



Duale Hochschule Baden-Württemberg Mannheim

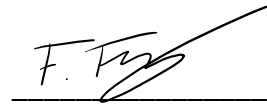
Data Exploration Project – Sentiment-Analyse auf Tweets zu Videospielen

Wirtschaftsinformatik Schwerpunkt Data Science

Verfasser (Matrikelnummer):	Florian Frey (7199749) Olena Lavrikova (5436924) Frederick Neugebauer (4521985) Anh Vu (1039624)
Kurs:	WWI 20 DSB
Modul:	Machine Learning Fundamentals
Studiengangsleiter:	Prof. Dr. Bernhard Drabant
Dozent:	Simon Poll
Bearbeitungszeitraum:	11.05.2022 - 13.07.2022

Ehrenwörtliche Erklärung

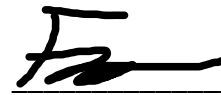
Wir versichern hiermit, dass wir die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben.




12.07.2022, Florian Frey



12.07.2022, Olena Lavrikova



12.07.2022, Frederick Neugebauer



12.07.2022, Anh Vu

Inhalt

1. Einleitung	4
2. Verwendete Technologien	4
3. Ergebnisse	5
4. Bewertung der Ergebnisse und Fazit	6
A. Anhang – Anmerkungen zum Quellcode	8
Literaturverzeichnis	9

1. Einleitung

Thema des Projekts ist die Durchführung einer Sentiment-Analyse von Tweets zu Videospielen, um die allgemeine Stimmung über den aktuellen Stand des Spiels zu untersuchen.

Unter einer Sentiment-Analyse versteht man die Bestimmung der Emotion in einem Text mit Hilfe von Natural Language Processing (NLP). Grundsätzlich geht man davon aus, dass die Emotion grob in „positiv“, „negativ“ oder „neutral“ klassifiziert werden kann (vgl. Siegel & Alexa 2020, S.5f.). Wendet man diese Analyse nun auf Äußerungen über Produkte, Firmen, Events an, können hieraus wichtige Schlüsse für die Unternehmen beziehungsweise Organisatoren gezogen werden.

Twitter hat sich zu einer Quelle vielfältiger Informationen entwickelt. Das Konzept von Twitter ist, dass dort Menschen in Echtzeit ihre Meinungen teilen, diskutieren und sich über aktuelle Themen äußern können. Dies geschieht in sogenannten Tweets, welche maximal 280 Zeichen enthalten. Unternehmen die Produkte und Dienstleistungen herstellen, haben angefangen diese Tweets, welche die für sie relevanten Keywords enthalten, abzufragen, um ein Bild von der allgemeinen Meinung über ihr Produkt zu erhalten.

Mithilfe unseres Projekts soll eine Analyse für Spieleentwickler bereitgestellt werden, anhand derer sie eine allgemeine Stimmungsübersicht über ihr Videospiel erhalten. Diese können sie beispielsweise nutzen, um auf Feedback einzugehen und ihre Spieler damit zufriedenzustellen. Analysiert man die Stimmung zu bestimmten Zeitpunkten oder in einem Zeitverlauf, können die Entwickler daraus schließen, wie beispielsweise neue Änderungen an dem Spiel ankommen und dementsprechend reagieren. Letztendlich können die Unternehmen dadurch an Spielerzufriedenheit gewinnen, bessere Spiele entwickeln und möglicherweise neue Publisher von sich überzeugen, um damit ihren Profit zu steigern.

Ein ähnliches Projekt wurde bereits von Chakraborty et al. im Jahr 2018 umgesetzt. Sie haben Modelle zur Sentiment-Analyse auf einem Datensatz von Game-Reviews trainiert, um die Modelle anschließend auf Tweets zu Videospielen anzuwenden. Zur Anwendung kamen hier unter anderem ebenfalls ein Naive Bayes Klassifizierer und eine Support Vector Machine. Über Sentiment-Analyse in anderen Gebieten gibt es zahlreiche Literatur und andere Medien.

2. Verwendete Technologien

Zum Programmieren wurde die Programmiersprache Python verwendet. Es wurde sich für Python entschieden, da eine große Zahl an fertigen Modulen vorhanden ist, mit deren Hilfe sich Machine-Learning Modelle entwickeln lassen. Sowohl Datenaufbereitung als auch das Trainieren des Modells konnte so durchgeführt werden.

