Programmierprojekt: Webscraper für Designerdrogen

In diesem Projekt, welches in Zusammenarbeit mit Dipl.-Chem. Johannes Kutzler, Doktorand am Universitätsklinikum Freiburg, durchgeführt wird, geht es darum, Daten zu Designerdrogen von einer Webseite zu laden, diese zu speichern und durchsuchbar zu machen. Dazu soll die Programmiersprache Python verwendet werden.

Hintergrund des Projekts

Neue Psychoaktive Substanzen (NPS) sind Designerdrogen, die seit den 2000er Jahren zunehmend an Bedeutung gewinnen, da sie häufig als Ersatz für tradionelle Drogen wie Heroin, Amphetamin oder Cannabis verwendet werden. Häufig wird die chemische Struktur von NPS, die den jeweiligen Suchtstoffgesetzen der EU-Mitgliedstaaten bereits unterstellt sind, gezielt so verändert (z. B. nach dem Baukastenprinzip oder anhand von Pharmapatenten), dass die neue Substanz nicht mehr diesen Regelungen unterliegt. Das hat zur Folge, dass der Markt enorm schnelllebig ist und regelmäßig neue Substanzen auf dem (Online-)Markt erscheinen.

Die Laboranalytik solcher Substanzen erfolgt u. a. mittels eines Datenbankabgleichs der Messdaten. Neue, bereits auf dem Markt befindliche Substanzen, die noch nicht in diesen Datenbanken enthalten sind, werden daher nicht erfasst und führen zu falsch-negativen Ergebnissen. Möglichkeiten, um stets aktuelle Informationen über neu aufgetretene Stoffe zu erhalten, sind beispielsweise Online-Testkäufe, dem Informationsaustausch mit Laboren oder der Verfolgung von Onlineforen-Diskussionen.

Viele Webseiten listen bereits Informationen zu NPS, jedoch bieten sie in der Regel weder eine API, um die Daten direkt extrahieren zu können, noch sind sie vollständig und korrekt. Die verschiedenen Plattformen verwenden unterschiedliche Formate, die Zugriffe auf die Daten erfolgt jeweils unterschiedlich und die Datenbestände überlappen sich zwar, sind aber größtenteils verschieden. Substanzen sind also durchaus auf mehreren Plattformen vorhanden, aber es gibt keinen Single-Point-of-Truth. Wer also Informationen beschaffen möchte, muss die Daten von verschiedenen Webseiten beschaffen und Webseiten-übergreifend suchen. Genau das zu automatisieren ist Ziel dieses Projekts.

Allgemeine Anforderungen

- **Programmiersprache**: Verwenden Sie Python.
- **Libraries**: Suchen Sie geeignete Bibliotheken, die Ihnen das Web-Scraping vereinfachen.
- **Teamarbeit**: Arbeiten Sie im Zweiter-Team, teilen Sie sich gegenseitig Ihre Aufgaben auf, verstehen Sie die Lösungen Ihrer/s Teampartner*in, arbeiten Sie selbstständig, übernehmen Sie keine Lösungen von anderen Gruppen und übernehmen Sie nicht blind KI-genierten Programmcode.
- **Git**: Legen Sie ein (privates oder öffentliches) Projekt auf GitHub oder dem OTH-GitLab an, arbeiten Sie zu zweit an Ihrem Projekt, verwenden Sie sinnvolle Commit-Messages.

- **Code Style**: Verwenden Sie einen guten Programmierstil. Verwenden Sie sinnvolle Variablennamen, machen Sie Ihren Code verständlich und gut wartbar.

- **Dokumentation**: Dokumentieren Sie Ihren Programmcode, verwenden Sie Kommentare, um z. B. zu beschreiben, was Ihre Funktionen tun, und erstellen Sie ausführliche Dokumentationen und READMEs auf GitHub/GitLab, sodass die Verwendung, Installation und Erweiterung Ihrer Software für andere so einfach wie möglich gemacht wird.

- **Modularisierung**: Schreiben Sie nicht einfach ein Script, welches von oben nach unten abgearbeitet wird, sondern unterteilen Sie Ihre Software sinnvoll in Funktionen; optional können Sie auch objektorientiert programmieren.

- **Testen**: Schreiben Sie für Ihre Funktionen Unit-Tests, um zu überprüfen, ob sie korrekt funktionieren.

- **Logging**: Verwenden Sie eine Logging-Library, sodass alle relevanten Aktionen, die Ihre Antwendung ausführt, protokolliert werden. Je nach eingestelltem Log-Level sollen die Log-Nachrichten in eine Datei geschrieben werden und der/die Anwender*in sieht die Log-Nachrichten als Konsolenausgabe.

- **Performance**: Zwar steht die Performance des Web-Scrapers (Teil 1) und der Suchmaschine (Teil 4) nicht an oberster Stelle, dennoch soll sie akzeptabel sein.

