

Code:

```
from flask import Flask, render_template, request
import pickle
```

```
# Load the model correctly with 'rb' mode
model = pickle.load(open('Fake_job6.sav', 'rb'))
```

```
app = Flask(__name__)
```

```
@app.route('/')
def loadPage():
```

```
    return render_template('DIC-3.html')
```

```
@app.route('/predict', methods=['POST'])
def predict():
```

```
    # Retrieve all form data
```

```
    a = request.form["telecommuting"]
```

```
    b = request.form["ratio"]
```

```
    c = request.form["min_salary"]
```

```
    d = request.form["max_salary"]
```

```
    e = request.form["has_company_logo"]
```

```
    f = request.form["has_questions"]
```

```
    g = request.form["mean_tfidf_score"]
```

```
    # Prepare the input feature array for prediction
```

```
    t = [[int(a), float(b), float(c), float(d), int(e), int(f), float(g)]]
```

```
    y = model.predict(t)
```

```

# Return results to DIC-1.html, including the prediction result

return render_template('DIC-1.html',    telecommuting=a,    ratio=b,
min_salary=c,    max_salary=d,    has_company_logo=e,    has_questions=f,
mean_tfidf_score=g, prediction=y[0])

if __name__ == "__main__":
    app.run(debug=True)

```

## **HTML CODE:**

```

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>FakeJobprediction</title>

    <style>

        @import url('../static/style.css');

    </style>

</head>

<body>

    <section class="main">

        <div class="nav">

            <h1><a href="/">FAKE_JOB_PREDICTION</a></h1>

        </div>

        <div class="main-container">

            <div class="model">

```

```

<h1>REAL(1) OR FAKE(0)</h1>
<br>
<div class="form">
  <form action="/predict" method="post">
    <div class="mg mg1">
      </div><br>
      <div class="group">
        <label>telecommuting:</label><br>
        <input type="number" name="telecommuting"
placeholder="Ex: 1" required>
      </div><br>
      <div class="group">
        <label>ratio:</label><br>
        <input type="number" name="ratio" placeholder="Ex: 12"
required>
      </div><br>
      <div class="group">
        <label>min_salary:</label><br>
        <input type="number" name="min_salary" min="0"
placeholder="3555" required>
      </div><br>
      <div class="group">
        <label>max_salary:</label><br>
        <input type="number" name="max_salary" min="0"
placeholder="123444" required>
      </div><br>
    </div>
    <div class="mg mg2">
      <div class="group">

