LONDON METROPOLITAN UNIVERSITY

Islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CU6051NI Artificial Intelligence

25% Individual Coursework

Submission: Milestone 1

Academic Semester: Autumn Semester 2025

Credit: 15 credit semester long module

Student Name: Shidharth Kharga

London Met ID: 23049367

College ID: NP01CP4A230377

Assignment Due Date: 07/01/2026.

Assignment Submission Date: 07/01/2026

Submitted To: Mr. Binod Bhattarai

GitHub Link	*https://github.com/justwannar0ck/Job_Recomendation_ML.git*

I confirm that I understand my coursework needs to be submitted online via MST Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1. INTRODUCTION

This coursework explores the application of machine learning and natural language processing for the recommendation of jobs. “Artificial Intelligence is the development of computer systems able to perform tasks that would typically require human intelligence.” “Machine Learning is a type of artificial intelligence that involves training models on data in order to make predictions or decisions without being explicitly programmed.” Supervised machine learning is a type of machine learning in which models learn to predict or classify data by training with labelled examples. Classification or Regression are some of the tasks that fall into supervised machine learning.

NLP is the area that deals with the application of ML to text-based job data such as job titles and descriptions with the aim of developing models for recommending jobs. NLP is applied using the job-skill set data from Hugging Face Datasets. This data set is made up of approximately 1.17K job entries with features such as job title, description, category (categorized into 5 broad categories of jobs), and finally the job skill set. (Batuhanmtl, 2024)

1.1. Concepts used in this project

Here it seems that the heart of this assignment is about Supervised Learning, a machine learning technique in which the models are developed on labelled data. In this scenario, it is assumed that the "label" is probably the Job Title or Category, with the "features" being the skill or description listed.

We will discuss three of these algorithms:

Naive Bayes: This is a probabilistic learning classifier that is linked with Bayes' theorem. It is widely used in NLP for spam filtering or document categorization. It is effective for dealing with high-text features. (Anon., 2022)

Logistic Regression: Ironically, this is a type of classification algorithm, despite its name. It is used to predict the probabilities of a categorical dependent variable, such as: "Is this a Data Scientist position? Yes/No." It is a good baseline for binary or multi-classification tasks. (Lee, n.d.)

Random Forest: An ensemble learning technique that builds many decision trees. It is less prone to overfitting and tends to provide more accurate predictions by aggregating the numerous decision trees. (Alam, 2024)

1.2. Problem Domain

The main objective here is to work out two related problem formulations and conduct experiments for each of these:

Multi-class job-category classification: from the job title and description, predict the job category (5 classes). This is a standard text classification task and can be used as a basic recommendation signal (e.g., recommend jobs in predicted categories).

Skill prediction/job-to-skill mapping (multi-label) job-skill set will be predicted; one job can have multiple skills. This is a multi-label prediction/information extraction task which will help in matching candidate’s skills to their job postings or recommending jobs given the user's skills.

2. BACKGROUND

2.1. Review of Existing Research

Although Job Recommendation Systems is an area that is widely explored as platforms such as LinkedIn and indeed aim for candidate automated matching systems. Currently, research work is more inclined towards a shift from mere word matching to semantic matching (meaning of skills, not words).

Research 1: Content Based Recommendation using NLP (IEEE, 2024)

- **Concept:** Recently, research work carried out by Suresh et al. in the journal IJRASET is on “Content-Based Filtering." This is where the candidate description and resume are looked upon as two text files.

- **Technique:** Term Frequency-Inverse Document Frequency is used by the researchers to convert these documents into vectors. Finally, the Cosine Similarity is calculated to determine how close the resume is to the description of the job.

- **Outcome:** This is an effective approach for use in certain positions, but it is less effective if the candidate describes themselves with different terminology than that of the recruiter (e.g., “App Development” versus “Software Engineering”).

Research 2: Classification Based Hybrid Techniques (Martinez, 2018)

- **Concept:** Other work by Karthikeyan et al. suggests employing supervised classification techniques such as Random Forests and SVM to predict employment role using explicit features such as skills and experience.

