Machine Learning Project

Report on

ACE THE SPACE

Modern way of Acing the job market

By

GARIMA MATHUR

VARUN PRASANNA RAO

AMIT BALASAHEB GANGANE

Supervisor

Dr Rahul Makhijani

Master's in Science of Business Analytics

University of California, Davis

1 Shields Ave, Davis

CA - 95616

Table of Content

List of Tables

List of Plots

1. Introduction

1.1 Problem Statement

In today’s fast-paced job market, having a resume that meets employer expectations is essential. Our recommendation system helps bridge the gap between what candidates offer and what employers need. By analysing resumes through advanced algorithms, we evaluate skills, experience, and education, comparing them with current job market trends. Our solution doesn’t just identify areas for improvement, it provides practical suggestions, like skills to acquire or achievements to highlight, helping candidates stand out. With features like detailed resume analysis, market trend comparisons, and personalized recommendations, we make it easier for job seekers to craft resumes that truly align with in-demand roles. Future updates will bring real-time insights to keep candidates ahead in the ever-changing job landscape.

1.2 Objectives

1. **Automate Resume Scoring**– Use Natural Language Processing (NLP) to extract and analyse skills, job titles, and experience from resumes.
2. **Enhance Candidate-Job Matching** – Implement machine learning models to match resumes with the most relevant job descriptions based on skill similarity and job role alignment.
3. **Identify Skill Gaps** – Highlight missing skills in resumes that are required for top job matches, helping candidates upskill and improve job readiness.
4. **Optimize Recruitment Efficiency** – Reduce the manual effort in shortlisting candidates, enabling recruiters to focus on high-potential profiles.
5. **Provide Personalized Job Recommendations** – Suggest the top 5 most suitable job roles for each candidate based on their resume content and job market trends.
6. **Evaluate & Improve Matching Accuracy** – Measure model performance using accuracy, precision, recall, and F1-score, ensuring high-quality recommendations.

2. Dataset Description

2.1 Resume Dataset

This dataset contains resumes along with extracted information such as skills, job titles, experience, and education.

Column Name	Description
Name	Candidate's name
Job Title	Current or previous job title
Extracted Skills	Extracted skills from the resume
Experience (Years)	Number of years of experience
Education	Highest degree earned
Certification	Any relevant certifications
Location	Candidate’s location

2.2 Job Description Dataset

This dataset contains job postings with details such as job title, skills required, job description, and salary.

Column Name	**Description**
Company Name	Name of the company posting the job
Job Title	Name of the job role
Job Description	Detailed job responsibilities and qualifications
Required Skills	List of skills needed for the job
Max Salary	Maximum salary offered for the role
Location	Job location
Currency	Currency in which salary is offered

3. Data Preprocessing

3.1 Handling Missing Values

- The **Resumes Dataset** contains 2,000 records and 7 features. All columns, except the **Certification** column, have no missing values. The **Certification** column has 517 missing values, indicating that not all resumes provide information on certifications.

- The **Job Descriptions Dataset** consists of 24,544 records and 8 features. However, several columns have missing values. Specifically, the **company_name** column has 478 missing entries, the **description** column is missing 2 values, and the **fips** column has 5,272 missing entries. More significantly, the **max_salary**, **currency**, and **normalized_salary** columns each have many missing values, with 19,143, 17,857, and 17,857 missing values, respectively. The **title** and **location** columns, on the other hand, do not have any missing values. These missing values need to be addressed in the preprocessing phase to ensure that the dataset is complete and ready for model training.

3.2 Text Cleaning & Preprocessing

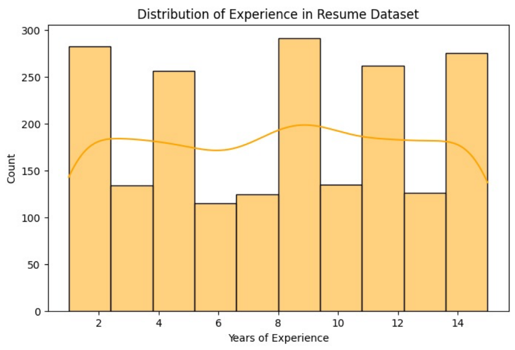
We applied text preprocessing techniques to the resume and job description datasets to prepare them for machine learning analysis.

