Übersicht Versionen und Updates

Von V1.0 an:

- **Crawlen** der Website nach Informationen über Seiten → learnpages.yaml
- **Organisieren** der Daten durch Zusammenführen von learnpages.yaml + data_complete.csv → data_clean.scv
- **Transformieren** der Daten in numerische Werte und Summieren nach Usern → data_trafo.csv
- **Präparieren** der Daten durch Hinzufügen weiterer Feature und Vorbereiten für clustern → data_prep.csv
- Erzeugen von **Lineplots** mit Informationen über Parameter der Aktivitäten der User im zeitlichen Verlauf
- **Clustern** der User über alle möglichen Feature mittel kMeans, Meanshift, Spectral, Agglomerative, Hierarchy
- **Principal Component Analyse** verschiedener Feature und Darstellung inklusive Loadings und Cluster
- 1D (**Histogram**) und 2D (**Scatterplot**) Darstellungen der Cluster über einzelne und kombinierte Feature
- Zusammenfassende Ausführung der Methoden mittels **main.py** Datei

Neu in V1.1:

- **Neue Feature**: DidTest, TimePerCat, BeforeTestTime, AfterTestTime, PredChance1, PredChance2, PredChance3, PredChance4, LearnTypeEntropy, CondEntropy1, CondEntropy2, CategoryEntropy
- Feature Typen und **Feature** können in main.py **per Name** (string) und nicht nur per Index gegeben werden
- **Feature** und Feature Typen können **beliebig kombiniert** werden (alle in [] gegebenen zusammen verwendet)
- Auch bei **dim>2** werden in main.py alle Kombinationen als **Scatterplots** gezeigt, die gezeigten Cluster entsprechen dann jeweils denen die in höherer Dimension mit allen gegebenen Featuren gefunden wurden
- **Zusätzliche Parameter in main.py**: c_**threshold**: Color threshold für Dendrogram (float, frei wählbar), **max_learntype**: Maximaler Lerntyp der in den Lineplots gezeigt wird
- **Readme** überarbeitet, **Files_Overview** überarbeitet, **Features_Overview** und **Versions_Overview** hinzugef.

Neu in V1.2:

- **Neue Feature:** TRegrFit, ÜRegrFit, BKRegrFit, CatTRegrFit, CatÜRegrFit, CatBKRegrFit, DiffLowRegrFit, DiffMedRegrFit, DiffHighRegrFit, KÜfRegrFit, KÜtwrRegrFit, KÜrRegrFit, TresQuantRegrFit, TsecSpentRegrFit, ÜSecSpentRegrFit, BKSecSpentRegrFit, nResponsTask, nTestResQual, nTestResQuant, ndifficulty, meanResponsTask, meanTestResQual, meanTestResQuant, meandifficulty
- **Neue Daten:** data_prep.csv enthält Schwierigkeiten der Aufgaben und Ergebnisse von Aufgaben und Tests. Dazu Verknüpfung mit difficulties.csv, task_results.csv und test_results.csv über Zeit/User Spalten.
- **Überarbeitete Lineplots:** Es wird die **originale Zeit** auf der Lernumgebung gezeigt (x-Achse), **Übersichtsseiten** sind enthalten und können ausgeblendet werden (s.u.). Seiten mit **Schwierigkeiten** (Übungen) werden mit **Dreiecken** dargestellt (Spitze unten: leicht, Spitze rechts: mittel, Spitze oben: schwer), alle anderen mit **Kreisen**. Die **Größe** (Fläche) dieser **Marker** entspricht der **verbrachten Zeit** auf der Seite. Ergebnisse von Aufgaben und Tests sind an den Markern angemerkt (+: richtig/gut bestanden, o: twr/bestanden, -: falsch/nicht bestanden). Bei den Tests stehen dazu in Klammern die erreichten Punkte.
- **Zusätzliche Parameter in main.py**: **n_bins**: Anzal der Bins in Histogram (1D) → für weniger als 10 unterschiedliche Werte wird jetzt Barplot statt Histogram gezeigt, **selected_learntypes** (ersetzt max_learntype): Lerntypen die in lineplots gezeigt werden sollen → geschachtelte Liste, innere Einträge werden aggregiert und die resultierenden Lerntypen im lineplot dargestellt, der Rest wird ausgeblendet.
- **Neue ausführbare Datei:** cluster_prep_plot.py → Erzeugt und speichert nur die Lineplots+

