Documentation of Resume Classification

Table of Contents

1. Exploratory Data Analysis………………………………………………….2
2. Data Preprocessing and Feature Extraction………………………………...4
3. Model Selection and Model Training……………………………………….5
4. Results and performance Analysis………………………………………….6
5. Script Development and Instruction………………………………………...7
6. **Exploratory Data Analysis**

The dataset contains the following columns:

1. **ID**: A unique identifier for each entry.
2. **Resume\_str**: The resume content in a string format.
3. **Resume\_html**: The resume content in an HTML format.
4. **Category**: The job category to which the resume belongs.

According to this finding the following exploratory data analysis is done.

 **Class Distribution Analysis**: To understand the distribution of different categories in the dataset, ensuring that the dataset is balanced or identifying the need for handling imbalances. The following figure represents class distribution analysis from the dataset.

A graph of a number of jobs

Description automatically generated

 **Text Length Distribution**: To analyze the variation in the length of the resumes (measured by the number of words or characters) across different categories. The following figure represents text length distribution and their frequency analysis from the dataset.

A graph of a distribution of resume text length

Description automatically generated

 **Missing Data Analysis**: To identify any missing or null values in the dataset that could affect model performance. Missing values in key columns like Resume\_str and Category were analyzed and handled by dropping the affected rows.

 **Word Cloud Visualization**: Word cloud gives a overall visualization of commonly used words across the classes. Word cloud for each class is visualized and this helped in refining the text preprocessing steps, like the removal of common but non-informative phrases. This figure represents word clouds of three classes.



 **N-gram Analysis**: To explore the occurrence of n-grams (combinations of two or more words) within the text data. This analysis represents the frequent words from the dataset. It helped to find important features from frequently used words. Following figure represents the top 20 n-grams in the dataset.

A screenshot of a computer

Description automatically generated

1. **Data Preprocessing and Feature Extraction**

**Preprocessing**:

* **Lowercasing**: All text data is converted to lowercase to maintain consistency and avoid treating the same words differently based on case.
* **HTML Tag Removal**: Any HTML tags present in the resumes are removed using regular expressions to retain only the textual content.
* **Special Character Removal**: Non-alphabetic characters are removed to focus on meaningful textual data.
* **Tokenization**: Text is split into individual words (tokens) using NLTK's word tokenizer.
* **Stop Words Removal**: Common words do not contribute much meaning in terms of training the model. (e.g., "and", "the") are removed using NLTK's stopword list.
* **Lemmatization**: Words are reduced to their base form (e.g., "running" becomes "run") using NLTK’s WordNetLemmatizer, which helps in reducing the dimensionality of the text data.
* **N-gram Removal:** From EDA it is observed that some of words or phrases showed up very frequently which has actually a very little relation in terms of resume classification (e.g. city state, company name etc). These n-grams words were removed for better classification performance. Here is the updated top n-gram words after removal,

A screen shot of a computer

Description automatically generated

**Feature Extraction**:

* **TF-IDF Vectorization**: The cleaned text data is converted into numerical features using the TF-IDF vectorization (Term Frequency-Inverse Document Frequency) method. This approach helps in identifying the importance of words in a document relative to the entire corpus, capturing the significance of words in distinguishing resumes from different categories.
* **N-grams**: Both unigrams and bigrams are used to extract the context from the resumes. This is useful in resume data where job titles, skills, and other phrases play a significant role for classify the data.

1. **Model Selection**

For this task a Voting Classifier combining various base models is selected. This is an ensemble method which involves a prediction considering multiple model’s output.

Ensemble Learning involves using a Voting Classifier to aggregate the forecasts of various models for enhanced effectiveness and resilience in classification tasks. For this task 5 different models were used. These are Logistic Regression, SVM, RandomForest, Gradient Boosting, and Multinomial Naive Bayes. By using a mix of models such as linear, tree-based, and probabilistic ones, the Voting Classifier is able to take into account different characteristics of the data, resulting in improved ability to predict on new data. By using soft voting by averaging predicted probabilities from each model helps the ensemble make better decisions and lowers the risk of overfitting compared to hard voting.