- **Technique:** Instead of pointing to a certain occupation, these models classify the user into what is called a Job Category (for example, Information Technology), with the lesson plan then narrowing down to that category.

- **Outcome:** This cuts down the complexity of calculations since the search space is reduced earlier on in the process.

Research 3: Solving the Cold Start Problem (Zhang, n.d.)

- **Concept:** One of the key challenges mentioned in the literature is that of Cold Start job recommendation to a new user with no history.

- **Technique:** This study makes use of Naive Bayes Classifiers for probabilistic predictions with less starting information (only 3-4 skills) for resume recommendations without requiring the entire resume history.

2.2. Dataset Information

Source: Hugging Face Datasets (Batuhanmtl, 2024)

Dataset Name: batuhanmtl/job-skill-set

This is a specially compiled data set for matching work skills with work titles. This is suited for the classification task that is supervised.

Structure: This data set has about 1,170 rows of training data.

Key Columns:

- **Job Title –** contains the target label that we would like to predict (e.g., Data Scientist, HR Manager).
- **Skills –** example of skills listed for the input text are Python, SQL, Management.
- **Job Description –** unstructured text about the role (useful for the extraction of advanced NLP features).

Characteristics of Data: Data is filled with text. Some preprocessing techniques that need to be applied before it is feasible for use by the machine learning models are - Tokenization, which involves Breaking the texts into words and, Stop Word Removal which refers to removing words like ‘and’, ‘the’ that do not add any significant.

3. SOLUTION

To address the issue of job titles recommendation depending on skill sets, the Supervised Multi-Class Classification System is suggested in the form of a conventional Natural Language Processing (NLP) pipeline. The data in the Job Skill Set dataset is in raw text (skills and descriptions), therefore, the system cannot process the data directly. The suggested strategy entails:

- **Data Cleaning:** Cleaning the text by removing noise (punctuation, special characters).

- **Vectorization:** Transforming the processed text into numerical vectors with the help of Term Frequency-Inverse Document Frequency. This puts special skills (such as TensorFlow) into focus and generic words into the background.

- **Model Training:** We will model three different models on the vectorized data – Naive Bayes, Logistic Regression and Random Forest.

- **Prediction:** This system will receive a new set of user skills, run the input through the same vectorizer, and give the job title with the highest probability.

3.1. AI Algorithms Used (Implementation Logic)

Naive Bayes (Multinomial): We are using the Multinomial variant, which is specifically created to work with text data, with features being word counts or frequencies. It determines the likelihood of a word (skill) to be found in a job category (e.g. how many times Java is found in Software Engineer).

Logistic Regression: We employ One-vs-Rest strategy. Because we have several job titles, the algorithm divides the problem into - “Is it Job A or not?”, “Is it Job B or not?”. It finds a linear boundary to divide these categories by the weights of TF-IDF of the skills.

Random Forest Classifier: The model develops hundreds of decision trees. The trees pose a set of questions (e.g., "Is the skill set in Python?”). A majority vote across all the trees determines the final prediction, and this is more effective in the job descriptions noise than the individual models.

4. PSEUDOCODE

4.1. Main System Pseudocode

START

IMPORT pandas, numpy, sklearn libraries

LOAD "Job Skill Set" dataset into DataFrame

DEFINE Function preprocess_text(text):

CONVERT text to lowercase

REMOVE punctuation and special characters

REMOVE stop words (e.g., "the", "and")

RETURN cleaned_text

APPLY preprocess_text to "Skills" column

INITIALIZE TfidfVectorizer

FIT_TRANSFORM cleaned_skills into Feature_Vectors (X)

SET Job_Titles as Target_Labels (y)

SPLIT data into Training_Set (80%) and Testing_Set (20%)

FOR EACH Model IN [Naive_Bayes, Logistic_Regression, Random_Forest]:

