**Job Matching Engine**

**Introduction**

This project aims to develop a job matching component that can automatically match job seeker profiles with job descriptions to provide the best job matches to job seekers. The project leverages machine learning techniques such as natural language processing and recommender systems to analyze and rank the relevance of job descriptions based on the profile information. The project uses AWS Sagemaker as the main platform for model development and deployment, and also integrates other AWS services such as Comprehend, Cloudwatch, etc. [The project also follows the best practices and guidelines from existing research on job recommender systems](https://www.wikihow.life/Prepare-Documentation-for-a-Project)[2](https://eugeneyan.com/writing/ml-design-docs/" \t "_blank).

**Data**

Datasets are;

[Resume Dataset | Kaggle](https://www.kaggle.com/datasets/snehaanbhawal/resume-dataset)

[Predicting Job Titles from Resumes | Kaggle](https://www.kaggle.com/datasets/thedevastator/predicting-job-titles-from-resumes)

[Updated Resume Dataset | Kaggle](https://www.kaggle.com/datasets/jillanisofttech/updated-resume-dataset)

[Linkedin Dataset | Kaggle](https://www.kaggle.com/datasets/heet9022/linkedin-dataset)

[resumes | Kaggle](https://www.kaggle.com/datasets/maitrip/resumes)

The data for this project consists of two main types: resumes and job descriptions. The resumes are collected from various sources such as Kaggle datasets, Linkedin profiles, etc. [The job descriptions are scraped from various job portals such as Indeed, Monster, etc4](https://www.datarevenue.com/en-blog/machine-learning-project-checklist). The data is stored in CSV files with the following schema:

| **Column** | **Description** | **Type** |
| --- | --- | --- |
| resume\_id | A unique identifier for each resume | String |
| resume\_text | The text content of the resume | String |
| location | The preferred location of the job seeker | String |
| experience | The cumulative years of relevant experience of the job seeker | Integer |
| education | The cumulative years of relevant education of the job seeker | Integer |
| work\_type | The type of work desired by the job seeker (remote, hybrid, on-site) | String |
| job\_title | The desired job title of the job seeker | String |
| visa\_sponsorship | The visa sponsorship requirements of the job seeker (yes, no) | String |

| **Column** | **Description** | **Type** |
| --- | --- | --- |
| job\_id | A unique identifier for each job description | String |
| job\_text | The text content of the job description | String |
| location | The location of the job posting | String |
| experience | The minimum years of relevant experience required for the job | Integer |
| education | The minimum years of relevant education required for the job | Integer |
| work\_type | The type of work offered by the job (remote, hybrid, on-site) | String |
| job\_title | The title of the job posting | String |
| visa\_sponsorship | The visa sponsorship availability for the job (yes, no) | String |

The data is cleaned and preprocessed by removing duplicates, missing values, outliers, and irrelevant information. The text data is also tokenized, normalized, and vectorized using various techniques such as TF-IDF, word embeddings, etc.

**Methodology**

* Summarize the methodologies that you employed in your work. [Include information such as the model architecture, algorithm, hyperparameters, evaluation metrics, etc](https://www.wikihow.life/Prepare-Documentation-for-a-Project)[3](https://reason.town/abstract-for-machine-learning-project/). You can also include diagrams or pseudocode to illustrate your approach.
* For example:

The methodology for this project consists of two main steps: feature extraction and matching. In the feature extraction step, we use natural language processing techniques to extract relevant features from the resume and job text data. We use AWS Comprehend to perform named entity recognition (NER), part-of-speech tagging (POS), and sentiment analysis on the text data. We also use word embeddings such as Word2Vec or BERT to represent the text data as numerical vectors. We then concatenate these features with the other profile information such as location, experience, education, etc., to form a feature vector for each resume and job description.

In the matching step, we use a recommender system algorithm to compute the similarity score between each resume and job description feature vector. We use a cosine similarity function to measure the similarity between two vectors. We then rank the job descriptions based on their similarity scores for each resume and return the top x matches as recommendations.

The model is trained and evaluated using a train-test split of 80:20. We use accuracy, precision, recall, and F1-score as the evaluation metrics to measure the performance of the model. We also use cross-validation and grid search to tune the hyperparameters of the model.

**Implementation**

* Discuss how you implemented and deployed your system. Include information such as the tools, frameworks, libraries, APIs, dependencies, etc., that you used in your work. [You can also include code snippets or screenshots to show how your system works1](https://www.wikihow.life/Prepare-Documentation-for-a-Project).
* For example:

The system is implemented and deployed using AWS Sagemaker as the main platform. We use Python as the programming language and Jupyter notebooks as the development environment. We use various libraries such as pandas, numpy, scikit-learn, nltk, gensim, transformers, etc., to perform data analysis, preprocessing, modeling, and evaluation. We also use AWS Comprehend as the natural language processing service and AWS Cloudwatch as the monitoring service.