- **Tokenization**: The **BERTembedding** automatically performs tokenization by splitting text into words or tokens.
- **Stopword Removal**: The BERTembedding is configured with stop_words='english', which removes common, unimportant words like "the," "is," and "and" that don't contribute much to the analysis.
- **Lemmatization**: While lemmatization isn't explicitly mentioned in the code, it is generally part of text preprocessing in NLP. However, in the code, this can be implicitly handled by configuring the vectorizer with stop_words='english' and by considering unigrams and bigrams that capture meaningful word combinations.
- **Vectorization**: The **BERT embedding** is used to convert text data into numerical format. It transforms the text in the **"Skills"** column from the resume dataset (**resumes_df['Skills']**) and the **"description"** column from the job description dataset (**job_descriptions_df['description']**) into dense vector representations.

4. Exploratory Data Analysis (EDA)

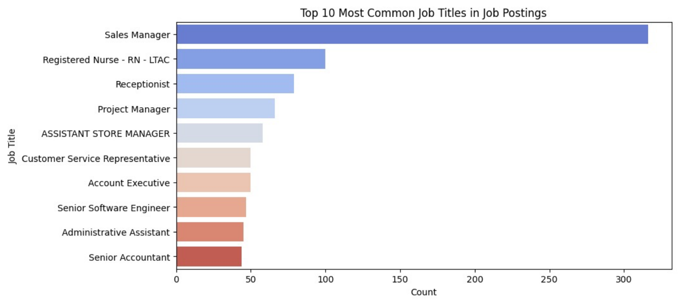
Here's the complete EDA breakdown with the **4.1** format, including the requested sections:

4.1 Distribution of Experience in Resume Dataset



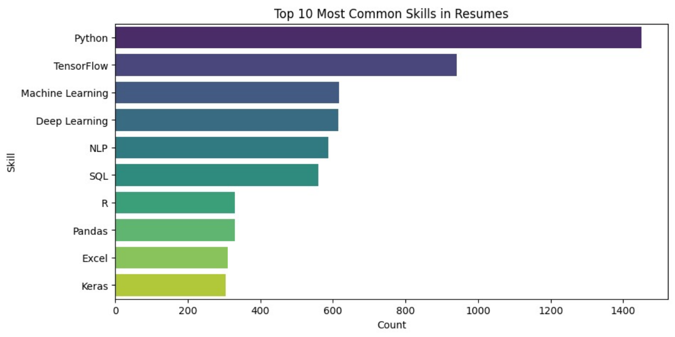
Insight: This plot shows the distribution of years of experience in the resume dataset. The data is evenly spread across different experience levels, with peaks at around 2, 4, 8, and 12-14 years.

4.2 Most Required Skills in Job Descriptions



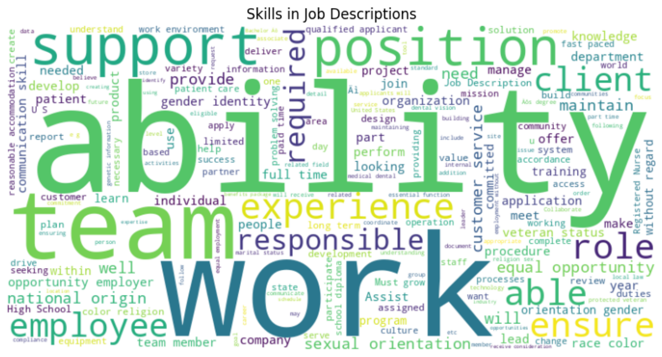
Insight: This bar plot shows the most frequently mentioned skills in job descriptions. It helps in understanding which skills are most in demand across different job listings. This provides actionable insights for candidates to focus on skills that employers are actively seeking.

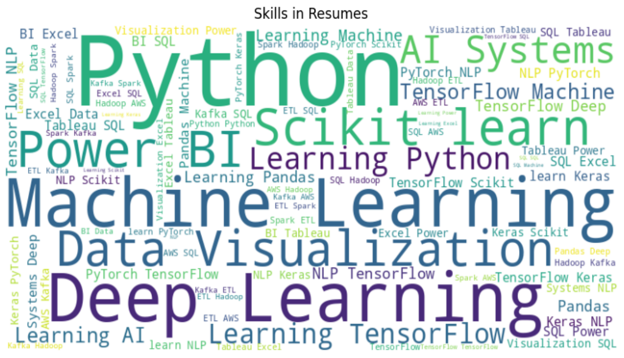
4.3 Most Frequent Skills in Resumes



Insight: This plot displays the most common skills listed in resumes. It helps in understanding which skills are most often emphasized by job seekers. By comparing this with the skills in job descriptions, candidates can identify potential gaps in their resumes.

