Modellierungsplan

Die Modellierung ist als mehrschrittiger Prozess zu verstehen, welcher sequenziell abläuft. Die dazugehörigen Prozessschritte sind:

- Cleaning
- Preprocessing
- Sprachfilter
- Datentransformation
- Modellauswahl
- Modelbereitstellung

1. Cleaning

Hier werden die Unreinheiten der Kaggle-Daten bereinigt, in dem Verlinkungen in den Nachrichten entfernt werden und die bestehende HTML-Codierung zu UTF-8 verändert wird.

2. Preprocessing

Das Preprocessing beginnt mit einer weiteren Datenaufbereitung, welche dafür sorgt, dass der vorhandene Text reduziert wird auf die in dem Text verwendeten Worte. Dies bedeutet, dass Sonderzeichen, Hashtags und weiteren Zeichen, die keine Informationen enthalten entfernt werden.

Anschließend werden die so genannten Stopwords entfernt, diese befinden sich in einer Textdatei innerhalb des Moduls und soll dazu dienen Worte zu entfernen, die keinerlei Mehrwert für die Klassifikation bilden.

Zuletzt werden die Wörter in ihr Lemmatisiert, also in ihre Grundform zurückgeführt. Dies dient dazu die Heterogenität der Texte zu reduzieren und Wortbedeutungen durch ihre Stammform zu vereinheitlichen.

Bei Bedarf: https://www.datacamp.com/community/tutorials/stemming-lemmatization-python

3. Sprachfilter

Durch die Bereinigung kann nun sichergestellt werden, dass Texte eine einheitliche erkennbare Form hat. Dies machen wir uns zu nutzen und bestimmen die in dem Text verwendete Sprache.

Da das überwiegende Format im Text in Englisch verfasst ist, schränken wir die Trainingsdaten auch entsprechend diese Sprache ein.

Sofern ein Artikel in einer anderen Sprache geschrieben ist, wird er verworfen.

4. Datentransformation

Um die vorhandenen Daten nun entsprechend in ein für die Modelle lesbares Format zu transformieren werden die verwendeten Worte entsprechend codiert.

Diese Transformation findet in 2 Schritten ab.

1. Count Vectorizer:
    - Beginnend wird die gesamte Datenmengen dazu verwendet zu bestimmen, welche Worte im Datensatz vorhanden sind.
    - Aus diesen Worten wird dann für jede Instanz des Datensatzes ein Dictionary erstellt. Die Keys werden durch die Worte aus der gesamten Datenmenge abgebildet. Die dazugehörigen Werte beschreiben, wie oft das jeweilige Wort in Instanz vorkommt.
    - Beispiel:
        - Datensatz:
            - Ich mag Züge
            - Es geht mir gut
            - DHBW finde ich nice nice
        - Dictionary für den dritten Satz ist entsprechend:
            - Ich: 1
            - Mag Züge: 0
            - mir: 0
            - …
            - DHBW: 1
            - finde: 1
            - nice: 2
2. Term-Frequency-Invers-Document-Frequency (TF-IDF):
    - TF:
        - $\frac{Anzahl\ wie\ oft\ ein\ Wort\ im\ Document\ vorkommt}{Anzahl\ aller\ Wörter\ im\ Dokument}$
        - Relative Wahrscheinlichkeit, dass ein zufälliges Wort aus dem Blogeintrag diesem Wort entspricht
    - IDF:
        - $\log_e(Anzahl\ an\ Dokumenten\ /\ Anzahl\ an\ Dokumenten\ mit\ dem\ Wort)$
        - Zeigt auf, wie ein Wort verwendet wird.
    - TF-IDF ist entsprechend gegeben durch TF*IDF
    - http://www.tfidf.com/

Wir werden im Rahmen der Datentransformation zum einen den obigen Prozess durchführen, aber auch Testweise die IDF deaktivieren, um zu erproben, ob dies zu besseren Ergebnissen führt.

5. Modellauswahl

Hier werden verschiedene Modelle verprobt, um deren Erfolg mit entsprechenden Metriken zu Quantifizieren.

Geplant ist, hier die Modelle SGDClassifier, XGBClassifier, LinearSVC für Klassifikationsaufgaben zu verwenden. Parameteroptimierung wird anschließend mittels Grid Search durchgeführt.

Des weiteren ist ein Unsupervised Modell zu erstellen, dieses wird verwendet, um Texte entsprechend ihrer Ähnlichkeiten zu gruppieren und ähnliche Texte vorzuschlagen

6. Modellbereitstellung

Die besten Modelle aus den obigen Parameteroptimierungen werden entsprechend in Form einer Binärdatei (picklefile) exportiert, so dass diese in eine API eingebunden werden können und somit nach außen verfügbar sind.