DHBW
Duale Hochschule
Baden-Württemberg
Mannheim

Twitter Sentiment Analyse für Cryptowährung Dogecoin

Prüfungsleistung Data Exploration Project

im Studiengang WWI-19DSB

an der Dualen Hochschule Baden-Württemberg Mannheim

von

Luca Fenucciu, 2512023

Kilian Ebi,

Miguel Sarasa-y-Zimmermann,

Sebastian Hildenbeutel,

Abgabedatum 12.07.2021

1 Projektbeschreibung

Für die Prüfungsleistung der Vorlesung „Data Exploration Project“ wird von der Gruppe eine Sentimentanalyse in Bezug auf die Tweets bezüglich der Cryptowährung Dogecoin durchgeführt. Hierzu bezieht die Gruppe einen geeigneten Datensatz mit Hilfe der Twitter API. Dieser Datensatz wird von der Gruppe bereinigt und zur Verarbeitung vorbereitet.

Für die Durchführung einer Sentimentanalyse wird ein Modell zur Vorhersage des Sentiments trainiert. Als Trainingsdaten dienen Trainings-Tweets des nltk Pakets in Python. Das Modell wird von der Gruppe gemäß den Machine Learning Standards trainiert, evaluiert und optimiert.

2 Umsetzung

2.1 Datenbeschaffung

Es gibt einige Voraussetzungen für das Beschaffen der Twitter Daten mit einem eigenen Zugang zur Twitter API.

1. Erstellen eines Twitter Developer Accounts, um überhaupt die notwendigen Berechtigungen zu erlangen über die Twitter API Tweets ziehen zu können.
2. Nachdem der Access zur Twitter API von Twitter gewährleistet wurde, muss eine App in dem Twitter Developer interface initialisiert werden.
3. Beim Initialisieren der APP werden einem direkt die Auth Tokens und die API Access Tokens generiert und angezeigt, diese unbedingt sofort speichern, da sie sonst neu generiert werden müssen.

Im Folgenden werden Befehle zur Installierung von Kafka Servern und den notwendigen Packages in der Syntax von Ubuntu Linux OS beschrieben, der Prozess ändert sich allerdings nicht mit einem anderen OS.

- Python3 Installation (für die Ausführung des Scripts) 'apt install python3'
- pip Installation (zum Installieren von Packages) 'apt install python3-pip'
- Tweepy Installation (Package speziell, um Twitterdatenmanipulation in Python zu vereinfachen) 'pip install tweepy'
- Kafka Installation (Server Kontaktpunkt kommuniziert mit der Twitter API und requested streamed Data nach welcher im Skript gefragt wird) 'pip install kafka-python'

- Python Twitter Installation (Python Interface für die Twitter API) 'pip install python-twitter'

Damit sind alle Voraussetzungen erfüllt.

Wir benötigen in den nächsten Schritten mehrere command line interfaces, denn wir möchten sowohl unseren Kafka Server, als auch unseren Listener im Hintergrund parallel zum Skript ausführen.

- window 1: zookeeper server: (zur Verwaltung und der Orchestrierung von eingehenden Daten und ausgehenden queries 'Management-tool') ./bin/zookeeper-server-start.sh -daemon config/zookeeper.properties (start des Zookeeper servers)

  kafka server: (Datenstromzugang Verbindung mit unseren gewünschten Tweets, erleichtert durch Aufbereitung nachfolgende Analyse) ./bin/kafka-server-start.sh config/server.properties

- window 2: topic creation: ./bin/kafka-topics.sh --create --bootstrap-server 192.168..:9092 --replication-factor 1 --partitions 1 --topic doge

  output file creation: (Text-Datei, in welcher wir unsere Daten speichern für die Weiterverarbeitung) ./bin/kafka-console-consumer.sh --bootstrap-server 192.168..:9092 --topic doge --from-beginning > cryptodoge.txt

Bearbeiten der Server.properties file um die richtigen Listener zu definieren Geräte IP eingeben an Stelle des Listerners (mit welcher IP wird verbunden) Verbindung zum Bootstrap Server Eingabe der eigenen IP mit Port, welcher im Fall von Kafka meist 9092 ist

- window 3: run python: (Skriptausführung hier geben wir an was wir genau wollen und wie wir es beziehen (sprich Server configs und Schlüsselwörter), außerdem verifizieren wir uns gegenüber Twitter am Anfang nach den Package Imports) python3 Tweets.py .

Tweets werden daraufhin in der hierfür angelegten Datei mit JSON Syntax gespeichert. Link zu unserer VM umgebung: https://drive.google.com/file/d/1zEigml1mJAgY2jle0-eTY0Pw3vBQvy5C/view?usp=sharing

2.2 Datenverarbeitung

Die Tweets beinhalten eine Reihe an Informationen, welche für die spätere Date Auswertung von Bedeutung sein könnten (mögliche Korrelationen). Aus diesem Grund ist es zumindest anfänglich von Vorteil die scheinbar überflüssigen Informationen nicht zu verwerfen. Hierfür empfiehlt sich die Arbeit mittels eines DataFrames dessen Spalten sich einfach auswerten/bearbeiten lassen.

