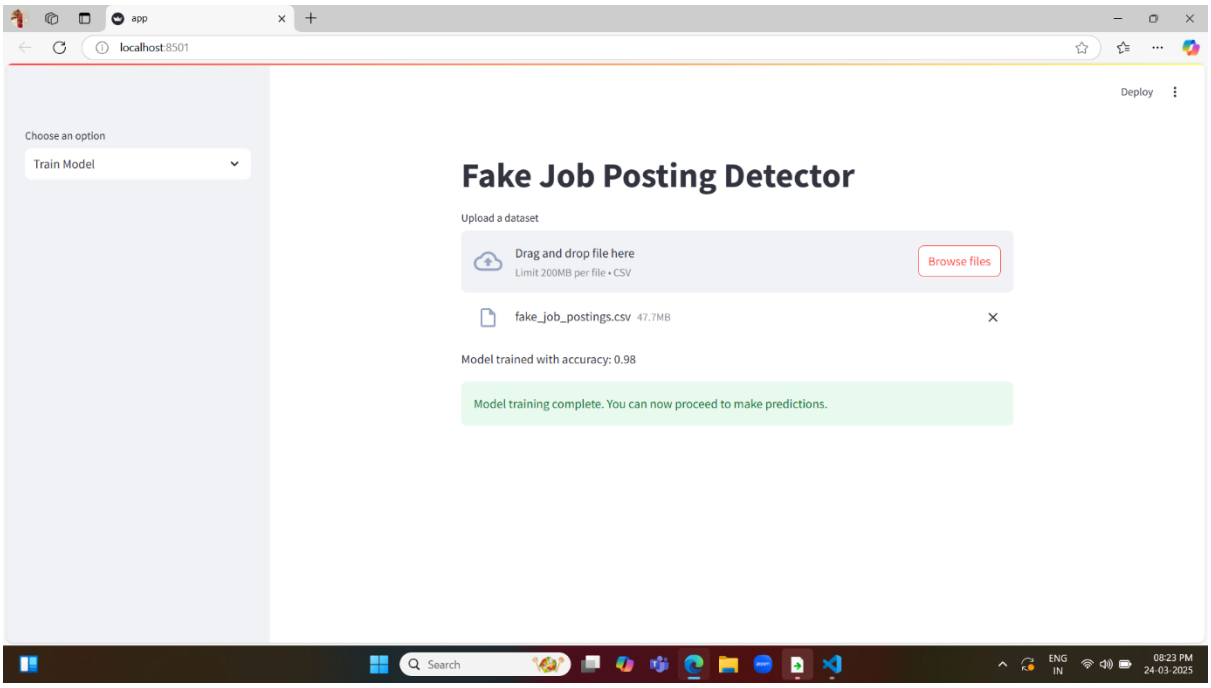


Figure 6. 3: Accuracy of Prediction

When the "Make Prediction" option is selected from the dropdown menu in the sidebar, the interface changes to allow the user to input job details to determine if a job posting is real or fake. The main content area displays a form with four text input fields labeled:

- **Job Title** – For entering the title of the job.
- **Company Profile** – For providing information about the company.
- **Job Description** – To describe the responsibilities and tasks associated with the job.
- **Job Requirements** – To specify the qualifications and skills needed for the position.

At the bottom of the form, there is a "Predict" button, which processes the entered data and provides a prediction on whether the given job posting is legitimate or fraudulent. This interface is designed to allow users to easily test the trained model on new job postings by providing relevant job details.