4.4 Word Cloud of Skills in Resumes and Job Descriptions





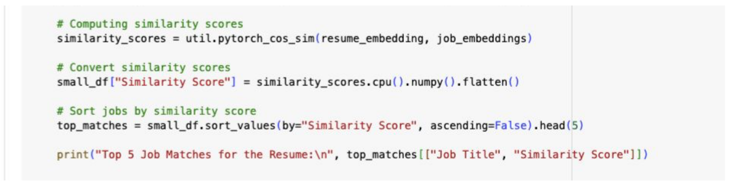
Insight: Word clouds for both resumes and job descriptions give a visual representation of the most common skills mentioned. This helps in quickly identifying key skills that candidates have and those required by employers. Word clouds offer an intuitive way to see trends in skillsets across both datasets.

5. Model Building

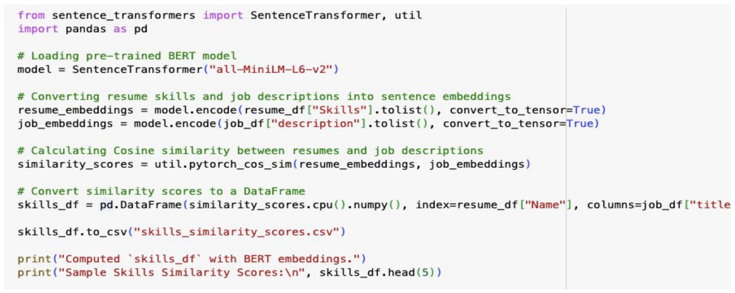
5.1 Feature Engineering

- **Resume Features**: Extracted skills from resumes.
- **Job Features**: Skills required in job descriptions.
- **TF-IDF Vectorization**: Convert text into numerical representation.

5.2 Similarity Calculation



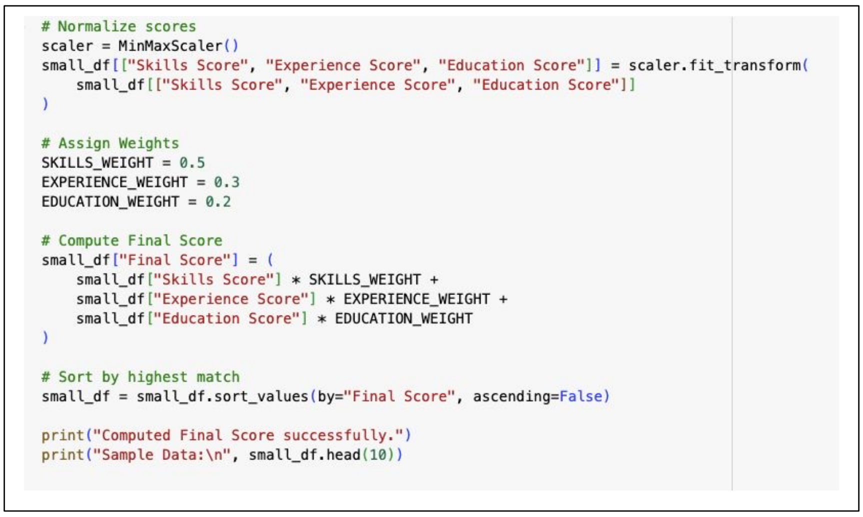
```
# Computing similarity scores
similarity_scores = util.pytorch_cos_sim(resume_embedding, job_embeddings)

# Convert similarity scores
small_df["Similarity Score"] = similarity_scores.cpu().numpy().flatten()

# Sort jobs by similarity score
top_matches = small_df.sort_values(by="Similarity Score", ascending=False).head(5)

print("Top 5 Job Matches for the Resume:\n", top_matches[["Job Title", "Similarity Score"]])
```



```
from sentence_transformers import SentenceTransformer, util
import pandas as pd

# Loading pre-trained BERT model
model = SentenceTransformer("all-MiniLM-L6-v2")

# Converting resume skills and job descriptions into sentence embeddings
resume_embeddings = model.encode(resume_df["Skills"].tolist(), convert_to_tensor=True)
job_embeddings = model.encode(job_df["description"].tolist(), convert_to_tensor=True)

# Calculating Cosine similarity between resumes and job descriptions
similarity_scores = util.pytorch_cos_sim(resume_embeddings, job_embeddings)

# Convert similarity scores to a DataFrame
skills_df = pd.DataFrame(similarity_scores.cpu().numpy(), index=resume_df["Name"], columns=job_df["title

skills_df.to_csv("skills_similarity_scores.csv")

print("Computed `skills_df` with BERT embeddings.")
print("Sample Skills Similarity Scores:\n", skills_df.head(5))
```

- **Cosine Similarity** is used to measure how closely a resume matches a job.