INITIALIZE Model

TRAIN Model using Training_Set

PREDICT Labels using Testing_Set

CALCULATE Accuracy, Precision, Recall

PRINT Performance Report

ENDFOR

END

4.2. Naive Bayes Pseudocode

START NaiveBayes_Classification

IMPORT MultinomialNB

SET vectorizer = CountVectorizer(binary=True)

X_train = vectorizer.fit_transform(train_texts)

X_test = vectorizer.transform(test_texts)

model = MultinomialNB

model.fit(X_train, y_train_category)

y_pred = model.predict(X_test)

probabilities = model.predict_proba(X_test)

OUTPUT accuracy, precision, recall, f1

SAVE model & vectorizer

END

4.3. Logistic Regression Pseudocode

START LogisticRegression_Classification

vectorizer = TfidfVectorizer(ngram_range=(1,2), max_features=2000)

X_train = vectorizer.fit_transform(train_texts)

X_test = vectorizer.transform(test_texts)

base = LogisticRegression(max_iter=1000)

model = OneVsRestClassifier(base)

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

y_proba = model.predict_proba(X_test)

evaluate metrics

END

4.4. Random Forest Pseudocode

```
START RandomForest_Classification
  vectorizer = TfidfVectorizer(max_features=2000)
  X_train = vectorizer.fit_transform(train_texts)

  model = RandomForestClassifier(n_estimators=200, random_state=42)

  IF task = multi_class:
     model.fit(X_train, y_train_category)
  ELSE IF task = multi_label:
     wrapper = MultiOutputClassifier(model, n_jobs=-1)
     wrapper.fit(X_train, Y_train_skills)

  y_pred = model.predict(X_test)
  evaluate metrics

END
```