Neu in V1.3:

- **Neue Feature:** Nst_Pre, Nst_Post, Nst_Diff, RF_Pre, RF_Post, RF_Diff, SWJK, Note, SWK, FSK, AV, ALZO, VLZO, LZO, Inte, KLO, KLW, MKL, RZM, RSA, RAK

- **Neue Struktur**: Alle Funktionalitäten sind jetzt in einem **pysrl Modul** gebündelt. Dieses liegt unter **src** im Projektordner. Daneben liegt ein Ordner **tests** für Unittests der Untermodule. Aller **input** liegt im Projektordner unter input, alle Ergebnisse werden unter **results** gespeichert. Alle zwischendurch erzegten Dateien werden in **data** abgelegt. Erläuterungen sind im **docs** Ordner gesammelt.
- **Neue Dateien:** Eine Reihe von Dateien für das Erzeugen eines standalone packages aus pysrl, namentlich setup.cfg, pyproject.toml, LICENSE, MANIFEST.in, sowie die bereits erzeugten Ordner dist und pysrl.egg-info. Diese Dateien können weitestgehend ignoriert werden. Außerdem mit **requirements.txt** eine Datei um benötigte packages bequem zu installieren. In src/tests liegen Unittests für die Module, allerdings gibt es davon bisher kaum welche.
- **Änderungen an main.py:** main.py lädt nicht mehr die Funktionen aus den Modulen direkt, sondern nur noch eine Klasse Analyst aus pysrl.analysis, die alle Funktionen implementiert. Dadurch wird main.py übersichtlicher. Parameter werden jetzt sofort an eine Instanz der Klasse übergeben. Die Übersicht der möglichen Feature is nur noch in main_explained.py enthalten, hier finden sich auch weitere Erläuterungen. main.py selber ist so kompakt gehalten wie möglich.
- **Zusätzliche Parameter in main.py:**
  an.performs (ersetzt perform): Ein dict mit den Funktionen die auszuführen sind (crawl, orga, trafo, prep, analyse). Setzte 1 um auszuführen, 0 um zu skippen.
  an.format: Das Format in dem Daten zwischengespeichert werden: ('csv', 'xlsx') oder ('csv',). Das speichern von xlsx erleichtert es die Daten mit Excel zu untersuchen, braucht aber deutlich länger als nur in csv.
  an.use_personal: True um Personenmerkmale als Feature mit aufzunehmen
  an.rm_incomplete: Bei der Auswertung der Personenmerkmale sind NaN Werte vorhanden, weil User nicht vorher und nachher Tests absolviert haben. Sollen diese User nicht berücksichtigt werden setze heir True.
  an.exclude_user: Liste mit user_ids deren User für die Auswertung unberücksichtigt bleiben sollen
  an.predict: Das Feature welches mittels Entscheidungsbaum klassifiziert werden soll, wenn classify True ist
  an.classify: True um die Klassifizierung eines Features mittels Entscheidungsbaum durchzuführen
  an.show_cluster_of: Wird hier eine Liste (oder Tupel) von Features gegeben, nach denen ein Clustering erfolgte, so werden in allen Scatterplots (sofern welche erzeugt) die user nach diesen Clustern eingefärbt. Das ist hilfreich um zu erkennen wie sich ein Cluster bestimmter Feature in anderen Features niederschlägt.
  an.clear_recent: Plots werden unter results/… in einem assenden Ordner gespeichert. Dazu werden sie in einem Odner results/recent gespeichert. Dieser kann geleert werden durch an.clear_recent=True.
  an.save_plots: True um erzeugte plots zu speichern.
  an.max_depth: Die maximale Tiefe von Entscheidungsbäumen
  an.min_leaves: Die minimale Anzahl von Usern in einem Blattknoten der Entscheidungsbäume
- **Neue Funktionalitäten:**
  - pysrl lässt sich jetzt als package mit pip installieren, allerdings können hier noch Fehler auftauchen
  - Die Klassifizierung mittels Entscheidungsbaum ist jetzt für ein beliebiges Feature möglich
  - Die personenmerkmale können wie alle anderen Feature auch in alle Analysen mit einbezogen werden