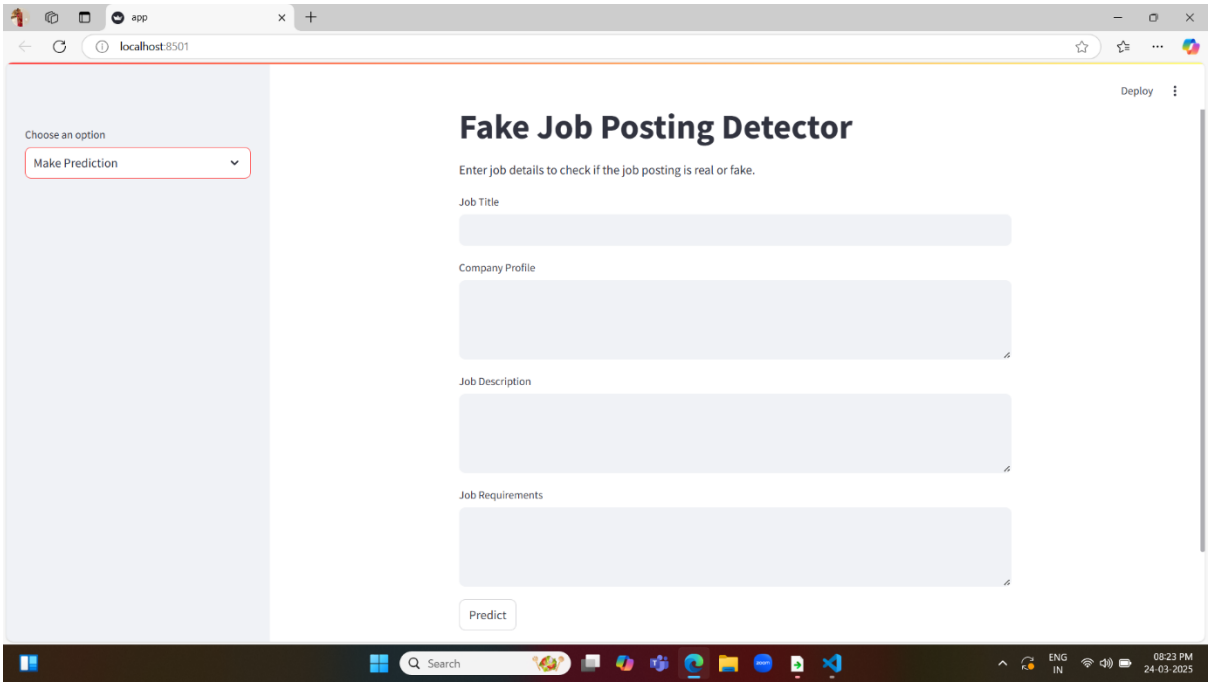


Figure 6. 4: Make Prediction

After providing the necessary input values, such as the Job Title, Company Profile, Job Description, and Job Requirements, and clicking the "Predict" button, the model processes the information and evaluates it using the trained algorithm. The model analyzes various features in the input data, comparing them to patterns learned during training to identify characteristics associated with real and fake job postings. Upon completing the analysis, the application displays the result, stating that **"This job posting appears to be real."**

This outcome suggests that the provided job details match the patterns of legitimate job postings, indicating that the listing is authentic. The model's ability to make this prediction is based on a high accuracy score (0.98) achieved during the training phase, making it highly reliable in identifying fraudulent or genuine listings. The prediction feature not only streamlines the validation process but also helps users make informed decisions about job postings by leveraging advanced machine learning techniques. Additionally, this functionality can be beneficial for job seekers, recruiters, and platforms aiming to safeguard against fake job advertisements, enhancing trust and security in the hiring process.