Für den Import der Daten wurden Pandas verwendet. Außerdem konnten Funktionen von Pandas benutzt werden, um Daten zum weiteren Verarbeiten zu formatieren.

Nltk ist ein Werkzeugkasten des NLP. Mithilfe dieses Moduls konnten die Twitter-Daten vorverarbeitet werden, um diese in einen Zustand zu bringen, mit dem der Machine-Learning-Algorithmus arbeiten und ein bestmögliches Ergebnis erzielen kann. Dazu gehören das Entfernen von Stopwords, also irrelevanten Worten, sowie Zeichen, aber auch das Aufteilen der Tweets in ihre einzelnen Wörter. Außerdem wurde probiert Wörter in ihre Stammform zurückzuführen, jedoch hat dies das Ergebnis teilweise verschlechtert. Außerdem wurde der TF-IDF-Vectorizer des Moduls benutzt, um den Wörtern einen numerischen Wert zuzuordnen. Anhand der Term-Frequency (TF) also der Häufigkeit der Wörter und der inversen Dokumenten-Häufigkeit (IDF) der Wörter wird hierbei eine Gewichtung für jedes Wort bestimmt.

Zum Trainieren und Analysieren des Modells wurde Scikit-Learn verwendet. Hierzu wurden der Multinomial-Naive-Bayes-Klassifizierer, sowie eine Support-Vector-Machine getestet. Der Naive-Bayes berechnet die wahrscheinlichste Klasse nach dem Satz von Bayes und geht dabei davon aus, dass alle Features unabhängig voneinander sind. Dieser Algorithmus ist sehr simpel und wird oft als Baseline-Algorithmus in Textklassifikation angesehen (vgl. Xu 2017), ist jedoch schnell und kann in manchen Fällen zu guten Ergebnissen führen.

Die Support-Vector-Machine berechnet eine Linie beziehungsweise Ebene, die die verschiedenen Datenpunkte im Raum unterteilt und somit einer Klasse zuweist. Diese sogenannte Hyperplane versucht einen möglichst großen Abstand zwischen den nächsten Trainingsdatenpunkten der verschiedenen Klassen zu erreichen, um dann neue Datenpunkte der richtigen Klasse zuzuweisen. (vgl. Gandhi 2018)

Um Tweets für die Analyse zu erhalten, wurde die Twitter API v2 verwendet. Zugang zu den Schnittstellen der API erhält man durch das Erstellen eines Twitter-Developer-Accounts. Die Abfrage via URL kann ebenfalls via Python durchgeführt werden, um die Antworten der API anschließend direkt weiterverarbeiten zu können. Zudem kann die Query, also die Sucheinstellung, ebenfalls einfach angepasst werden.

3. Ergebnisse

Die Genauigkeit des verwendeten Klassifizierers zum Analysieren der Stimmung von Tweets erreichte eine Genauigkeit von 71% bei einem Trainingsdatenanteil von 80%, zu einem Testdatenanteil von 20%. Auf den selbst gelabelten Testdaten wurde eine Genauigkeit von 63% erreicht. Bei diesen Tweets handelt sich um die von der Twitter API abgefragten Tweets. Sie wurden genau wie der Trainingsdatensatz vorverarbeitet. Zu Beginn des Projekts wurde zunächst ein Klassifizierer nur mit den Klassen positiv und negativ trainiert. Jedoch stellte sich schnell heraus, dass zum einen dieser Ansatz weit von der Realität entfernt ist, als auch die Ergebnisse mit 76% Genauigkeit nicht zufriedenstellend waren. Dabei muss angemerkt werden, dass eine Genauigkeit von 71% mit drei Klassen schwieriger zu erreichen ist als 76% bei lediglich zwei Klassen. Zudem wurden die 71% mit einer Support Vector Machine erzielt, während die 76% im binären Fall mit dem Naive Bayes Klassifizierer erzielt wurden. Die Tweets zum Trainieren stammen von zwei gelabelten Datensätzen von Kaggle. Im Fall des binären

Klassifiziertes handelt es sich um 800.000 positiv, sowie 800.000 negativ gelabelte Tweets. Beim Datensatz mit drei Klassen gab es 11.118 neutral, 8582 positiv und 7781 negativ gelabelte Tweets. Bei beiden Ansätzen konnte die Genauigkeit der Support Vector Machine um einige Prozentpunkte durch Hyperparametertuning verbessert werden.

