CS 410 Project Investy Documentation

Team:
Name: Investors
Team captain: Akash Patel - apate400
Members: Rushil Thakkat - rthakk20, George Obetta -

Overview

The main function of our webapp is to execute a stock market sentiment analysis
to predict stock appreciation or depreciation based on tweets gathered from Twitter on
the day a specific stock searched by the user. We display to the user the percentage of
positive, negative and neutral tweets which will indicate how confident our prediction is.

Implementation Details

Frontend:
For the frontend of the application, we created a html template and styled it using
bootstrap css. We used this template and styling in conjunction with the python flask
application. The predicted sentiment gotten from the model was then formatted and
displayed on this template in a tabular format.

API/Middleware:

To create an endpoint which can be used by the frontend we created a Flask app with
an API that has one simple get request url that takes one query parameter which would
be the name of the stock.

After retrieving the stock name, the Python requests library is used to make a get
request to the search endpoint of the Twitter API. Proper authentication needed to be
done to access Twitter API and also add query parameters to change how many
resulting tweets we would get and also the time frame from when they were posted.
Once the tweets are retrieved they are passed over to the model to perform the
sentiment prediction and display the conclusions on to the frontend.

Backend:

For the sentimental analysis, we used Naive Bayes classifier to perform sentiment
prediction on tweets collected from Twitter API for the day the user searches the stock

on. Initially, we were going to use the BERT model for this part, but then we checked the accuracy and robustness of that model and it was lower compared to Naive Bayes and that is why we chose to go with it for our final implementation. The initial accuracy with the training data we got from Kaggle dataset of tweets for Naive Bayes model was around 75%, but then we decided to improve this by also using NLTK's movie reviews dataset to also include in our training to make the model more robust and effective in the sentiment prediction. After this, we were able to increase our accuracy to 85% on the test set. We also performed a lot of clean up on the dataset before the training such as removing stopwords, hashtags, emojis, phrases such as "retweets", etc to perform better training. Finally, we perform sentiment prediction on tweets received from Twitter API and aggregate the predictions as positive, negative, and neutral as percentage to display them to the user to help them understand if they should invest in the stock or not.

Instructions

Installation:

Use the pip command below and replace the package with the packages you want to install. Packages need for this project are pandas, numpy, python-twitter, nltk, seaborn, requests

Kaggle Dataset: https://www.kaggle.com/utkarshxy/stock-markettweets-lexicon-data

Download the dataset above from the given link. Make sure to switch all paths that point to the local copy of the dataset in the code to the path you have downloaded to. Switch path on line 67 twitterapi.py

Use the command "python -m nltk.downloader all" to download the movie reviews dataset for training.

pip install package

How to Run:

To start the webapp run the following command "python app.py" in WebDev-master directory. After the Flask app has successfully run, follow the link in the terminal to the running webapp. Navigate to the browser page running the app and type the name of

the stock in the search bar and press submit to get the results from the model as described above.

Warning: After pressing submit, it might take up to a 1 or 2 min to get back the results because of the Twitter API call, parsing the data, training, and predicting results.

Team Member Contributions

Akash Patel
- Researched how to use Flask to build webapp
- Made a get request to the Twitter API through Postman to test response
- Developed Python code that makes Twitter API Request
- Connected to frontend and also to the backend model

Rushil Thakkar
- Researched different sentimental models
- Cleaned the training and testing dataset by removing stopwords, emojis, etc
- Developed the model on backend using Naive Bayes for sentiment prediction
- Improved the accuracy by leveraging NLTK's movie reviews dataset for sentiment analysis training