1. Konzeptionsphase

Als Basis für dieses Portfolio dient ein Datensatz, der auf der Plattform kaggle.com zur Verfügung gestellt wurde. Er beinhaltet 10000 Restaurant Reviews[1] mit acht Variablen. Zur Vorverarbeitung der Bewertungen werden alle Buchstaben in Kleinbuchstaben umgewandelt, wodurch eine Konsistenz hergestellt und Redundanzen vermieden werden. Als Beispiel können somit „restaurant“ und „Restaurant“ als ein Wort betrachtet werden. Zusätzlich werden Stoppwörter wie „und“, „der“, „die“ entfernt, da sie die Analyse erschweren können.

Benötigte Python-Bibliothek: nltk (mit stopwords-Modul)

Zur Konventierung der Daten in numerische Vektoren werden die Methoden BoW (Bag of Words) und TF-IDF (Term Frequency – Inverse Document Frequency) verwendet. Beide Methoden werden in dem Modul „DLBDSEDA01 – Advanced Data Analysis“ erläutert. Mit Hilfe der BoW Methode wird ein Dokument als Sammlung von Wörtern ohne Berücksichtigung der Reihenfolge betrachtet. Jedes Wort wird in einen Vektorraum abgebildet, wobei die Dimensionen durch den Wortschatz definiert sind. Im Gegensatz dazu gewichtet das TF-IDF Modell die Wörter basierend auf ihrer Häufigkeit in einem Dokument und ihrer Seltenheit in der gesamten Sammlung. Der TF-IDF-Wert eines Wortes wird berechnet, indem die Häufigkeit des Wortes im Datensatz mit der Inversen Häufigkeit der Dokumente, die das Wort enthalten multipliziert wird. Dies hilft, häufige Wörter wie "und" oder "die", die wenig Informationsgehalt haben, zu reduzieren und hebt relevantere Wörter hervor, die für das Dokument spezifischer sind.

Benötigte Python-Bibliothek: scikit-learn

Zur Extraktion der am häufigsten vorkommenden Themen werden die Techniken LSA (Latent Semantic Analysis) und Latent Dirichlet Allocation (LDA) angewendet. LSA hilft dabei, verborgene Themen zu extrahieren, indem sie semantische Beziehungen zwischen Wörtern aufdeckt. LDA hingegen ist ein statistisches Modell zur Themenmodellierung, das Dokumente als Mischung verschiedener Themen und Themen als Mischung von Wörtern beschreibt.

Benötigte Python Bibliothek: scikit-learn

[1] 10000 Restaurant Reviews (kaggle.com)

2. Erarbeitungsphase / Reflexionsphase

Die Ausarbeitung sowie die Quelle der verwendeten Daten wurden in Github abgelegt und sind unter dem folgenden Link zu erreichen: palimpalim08/Phase2_Data_Analysis (github.com)

Zu Beginn wird der Datensatz in Python mithilfe der Bibliothek pandas eingelesen. Der Datensatz enthält Rezensionen, die später analysiert werden.

Verwendete Bibliotheken:

- pandas: zum Einlesen und Bearbeiten der Daten
- re: zur Textbearbeitung und Mustererkennung (z.B. Entfernen von Sonderzeichen)

Um Textdaten für maschinelle Lernmodelle nutzbar zu machen, müssen sie in numerische Form umgewandelt werden. Dies geschieht durch die **Vektorisierung**. Wie in Phase 1 beschrieben sind zwei gängige Techniken hierfür **Bag of Words (BoW)** und **TF-IDF**.

Bei der **Bag of Words (BoW)** Technik wird jedes Dokument in eine Repräsentation umgewandelt, bei der gezählt wird, wie oft jedes Wort im Text vorkommt. Anschließend wird eine Matrix wird erstellt, in der jede Zeile und jede Spalte ein Wort darstellt, gefüllt mit der Häufigkeit des jeweiligen Wortes. Die Anzahl der Wörter kann durch einen Parameter wie max_features begrenzt werden (z.B. 500 häufigste Wörter).

Bei der **TF-IDF (Term Frequency-Inverse Document Frequency)** Technik wird die Häufigkeit eines Wortes mit seiner Seltenheit in dem gesamten Dokument gewichtet. Häufige, aber wenig bedeutende Wörter (z.B. "der", "und") werden so niedriger bewertet als seltene, aber bedeutende Wörter. Analog zu BoW wird hier eine Matrix erstellt, aber mit den TF-IDF-Werten anstelle der bloßen Häufigkeit. Die Ergebnisse der BoW- und TF-IDF-Analysen können dann als CSV-Dateien gespeichert werden.

Der **Coherence Score** bewertet, den Zusammenhang der identifizierten Themen. Ein höherer Coherence Score deutet auf sinnvollere Themen hin.

- **Bibliotheken**:
  - gensim.models.coherencemodel.CoherenceModel: zur Berechnung des Coherence Scores.
  - gensim.matutils.Sparse2Corpus: zur Umwandlung der BoW-Matrix in ein Format, das gensim versteht.
  - gensim.corpora.Dictionary: zur Erstellung eines Wörterbuchs.

