Expose V1: PandaRec - Recommendations for the Pandas Library

Introduction

Code Recommendation in Software Development help developers write good code faster and easier, while optimally also suggesting solutions that are fast and easily maintainable.

Pandas is a library for Python that helps with the efficient cleaning, reshaping, visualization and merging of small or large datasets. While it is powerful and flexible, it is equally complex.

Problem Description

Because of pandas complexity, it is hard to use for people without much programming knowledge and/ or practice in data manipulation. In essence, the library has a sharp learning curve.

Solution

By recommending code snippets based on a search term and specific information on the data, the learning curve should be flattened.

Methods/ Libraries

Using jupyter notebooks we can create a gui that is easy to set up (e.g. in google colab) and recommends predefined recipes/ snippets. The GUI should have an overview over the data with e.g. the ipydatagrid library. It should also have a search bar and the recommended code snippets. As long as this is kept expandable, this solution might be applied to other contexts, such as coding in an IDE with recommendations based on the current dataframes loaded and the current line as a search term. For recommending code snippets, different algorithms should be implemented and compared.

Workpackages

The work can be grouped into the following Workpackages.

1 Setup

Development of a snippet data structure, importing of regularly used data processing operations into this data structure, preferably automated.

Development of a basic ranking algorithm such as keyword matching or a basic string search algorithm.

At this point we should have a usable base for implementing more complex features. The requirements for this Workpackage are only a working python installation and the pandas library.

2 Interface

Development of the interface by adding a table for the data, a search bar and space for the recommendations.

The current context should be linked to the recommender and the resulting snippets back to the gui.

This step will require a dataframe visualizer (ipydatagrid) and the ipywidgets library.

3 Recipes

The recipes are an important backbone of the project. Curating meaningful and useful snippets will increase the usefulness of the project by a lot. Snippets can be generated via the documentation, the dir() function and collected usage data. Handcrafted snippets could further increase overall quality.

4 Ranking Algorithms

Multiple ranking algorithms should be considered. A name search or keyword search was already mentioned. Additionally, using immediate user feedback, i.e. looking at what snippets a user picked earlier, and using the structure of the DataFrame (types, groups column names, values) we should be able to improve the algorithm. The best results will probably be given by a combination of multiple algorithms. The different algorithms will be implemented via a strategy pattern.

Additionally, offloading the ranking to a server as a microservice could help by collecting more user feedback from multiple users. The data collected with such an approach, together with data of usage of the pandas library collected online, could be used to train a model so that we can leverage machine learning for an algorithm.

5 Analysis

This solution could be analyzed by comparing different ranking algorithms, comparing different snippet sets, and comparing the whole solution to existing ones.

6 Extension

The project could be extended further by using the algorithms to recommend snippets in vscode. Additionally, other the project could be expanded to recommend snippets for different libraries. Later work could use the system set in place with this project to use develop different recommendation algorithms.