Success Criteria Test (Resume ATS Scorer and Skills Matcher)

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr** | **Success Criteria** | **Expected** | **Result** |
| 1 | Developed a Resume ATS Scorer and Skill Matcher tool using Python libraries like spacy and nltk for natural language processing. Successfully built an algorithm to score resumes against job descriptions and match relevant skills across various industries. This project streamlined the candidate screening process by providing actionable insights based on skill sets, helping recruiters improve hiring efficiency and candidate-job compatibility. | **Group 1 Task 1:** Identify a suitable job portal for scraping job descriptions. For example, Scrape job descriptions from "<https://www.examplejobportal.com>". | PASS |
| **Group 1Task 2**: Research and select appropriate web scraping tools and libraries. For example, Consider using BeautifulSoup or Scrapy for extracting data from websites. | PASS |
| **Group 1 Task 3**: Handle website-specific challenges, such as dynamic content or pagination. For example, Create a Python script using BeautifulSoup to extract news articles from “<https://www.examplejobportal.com>” , taking into account the website's structure and complexity. | PASS |
| **Group 1 Task 4**: Identify authentication or access restrictions on the target website. For example, Check if ”<https://www.examplejobportal.com>” requires login, CAPTCHA, or API access for scraping. | FAIL |
| **Group 1 Task 5**: Check for handling of edge cases like missing or malformed data | FAIL |
| **Group 1 Task 6**: Detail the challenges faced during web scraping | FAIL |
| 2 | Designed an efficient NLP preprocessing pipeline utilizing popular libraries such as NLTK, stopwords, and emoji to perform core tasks like lowercasing, punctuation removal, number filtering, emoji handling, stopword removal, and whitespace stripping. Successfully implemented diverse tokenization techniques including word | **Group 2 Task 1:** Test tokenization methods such as word tokenization, sentence tokenization, regex-based tokenization, and whitespace tokenization Example: Verify that NLTK word\_tokenize, NLTK sent\_tokenize, and Python's split function correctly tokenize text as intended**.** | PASS |

|  |  |  |  |
| --- | --- | --- | --- |
|  | tokenization, regex-based tokenization, and whitespace tokenization using functions like NLTK word\_tokenize, NLTK sent\_tokenize, and Python's split function, ensuring clean and structured text data for further analysis. | **Group 2 Task 2**: Process the job articles scraped from “<https://www.examplejobportal.com>” using the developed NLP preprocessing pipeline. | PASS |
| **Group 2 Task 3**: Check that the tokenized output accurately reflects the intended word and sentence boundaries, and confirm the effectiveness of tokenization techniques. | FAIL |
| **Group 2 Task 4**: Include sample text before and after each preprocessing step to illustrate the effects and improvements made by the pipeline. | PASS |
| **Group 2 Task 5**: Implement techniques like stemming and lemmatization on a sample text dataset to prepare it for analysis. | PASS |
| **Group 3 Task 1**: Evaluate and document the performance of each tokenization method on sample text to determine their accuracy and suitability for different types of text data. | PASS |
| **Group 3 Task 2**: Investigate methods for segmenting text into sentences, including library-based techniques and custom implementations. | FAIL |
| 3 | Developed an advanced text preprocessing pipeline incorporating stemming algorithms like Porter Stemmer, Snowball Stemmer, and Lancaster Stemmer, along with lemmatization using the WordNetLemmatizer library, to reduce tokenized words to their base or root forms. Additionally, utilized libraries such as NLTK and spaCy for Part of Speech (POS) tagging, accurately assigning grammatical labels to each tokenized word, thereby enhancing the quality and structure of the processed text for further linguistic analysis. | **Group 3 Task 3**: Apply Porter Stemmer, Snowball Stemmer, and Lancaster Stemmer to a sample dataset to reduce words to their base or root forms. | PASS |
| **Group 3 Task 4**: Utilize NLTK and spaCy for Part of Speech tagging on the preprocessed data, and evaluate the accuracy of grammatical labeling and its impact on text quality. | FAIL |
| **Group 3 Task 5**: Record the results and performance metrics of different tokenization methods (e.g., whitespace tokenization, regex-based tokenization, and NLP library-based tokenization) on the preprocessed data. | FAIL |
|  |  | **Group 3 Task 6:** Verify the correctness of the Document-Term Matrix (DTM) | PASS |

|  |  |  |  |
| --- | --- | --- | --- |
| 4 | Implemented text-to-numerical feature transformation using scikit-learn’s, CountVectorizer and TF-IDFVectorizer libraries, enabling the construction of key matrices such as the Document-Term Matrix (DTM), Term-Document Matrix (TDM), and Term Frequency-Inverse Document Frequency (TF-IDF) matrix. This process facilitated efficient extraction of insights from textual content, improving the model's ability to interpret and analyze text-based data | **Group 4 Task 1:** Use spaCy’s POS tagging capabilities to label tokens in a sample text, utilizing spaCy’s built-in tagger to assign grammatical tags. | PASS |
| **Group 4 Task 2**: Assess how well the TF-IDF matrix supports the extraction of meaningful insights from textual content and improves text-based data analysis. | PASS |
| **Group 4 Task 3**: Apply WordNet to identify and retrieve synonyms for words in the text, enhancing the understanding of semantic relationships. | PASS |
| **Group 4 Task 4**: Analyze and compare the results of POS tagging and WordNet analysis from NLTK, spaCy, or other libraries. | PASS |
| **Group 4 Task 5**: Compare the results from different NLP tools and resources | PASS |
| 5 | Conducted text similarity analysis by collecting five news articles from different websites within the same genre and applying the cosine\_similarity() function to measure similarity between the texts. The higher the cosine similarity value, the smaller the angle between vectors, indicating greater sentence similarity. This approach enabled effective comparison of news content across multiple sources for deeper insights. | **Group 2 Task 6**: Verify that the cosine similarity values accurately reflect the degree of similarity between the text data, with higher values indicating greater similarity. | PASS |