5.3 Matching scores for Resumes to Jobs



```
# Normalize scores
scaler = MinMaxScaler()
small_df[["Skills Score", "Experience Score", "Education Score"]] = scaler.fit_transform(
    small_df[["Skills Score", "Experience Score", "Education Score"]]
)

# Assign Weights
SKILLS_WEIGHT = 0.5
EXPERIENCE_WEIGHT = 0.3
EDUCATION_WEIGHT = 0.2

# Compute Final Score
small_df["Final Score"] = (
    small_df["Skills Score"] * SKILLS_WEIGHT +
    small_df["Experience Score"] * EXPERIENCE_WEIGHT +
    small_df["Education Score"] * EDUCATION_WEIGHT
)

# Sort by highest match
small_df = small_df.sort_values(by="Final Score", ascending=False)

print("Computed Final Score successfully.")
print("Sample Data:\n", small_df.head(10))
```

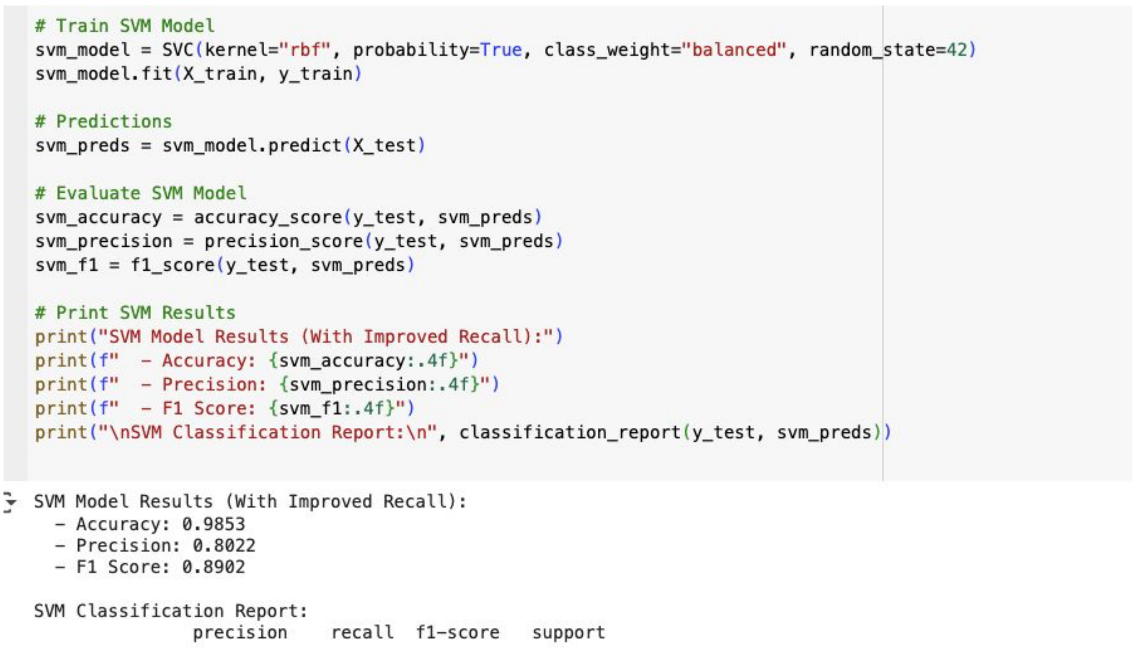
6. Model Evaluation

6.1 Performance Metrics

To evaluate the model, we compare the predicted job titles with actual job roles using:

Models	**TF-IDF with RF**	**Bert with RF**	**Bert with SVM (Best Model)**
Accuracy	0.2970	0.5149	0.9853
Precision	0.6373	0.7529	0.8022
F1-score	0.2970	0.5149	0.8902