Neu in V1.3.1:

- **Fixed problem:** Aggregierte Feature werden jetzt by default normiert (ansonsten müssen neu hinzugefügte auch immer unter ‘norm’ bei get_columns ergänzt werden, was zu Fehlern führen kann)

Neu in V1.3.2:

- **Fixed problems:**
  → existing_features.txt import error
  Umlaute werden nicht korrekt geladen → Fix: Die Datei wird nicht mehr verwendet, stattdessen wird anhand von self.df.columns in Analyser getestet ob die gewählten Feature in den Daten existieren → Beeinflusst: existing_feature_types.txt, die Datei wird umbenannt in feature_types.txt und weiterhin verwendet, da hier keine Umlaute vorkommen.
  → messed up ROOT etc.
  Die Ordner input, data, result sollen in ROOT liegen, wobei ROOT unklar definiert war. Jetzt wird bei Ausführen von main.py der Pfad von main.py in src/pysrl/config/root.txt geschrieben und als ROOT verwendet, sodass input, data und result relativ dazu im gleichen Ordner wie main.py erzeugt werden. ROOT kann in set_root() auch als Argument manuell angegeben werden um den Speicherort zu ändern. Sollte irgendeine Funktion aus pysrl anders als über main aufgerufen werden und dabei auf ROOT zugreifen, wird immer der Pfad aus root.txt verwendet. Wurde main zuvor nicht ausgeführt gibt es einen Fehler der auffordert ROOT in root.txt zu spezifizieren.
- **Entfernt:**
  → Möglichkeit Submodule als standalone Skripts auszuführen

Import innerhalb des packages sind in der Regel relativ (from .xxx import yyy), gemäß PEP8:

Absolute imports are recommended, as they are usually more readable and tend to be better behaved (or at least give better error messages). [...] However, explicit relative imports are an acceptable alternative to absolute imports, especially when dealing with complex package layouts where using absolute imports would be unnecessarily verbose.

Ausführen von Submodulen als standalone Skripts funktioniert aber nicht mit relativen Imports nach Pep338:

... relative imports rely on __name__ to determine the current module's position in the package hierarchy. In a main module, the value of __name__ is always '__main__', so explicit relative imports will always fail (as they only work for a module inside a package)

Diese Funktionalität ist aber auch nicht zwangsläufig notwendig (oder gewünscht), nach Guido von Rossum:

I'm -1 on this and on any other proposed twiddlings of the __main__ machinery. The only use case seems to be running scripts that happen to be living inside a module's directory, which I've always seen as an antipattern. To make me change my mind you'd have to convince me that it isn't.

Und wird daher hier nicht weiter unterstützt. Ausführen von z.B. crawler als standalone kann ohne weiteres 'simuliert' werden, indem main ausgeführt wird und nur crawl unter an.performs aktiviert wird.