Bevor die Daten in das Modell eingeschleust werden können, sollten sie bereinigt werden. Hierfür betrachtet man zuerst den Texten und macht jegliche Zeichenkombination aus welche man herausfiltern möchte. Oft sind das Twitternamen, URL´s… Obwohl auf den ersten Blick Emojis nicht dazu gehören werden diese auch herausgefiltert da das Modell mittels Standard Tweets trainiert wird, die Crypto Twitter-community jedoch ihre ganze eigenen Emojicon Sprache besitzt. Aus demselben Grund sollte man auch alle nicht englischen Tweets löschen da das Modell auf einem rein englischen Datensatz trainiert wurde.

2.3 Erstellung des Modells

Um die Sentimentanalyse durchzuführen, wird ein Machine-Learning-Modell trainiert. Die Entscheidung ist dabei auf einen Naive-Bayes Klassifikator mit multinomialer Verteilung gefallen: multinomialNB aus dem Paket sklearn. Dieser Klassifikator kann gut mit gezählten Wörtern umgehen und benötigt deshalb als Eingabe die Anzahl bestimmter Wörter pro Tweet. Dabei müssen alle verwertbaren Wörter bereits beim Training bekannt sein. Das wird erreicht, indem ein Vectorizer vorgeschaltet wird. Der TFIFD-vectorizer von sklearn spaltet die Tweets in Wörter (oder Kombinationen von Wörtern) und gewichtet diese Tokens invers zu ihrer Häufigkeit in den Daten. Beides zusammen wird in einer sklearn-pipeline vereint, welche im weiteren Verlauf unser Modell wird. Für das Training wird der sample_tweets-Datensatz von nltk verwendet.

2.4 Optimierung des Modells

Das Modell liefert bereits mit direktem Training gute Ergebnisse auf den Testdaten. Diese Ergebnisse können jedoch durch Hyperparameteroptimierung ver-

bessert werden. Dafür werden die Ansätze Grid Search und Random Search mit dem ursprünglichen Modell verglichen.

Beide Ansätze verlangen ein Parameter-Grid, in dem die Kandidaten für Hyperparameter angegeben werden. Es enthält hier sowohl Parameter für den Vectorizer, als auch für den MultinomialNB-Klassifikator.

Bei MultinomialNB werden die Parameter alpha und fit_prior optimiert.

Alpha gibt an, wie viel Laplace-Smoothing angewandt wird.

Fit_prior gibt an, ob A-Priori-Wahrscheinlichkeiten gelernt werden sollen.

Der TFIFD-Vectorizer kann aus den verwendeten Texten sehr unterschiedliche Tokens generieren.

Mit dem Parameter use_idf wird angegeben, ob oft vorkommende Tokens geringer schwer gewichtet werden oder nicht.

Ngram_range gibt an, wie viele Wörter zu einem Token zusammengefasst werden.

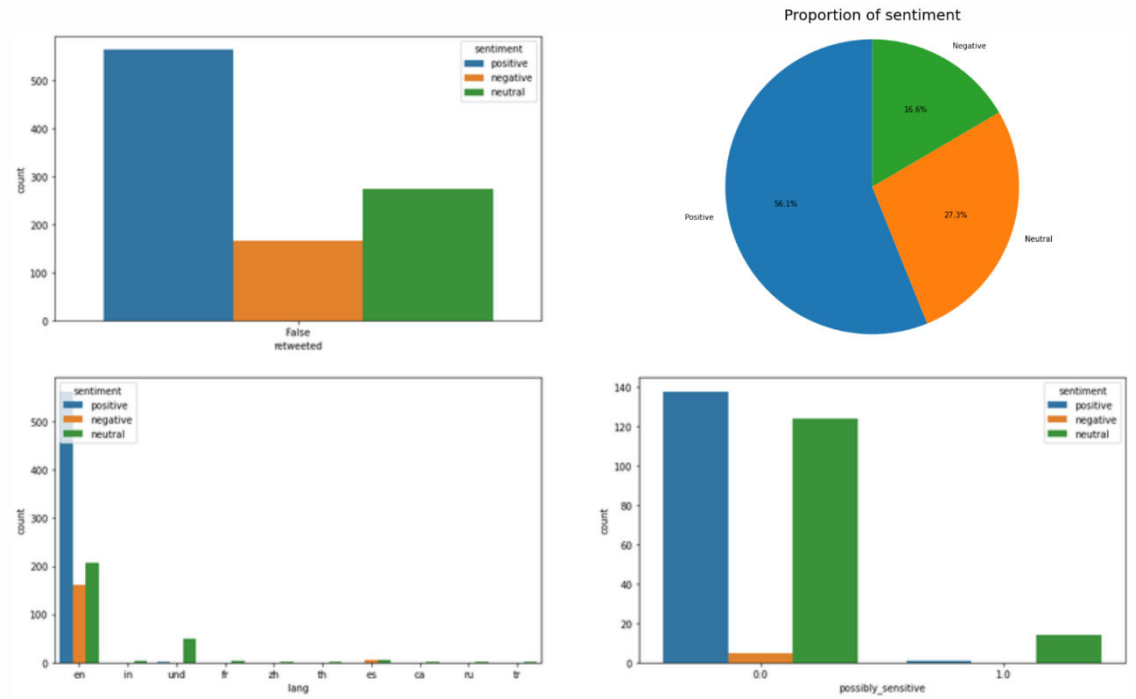
Im Parameter stopwords wird optional eine Liste von zu ignorierenden Wörtern übergeben. Diese Liste wurde von https://countwordsfree.com/stopwords übernommen. Je nach Trainingsdatenlage entscheidet die Hyperparameteroptimierung, ob das sinnvoll ist.