4. Bewertung der Ergebnisse und Fazit

Das Ergebnis des Modells auf dem verwendeten Trainings- und Testdatensatz mit positiver, negativer und neutraler Klasse ist mit circa 71% Genauigkeit nach Hyperparametertuning mit dem verwendeten Algorithmus zufriedenstellend. Möglicherweise wäre mit einem neuronalen Netz oder anderen komplexeren Algorithmen ein noch besseres Ergebnis möglich gewesen, jedoch lag der Aufwand für die Umsetzung eines solchen über dem Projektumfang.

Kritisch zu bewerten ist die Qualität der Videospiel-Tweets, auf welche das Modell letztlich angewendet wurde, um Ergebnisse für den Use-Case des Projekts zu erhalten. Beim manuellen Labeln einiger dieser Tweets ist aufgefallen, die Interpretation des Sentiments oft schwierig ist. Dies liegt zum einen daran, dass ein großer Anteil der Tweets von Bots automatisch generiert wurde und lediglich Werbung oder betrügerische Links enthält. Zum anderen auch daran, dass Videospiele im Allgemeinen ein popkulturelles Thema sind und viele Äußerungen von echten Menschen keine eigentliche Meinung direkt zu dem Spiel enthalten, sondern das Spiel nur anderweitig erwähnen.

Die wohl größte Herausforderung bei der Umsetzung des Projekts war also der fehlende „Academic Research“ Zugang zur Twitter API. Dadurch fehlten Funktionen wie „full-archive-search“ und erweiterte Filter-Operatoren.¹ Mit diesen hätte man zum einen statt nur auf die chronologisch aktuellsten Tweets, auf das gesamte Archiv an Tweets zurückgreifen können. Zudem wäre es möglich gewesen die Qualität der Tweets durch weitere Filter direkt in der Abfrage bei der API zu steigern, indem beispielsweise verschiedene Sources² herausgefiltert werden. Diese Filterung konnte teilweise auch anschließend im Python-Code vorgenommen werden, jedoch wurde dadurch die bereits kleine Menge an Tweets noch weiter verschmälert. Denn auch die Anzahl an Tweets wurde durch die Zugangsstufe der Twitter API auf gerade einmal 100 Tweets pro Abfrage beschränkt.

Bei der Anwendung des trainierten Modells auf die circa 600 selbst gelabelten Tweets ist auch aufgefallen, dass das Modell ein Bias auf neutrale Vorhersagen aufweist (vgl. Abbildung 1). Das heißt, dass sowohl negative also auch positive Sentiments häufig (zu 59 beziehungsweise 73 Prozent) vom Modell als neutral bewertet werden. Ursache hierfür könnte sein, dass sich die Videospiel-Tweets stark von den Trainingsdaten unterscheiden, oder dass das Sentiment beim Labeln unterschiedlich interpretiert wurde. Auch dass der Trainingsdatensatz größtenteils aus neutralen Tweets besteht, könnte hierauf Einfluss haben. Zudem beinhaltet

¹ Tabellarische Übersicht über die Twitter API Access Levels und deren Funktionen unter <https://developer.twitter.com/en/docs/twitter-api/getting-started/about-twitter-api#v2-access-level> verfügbar

² Die Source beschreibt die Quelle, von welchem der Tweets abgesendet wurde (beispielsweise aus der offiziellen Twitter App oder von anderer Software). Diese konnten bei der API-Abfrage mit abgerufen werden.

der Trainingsdatensatz nicht hauptsächlich Tweets, welche sich auf Videospiele beziehen und popkulturelle Sprache aufweisen.

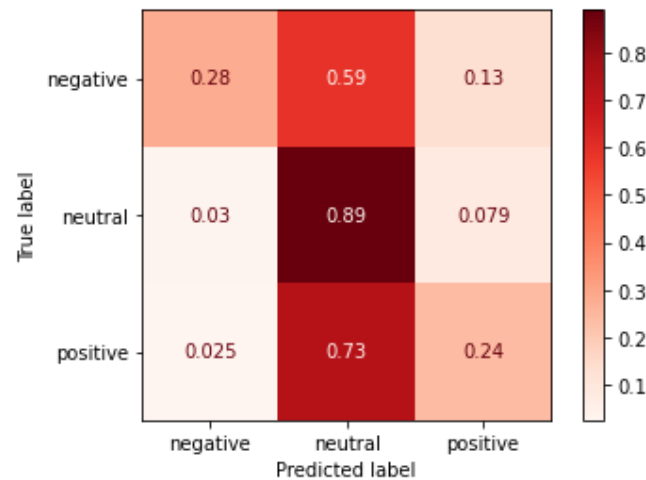


Abbildung 1: Confusion Matrix der Support Vector Machine auf den selbst gelabelten Gaming-Tweets