  Dieses Dokument beschreibt hauptsächlich, *was* Ihre Anwendung können soll. *Wie* genau Sie das umsetzen und welche nicht hier aufgeführten Features, die Sie noch sinnvoll finden, Sie zusätzlich umsetzen, ist Ihnen überlassen.

Teil 1: Web-Scraper

- Im ELO-Kurs sehen Sie die URL zu der Webseite, um die sich Ihre Gruppe kümmern soll. Jede Gruppe verwendet eine andere Webseite.
- Schreiben Sie ein Script, welches alle Substanzen von ebendieser Webseite lädt und in eine JSON-Datei abspeichert.
- Relevante Attribute der Substanzen, die gespeichert werden sollen, sind: Name, Synonyme, Kategorien, Direkt-URL auf der jeweilgen Webseite, diverse technische Informationen (sofern angegeben): formeller Name, CAS-Nummer, Molekulare Summenformel, Formelgewicht, SMILES, InChI-Code, InChI-Key).
- Einigen Sie sich zusammen mit den anderen Gruppen auf ein einheitliches Format.
- Beachten Sie, dass nicht jede Webseite alle Informationen zu einer Substanz liefert.
- Beachten Sie, dass eine Substanz mehrere Kategorien und mehrere Synonyme haben kann.
- Beachten Sie, dass die SMILES-Notation nicht eindeutig ist. Beispielsweise spezifizieren CCO, OCC and C(O)C ein und dieselbe Substanz. Verwenden Sie das Python-Paket RDKIT, um einen SMILES-String in einen kanonischen SMILES-String umzuwandeln und speichern sie diesen, da dieser eindeutig ist.
- Sie können sich überlegen, ob Sie zusätzlich zu den JSON-Dateien noch Hilfsdateien oder eine SQLite-Datenbank einsetzen, um später bei Teil 3 (Inkrementelles Laden) und Teil 4 (Suchmaschine) effizienter zu sein.

Teil 2: Validation

- Es ist durchaus möglich, dass die Daten auf der Webseite fehlerhaft sind.
- Verwenden Sie das Python-Paket RDKIT, um aus der SMILES-Notation Formelgewicht und Summenformel zu bestimmen.
- Vergleichen Sie Ihre berechneten Werte mit den auf der Webseite angegebenen Werten, um eine Aussage über die Datenqualität zu machen.

- Speichern Sie in Ihren Daten die Informationen über die Datenqualität ab.

Teil 3: Inkrementelles Laden

- In unregelmäßigen Abständen kommen auf der Webseite neue Substanzen hinzu und existierende Substanzen werden geändert, um Fehler zu korrigieren.
- Passen Sie Ihr Programm so an, dass es erneut die Webseite durchsuchen kann, ohne alle bisherigen Resultate wegzuwerfen.
- Nur die Änderungen seit dem letzten Vorgang sollen geladen werden.
- 3 einstellbare Modi: (1) alles komplett neu laden, (2) nur neu hinzugekommende Substanzen laden, (3) neue Substanzen laden, geänderte anpassen.

Teil 4: Suchmaschine

- Entwickeln Sie eine Anwendung, die eine Suche auf Ihren Daten ermöglicht.
- Entscheiden Sie selbst, wie Sie Ihre Suchmaschine gestalten: Als Konsolenprogramm, GUI-Anwendung, Web-Anwendung, App, …
- Folgende Suchen sollen unterstützt werden:
  - Finde Substanzen für einen bestimmte SMILES-String.
  - Finde Substanzen, deren Summenformel einen bestimmten String enthält.
  - Finde Substanzen, die in einem bestimmten Massebereich liegen, z. B. 300.5 +/- 0.5

Teil 5: Integration der Daten der anderen Gruppen

- Stellen Sie Ihre Daten den anderen Gruppen zur Verfügung und intergrieren sie deren Daten ebenfalls in Ihre Software.
- Ihre Suchmaschine soll alle Daten (Ihre und die der anderen Gruppen) durchsuchen.

Termine und Abschlusspräsentation

- Nutzen Sie die Treffen am 23. April und 28. Mai, jeweils von 17:15 Uhr bis 18:45 Uhr in K221, um Ihre Fragen zu stellen und um Tipps, Hilfe und Feedback zu bekommen.
- Präsentieren Sie Ihre Software, Ihre Herausforderungen und Lösungen am 2. Juli. Die Präsentationen finden von 17:15 Uhr bis 20:30 Uhr in K221 statt.
- Zeigen Sie sowohl eine Folien-Präsentation als auch eine Live-Demo Ihrer Suchmaschine (Teil 4).
- Beide Teampartner sollen präsentieren.
- Präsentationszeit: 20 Minuten.
- In die Bewertung geht sowohl Ihre Abschlusspräsentation ein, als auch Ihre Software und deren Dokumentation.
- Zur Bewertung der Software werden die einzelnen in diesem Dokument befindlichen Punkte herangezogen und bewertet, in wie fern die einzelnen funktionalen und nicht-funktionalen Anforderungen erfüllt wurden.