AI Project: Resume Description Matcher

**Project Description:**

In today's competitive job market, efficiently matching job seekers with suitable positions is crucial for both employers and candidates. Our proposed project aims to tackle this challenge by developing an intelligent system that leverages natural language processing (NLP) techniques to match resumes to job requirements. By analyzing the content of resumes and job descriptions, our system will identify the most relevant candidates for specific job roles. And for this we train 2 models one by generating embeddings using TF-IDF and one through BERT. Our model take Resume and Description and tell they Matched or Not Matched.

**Work Flow:**

First collect data from data sources clean and preprocess it and train model and then test model.

**Data Set Sources and Preprocessing Steps:**

1. Resume Data Collection:
   * Scraping from Websites: Utilize web scraping techniques to gather resume data from various job portals, career websites, and online platforms where users upload their resumes.
   * API Integration: Access APIs provided by HuggingFace.
   * University Students: Collaborate with universities to collect resumes from students and graduates.
2. Job Requirement Data Collection:
   * Scraping LinkedIn: Scrape job requirements from LinkedIn job postings using web scraping techniques.

Preprocessing Steps for Resume and Job Requirement Data:

1. Text Cleaning:
   * Handle Special Characters: Handle special characters, symbols, and non-alphanumeric characters by either removing them or replacing them with appropriate tokens.
   * Remove Punctuation: Eliminate punctuation marks from the text as they may not contribute significantly to the semantic meaning.
   * Stop Word Removal: Filter out common stop words (e.g., "the," "and," "is") from the text data as they occur frequently but carry little semantic value.
2. Tokenization:
   * Tokenize Text: Split the text into individual words or tokens to create a structured representation of the textual data.
   * Word Tokenization: Use techniques such as whitespace tokenization or word tokenization libraries (spaCy) to segment the text into tokens.
3. Normalization:
   * Convert to Lowercase: Convert all text to lowercase to ensure consistency in text representation and reduce the vocabulary size.
   * Stemming or Lemmatization: Apply stemming or lemmatization techniques to normalize words by reducing them to their base or root form. This helps in reducing the dimensionality of the data and capturing the core meaning of words.
4. Data Vectorization:
   * TF-IDF Vectorization: Convert the preprocessed text data into numerical representations using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. This technique assigns weights to words based on their frequency in a document relative to the entire corpus, capturing their importance in distinguishing documents.
   * Secondly we use BERT to generate vectors embeddings

**Approaches and Methodologies:**

**Data Collection:**

* Utilized **Selenium** and **Beautiful Soup** for web scraping to collect data from websites.
* Leveraged **Hugging Face API's** to gather additional data.
* Loaded data from PDF and DOC files to augment the dataset.

**Preprocessing:**

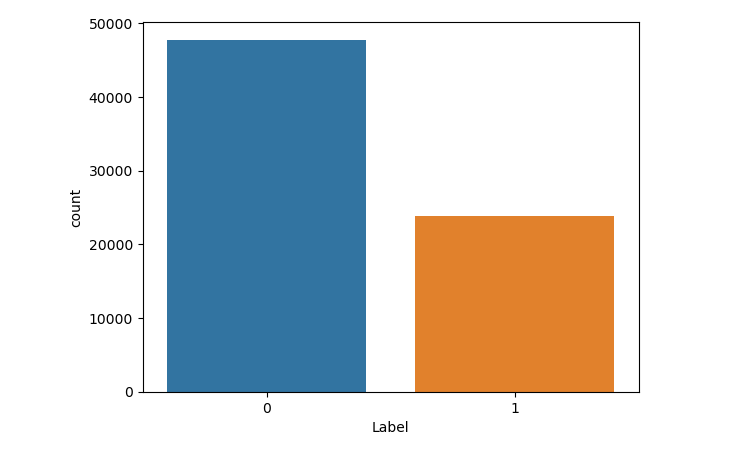
* Used **Spacy** with its English model **en\_core\_web\_sm** for **tokenization**, **stemming**, and **lemmatization**.
* Perform text preprocessing to clean and normalize the data.