```

```
<label>has_company_logo:</label><br>
<input      type="text"      name="has_company_logo"
placeholder="0" required>
</div><br>
<div class="group">
  <label>has_questions:</label><br>
  <input  type="number"  name="has_questions"  min="0"
placeholder="1" required>
</div><br>
<div class="group">
  <label>mean_tfidf_score:</label><br>
  <input      type="text"      name="mean_tfidf_score"
placeholder="12.344" required>
</div><br>
</div>
<div class="submit">
  <input type="submit" value="Predict">
</div>
</form>
<br><br>
</div>
<br>
</div>
</div>
</section>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Fake_job_prediction</title>
  <style>
    @import
url('https://fonts.googleapis.com/css2?family=Barlow+Condensed&display=sw
ap');
    * {
      padding: 0;
      margin: 0;
      box-sizing: border-box;
    }
    body {
      font-family: 'Barlow Condensed', sans-serif;
    }
    .main {
      height: 100vh;
      width: 100vw;
    }
    .main .nav {
      background: #123456; /* Adjust the color to fit the credit risk theme */
      color: white;
      padding: 1rem;
```

```

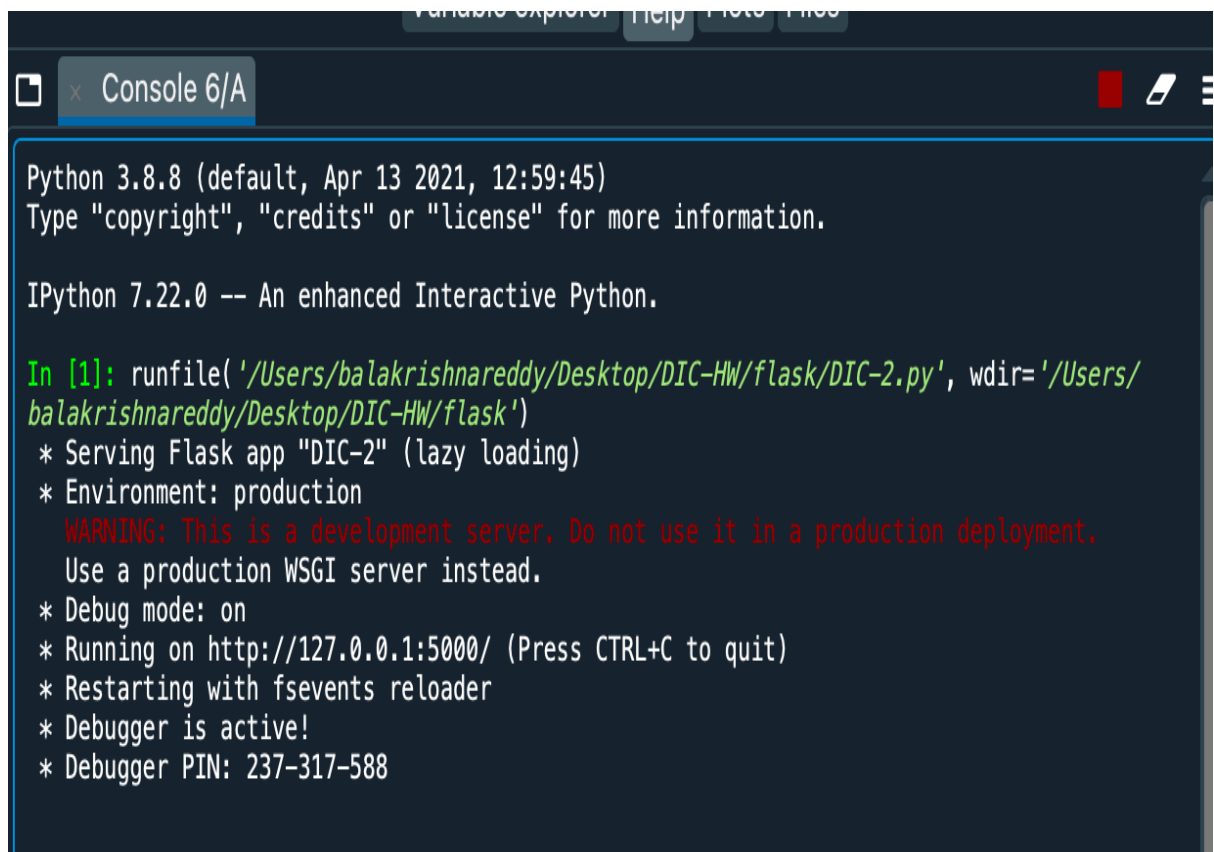
    }
    .main .outer {
        height: 90vh;
        width: 100vw;
        display: flex;
        justify-content: center;
        align-items: center;
    }
    a {
        text-decoration: none;
        color: white;
    }
</style>
</head>
<body>
    <section class="main">
        <div class="nav">
            <h2><a href="/">FAKE JOB PREDICTION</a></h2>
        </div>
        <div class="outer">
            <div class="inner">
                <div class="res">
                    <h1>THE GIVEN JOB POST IS {{ prediction }}</h1>
                </div>
            </div>
        </div>
    </section>

```

</body>

</html>

## ScreenShots:



```
Python 3.8.8 (default, Apr 13 2021, 12:59:45)
Type "copyright", "credits" or "license" for more information.

IPython 7.22.0 -- An enhanced Interactive Python.

In [1]: runfile('/Users/balakrishnareddy/Desktop/DIC-HW/flask/DIC-2.py', wdir='/Users/
balakrishnareddy/Desktop/DIC-HW/flask')
* Serving Flask app "DIC-2" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with fsevents reloader
* Debugger is active!
* Debugger PIN: 237-317-588
```