6.2 Model Performance



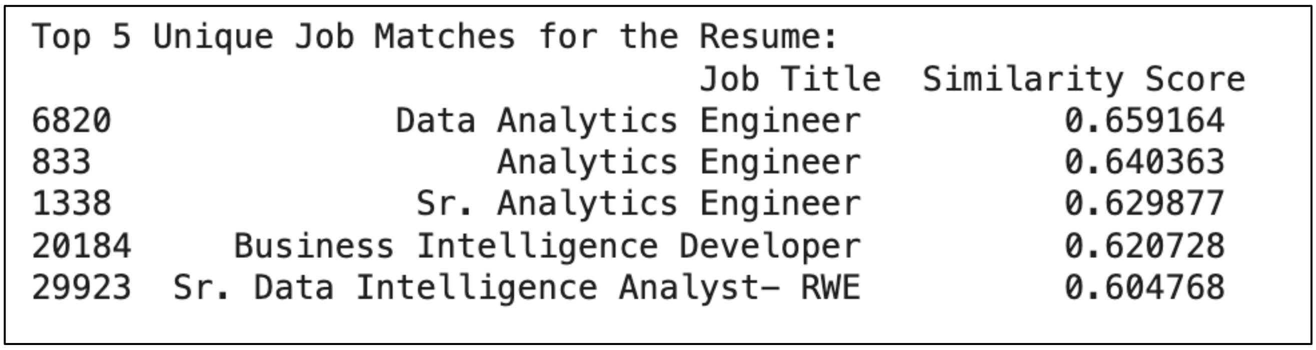
```
# Train SVM Model
svm_model = SVC(kernel="rbf", probability=True, class_weight="balanced", random_state=42)
svm_model.fit(X_train, y_train)

# Predictions
svm_preds = svm_model.predict(X_test)

# Evaluate SVM Model
svm_accuracy = accuracy_score(y_test, svm_preds)
svm_precision = precision_score(y_test, svm_preds)
svm_f1 = f1_score(y_test, svm_preds)

# Print SVM Results
print("SVM Model Results (With Improved Recall):")
print(f"  - Accuracy: {svm_accuracy:.4f}")
print(f"  - Precision: {svm_precision:.4f}")
print(f"  - F1 Score: {svm_f1:.4f}")
print("\nSVM Classification Report:\n", classification_report(y_test, svm_preds))
```

```
SVM Model Results (With Improved Recall):
  - Accuracy: 0.9853
  - Precision: 0.8022
  - F1 Score: 0.8902

SVM Classification Report:
               precision    recall  f1-score   support
```

7. Results

7.1 Key Findings

- The model successfully **matches resumes to job descriptions** using textual similarity.
- The system identifies **missing skills** that job seekers can improve.
- Skill gaps are prevalent in areas such as **Python, Cloud Computing, and Data Science**.



```
Top 5 Unique Job Matches for the Resume:
                                 Job Title  Similarity Score
6820              Data Analytics Engineer          0.659164
833                    Analytics Engineer          0.640363
1338               Sr. Analytics Engineer          0.629877
20184      Business Intelligence Developer          0.620728
29923  Sr. Data Intelligence Analyst- RWE          0.604768
```

8. Conclusion & Future Work

8.1 Conclusion

Navigating today's dynamic job market can be challenging, but our AI-powered job role prediction system simplifies the process. By leveraging Natural Language Processing (NLP) and Machine Learning, it analyses skills and job descriptions to accurately match candidates with the most suitable roles. This smart, data-driven approach helps job seekers find the perfect fit while enabling employers to connect with top talent—making hiring faster, fairer, and more efficient.

➢ What makes this project impactful?

- **Efficiency**: Automates manual job-matching efforts.
- **Scalability**: Can be expanded for multiple industries and roles.
- **Future-Ready**: With advancements like BERT and Deep Learning, accuracy and relevance can be further improved.

8.2 Future Enhancements

❖ **Enhancing Recommendation System for Future Implementation**

- To further improve our job role prediction system, we plan to integrate **personalized job recommendations** that align more closely with market demands and individual career trajectories.

❖ **Exploring Deep Learning Models for Advanced Textual Analysis**

- To enhance the accuracy and efficiency of job role prediction and resume matching, we aim to integrate **deep learning models** for more advanced **textual analysis**.

❖ **Expanding Dataset Size for Improved Model Performance**

- We plan to **increase the dataset size**, ensuring the model learns from a diverse and comprehensive set of resumes and job descriptions. A larger dataset will enable better **pattern recognition, contextual understanding, and robust decision-making** in candidate-job matching.