5. FLOWCHARTS

5.1. Main System Flowchart

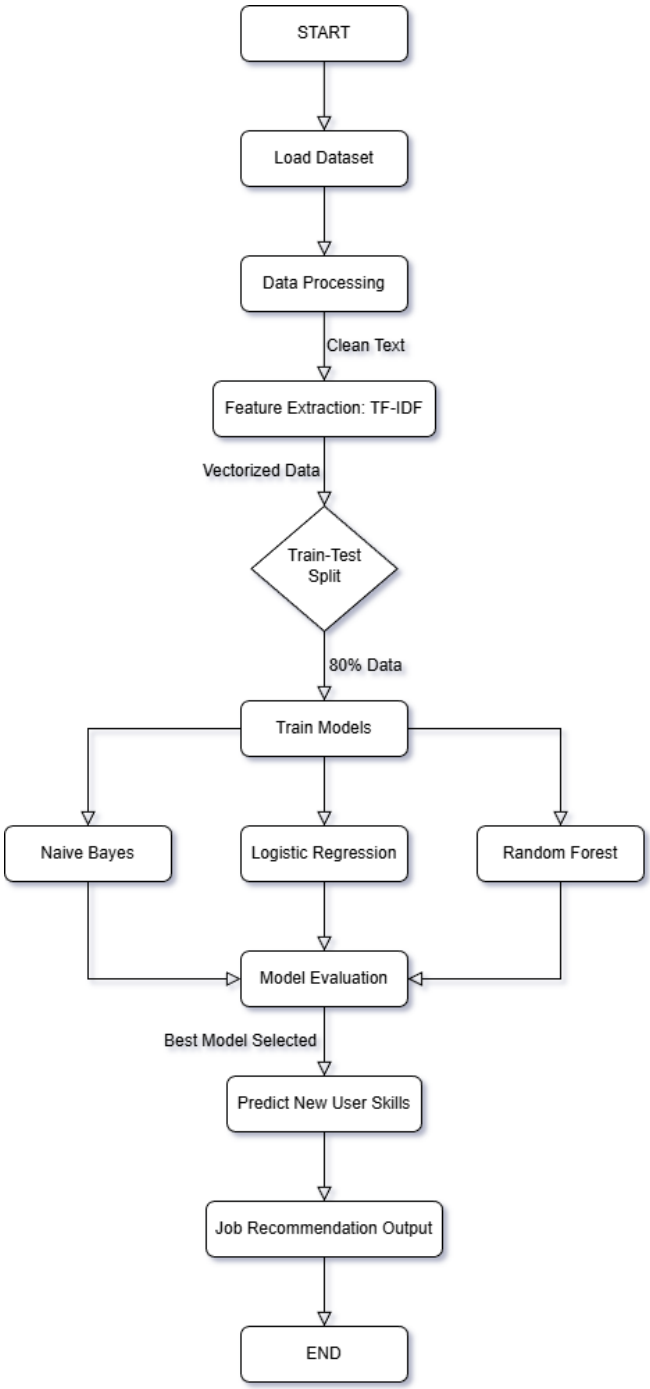

Figure 1: Main System Flowchart

5.2. Naive Bayes Flowchart

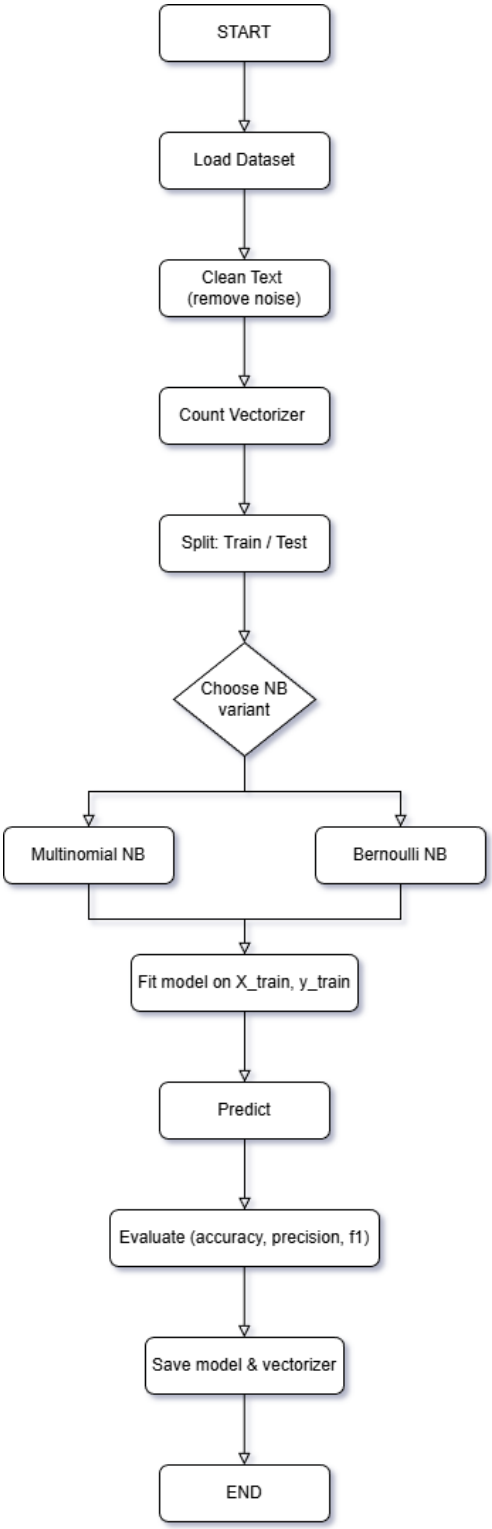

Figure 2: Naive Bayes Flowchart

5.3. Logistic Regression Flowchart

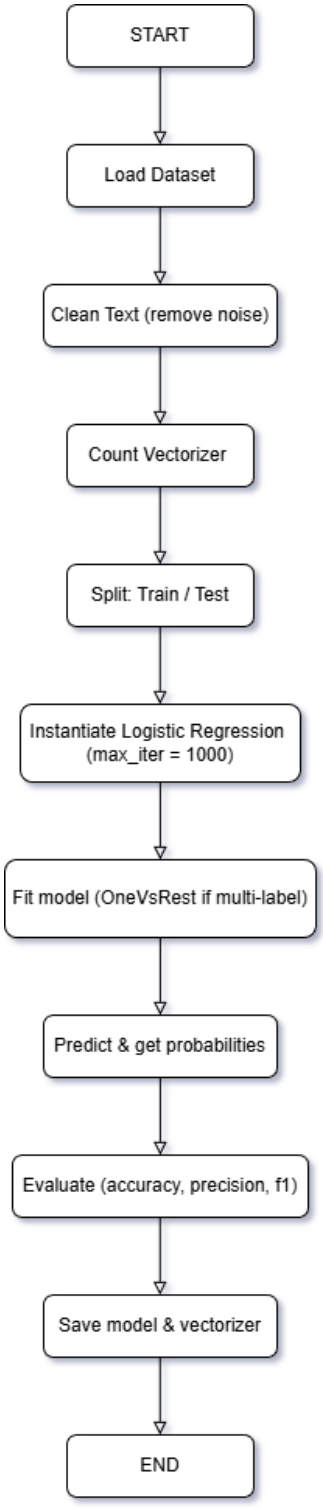

Figure 3: Logistic Regression Flowchart

5.4. Random Forest Flowchart

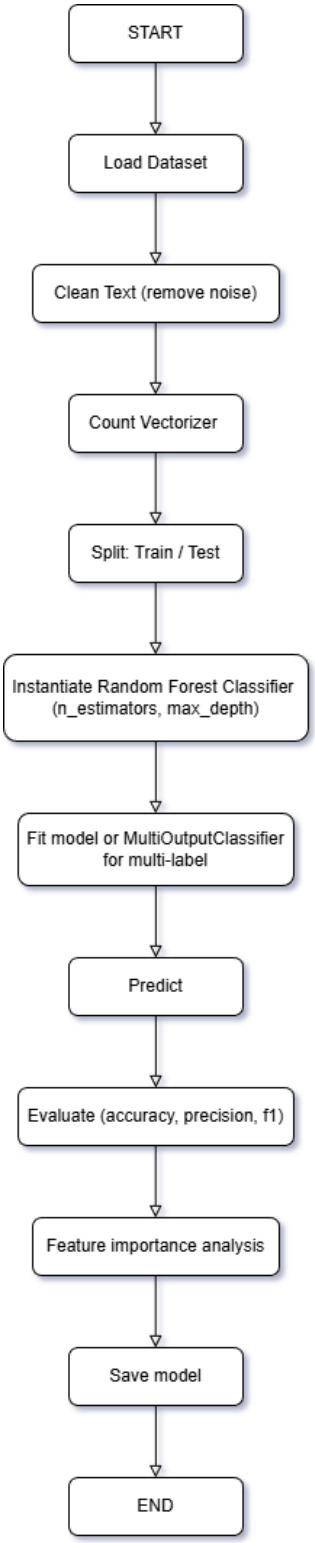

Figure 4: Random Forest Flowchart

6. TOOLS AND TECHNOLOGIES USED

To create the Job Recommendation System, Python-based technology stack was chosen because it has a wide range of data science and machine learning libraries. The following tools and libraries were used:

Table 1: Tools & Technologies Used

TOOL / LIBRARY	ROLE IN PROJECT	JUSTIFICATION FOR SELECTION
Python	Core programming language	The standard in the industry in terms of AI because it is easy to read and has an enormous collection of ready-to-use data science packages.
Pandas	Data Manipulation	Used to load the dataset into a structured Data-Frame. It enabled the processing of data with efficient data cleaning (dropping of nulls, string operations).
Scikit-Learn (sklearn)	Machine Learning Framework	The library was used for: 1. Vectorization (TF-IDF): Text to numbers.