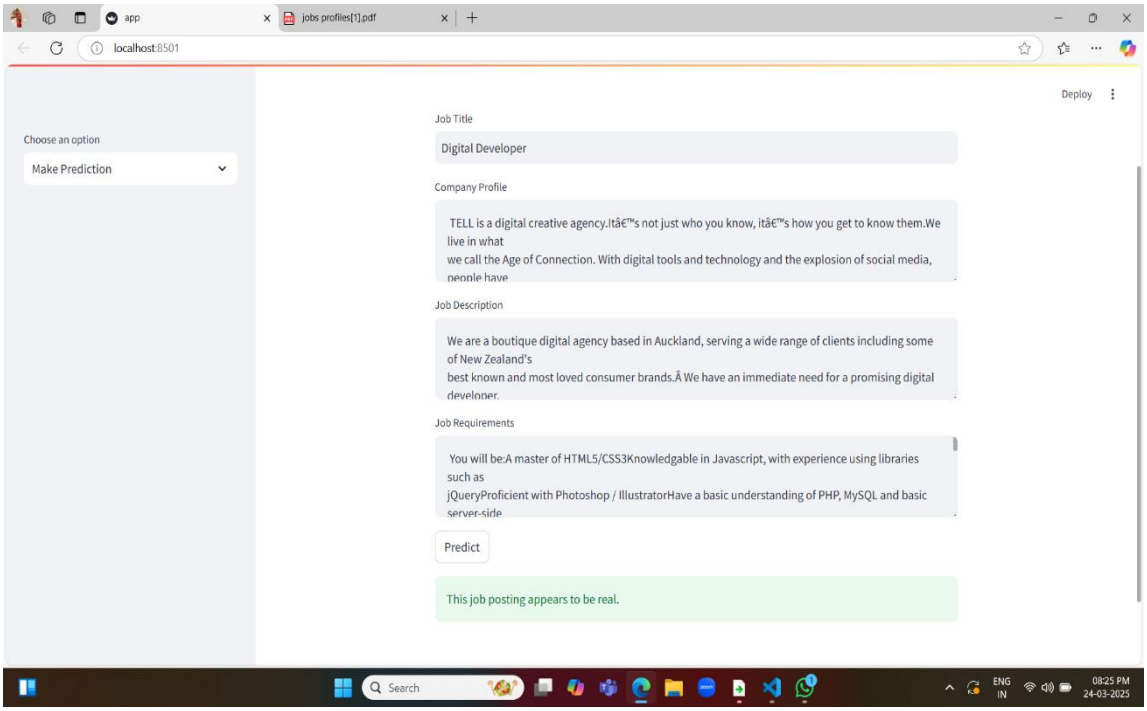


Figure 6. 5: Predict Output 1

After providing a different set of input values in the fields for Job Title, Company Profile, Job Description, and Job Requirements, and clicking the "Predict" button, the model processes the new information using the trained algorithm. It analyzes the input data and compares it to the patterns learned during training, which helps it identify potential indicators of fraudulent job postings. Following this analysis, the application displays a warning message stating, **"Warning: This job posting is likely fake!"**

This result suggests that the provided job details exhibit characteristics commonly associated with fake or fraudulent job postings. The warning indicates that the model has identified suspicious patterns, such as vague descriptions, unrealistic requirements, or missing critical information, which are often linked to fake listings. Since the model achieved a high accuracy of 0.98 during training, this prediction is likely reliable.

This feature serves as a valuable safeguard for users, helping them identify potentially harmful or misleading job listings before engaging with them. By issuing a clear warning, the application empowers job seekers, recruiters, and platforms to take caution and avoid falling victim to fraudulent schemes, thereby enhancing safety and trust in the hiring process.

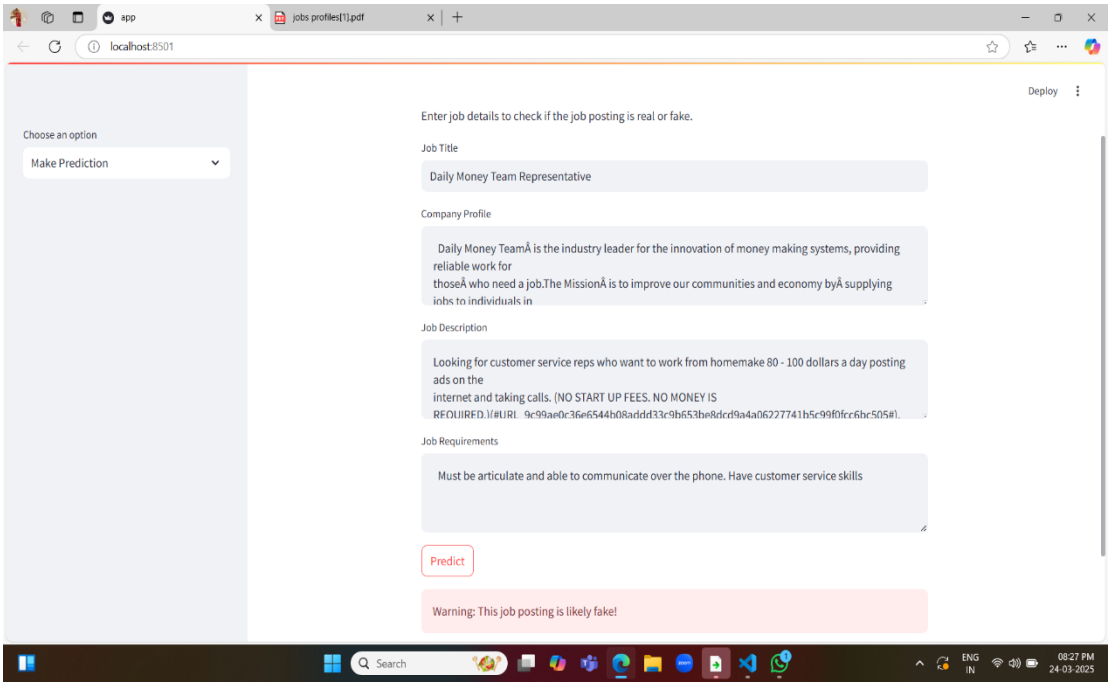


Figure 6. 6: Predict Output 2

To stop the Streamlit server running in VS Code, you need to press Ctrl + C in the terminal where the application is running. When you launch a Streamlit app, it starts a local server, typically accessible at http://localhost:8501. The terminal displays logs indicating that the application is running and provides the URL to access the web interface.