Zur Berechnung wird die BoW-Matrix in ein Gensim-kompatibles Format umgewandelt (Sparse2Corpus). Anschließend wird ein LDA-Modell für verschiedene Anzahlen von Themen (z.B. 2 bis 10) trainiert. Der Coherence Score wird für jede Themenanzahl berechnet, indem überprüft wird, wie gut die Wörter innerhalb der Themen zusammenpassen. Die Themenanzahl mit dem höchsten Coherence Score wird als optimal gewählt.

3. Finalisierungsphase

Das Projekt kann unter dem Link palimpalim08/Phase2_Data_Analysis (github.com) abgerufen werden. Zunächst erfolgt eine technische Reflexion des Projektes.

Zur korrekten Anwendung des Codes muss im ersten Schritt der Pfad zu der auszuwertenden Datei angepasst werden. Hierzu ist zunächst Zeile 6 aus „BoW and TF-IDF Analysis“ zu überprüfen: `file_path = 'restaurant_reviews.csv'` Der Name des auszuwertenden Files muss mit dem dort hinterlegten Namen übereinstimmen. Die in dem Beispiel verwendete Datei liegt in dem Repository unter „Data Source“ ab. In der darauffolgenden Zeile 7 `df = pd.read_csv(file_path)` wird die Datei eingelesen und anschließend ausgewertet. Der Python Code enthält vor jedem Schritt eine Kommentarzeile aus der ersichtlich wird, welcher Arbeitsschritt folgen wird.

Bei Bedarf kann der Code um folgende Zeilen erweitert werden:

```
# Speichere die Ergebnisse als CSV – Dateien
bow_df.to_csv(bag_of_words_results.csv', index=False)
tfidf_df.to_csv('tfidf_results.csv', index=False)
```

Hierdurch werden nach erfolgreicher Auswertung automatisch .csv Dateien mit den Ergebnissen erstellt und im Arbeitsverzeichnis abgespeichert. Durch den zusätzlichen Schritt wird eine bessere Dokumentation gewährleistet – was allerdings optional und nicht Vorgabe dieses Projektes ist.

Analog hierzu ist mit „Create Calculation of Coherence Score“ vorzugehen. Der Code liest in der Zeile 11 ebenfalls die Datei ein – die Überprüfung der Bezeichnung erfolgt in diesem Fall in Zeile 10. Der Aufbau des Codes ist identisch strukturiert und erklärt anhand der #Kommentarzeile welcher Arbeitsschritt durchgeführt wird.

Im nächsten Schritt erfolgt die Projektreflexion auf persönlicher Ebene.

Zur Transparenz möchte ich die Punkte in High- und Lowlights unterteilen. Beginnend mit den Lowlights sind folgende Punkte während der Bearbeitung des Projektes aufgefallen:

1. Programmierkenntnisse

Herausforderung: Aufgrund von nur geringen Programmierkenntnissen hat mich die Aufgabenstellung der Projektarbeit zu Beginn vor Probleme gestellt. Es fiel mir schwer einen Einstieg und eine geeignete Struktur zu finden.

Lösung: Als Gegenmaßnahme habe ich das Buch „Python 3 das umfassende Handbuch“ von Johannes Ernesti und Peter Kaiser gekauft. Das Buch bietet eine umfangreiche Einführung in die Programmierung mit Python und eignet sich zur Bearbeitung der Aufgabenstellung.

2. Projektstruktur

Herausforderung: Die Aufgabenstellung hat klar vorgegeben was gefordert wird und welche Schritte in den einzelnen Phasen bearbeitet werden, allerdings fiel es mir zu Beginn schwer die Anforderungen in eine strukturierte und nachvollziehbare Form zu bringen.

Lösung: Das Skript „DLBDSEDA01 – Advanced Data Analysis“ bietet eine gute Übersicht der einzelnen Techniken und hat zu Beginn des Projektes geholfen ein Vorgehen sowie die benötigten Komponenten festzulegen.

3. Codeaufbau

Herausforderung: Bei der Entwicklung von Programmen ist es hilfreich, wenn der Code auch von Entwicklern nachvollzogen werden kann, die ihn nicht programmiert haben. Aufgrund meiner unter Punkt 1 nur rudimentären Programmierkenntnissen habe ich mehrere Anläufe gebraucht, um eine übersichtliche Struktur zu erstellen.

Lösung: Mit Hilfe der Kommentarfunktion möchte ich den Code transparent und für alle nachvollziehbar darstellen.

Trotz der genannten Lowlights gab es auch Highlights, die während der Projektbearbeitung aufgefallen sind:

1. Programmieren

Trotz anfänglicher Schwierigkeiten konnte ich im Verlauf des Projektes ein größeres Verständnis für die Zusammenhänge und die Möglichkeiten des Codes aufbauen, wodurch die theoretischen Inhalte aus dem dazugehörigen Modul DLBDSEDA01 – Advanced Data Analysis nachvollziehbarer wurden

2. Praxisbezug

Das Modul legt den Fokus auf die praktische Anwendung des zuvor theoretisch erlernten Inhalts und bietet dadurch eine interessante Abwechslung zu vielen restlichen Modulen. Aus meiner Sicht sind diese Art von Modulen für den beruflichen Einstieg sehr wichtig, da Programmierkenntnisse / Projektmanagement Skills in diversen Stellenausschreibungen gefordert werden.

Zusammengefasst hat das Modul einen positiven Eindruck hinterlassen und gezeigt, wie das theoretische Wissen in der Praxis angewendet werden kann.