ProSRL Datenanalyse (pysrl) 0.3.6

Das ist ein Projekt zur Datenanalyse von selbstreguliertem Lernen (SRL) mittels einer digitalen Lernplattform. Im Zentrum steht das package pysrl, das Modul zum Crawlen der Website, Vorbereiten der Daten, Clustering und Klassifizierung enthält. Es lässt sich über die main.py Datei ansteuern.

Vorbemerkungen

Das pysrl Package besteht aus einer analysis.py Datei, die eine Analyst Klasse enthält, die alle Funktionalitäten des Packages bündelt. Dazu gibt es eine Reihe Module (classifier, cluster, ...), die diverse Submodule und Funktionen enthalten. Diese lassen sich durch direkte Importe aus anderen Dateien nutzen. Standardmäßig liegt das pysrl package in einem Projektordner (in dem auch diese README liegen sollte). Aus diesem Projektordner können Python Dateien (oder auch Jupyter Notebooks) durch

```
from pysrl.SUBPACKAGE.MODULE import FUNKTION
```

alle beliebigen Funktionen importieren (* statt FUNKTION importiert jede Funktion des Moduls). Beispiel:

```
from pysrl.preparer.prep_fcts import load_transformed_data
```

Alternativ kann das package auch mit pip installiert werden, sodass jede Python Datei im System, alle Funktionen importieren kann, mehr dazu unten.
Für diese Installation muss im Projektordner ein build von pysrl stattgefunden haben, was standardmäßig der Fall ist. Wie ein manueller (re-)build erfolgen kann wird unten erklärt.

Installation

Am einfachsten ist es, den Projektordner zu verwenden und nur mit Dateien daraus (z.B. main.py) auf das Package zuzugreifen.

Achtung: pysrl benötigt Python >= 3.9 zur Ausführung.

Die benötigten packages für die Ausführung sind in setup.cfg unter install_requires aufgeführt. Als shortcut bietet es sich an, alle Pakete direkt mithilfe der requirements.txt Datei zu installieren. Nutze hierzu im Terminal innerhalb des Projektordners:

```
pip install -r requirements.txt
```

Dabei muss pip zwangsläufig Python3.9 (oder 3.10) zugeordnet sein, prüfe das zuvor mit

```
pip --version
```

Sollte die Version nicht Python3.9 (oder 3.10) entsprechen, nutze für die Installation von requirements.txt (und allen anderen pip Installationen für dieses Projekt):

```
python3.9 -m pip install -r requirements.txt
```

(Bzw. mit 3.10, falls diese installiert und genutzt ist)

Achtung: requirements.txt enthält mehr Pakete als zur standardmäßigen Ausführung notwendig. Die Installation kann daher einige Zeit in Anspruch nehmen. Sollten Zeit-/Speicherbedarf zu hoch sein, wird eine manuelle Installation der Pakete aus setup.py → install requires empfohlen.

Alternativ kann pysrl auch als package in Python eingebettet werden. Hierzu im Projektordner ausführen:

```
pip install .
```

Achtung: Die Installation erfolgt in den site_packages Ordner, der dem verwendeten pip installer zugeordnet ist. Verwende in jedem Fall pip für Python >= 3.9 und ggbf. pip für Conda, falls das package in einem Conda environment installiert werden soll.

pysrl steht jetzt als package zur Verfügung und kann überall importiert werden. Die main.py Datei kann also auch außerhalb des Projektordners ausgeführt werden.

Achtung: Die Verwendung von pysrl als package is instabil. Importfehler insbesondere durch die Pakete numpy und Pillow können auftreten und müssen manuell behoben werden. Im Zweifelsfall ist das Ausführen von main im Projektordner und importieren von pysrl direkt aus dem Unterordner sicherer.

Verwendung

In der main Datei können Parameter eingestellt werden, die sich auf die Auswertung der dort importierten Objekte aus pysrl auswirken. In main_explained.py sind die Funktionalitäten genauer erklärt. Es gibt einen docs Ordner mit weiteren Informationen (Versionsübersicht, Featureübersicht, Dateiübersicht, etc.).

Die Auswertung erfolgt vollautomatisch anhand der in main festgelegten Parameter. Allerdings braucht pysrl dazu input Daten über

1. Die Aktivitäten der User (csv Datei mit User, Date/Time, Link)
2. Die Personenmerkmale (csv Datei mit x Spalten und einer Reihe für jeden User)
3. Die Zuordnung zwischen User, Code und id (csv Datei mit User, Person, user_id)
4. Die Zuordnung der Aufgabenschwierigkeiten (csv Datei mit Label, Niveau)
5. Die Ergebnisse der Aufgaben (csv Datei mit Task, Date/Time,User, ResponsTask)
6. Die Testergebnisse (csv Datei mit Test, Date/Time, User, Testergebnis 1, Testergebnis 2)

Achtung: Es wird dringend empfohlen die Dateien in einem Ordner 'input' im Projektordner zu speichern und sie wie folgt zu benennen: 1) data_complete.csv 2) person_features.csv 3) relation_user_code.csv 4) task_difficulties.csv 5) task_results.csv 6) test_results.csv. Liegen die Dateien in einem anderen Ordner im Projektordner oder sind anders benannt wird pysrl trotzdem versuchen sie zu finden und zu laden, Erfolg ist aber nicht garantiert.

Für weitere Erläuterungen über die Funktionalitäten und Features wird auf den docs Ordner im Projektordner verwiesen. Hier findet sich neben Übersichtsdateien die pysrl.html Datei mit der vollständigen pdoc Dokumentation inklusive source-code.

Build

Manueller (re-)build von pysrl ist möglich, wenn im Projektordner korrekte MANIFEST.in, pyproject.toml und setup.cfg Dateien liegen. Dann bedarf es folgender Schritte:

1. Installieren des build packages mittels pip (Prefix python3.x für die korrekte Python Version (>=3.9)):

```
python3.10 -m pip install --upgrade build
sudo apt install python3.10-venv
```

2. Package build (im Projektordner)

```
python3.10 -m build
```

3. Re-build der html docs ist aus dem Projektordner möglich mittels

```
pdoc -o ./docs -d google pysrl
```

Achtung: Vor dem re-build muss pip install . ausgeführt werden. Jeder korrupte Import wird pdoc abbrechen lassen. Korrigiere diese Imports, entferne die Dateien oder ignoriere sie für die docs mittels:

```
pdoc -o ./docs -d google pysrl !pysrl.MODUL_DAS_IGNORIERT_WERDEN_SOLL
```

Jetzt kann es mit der Installation wie oben beschrieben weitergehen.