Als tokenizer wird bei Bedarf, der von ntlk bereitgestellte WordNetLemmatizer verwendet. Dieser bildet den Stamm von konjugierten Wörtern und nutzt ihn als Token, wodurch ähnliche Wörter vergleichbarer werden. Auch hier entschiedet die Hyperparameteroptimierung, ob Lemmatisierung das Ergebnis verbessert.

Optimierung des Modells

	Unoptimiert	Grid Search	Random Search
Precision negatives Sentiment	76.20%	76,99%	77,66%
Precision positives Sentiment	77.18%	77.15%	77.95%
Recall negatives Sentiment	78.80%	78.40%	79.19%
Recall positives Sentiment	74.46%	75.68%	76.36%
F1-Score negatives Sentiment	77.48%	77.69%	78.42%
F1-Score positives Sentiment	75.80%	76.41%	77.14%
Accuracy	76.67%	77.07%	77.80%

2.5 Auswertung



Man kann hier generell ein sehr positiv konnotiertes Sentiment erkennen, das liegt vor allem daran, dass sich vor allem Leute, die einen interessierten und begeisterten Bezug zu den auf Twitter debattierten Themen haben, zu den jeweiligen Themen aussprechen.

Zusätzlich sind unsere Vorliegenden Daten von einer Quelle, nämlich Twitter, sollte man die Datenquelle ausweiten auf mehrere Foren und Newsfeed Dienste würde sich auch auf der negativen Seite ein größeres Sentiment wieder fin-

den. Denn wenn Menschen sich die Zeit nehmen einen Tweet oder Beitrag zu verfassen, dann wollen sie mit der Veröffentlichung auch andere Menschen erreichen, hierbei über ein Thema zu sprechen, das einem gefällt fällt den meisten Menschen auch erheblich einfacher, als fundierte und konstruktive negative Kritik zu äußern.

3 Fazit und Ausblick

Die Durchführung des Projekts hat mehrere Probleme einer Sentimentanalyse in Bezug auf Tweets offenbart. Ein Problem ist, dass ein Machine Learning Algorithmus Sarkasmus, Memes und Humor nur schlecht erkennen kann. Dadurch können manche Tweets durch den Algorithmus nicht richtig zugeordnet werden. Des Weiteren ist es in diesem Modell nicht möglich Bilder zu interpretieren, was bedeutet, dass viele Tweets nicht analysiert werden können.

Auch Werbung zu neuen Crypto Coins stellen ein weiteres Problem dar, denn die Developer einer neuen Währung sprechen sich natürlich positiv bezüglich dieser aus und taggen, um mehr Leser zu erreichen mit ihrer Werbung auch bereits etablierte Coins, diese Tweets werden auch von uns erfasst.

Die Gruppe sieht in dem Modell Potential. Das Modell kann noch erweitert werden, um beispielsweise Bilder noch zu erkennen und Sarkasmus auch teilweise besser interpretieren zu können. Dies ist aufgrund der begrenzten Durchführungszeit allerdings noch nicht möglich gewesen. In Bezug auf den wirtschaftlichen Nutzen ist es möglich mit dem Modell das Sentiment in Korrelation zu dem aktuellen Kurs zu setzen. Hierzu ist es möglich eine Webapp zu programmieren in dem automatisch täglichen Tweet gezogen werden und das Sentiment automatisch analysiert wird. Dies wiederum bietet dann die Möglichkeit durch die dadurch gewonnen Erkenntnissen ein Modell zur Vorhersage des Kurses zu trainieren.