To terminate the server:

1. Click on the terminal window in VS Code where the Streamlit app is running.

2. Press Ctrl + C on your keyboard.

3. You will usually be prompted with a confirmation asking if you want to terminate the process. Press Y to confirm and stop the server.

Once stopped, the local server will shut down, and the web interface will no longer be accessible until the server is restarted by running the streamlit run <app_name.py> command again. This process is useful when you need to make changes to the application code or reset the server for any reason.

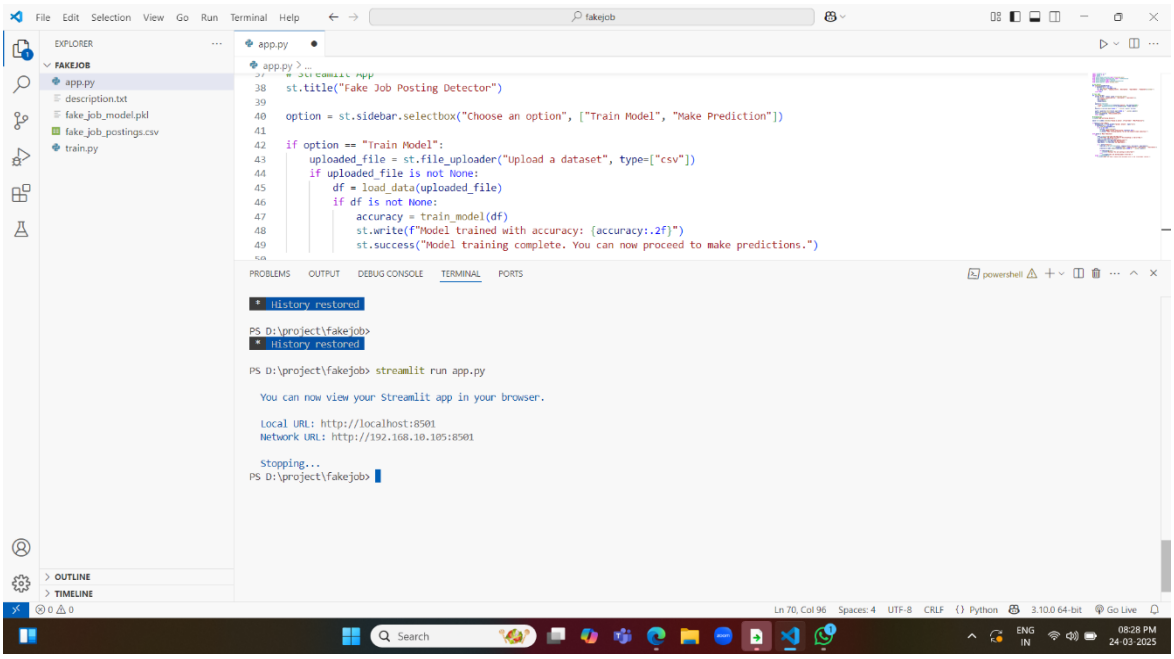


Figure 6. 7: Stopping Streamlit App

This error typically occurs when the Streamlit server is terminated, often by pressing Ctrl + C in the terminal where the application was running. Since the server is no longer active, the frontend (web interface) cannot communicate with the backend to process requests or generate responses. To resolve this issue, the user needs to restart the application by running the Streamlit command in the terminal, specifying the name of the script (e.g., app.py or your_script.py). Once the server is restarted, the application will become accessible again through the same localhost URL.