**Vector Embeddings:**

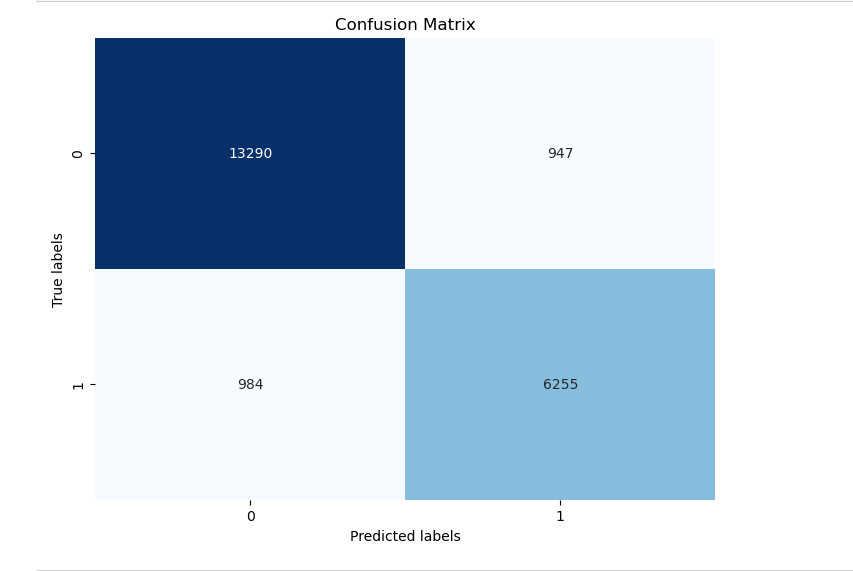
* Employed **TF-IDF** and **BERT** vectorization to generate numerical representations for the data.

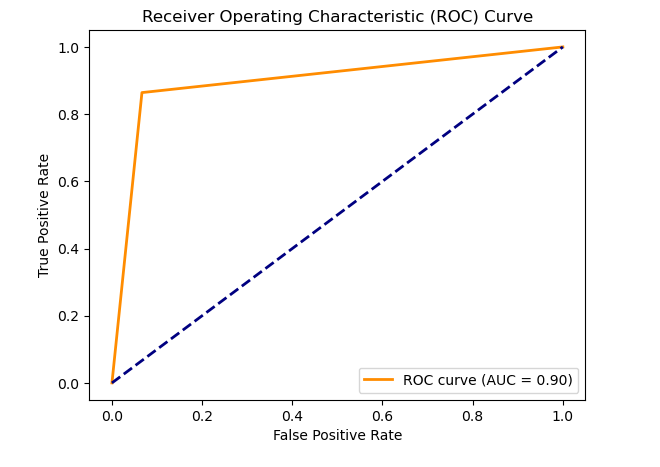
**1- TF-IDF Model:**

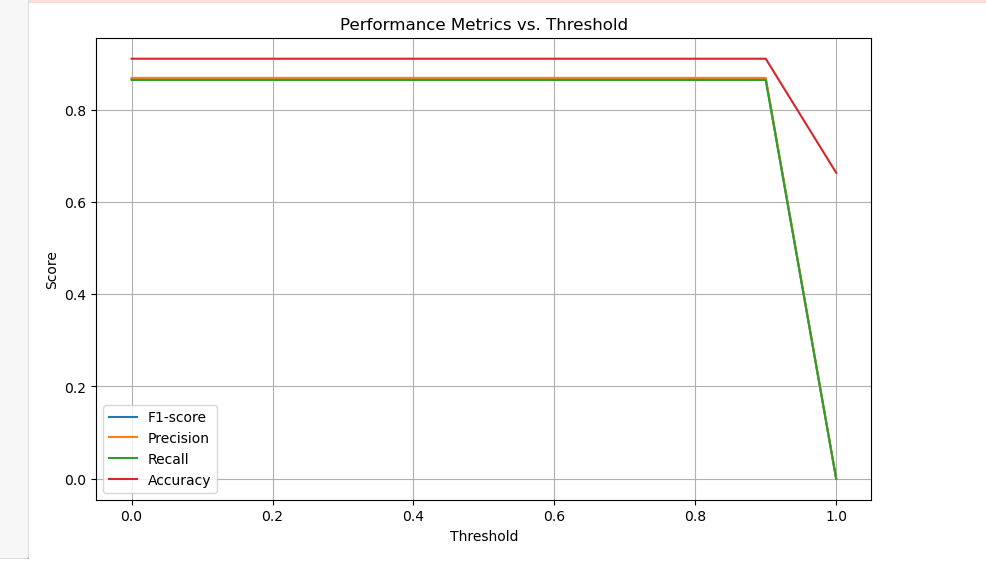
* Compared resumes with job descriptions using TF-IDF embeddings.
* Set a threshold to determine label (0 or 1) based on the cosine similarity score.
* Distribution of our Data set in labels 0 and 1