		2. Modelling: applying Naive Bayes, Logistic Regression and Random Forest. 3. Assessment: Accuracy measure and division of training data.
Hugging Face (dataset)	Data source	Facilitated the access to the dataset of the "Job Skill Set".
NumPy	Numerical Operations	Utilized for calculating the maximum probability scores from the model's prediction output to determine match confidence.
Regex (re)	Text Normalization	Utilized to generate a cleaning operation that removes punctuations and non-alphanumeric characters, which provide the AI with high-quality input.

7. DEVELOPMENT PROCESS

The prototype development was carried out in a typical Machine Learning Pipeline, which consisted of phases which are as follows:

7.1. Data Acquisition and Inspection

The process started with loading the batuhanmtl/job-skill-set dataset in a parquet file into a Pandas Data-Frame. To maintain the integrity of the data, a strict cleaning procedure was conducted to detect and delete any rows that had missing values (NaN) in the Job Title, Skills or Description columns. This avoided the possibility of errors in the later process of vectorization.

```
import pandas as pd
import numpy as np
import re
```

```
df = pd.read_parquet("train-00000-of-00001.parquet")
df = df.dropna(subset=['job_title', 'job_skill_set', 'job_description'])
print(df.head())
```

```
       job_id category                                job_title  \
0  3902668440       HR              Sr Human Resource Generalist
1  3905823748       HR                   Human Resources Manager
2  3905854799       HR                Director of Human Resources
3  3905834061       HR              Chief Human Resources Officer
4  3906250451       HR  Human Resources Generalist (Hybrid Role)

                                  job_description  \
0  SUMMARY\nTHE SR. HR GENERALIST PROVIDES HR EXP...
1  BE PART OF A STELLAR TEAM AT YSB AS THE MANAGE...
2  OUR CLIENT IS A THRIVING ORGANIZATION OFFERING...
3  JOB TITLE: CHIEF HUMAN RESOURCES OFFICER (CHRO...
4  DESCRIPTION\n\n WHO WE ARE \n\nAVI-SPL IS A DI...

                                     job_skill_set
0  ['employee relations', 'talent acquisition', '...
1  ['Talent Acquisition', 'Employee Performance M...
2  ['Human Resources Management', 'Recruitment', ...
3  ['talent management', 'organizational developm...
4  ['Microsoft Office', 'Data analysis', 'Employe...
```

Figure 5: Data Acquisition and Inspection

7.2. Dataset Optimization through Thresholding

Preliminary analysis of the job titles showed that there was a lot of noise in the data with most of the job titles only occurring once or twice, and thus impossible to be learned by the AI. A frequency threshold of 10 was used to address this. The fact that the dataset was filtered to only contain job titles that had 10 or more samples reduced the scope of the model to a set of learnable categories, which greatly increased the overall accuracy of the final classification.

```
job_counts = df['job_title'].value_counts()
threshold = 10
titles_to_keep = job_counts[job_counts >= threshold].index

df_filtered = df[df['job_title'].isin(titles_to_keep)].copy()

print(df_filtered.head())
```

```
        job_id category                    job_title  \
1   3905823748       HR     Human Resources Manager
5   3901389277       HR     Human Resources Manager
6   3902348043       HR  Human Resources Generalist
8   3891070825       HR  Human Resources Generalist
12  3894573937       HR  Human Resources Generalist

                                     job_description  \
1   BE PART OF A STELLAR TEAM AT YSB AS THE MANAGE...
5   JOB DESCRIPTION: · THE HR MANAGER WILL SUPPORT...
6   DRIVE YOUR FUTURE WITH TURN 14 DISTRIBUTION! N...
8   DIRECT-HIRE, $65,000 SALARY\nTHE IDEAL CANDIDA...
12  COMPANY INFORMATION\n\nFOR MORE THAN 20 YEARS,...

                                       job_skill_set
1   ['Talent Acquisition', 'Employee Performance M...
5   ['HR management', 'talent acquisition', 'labor...
6   ['Microsoft Office', 'communication', 'attenti...
8   ['recruitment', 'payroll administration', 'com...
12  ['HRIS systems', 'Microsoft Office', 'data ana...
```

Figure 6: Cleaning the data

7.3. Feature Engineering and Text Normalization

The raw text was cleaned, and a feature creation pipeline was run before training:

- **Contextual Merging –** the columns of job_skill_set and job_description were merged into one feature. This gave the model specific technical keywords, and the general contextual language applied in job advertisements.