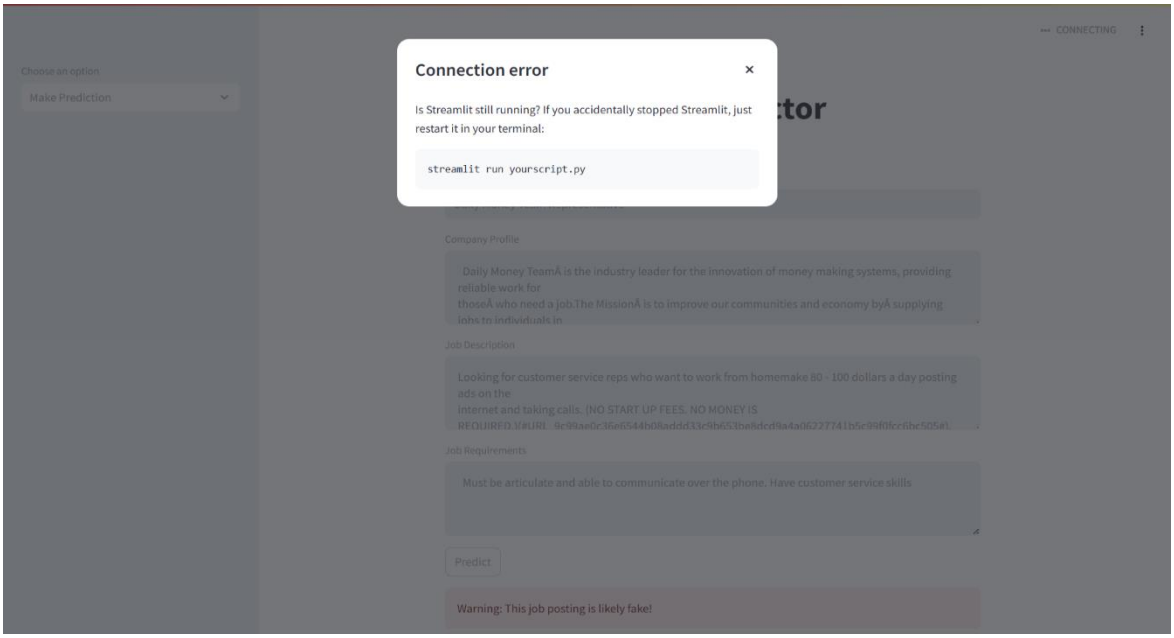


Figure 6. 8: Connection Error

Test Cases:

Test cases help evaluate the accuracy and reliability of the machine learning model in detecting fake job postings. Below are different types of test cases:

1. **Valid Job Posting (Genuine Case)**
   - Input: Job description with clear details, company name, proper recruiter email, realistic salary.
   - Expected Output: Classified as Real**.**
2. **Fake Job Posting (Fraudulent Case)**
   - Input: Job description with vague details, unverified company name, suspicious email (e.g., free domains like Gmail/Yahoo), unrealistic salary.
   - Expected Output: Classified as Fake.
3. **Missing Information**
   - Input: Job posting with missing salary details or recruiter information.
   - Expected Output: Model should still classify based on available data without errors.
4. **Spam Keywords in Job Description**
   - Input: Job post containing spam phrases like "Work from home and earn $500 daily" or "No experience required, instant approval."
   - Expected Output: Classified as Fake.
5. **Imbalanced Data Handling**
   - Input: Dataset with very few fake job postings compared to real ones.
   - Expected Output: Model should maintain accuracy without bias toward real jobs.
6. **Edge Case** – Newly Posted Job with No Reviews
   - Input: Job posting from a newly registered recruiter with limited history.
   - Expected Output: Model should classify based on other features rather than historical data alone.

These test cases help ensure the model performs well across different scenarios and accurately detects fraudulent job listings.

7.CONCLUSION & FUTURE SCOPE

Conclusion:

The "Fake Job Posting Detector" is an effective web application designed using Streamlit to help users identify fraudulent job postings by leveraging machine learning models. The application offers an intuitive and seamless interface that simplifies the process of training a model and making predictions based on job data. By providing two main options—"Train Model" and "Make Prediction"—the application ensures that users can easily train the model with their own dataset and assess new job postings for authenticity.

The "Fake Job Posting Detector" is a valuable tool that contributes to improving trust and security in the job market by protecting users from fraudulent job listings. Its high accuracy, combined with a user-friendly interface, makes it an effective solution for job seekers, recruiters, and hiring platforms. However, to maintain and enhance the model's performance, periodic retraining with updated datasets is recommended. Additionally, incorporating advanced techniques such as natural language processing (NLP) and sentiment analysis could further refine the model's ability to detect subtle indicators of fake job postings. Overall, the application provides a robust mechanism for safeguarding users against job-related scams and enhancing transparency in the hiring process.

Future Scope:

The future scope of the Fake Job Posting Detector is promising, with several opportunities to enhance its capabilities, improve accuracy, and expand its application across various domains. As job scams become increasingly sophisticated, incorporating advanced technologies and methodologies can significantly strengthen the model's ability to detect fraudulent job postings more effectively.

To keep up with the ever-evolving landscape of online job scams, continuous model retraining with updated and diverse datasets is essential. Fraudulent patterns change over time, and models trained on outdated data may struggle to detect new types of scams. Establishing an automated system to retrain the model periodically using recent job posting data will ensure that the model remains relevant and accurate.