The system is deployed as a RESTful API that can accept a profile as input and return the top x job descriptions as output. The API is secured using AWS IAM and API Gateway. The system is also scalable and fault-tolerant using AWS Lambda and S3.

The following code snippet shows how to use the API to get recommendations for a sample profile:

import requests

import json

# Define the sample profile

profile = {

"resume\_text": "I am a data scientist with 3 years of experience in machine learning, natural language processing, and data visualization. I have a master's degree in computer science and a bachelor's degree in mathematics. I am proficient in Python, R, SQL, TensorFlow, Keras, and Tableau. I am looking for a remote or hybrid position as a data scientist or machine learning engineer.",

"location": "New York",

"experience": 3,

"education": 6,

"work\_type": "remote or hybrid",

"job\_title": "data scientist or machine learning engineer",

"visa\_sponsorship": "no"

}

# Define the API endpoint

endpoint = "https://xxxxxxxxxx.execute-api.us-east-1.amazonaws.com/prod/recommend"

# Make a POST request with the profile as payload

response = requests.post(endpoint, data=json.dumps(profile))

# Print the response status code and body

print(response.status\_code)

print(response.json())

The output of the code snippet is:

200

{

"recommendations": [

{

"job\_id": "123456",

"job\_text": "We are looking for a data scientist to join our team and help us build innovative solutions for our clients. You will be responsible for developing and deploying machine learning models using various techniques such as natural language processing, computer vision, recommender systems, etc. You will also be responsible for creating data visualizations and reports using tools such as Tableau, Power BI, etc. You will work closely with other data scientists, engineers, and business analysts to deliver high-quality and efficient solutions.",

"location": "New York",

"experience": 2,

"education": 4,

"work\_type": "remote or hybrid",

"job\_title": "data scientist",

"visa\_sponsorship": "no",

"similarity\_score": 0.87

},

{

"job\_id": "234567",

"job\_text": "We are seeking a machine learning engineer to join our growing team and help us create cutting-edge products for our customers. You will be responsible for designing and implementing machine learning pipelines using frameworks such as TensorFlow, Keras, PyTorch, etc. You will also be responsible for optimizing and testing the performance of the models using tools such as AWS Sagemaker, Cloudwatch, etc. You will work collaboratively with other machine learning engineers, software engineers, and product managers to deliver scalable and robust solutions.",

"location": "New Jersey",

"experience": 3,

"education": 5,

"work\_type": "remote or hybrid",

"job\_title": "machine learning engineer",

"visa\_sponsorship": "no",

"similarity\_score": 0.85

},

{

"job\_id": "345678",

"job\_text": "We are hiring a data analyst to join our data team and help us analyze and interpret data for various business needs. You will be responsible for collecting and cleaning data from various sources such as databases, APIs, web scraping, etc. You will also be responsible for performing exploratory data analysis, statistical analysis, and hypothesis testing using tools such as Python, R, SQL, etc. You will also be responsible for generating insights and recommendations using tools such as Excel, Tableau, etc. You will work closely with other data analysts, data scientists, and business stakeholders to support data-driven decision making.",

"location": "New York",

"experience": 2,

"education": 4,

"work\_type": "on-site",

"job\_title": "data analyst",

"visa\_sponsorship": "no",

"similarity\_score": 0.82

}

]

}

**Alternatives**

* Discuss any alternative solutions that you considered and rejected in your work. [Explain why you chose your proposed solution over the alternatives2](https://eugeneyan.com/writing/ml-design-docs/).
* For example:

We considered several alternative solutions for this project such as:

Using a rule-based approach instead of a machine learning approach to match the resumes and job descriptions based on their semantic similarity. We rejected this approach because it would require a lot of manual effort to define and maintain the rules, and it would not be able to handle the variability and complexity of natural language.

* Using a collaborative filtering approach instead of a content-based approach to recommend jobs based on the ratings or preferences of other similar users. We rejected this approach because it would require a large and reliable user base to provide ratings or feedback, and it would also suffer from the cold start problem for new users or jobs.
* Using a different platform or framework instead of AWS Sagemaker to develop and deploy the model. We rejected this approach because AWS Sagemaker provides a comprehensive and integrated solution for machine learning that offers various benefits such as scalability, security, reliability, and cost-effectiveness. AWS Sagemaker also allows us to easily use other AWS services such as Comprehend, Cloudwatch, etc., to enhance our system.

We chose our proposed solution because it offers the following advantages:

* It uses a natural language processing technique to extract relevant features from the text data that capture the meaning and context of the resumes and job descriptions.
* It uses a recommender system algorithm to compute the similarity score between the feature vectors that reflect the relevance and suitability of the job descriptions for each resume.
* It uses AWS Sagemaker as the main platform for model development and deployment that provides a convenient and efficient way to build, train, test, monitor, and deploy machine learning models.