Neu in V1.3.3:

- **Fixed problems:**
  → folder creation:
  Datenordner für Clusterergebnisse wurden nicht korrekt erzeugt, da falscher Pfad gegeben. Fixed.
  → Histogram empty:
  Histogram wurde nicht korrekt angezeigt, da versucht wurde show_clusters_of zu verwenden, was nur in Scatterplots genutzt werden sollte. Entsprechende Zeilen wurden entfernt. Fixed.
- **Anpassungen:**
  → Ergebnis Daten des Clusterns:
  Werden jetzt in Subordnern für jeweilige Dimension und Zahl der Cluster gespeichert und mit korrekter user_id sowie Personencode und Nutzername versehen (siehe z.b. labels.xlsx)
  → Bedingte Entropie und PredChance flexibler:
  Es kann jetzt gewählt werden, welches Feature hierfür verwendet wird und wie die eindeutigen Werte des Features gemapped werden sollen (z.B. auf nur 2 Kategorien). Außerdem kann für die bedingte Entropie das maximale N gegeben werden, es wird dann bestimmt: CondEntropy1, CondEntropy2, …, CondEntropyN.
- **Neue Funktionen:**
  → Evaluieren von Clustern:
  Elbow plots und silhouette analysis lassen sich jetzt für gewählte Parameter zu den Cluster Analysen ausgeben.
- **Neue Parameter in main:**
  → an.use_time: Ob für die Normierung von Tests, Beispielen, … die Zeit verwendet werden soll (ansonsten Zahl der Aktivitäten)
  → an.prep_feature: Das Feature für die bedingte Entropie und PredChance berechnung in prep
  → an.prep_mapper: Das mapping der Werte für dieses Feature
  → an.max_n_cond_entropy: Die maximale zu berechnende bedingte Entropie CondEntropyN

Neu in V1.3.4:

- **Fixed problems:**
  → Lineplots:
  Werden jetzt im Rohfofrmat auch mit korrekten Zusammenfassungen der Kategorien gezeigt
- **New Features:**
  LearnActs2 (ohne B) und InfoActs2 (mit B)

Neu in V1.3.5:

- **Features überarbeitet:**

  Per default werden Feature jetzt nach Zeit auf der Seite bemessen (z.B. bei Tests, Kurzaufgaben, etc.).

Die zugehörigen Feature werden nicht mehr mit <Feature>Time benannt, sondern nur noch mit <Feature>.

Bsp: TotTime wird zu Tot

Dafür gibt es für die Anzahl der Aktionen zusätzliche Feature, bezeichnet mit <Feature>Acts.

Bsp: TestsActs

Außerdem gibt es für alle <Feature>Acts zugehörig die Feature <Feature>_TimePerAct, die sich ergibt durch Berechnung von <Feature> / <Feature>Acts.

Bsp: df['K+Ü+T_TimePerAct'] = df['K+Ü+T'].div(df['K+Ü+TActs'], axis=0)

Neu in V1.3.6:

- **Neue Feature:** backfrac, foolfrac1, foolfrac2, foolfrac12
- **Plotting überarbeitet**:
  → Eigener Ordner für plotting in pysrl
  Notbook für schnelles Erzeugen aller möglichen Plots von Usern (Lineplots, Timedependent, etc.)
  Ordner für Ergebnisse darin (results), um alle Plots an einem Ort zu speichern
  → Plotfunktion für zeitabhängige Feature
  Für verschiedene zeitabhängige Feature können jetzt Plots dargestellt werden. Dafür sind alle Feature nutzbar, die sich zeitlich ändern (z.B. Anteil der verschiedenen Lerntypen, Richtigkeit von Aufgaben, Sekunden pro Seite, …). Dazu wird time_dependent_features aus prep_fcts mit der plot=True flag genutzt.
  Für beispielhafte Anwendungen siehe user_plots.ipynb in pysrl/plotting.
  → Plotfunktion für Vergleich mit intuitiver Seitenreihenfolge
  Außerdem gibt es jetzt Plots die User-Aktivitäten mit der intuitiven Anordnung der Seiten auf der Lernplattform vergleichen. Dazu wird plot_lines aus prep_help_fcts verwendet.
  Für beispielhafte Anwendungen siehe user_plots.ipynb in pysrl/plotting.