IE University

School of Sciences & Technology

Bachelor of Computer Science and Artificial Intelligence

AI: Chatbots and Recommendation Engines

Code Compass

Midterm Progress Review

Luca Cuneo

lcuneo.ieu2021@student.ie.edu

Gabriel de Olaguibel

gdeolaguibel.ieu2021@student.ie.edu

Keti Sulamanidze

ksulamanidze.ieu2021@student.ie.edu

Miranda Drummond

mdrummond.ieu2021@student.ie.edu

Felix Gomez-Guillamon

fgomezguilla.ieu2021@student.ie.edu

Maud Helen Hovland

mhovland.ieu2021@student.ie.edu

22.02.2024

This document provides a comprehensive overview of the progress made on the Code CompassGitHub, a Github Recommendation System project. The project team has worked diligently across various fronts, including data engineering, testing, exploratory data analysis (EDA), feature engineering, and model development, culminating in the initial stages of frontend development. Below is a detailed account of our achievements to date.

Project Management and Development Model

Backlog Management

- Current Status: Approximately 50% of the items in our GitHub project backlog have been completed, marking significant progress towards our project milestones and keeps on track to deliver our product by the target date, April 4th 2024

Development Model

- Branch Management: Adhered to a strict protocol where development is carried out in short-lived branches, with direct pushes to the main branch prohibited, and branches deleted post-merge.
- Pull Requests (PRs): Enforced a rigorous review process requiring at least one member's review and the passing of automated workflow tests before merging.
- Issues: Integrated with the GitHub project backlog, facilitating transparent communication and collaboration on tasks and fixes.

Data Engineering

Data Retrieval and Preparation

- API Integration: Developed scripts within codecompasslib/API to leverage the GitHub API for extracting detailed repository and profile information.
    - get_bulk_data.py focuses on fetching extensive data features and saving the results in CSV format.
    - helper_functions.py includes essential utility functions for authentication token loading and data storage.

Data Engineering Output

- Data Storage: Extracted data is stored in the /Data/original folder, providing Data Scientists with a rich dataset for subsequent modelling efforts.

Automated Testing and Workflow Integration

- Testing Suite: Implemented tests in /tests to verify the functionality and reliability of data retrieval scripts, covering both success and failure scenarios.
- Workflow Automation: Configured .github/workflows/pr_gate.yml to trigger the test suite when pull request to main is opened

Data Analysis and Model Development

Exploratory Data Analysis and Feature Engineering

- Data Analysis: Conducted thorough EDA, including data cleaning, visualisation, and correlation analysis, to inform the model development process.
- Feature Engineering: Applied one-hot encoding, text vectorization (TfidfVectorizer), and normalisation to prepare the data for machine learning models.

Recommendation Models

- Cosine Similarity Model (cosine_modelling_repos.ipynb): Utilizes cosine similarity to recommend repositories based on language preferences and text data characteristics.
- Repository Interaction Model (modelling_repos.ipynb): Investigates the combination of repository data and user interactions to generate recommendations.
- Differentiating Repositories Model (modelling_differentiating_repos.ipynb): Employs domain-specific embeddings for the most suitable vectorization for natural language, finds similar users, and finds a differentiating project among those similar users.

Frontend & Backend Development

- Initial Implementation: Developed a preliminary html frontend allowing users to input their GitHub username and receive personalised repository recommendations.
- Flask Integration: Established a Flask connection to the backend recommendation model, facilitating dynamic user interactions with the system.
- The frontend currently enables users to input a username in the form (which must be in our existing database), then via flask takes the input, feeds it through the model, returning the repository based on our recommendation model.

Libraries and Repository Design

- Code Organization: Adhered to a structured repository design, with notebooks, scripts, and library calls neatly organised following Miguel's "Project Template" design.
- Library Usage: The project extensively utilises the codecompasslib library for internal functionalities and external libraries for data processing and model development.
- Deployed the first official release v1.0.0 - Midterm Review.

.