Kritischer als das neutrale Bias wären jedoch Tweets, die ein positives Sentiment beinhalten und vom Modell als negativ bewerten oder vice versa, da diese die aus der Analyse abgeleiteten Entscheidungen beeinflussen würden. Diese Fehlklassifizierung halten sich jedoch in einem angemessenen Rahmen (vgl. Abbildung 1).

Um für unseren in der Einleitung beschriebenen Business-Use-Case ein gutes Produkt anbieten zu können, müsste die Genauigkeit des Modells noch weiter verbessert werden. Beispielsweise indem man das Modell auf videospiegelbezogenen Tweets trainiert, andere Algorithmen testet und die Datenvorverarbeitung weiter ausbaut. Zudem könnte die Abfrage der API und die Analyse der Tweets automatisiert werden. Die Bereinigung der Tweets müsste ebenfalls noch erweitert werden, damit noch mehr Spam und für das Sentiment irrelevante Posts herausgefiltert werden können. Für die Kunden könnten die Ergebnisse schließlich grafisch aufbereitet und auf einem Dashboard bereitgestellt werden.

A. Anhang – Anmerkungen zum Quellcode

Um den Quellcode auszuführen eine funktionierende Python 3.x Installation benötigt. Mit folgendem Befehl können im Terminal, welches das Verzeichnis des GitHub-Repositories des Projekts ausgewählt hat, die benötigten Module installiert werden:

```
pip install -r requirements.txt
```

Hierbei wird auch das Jupyter Notebook Modul installiert. Dieses kann anschließend mit dem Befehl `notebook` gestartet werden. Alternativ kann eine Erweiterung in Visual Studio Code oder einer anderen Entwicklungsumgebung installiert werden, um die Jupyter Notebooks zu öffnen.

Um mit dem Notebook „`twitter_search_request.ipynb`“ Antworten von der API zu erhalten, muss ein Bearer Token in der zweiten Code-Zelle eingefügt werden. Den Bearer Token erhält man beim Registrieren eines Projekts im Twitter Developer Portal. Der Token dieses Projekts kann auf Anfrage zur Verfügung gestellt werden.

Das Notebook „`tweets_to_excel.ipynb`“ fügt alle API-Antworten in eine Excel-Datei zusammen welche anschließend analysiert werden kann. Ohne vorherige API-Abfragen ist dieses Notebook also nutzlos.

„`training_binary.ipynb`“ beinhaltet das Training auf dem binären Datensatz. Da diese Modelle jedoch nicht verwendet wurden sollte eher das Notebook „`training_multiclass.ipynb`“ angeschaut werden. Alle Bestandteile aus dem ersten Code werden hier ebenfalls wieder verwendet. Die Zellen sollte alle ohne Probleme auszuführen sein. Eventuell muss der Pfad zum Auslesen der Trainings- und Analysedateien angepasst werden. Die Zelle, welche das Hyperparameter tuning mit GridSearch beinhaltet, ist auskommentiert, da diese sehr viel Zeit in Anspruch nimmt.

Weitere Erklärungen zu den genauen Aufgaben der einzelnen Code-Abschnitte sind durch Markdown oder Kommentare in den Jupyter Notebooks selbst enthalten.

Literaturverzeichnis

- S. Chakraborty, I. M. (2018). *Rating Generation of Video Games using Sentiment Analysis and Contextual Polarity from Microblog*. International Conference on Computational Techniques: Electronics and Mechanical Systems (CTEMS).
- Siegel, M., & Alexa, M. (2020). *Sentiment-Analyse deutschsprachiger Meinungsäußerungen*. Wiesbaden: Springer Fachmedien Wiesbaden.
- Xu, S. L. (2017). Bayesian Multinomial Naïve Bayes Classifier to Text Classification. *Park, J., Chen, SC., Raymond Choo, KK. (eds) Advanced Multimedia and Ubiquitous Engineering. FutureTech MUE 2017 2017. Lecture Notes in Electrical Engineering, vol 448*. Singapore: Springer.