A Candidate-Job Matching Prototype Using NLP and Machine Learning

Objectives

The primary objective of this project was to develop a prototype system for matching job seekers' resumes with relevant job descriptions using Natural Language Processing (NLP) and Machine Learning techniques. The goal was to create an efficient and accurate method for identifying suitable job opportunities based on a candidate's skills and experience.

Task Description

The task involved creating a system that could process and analyze both resumes and job descriptions. Convert text data into a format suitable for comparison (vectorization).Calculate similarity scores between a resume and multiple job descriptions.Evaluate the effectiveness of the matching process and provide insights into the matching results through various analyses and visualizations.

Approach to Solving the Problem

Data Preparation

One of my first tasks was to clean and preprocess the data. The Jobs were in a CSV format with several inconsistencies and unenecessary information that would increase the processing time and reduce the accuracy of the results.

Therefore using NLTK data the job descriptions and resume text were preprocessed.

- Convert to Lowercase: In order to ensure uniformity in text by lowering all characters
- Remove Non-Alphabetic Characters:Strips out numbers and special symbols.
- Tokenization: Breaks down text into individual words.
- Stopwords Removal: Removes common words (like "the", "is", "and") which contribute little to the overall results.
- Lemmatization: Reduces words to their base form (e.g., "running" to "run"), improving consistency.

The processed data was combined for each job listing to consolidate relevant information.

Vectorisation

The data currently was in a text format and had to be vectorised. After much research it was decided the the TF-IDF function would be used to represent text data numerically.

TF-IDF is a statistical measure used to evaluate the importance of a word in a document within a collection of documents. It consists of two parts:
Term Frequency (TF): This measures how frequently a term appears in a document. It's calculated as:

- TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document)
- Inverse Document Frequency (IDF): This measures how important a term is across all documents. It's calculated as: IDF(t) = loge(Total number of documents / Number of documents with term t in it)

The TF-IDF score for a term is the product of its TF and IDF scores: TF-IDF = TF * IDF

Matching Algorithm

After vectorization, I decided to use cosine similarity to compare the resume vector with job description vectors. Cosine similarity measures the cosine of the angle between two vectors, providing a similarity score between -1 and 1. In the context of TF-IDF vectors, this score is always between 0 and 1, where 1 indicates perfect similarity and 0 indicates no similarity. The cosine similarity is calculated as:

$$cos(\theta) = (A * B)/(||A|| * ||B||) \tag{1}$$

This approach allows us to quantify how similar the resume is to each job description, forming the basis of our matching system.

Evaluation Methods

To assess the performance of our job matching prototype without labeled data, I had to figure out various different evaluation methods. Unfortunately I could not find any corresponding labelled data to compare accuracy against so I had to figure out more creative methods to correctly evaluate data. Eventually I decided to use 4 key evaluation methods:

- Cross-Validation: The job description was split into training (80%) and test (20%) sets, comparing the resume's similarity scores across both. This would help assesses how well our model generalise to unseen data and detect any overfitting
- Clustering Analysis: K-Means clustering was used to group similar job descriptions into 5 clusters. This was decided because clustering helps identify which job types the resume matches best.
- Keyword Overlap Analysis: This method would calculate how many shared word existed between the resume and each job description, providing relevance between the two
- Feature Importance Analysis: The top TF-IDF score in the resume vector were identified as it would interpret why certain scores were higher.

These methods were chosen for their ability to provide a comprehensive evaluation of the matching system without requiring labeled data. They combine quantitative metrics with qualitative insights, allowing me to assess technical performance, understand semantic groupings, and provide actionable insights for resume improvement.

Visualisation

After evaluation I decided to visualise the results so that a much more graphic representation could provide a better understanding of how well the prototype was performing.

Visualisation methods such as Historgrams and Barplots were used to present the data.

Implementation Details

Used Python with libraries such as scikit-learn, numpy, and pandas for data processing and analysis. Employed TF-IDF vectorization with 491 features for both resume and job descriptions. Implemented K-means clustering with 5 clusters to group similar job descriptions. Utilized matplotlib and seaborn for data visualization.

Results and Analysis

Cross-Validation

- **Results:**
  - Average similarity on training set: 0.0480
  - Average similarity on test set: 0.0496
- **Analysis:** The average cosine similarity between the resume and job description is low for both training and test set. The similarity is slightly higher in the test set, which is good as it shows the model generalises well. But the low similarity suggests the resume might not match the job description that well.