- **Normalization –** a custom cleaning operation transformed all the text to lowercase and used regular expression (re) to remove punctuations and additional whitespaces.

- **N-gram Vectorization –** the text was cleaned and then converted with the help of TF-IDF. More importantly, the ngram range of (1, 2) was employed, and the model treated phrases such as data scientist or project manager as tokens and not individual and unrelated words.

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
```

```
def clean(text):
    text = str(text).lower()
    text = re.sub(r'[^\w\s]', ' ', text)
    text = " ".join(text.split())
    return text

df_filtered['job_title'] = df_filtered['job_title'].str.lower()
df_filtered['clean_skills'] = df_filtered['job_skill_set'].apply(clean)
df_filtered['clean_desc'] = df_filtered['job_description'].apply(clean)

df_filtered['combined_features'] = df_filtered['clean_skills'] + " " + df_filtered['clean_desc']
df_final = df_filtered[['combined_features', 'job_title']].copy()
df_final.columns = ['skills', 'job_title']

tfidf = TfidfVectorizer(max_features=2000, stop_words='english', ngram_range=(1, 2))

print(df_final.head())
```

```
                                              skills  \
1   talent acquisition employee performance manage...
5   hr management talent acquisition labor complia...
6   microsoft office communication attention to de...
8   recruitment payroll administration compliance ...
12  hris systems microsoft office data analysis pr...

                   job_title
1     human resources manager
5     human resources manager
6   human resources generalist
8   human resources generalist
12  human resources generalist
```

Figure 7: Feature Engineering & Text Normalization

7.4. Model Training & Evaluation

Three supervised learning algorithms were trained on an 80/20 split of the optimized dataset to compare the performance:

- **Naive Bayes –** this is used to provide a probabilistic baseline in text classification.
- **Logistic Regression –** provided a more intensive linear method that is more accurate.
- **Random Forest –** this was an ensemble of 200 decision trees that were used to capture non-linear relationships among skills.

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

X = tfidf.fit_transform(df_final['skills'])
y = df_final['job_title']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)
nb_pred = nb_model.predict(X_test)
nb_acc = accuracy_score(y_test, nb_pred)

lr_model = LogisticRegression(max_iter=1000)
lr_model.fit(X_train, y_train)
lr_pred = lr_model.predict(X_test)
lr_acc = accuracy_score(y_test, lr_pred)

rf_model = RandomForestClassifier(n_estimators=200, random_state=42)
rf_model.fit(X_train, y_train)
rf_pred = rf_model.predict(X_test)
rf_acc = accuracy_score(y_test, rf_pred)

print("--- Model Accuracy Results ---")
print(f"Naive Bayes Accuracy: {nb_acc * 100:.2f}%")
print(f"Logistic Regression Accuracy: {lr_acc * 100:.2f}%")
print(f"Random Forest Accuracy: {rf_acc * 100:.2f}%")
```

```
--- Model Accuracy Results ---
Naive Bayes Accuracy: 42.31%
Logistic Regression Accuracy: 57.69%
Random Forest Accuracy: 65.38%
```

Figure 8: Model Training & Evaluation

7.5. Interactive Interface and Confidence Logic

The last phase was the creation of Command Line Interface (CLI) to enable the model to be accessible to users:

- **Inference Logic –** a recommend_job() procedure was constructed to clean the user input and convert it to the same TF-IDF space as the training data and make a prediction.
- **Probability Scoring –** the system does not merely give a job title but rather the system uses predict_proba to determine a match confidence percentage. This will enable the user to view the degree of match between their skills and the most probable suggestion of the AI.

```
def recommend_job(input_skills):
    cleaned_input = clean(input_skills)
    input_vector = tfidf.transform([cleaned_input])

    prediction = rf_model.predict(input_vector)
    probabilities = rf_model.predict_proba(input_vector)
    confidence = np.max(probabilities) * 100

    return prediction[0], confidence
```

```
print("Welcome to the AI Career Assistant!")
print("Enter 'quit' to exit.")

while True:
    user_input = input("\nPlease enter your skills (separated by commas): ")

    if user_input == 'quit':
        print("Closing application. Good luck with your job search!")
        break

    job, score = recommend_job(user_input)

    print("-" * 30)
    print(f"AI ANALYSIS:")
    print(f"Based on your skills, you are a great fit for: {job.upper()}")
    print(f"Match Confidence: {score:.2f}%")
    print("-" * 30)
```

```
Welcome to the AI Career Assistant!
Enter 'quit' to exit.

Please enter your skills (separated by commas): ↑↓ for history. Search history with c-↑/c-↓
```

Figure 9: Interactive Interface & Confidence logic

8. CONCLUSION

A viable job recommendation pipeline using the batuhanmtl/job-skill-set dataset was effectively created and tested in this project. The project was transformed to a simple keyword-matching system to a context-sensitive machine learning system. The frequency threshold of 10 was used to remove noise caused by singleton classes and this gave a more consistent training environment.

The suggested solution will solve the key issues of the contemporary recruitment environment by creating actionable hints to align the candidates with the possible positions. The application uses the probability output of the model (predict_proba) to give a score of the Match Confidence, which can be ranked by top-K and saves time that would otherwise be spent on screening resumes manually. Moreover, the tool provides a more comprehensive skill-to-job mapping by combining job titles and descriptions, and this is more holistic and considers the context in which a skill is applied, hence, making it possible to make better hiring and career development decisions.

Future work should be aimed at improving robustness and semantic depth. Although the existing threshold-based method achieved much better accuracy, the next round can be based on hyperparameter optimization with stratified cross-validation to make sure that the 65.38% accuracy is stable on all folds of data. Also, the incorporation of Deep Learning embeddings (e.g., BERT or Word2Vec) might enhance the level of semantic comprehension, particularly in the case of low-confidence inputs, such as the one in the case of the salary survey, which was noted during testing. Lastly, the system must be audited in terms of fairness and A/B tested to the user to guarantee that any offline metric gains are reflected in the production environment as unbiased and effective career recommendations.

9. BIBLIOGRAPHY

Alam, M., 2024. *Data Science Dojo.* [Online] Available at: https://datasciencedojo.com/blog/random-forest-algorithm/ [Accessed 16 December 2025].

Anon., 2022. *machine learning plus.* [Online] Available at: https://www.machinelearningplus.com/predictive-modeling/how-naive-bayes-algorithm-works-with-example-and-full-code/ [Accessed 16 December 2025].

Batuhanmtl, 2024. *Hugging Face.* [Online] Available at: https://huggingface.co/datasets/batuhanmtl/job-skill-set/viewer/default/train?views%5B%5D=train [Accessed 14 December 2025].

IEEE, 2024. *IEEE Explore.* [Online] Available at: https://ieeexplore.ieee.org/document/10714763 [Accessed 16 December 2025].

Lee, F., n.d. *IBM.* [Online] Available at: https://www.ibm.com/think/topics/logistic-regression [Accessed 16 December 2025].

Martinez, J., 2018. *Research Gate.* [Online] Available at: https://www.researchgate.net/publication/324652918_Recommendation_of_Job_Offers_Using_Random_Forests_and_Support_Vector_Machines [Accessed 16 December 2025].

Zhang, M., n.d. *Tsinghua.edu.cn.* [Online] Available at: https://keg.cs.tsinghua.edu.cn/jietang/publications/SIGIR14-Zhang-et-al-cold-start-recommendation.pdf [Accessed 16 December 2025].