The application can also benefit from real-time monitoring and alert systems that flag suspicious job postings as soon as they are detected. By integrating APIs with job boards and online platforms, the model can scan and assess job postings in real-time, alerting users or administrators about potentially fraudulent listings. This proactive approach can prevent job seekers from engaging with fake job offers, thereby enhancing security and trust in the hiring process.

Expanding the application to support multiple languages and international job markets is another future direction. As fraudulent job postings are not limited to a single region or language, enabling the model to analyze job listings in different languages and cultural contexts will make it more versatile and globally applicable.

Another area of growth is user feedback and reinforcement learning. By allowing users to report suspicious job postings and incorporating this feedback into the model, the system can learn from real-world scenarios and improve its accuracy over time. This feedback loop can refine the model's ability to distinguish between legitimate and fraudulent listings more effectively.

Lastly, incorporating explainable AI (XAI) techniques will make the model's predictions more transparent and interpretable for users. By providing explanations for why a job posting is classified as real or fake, the application can build greater trust and allow users to make informed decisions.

In conclusion, the future scope of the Fake Job Posting Detector is vast, with ample opportunities to enhance its functionality, scalability, and accuracy. By adopting advanced technologies and continuously refining its capabilities, the application can become an indispensable tool in protecting job seekers and improving the safety of online job platforms.

8. BIBLIOGRAPHY

[1] Van Huynh, Tin, Kiet Van Nguyen, Ngan Luu-Thuy Nguyen, and Anh Gia-Tuan Nguyen. "Job prediction: From deep neural network models to applications." In 2020 RIVF International Conference on Computing and Communication Technologies (RIVF), pp. 1-6. IEEE, 2020.

[2] Habiba, Sultana Umme, Md Khairul Islam, and Farzana Tasnim. "A comparative study on fake job post prediction using different data mining techniques." In 2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), pp. 543-546. IEEE, 2021.E. F. Ohata et al., "Automatic detection of COVID-19 infection using chest X-ray imagesthrough transfer learning," IEEE/CAA J. Autom. Sin., vol. 8, no. 1, pp. 239–248, 2021

[3] Nasser, Ibrahim M., Amjad H. Alzaanin, and Ashraf Yunis Maghari. "Online Recruitment Fraud Detection using ANN." In 2021 Palestinian International Conference on Information and Communication Technology (PICICT), pp. 13-17. IEEE, 2021.

[4] Mehboob, Asad, and M. S. I. Malik. "Smart fraud detection framework for job recruitments." Arabian Journal for Science and Engineering 46, no. 4 (2021): 3067-3078.

[5] Keerthana B, Reddy AR, Tiwari A (2021) Accurate prediction of fake job offers using machine learning. In: Bhattacharyya D, Thirupathi Rao N (eds) Machine intelligence and soft computing, pp 101– 112.Springer

[6] Srinivas, J., K. Venkata Subba Reddy, G. J. Sunny Deol, and P. VaraPrasada Rao. "Automatic Fake News Detector in Social Media Using Machine Learning and Natural Language Processing Approaches." In Smart Computing Techniques and Applications, pp. 295-305. Springer, Singapore, 2021.

[7] Tabassum, Hridita, Gitanjali Ghosh, Afra Atika, and Amitabha Chakrabarty. "Detecting Online Recruitment Fraud Using Machine Learning." In 2021 9th International Conference on Information and Communication Technology (ICoICT), pp. 472-477. IEEE, 2021.

[8] Amaar, Aashir, Wajdi Aljedaani, Furqan Rustam, Saleem Ullah, Vaibhav Rupapara, and Stephanie Ludi. "Detection of Fake Job Postings by Utilizing Machine Learning and Natural Language Processing Approaches." Neural Processing Letters (2022): 1-29.

[9] Jagadeesh, C., Pravin R. Kshirsagar, G. Sarayu, G. Gouthami, and B.

Manasa. "Artificial intelligence based Fake Job Recruitment Detection Using Machine Learning Approach." Journal of Engineering Sciences 12 (2021): 0377-9254.

[10] Ranparia D, Kumari S, Sahani A (2020) Fake job prediction using sequential network. In: 2020 IEEE 15th international conference on industrial and information systems (ICIIS), pp 339–343

[11] Shibly, F. H. A., Sharma Uzzal, and H. M. M. Naleer. "Performance comparison of two class boosted decision tree snd two class decision forest algorithms in predicting fake job postings." (2021).