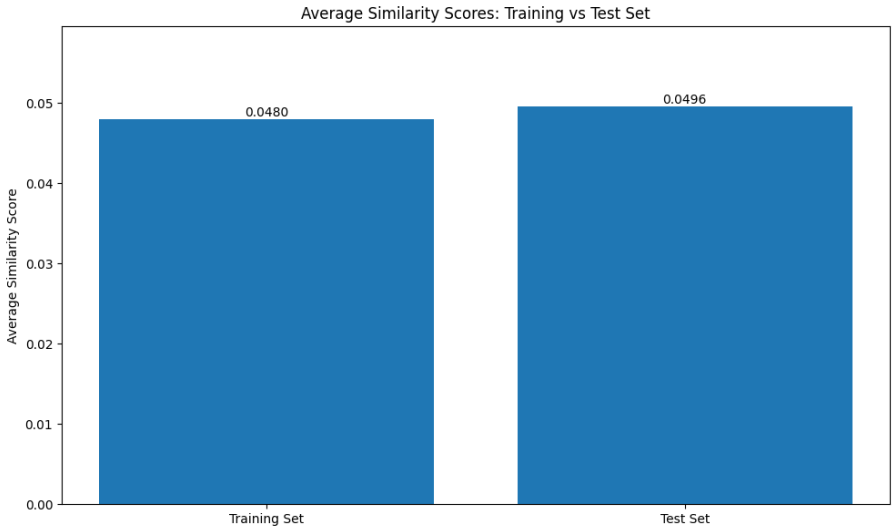

Figure 1: Caption

Clustering Analysis

- **Results:**
  - Cluster 3 (support, data, business) showed the highest average similarity (0.1319).
- **Analysis:** As can be seen the resume has higher similarity with the cluster water management and environmental services and the Business support and data-related roles (13.19%). This is very crucial for us to know as we can tell that this resume is well suited for this type of role.

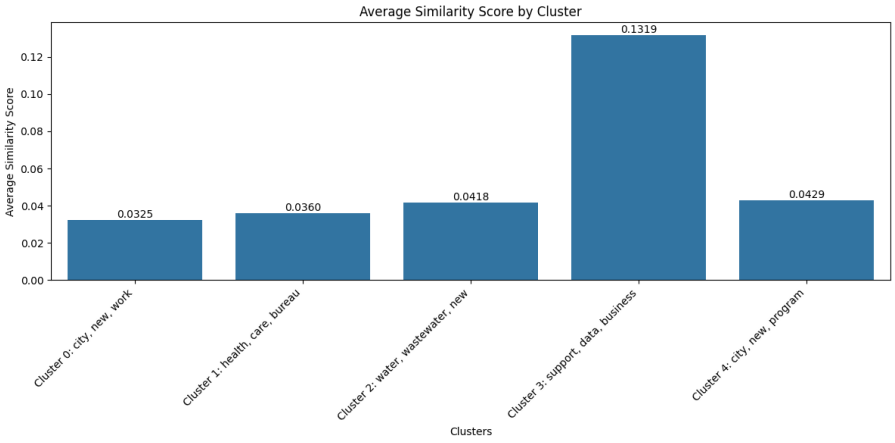

Figure 2: Caption

Keyword Overlap Analysis

- **Results:**
  - Average keyword overlap: 50.97
  - Maximum keyword overlap: 128
- **Analysis:** We can see that on average about 51 words from the resume appear in each job description. The job with the highest overlap shares 128 words with the resume. This suggests a moderate level of keyword matching.

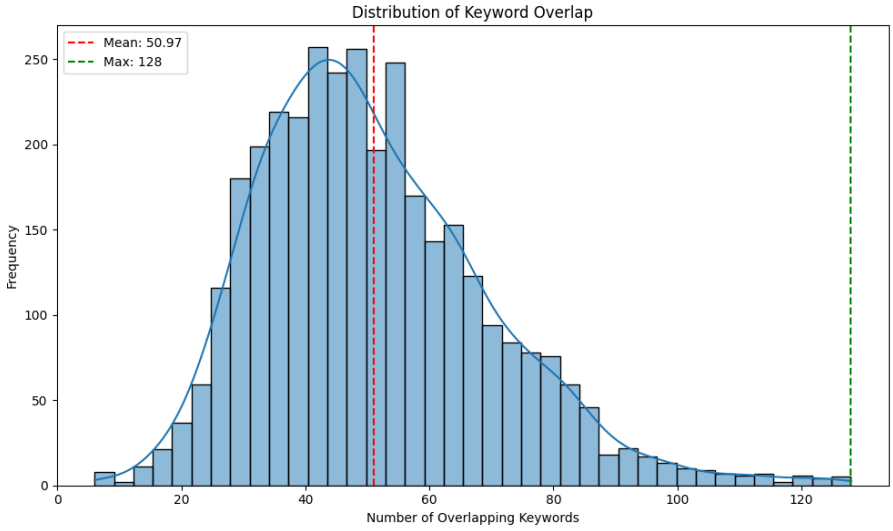

Figure 3: Caption

Feature Importance Analysis

- **Results:** Top features included sql, pl, oracle, table, script.
- **Analysis:** We can see that the most important terms in the resume according to the TF-IDF scores emphasise SQL, PL, Oracle databases and scripting skills. WIth the highest feature being the need for the SQL skill (48.8%).