## FAKE JOB PREDICTION

**REAL(1) OR FAKE(0)**

telecommuting:

ratio:

min\_salary:

max\_salary:

has\_company\_logo:

has\_questions:

mean\_tfidf\_score:





THE GIVEN JOB POST IS 1

## **Installation and setup instructions:**

Here are the instructions to be followed for running the code and flask-based web application and processes developed during phases 1 and 2 of our project

pip install Flask

pip install numpy

pip install pandas

## **Running the application:**

Run the following command in the CLI(command line interface) to start the Flask server

python app.py

open the web browser and visit <http://127.0.0.1:5000/>

this will load the homepage of web application.

Depending on the particular requirements and limitations of our project, there may be multiple benefits to using an SGD (Stochastic Gradient Descent) classifier rather than a Random Forest model. The SGD classifier may be a better fit for you for the following reasons, particularly if you want to deploy it in a server environment:

**Scalability and Efficiency:** Because SGD classifiers update the model incrementally with each iteration using either a single sample or a mini-batch of samples, they are often faster and more scalable for large datasets. This might be especially helpful when working with streaming data or when you have limited memory and processing power.

As we pointed out, utilizing an SGD classifier as opposed to a Random Forest may result in less inputs being needed. This is frequently due to the possibility that Random Forests may need to create and choose a wider variety of features in order to efficiently capture interactions and non-linear relationships. On the other hand, SGD and other linear models may function well with fewer features or with less feature engineering.

SGD classifiers don't require completely retraining in order to adjust to fresh data. This functionality is especially useful in settings where data is updated or changes often. Conversely, when fresh data is received, Random Forests typically need to be retrained or updated progressively.

Compared to ensemble models like Random Forests, linear models including those trained using SGD are frequently easier to understand and debug. This simplicity can facilitate maintenance and troubleshooting, particularly in a production setting where it's critical to comprehend how inputs impact outputs.

**Performance in High-Dimensional Spaces:** When the number of samples is not abnormally high in relation to the number of features, linear models can work effectively in high-dimensional spaces (many features). Because of this property, SGD may be a better option in some applications—such as text classification—where the feature space is large yet sparse.

**Resource Usage:** Linear models are better suited for deployment on servers with constrained resources because they typically consume less memory and CPU during training and inference.

It's crucial to weigh these benefits against any potential downsides, though. Complex patterns in the data may not be as well captured by linear models—including those trained with SGD—as they are by ensemble models like Random Forests. Furthermore, SGD can be sensitive to feature scaling and needs precise tweaking of hyperparameters like the learning rate and regularization terms.

It seems like a sensible decision if the SGD classifier is providing you with enough accuracy and is easier to manage in your server setup. To make sure the model functions properly in all anticipated situations and data variances, it is important to always think about extensively testing and validating the model.

Based on our problem statement, which concerns utilizing data analytics and machine learning to detect and mark fake job posts, the following comprehensive suggestions may be obtained from the examination, in addition to possible project expansions:

Resources for Job Seekers to Learn:

Engaging Education Module: Provide a training module on employment sites that instructs users on how to spot typical warning signals of phony job postings.

Real-Time Advice: Introduce a function that gives consumers real-time advice when perusing job postings, emphasizing warning signs right on the employment platform.

Improved Reporting Systems:

Reporting Interface Simplified: Improve the reporting interface so that people may report questionable job advertisements more easily. Fast reporting can facilitate more rapid analysis and action on potentially fraudulent listings.

Feedback on Reports: To promote active engagement and alertness among the job-seeking community, give users who report job scams feedback, including the results of their report.

Combining External Verification Tools with Integration:

Cross-Checking Information on Authentic Job Boards: To ensure the legitimacy of job posts, integrate the system with official job boards or company websites.