1. **Results Analysis**

During the experiment different models were tried to find the best solution for the task. Here is the results of those models that were tried during the experiment,

|  |  |
| --- | --- |
| Model | Result |
| Linear Support Vector Classifier | 65% |
| Artificial Neural Network | 58% |
| LogisticRegression | 63% |
| Support Vector Machine | 64% |
| Voting Classifier (Using 3 models) | 72% |
| **Voting Classifier (Using 5 models)** | **77%** |
| Voting Classifier (Using 7 models) | 75% |

From the result it is clear that the selected model achieved the highest accuracy. Now lets see the detailed evaluation of selected model.

Accuracy: 0.77

Precision: 0.78

Recall: 0.77

F1-Score: 0.77

This table represents the model’s classification report:

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Precision** | **Recall** | **F1-Score** |
| ACCOUNTANT | 0.88 | 0.97 | 0.92 |
| ADVOCATE | 0.71 | 0.8 | 0.75 |
| AGRICULTURE | 0.5 | 0.25 | 0.33 |
| APPAREL | 0.79 | 0.55 | 0.65 |
| ARTS | 0.47 | 0.44 | 0.46 |
| AUTOMOBILE | 0.33 | 0.17 | 0.22 |
| AVIATION | 0.9 | 0.9 | 0.9 |
| BANKING | 0.76 | 0.7 | 0.73 |
| BPO | 1 | 0.5 | 0.67 |
| BUSINESS-DEVELOPMENT | 0.96 | 0.93 | 0.94 |
| CHEF | 0.86 | 0.79 | 0.83 |
| CONSTRUCTION | 0.91 | 0.91 | 0.91 |
| CONSULTANT | 0.85 | 0.55 | 0.67 |
| DESIGNER | 0.81 | 0.89 | 0.85 |
| DIGITAL-MEDIA | 0.8 | 0.8 | 0.8 |
| ENGINEERING | 0.71 | 0.81 | 0.76 |
| FINANCE | 0.71 | 0.79 | 0.75 |
| FITNESS | 0.93 | 0.68 | 0.79 |
| HEALTHCARE | 0.61 | 0.55 | 0.58 |
| HR | 0.65 | 0.83 | 0.73 |
| INFORMATION-TECHNOLOGY | 0.63 | 1 | 0.78 |
| PUBLIC-RELATIONS | 0.87 | 0.76 | 0.81 |
| SALES | 0.86 | 0.83 | 0.84 |
| TEACHER | 0.72 | 0.82 | 0.77 |
| **Overall Metrics** | | | |
| **Accuracy** | **0.77** | | |
| **Macro Avg** | **0.76** | **0.72** | **0.73** |
| **Weighted Avg** | **0.78** | **0.77** | **0.77** |

1. **Script Development and Instruction**

As a final product a script named script.py is developed. To use the script a version of **python 3.10 or higher** must be installed in the system. Then open an command shell/ command line follow these steps,

1. **Create a Virtual environment (optional)**: It is recommended to create a virtual environment while using this script. The script is also executable without virtual environment as well. In order to create a virtual environment run these commands,

$python -m venv /path/to/new/virtual/environment

# Activate the virtual environment   
/path/to/virtual/environment\Scripts\Activate.ps1

1. **Clone git repository**: Then Clone the repository in the folder where venv is created.

$git clone https://github.com/Debopom/Resume-Summarization.git

1. **Install required packages**: The script requires some specific versions of python packages. To install these packages just run this command,

$pip install -r requirements.txt

1. **Run the script** : Finally run the script using following command,

$python script.py path/to/resume.csv

1. **Expected Outputs**:
   * Categorized Resumes: The script will create a folder named categorized\_resumes in the current working directory. Inside this folder, subfolders will be created for each predicted category, and the resumes (as HTML files) will be saved within their respective category folders. The filenames will be based on the ID column from the input CSV.
   * Output CSV: The output file named predicted\_categories.csv would contain columns for Filename and Category.