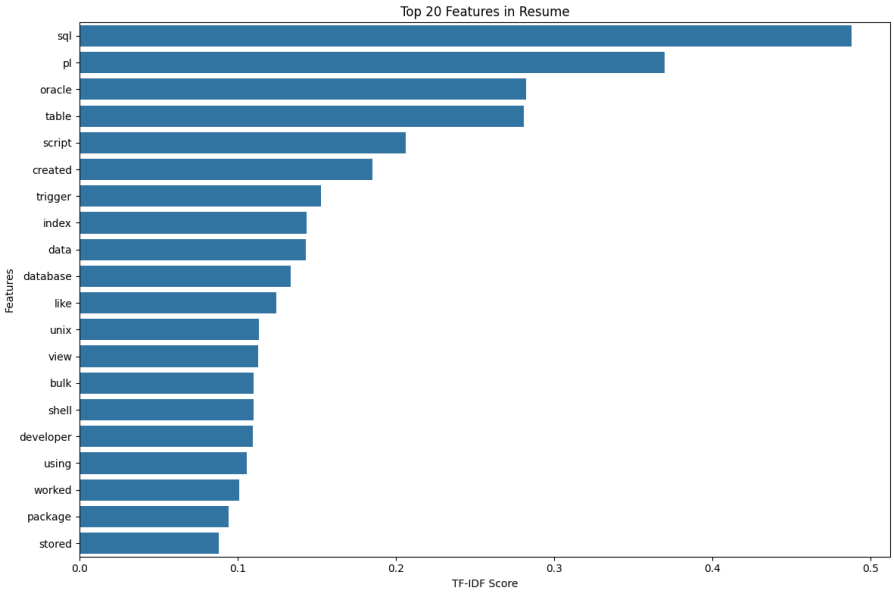

Figure 4: Caption

Challenges Faced

One of the biggest challenge I faced was during the preprocessing stage. The data of the resume and the data of the jobs were in different type formats. This meant that they would have to be extracted and pre-processed seperately. Moreover during the vectorisation phase I had to ensure I did more in depth research to ensure that the jobs were not vectorised in the a 2D array format while the resume data being vectorised in a list format. Since they both had to be uniform, I decided that the data from the jobs would all be combined and converted to a list format before vectorisation. The balance between capturing important details and avoiding feature selection was crucial. After that, the cosine similairty was relatively easy but the difficulty came again during evaluation. As mentioned before I undertook a lot of research but I could not find other similar results I could compare with. I also spent a lot of time at other job boards, trying to use their API calls to get a list of job data myself. I tried Jooble, Adzuna, Reed etc, however either the APIs had limitiation as to what could be extracted or there was some legal barriers to web scraping present. This meant that I was limited to the Job CSV file I was provided and could not test out this model with other job datas. This reduced my chance at a fair evaluation, as it would have been a good test to check how the model would generalise with multiple different job boards on the same resume. I did however have access to multiple other resumes which I tested and was able to get a good idea for accuracy. However since the model depended mainly on training the keyword from the job data it was of no practical use to try multiple resumes except to double check if the model performed on different datasets. However, after researching ways to check overfitting it was decided to use the evaluation methods described above as a way to check the model. It was found the model generalised well and that it was quite accurate in determining the type of job the resume was suitable for.

Conclusion and Future work

The prototype demonstrates potential in matching resumes to job descriptions, particularly in identifying relevant job clusters. However, the overall low similarity scores suggest room for improvement. Future work could include incorporating more advanced NLP techniques (e.g., word embeddings, transformer models such as BERT) could increase the accuracy of the model. I did not have the time to test out other models and compare them, but I am sure that would be a fair and proper evaluation. Additionally developing a method for obtaining labeled data to enable more rigorous evaluation would be very useful. If other labelled data was found online or generated over time, the model could be compared and any bias or overfitting could be solved. I also think a user inteface with the matching system would be much easier to access and understand. Finally exploring ways to incorporate srtucture data such as years of experience, education level etc into the matching process would add an extra level of accuracy into the model. It would ensure the model was robust and able to adapt to any type of position. Implementing a user interface for easy interaction with the matching system.