* Applied **binary** **classification** using **TensorFlow** and **Keras**.
* Utilized a **Dense** **neural** **network** architecture for binary classification.

**Accuracy Measures for TF-IDF: Confusion Metrix**

**ROC -Curve:**

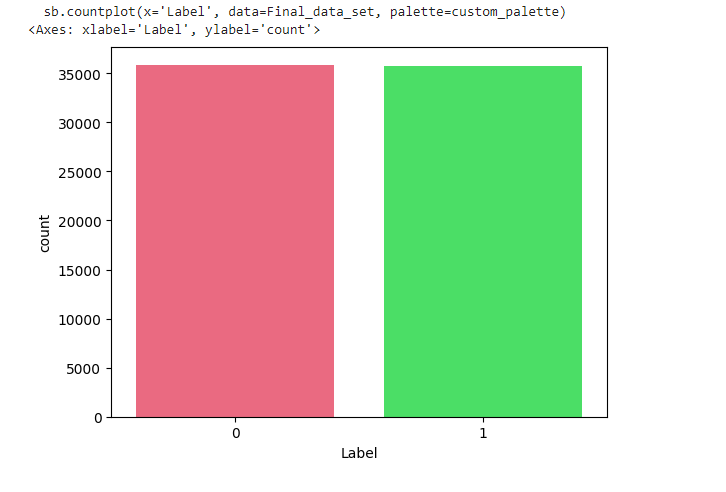
**Performance Grapgh:**

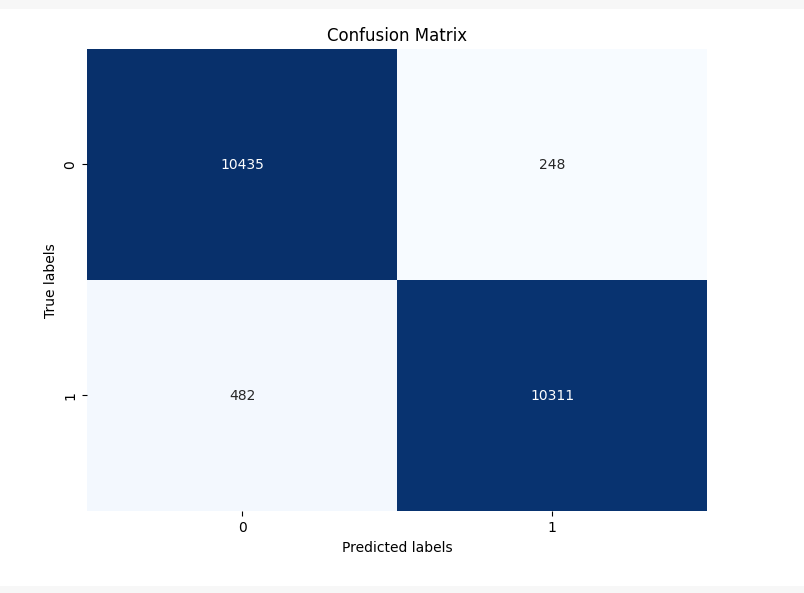
**2-BERT Embeddings:**

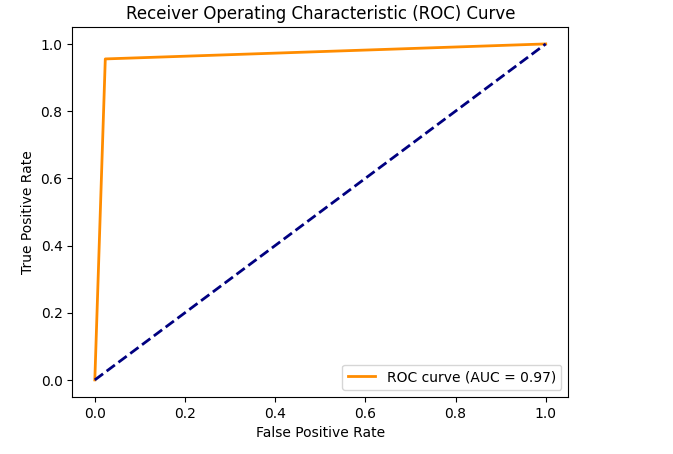
* Utilized **BERT** **embeddings** for contextual representation of the data.

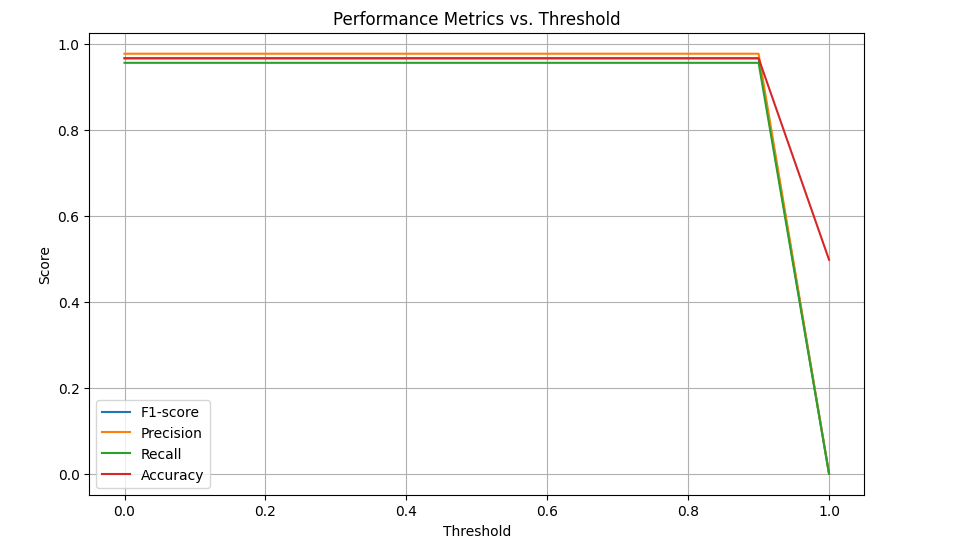
**BERT Model:**

* Matched resumes with job descriptions using BERT embeddings.
* Set a threshold to determine label (0 or 1) based on the cosine similarity score.



**Accuracy Measures for BERT: Confusion Metrix**

**ROC Curve**

**Performance Measures**

* Employed binary classification using TensorFlow and Keras.
* Constructed a Dense neural network architecture for binary classification.

**Challenges Encountered and Solutions:**

**1. Selection of Threshold for Similarity Check:**

* Challenge: Determining the appropriate threshold for cosine similarity to categorize data labels (0 or 1) after obtaining embeddings.
* Solution: Experimented with various threshold values and tested their impact on model performance. Evaluated each threshold by analyzing its effect on accuracy metrics and selected the threshold that yielded the best results.

**2. Unbalanced Initial Data Collection:**

* Challenge: The initial dataset collected from various sources was unbalanced, leading to biased model training.
* Solution: Gathered additional data from diverse sources to balance the dataset. Collected data from multiple sources to ensure a representative sample and enhance model generalization.

**3. Selection of Similarity Measure:**

* Challenge: Identifying the most suitable similarity measure for comparing resumes with job descriptions.
* Solution: Experimented with different similarity measures such as cosine similarity, Manhattan similarity, and Euclidean distance. Evaluated each measure's effectiveness in capturing semantic similarity between documents and selected the one that best described the data characteristics.

**Comparison with Existing Methods:**

* Compared to traditional keyword-based matching systems, our NLP-based approach offers superior accuracy and semantic understanding, resulting in more precise resume-job matches.

**Group Members:**

Muaaz (21L-5467)

Muhammad Hamza(21L-5382)

Ayyad Asif(21L-7561)