Dr. Konrad Völkel
Heinrich-Heine-Universität Düsseldorf
Wintersemester 2024/25

Data Science 2

Blatt 3, Abgabe am 06.11.2024 um 12:00

Bitte geben Sie Ihren Code (als .py-Datei(en) und ggf. 1 Notebook) zusammen mit einem PDF sowie den 2 generierten Spielzeugdaten (CSV) aus Aufgabe 2 ab.

Aufgabe 1. *Im vorigen Übungsblatt haben Sie bereits explorative Datenanalysen durchgeführt (mit dem italienischen Housing-Datensatz von Kijiji und dem berühmten Enron-Email-Datensatz). Nachdem Sie sich mit anderen Teams ausgetauscht haben, was können Sie in Ihrer explorativen Datenanalyse ergänzen?*

Aufgabe 2. *Generieren Sie aus beiden Datensätzen bereinigte, reduzierte Datensätze zum schnellen evaluieren, mit nur 10 Instanzen, als CSV gespeichert. Wir benötigen dabei zu jeder Instanz nur eine UID (z.B. Index oder URI) um den entsprechenden Eintrag im urpsrünglichen Datensatz wieder zuordnen zu können, und den String, den wir später mit LSH deduplizieren wollen / semantisch ähnliche finden wollen. Beim Housing-Datensatz ist dazu das längere Textfeld oder eine Konkatenation der Textfelder sinnvoll, beim Email-Datensatz der Volltext oder auch der BODY (also ohne Email-Header).*

Tipp: Wenn Sie sich Arbeit sparen möchten, schreiben Sie ein Programm, welches solche bereinigten Daten aus den Rohdaten generiert für eine beliebige Anzahl an Instanzen. Das können Sie später nutzen.

Bonusaufgabe: Wenn Sie Herausforderungen nicht scheuen, können Sie versuchen, den Enron-Datensatz mit MapReduce einzulesen (sodass jede Datei auf einer Storage Node liegt und zuletzt die Zieldatei von einem Reducer geschrieben wird). Das ist hauptsächlich eine Übung darin, MRJob anzusteuern.

*Tipp 2: Wenn Sie diese Aufgabe nicht schnell lösen können, besorgen Sie sich das Ergebnis von Kommiliton*innen, um erstmal weitermachen zu können.*

Aufgabe 3. *Programmieren Sie Shingling als Python Generator Objekt: implementieren Sie eine Methode* `shingle(s, k=3)`*, die mit Hilfe des* `yield`*-Schlüsselworts einen Generator über alle k-Shingles zurückgibt, sodass z.B.*
`list(shingle('test', k=2)) == ['te', 'es', 'st']`
ist. Dabei wählen wir also UTF8-Zeichen als kleinste Einheit (nicht Worte).

Testen Sie Ihre Methode von Hand an der ersten Instanz des Spielzeugdatensatzes aus der vorigen Aufgabe. Beschreiben Sie, ob es sich wie erwartet verhält. Was passiert für seltsame Werte von k wie etwa 0 oder $k \geq len(s)$? Was, wenn `s=''`*? Überlegen Sie (und schreiben Sie das Ergebnis der Überlegung auf), ob test cases (etwa via Python 'doctest'), Assertions im Code, Typendeklarationen oder etwas anderes geeignet wären, Ihren Code robuster gegenüber unerwarteten Eingaben zu machen bzw. die Korrektheit sicherzustellen.*

Aufgabe 4. *Geben Sie zu den Testdatensätzen aus Aufgabe 2 die 10 häufigsten 4-Shingles an, indem Sie diese Zählung mit MapReduce durchführen. Dabei dürfen Sie sich an die Beispiele mit WordCount erinnern und den Code der vorigen Aufgabe verwenden.*

Aufgabe 5. *Stellen Sie einen Plan auf, wie MinHash (und perspektivisch LSH) auf den beiden Datensätzen ohne und mit MRJob implementiert und getestet werden kann. Formulieren Sie insbesondere das Ziel und die wesentlichen Arbeitsschritte/Meilensteine dahin. Konsultieren Sie ggf. auch das Buch MMDS. Da Sie den Plan hinterher ausführen sollen, machen Sie hier keine Arbeit umsonst.*