Use of External APIs: To confirm the existence and legitimacy of the businesses providing the positions, make use of APIs from business registration entities.

Resolutions

**Deployment of Machine Learning Models:** Apply the created SGD classifier model to real-time job posting analysis, flagging submissions that match known fraud criteria. This technique keeps frauds from ever reaching potential victims, which lowers the risk associated with them.

**Frequent Updates and Model Training:** As con artists refine their techniques, keep the machine learning model up to speed with fresh information on fraudulent activities. Frequent retraining guarantees the model's continued relevance and effectiveness.

**User Feedback Loop:** Put in place a system that uses user input regarding the fraud detection system's accuracy to hone and train the model even more. This increases the accuracy of the model and motivates users to uphold the integrity of the platform.

**Extensions and Additional Study Expanded Use to Other Types of Internet Fraud:**

Expand the capabilities of the underlying technology to identify more types of online fraud, such as phishing attempts in emails or phony online classified ads.

**Improved Natural Language Processing (NLP):** Apply more sophisticated NLP methods to examine the nuances in job posting language that may point to fraud or false information.

**Geographical Analysis:** Using geographic data, examine trends in employment frauds. This could indicate location-based trends in scam techniques and assist in identifying areas with greater fraud rates.

**Industry Collaboration Initiatives:**

Form alliances with other employment boards to compile a database of recognized con artists and scammers for the sector. Across platforms, shared knowledge can aid in the quicker detection and avertance of employment frauds.

**Regulatory Influence and Policy Creation:** Work with legislators to develop rules requiring job posting verification, which will improve the general security of the job-seeking system.

These suggestions and additions provide a thorough approach to improve the general dependability and security of online job hunting in addition to addressing the pressing issue of employment scams. Through constant evolution to include new technologies and approaches, the project can stay ahead of fraudulent strategies and lessen their influence on job searchers.

Here are specific details on what users can learn from the product and how it helps them deal with the issues presented by employment scams, based on the problem statement of recognizing and flagging fake job advertisements using data analytics and machine learning:

**What Consumers Can Discover About Product Fraud Indicator Awareness:** Users will get knowledge about typical traits and warning signs of phony job listings. The tool might draw attention to particular dubious elements of a job advertisement, such as ambiguous job descriptions, excessive compensation for the position, or the need for needless personal data.

**Safe Job Searching Techniques:** Users will discover the best techniques for safe job searching by interacting with the site and the educational materials offered. This entails knowing their rights as job seekers, identifying safe and risky communication routes, and confirming the validity of the company through other sources.

**Gaining an understanding of automated security measures** helps users understand how technology might help protect their job hunt. Students gain knowledge about the application of data-driven techniques to improve digital platform security as well as the function of machine learning in real-time threat detection.

**How the Product Aids in Problem-Solving**

**Real-Time Fraud Detection:** The software evaluates job posts as they are posted by incorporating machine learning models such as the SGD classifier.

**Enhanced Confidence and Trust:** When using job-search sites that make use of such sophisticated detection technologies, users can feel more safe. Users are more likely to trust the platform and the veracity of the postings accessible when they are aware that the ads have been checked for any scams.

**Decreased Risk of Financial and Personal Loss:** The technology immediately helps consumers experience a lower risk of financial and personal data loss by efficiently detecting fake postings. This is essential to shielding users from the potentially disastrous consequences of employment scams.

**Empowerment Through Reporting Tools:** Users may report questionable job ads with ease thanks to the product's user-friendly reporting tools. These tools enable users to enhance the machine learning model and contribute to the platform's safety.

Compliance and Ethical Standards: Employing such a solution for employment platforms guarantees adherence to ethical and legal requirements for consumer protection. It exhibits a dedication to upholding a secure and reliable atmosphere for prospective employees.

Through the implementation of these learning objectives and problem-solving techniques, the program not only improves the job seekers' immediate security but also cultivates a community of users who are more knowledgeable and circumspect. Using both technology-driven prevention and education is essential to effectively combating work frauds in the digital era.