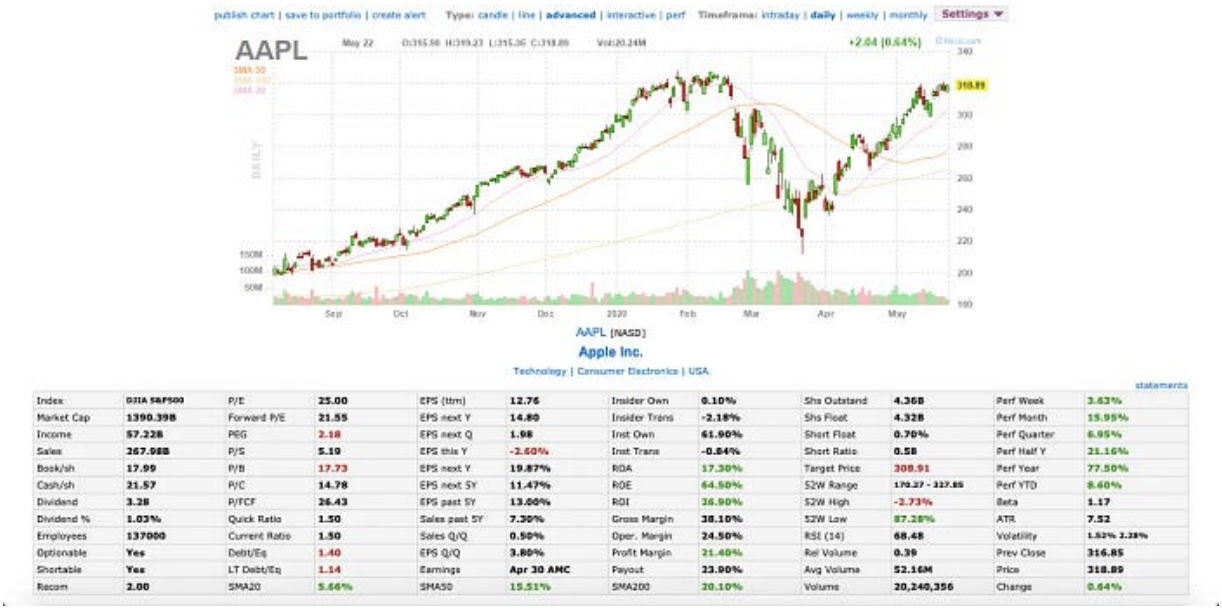


Analyzing Stock Price News Sentiment with Machine Learning Models in Python

Introduction: In today's fast-paced stock market, news sentiment plays a crucial role in shaping investor behavior and influencing stock...

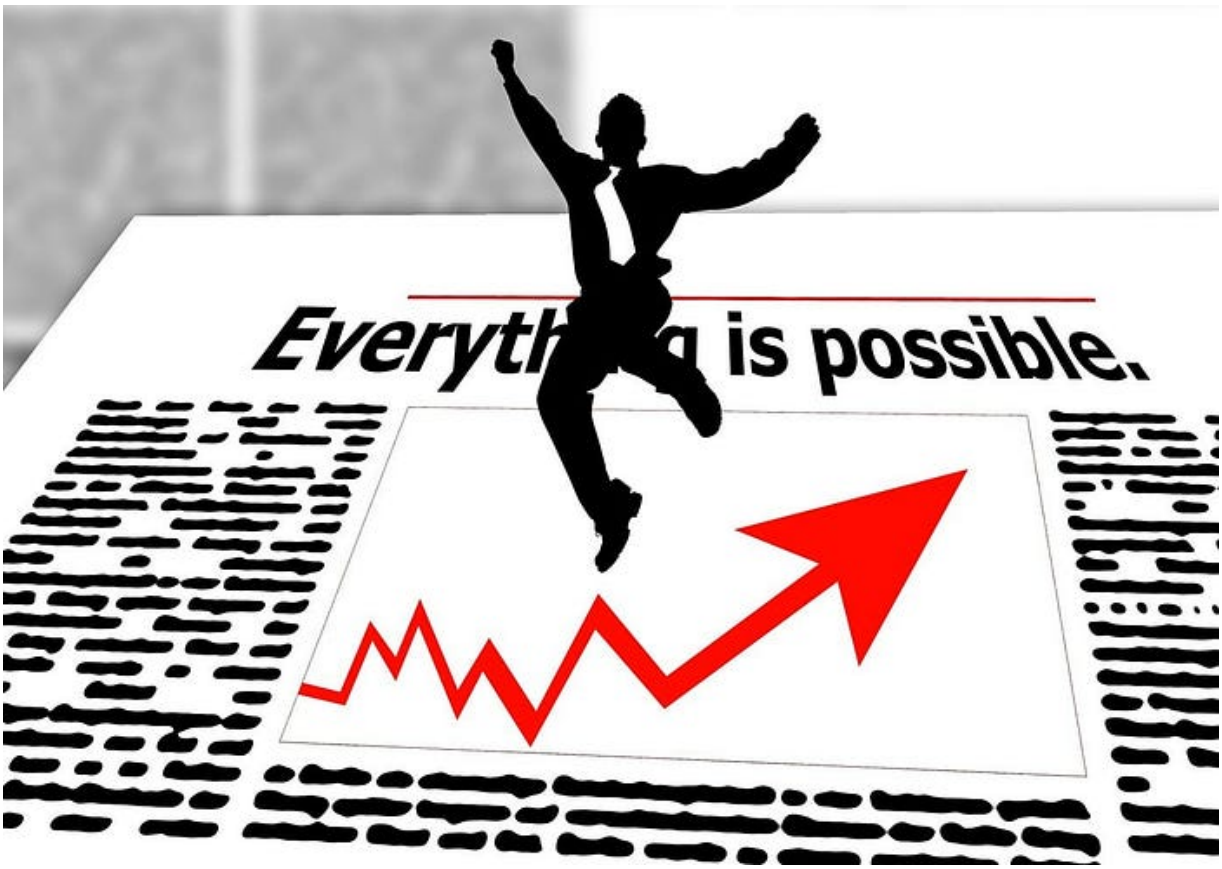
Sahaj Godhani

Follow

androidstudio · June 4, 2023 (Updated: August 16, 2024) · Free: No

Introduction: In today's fast-paced stock market, news sentiment plays a crucial role in shaping investor behavior and influencing stock prices. By analyzing the sentiment behind stock-related news articles, investors can gain valuable insights to make informed trading decisions. In this blog, we will explore how machine

training ML models to predict sentiment, enabling investors to stay ahead of market trends.



News Sentiment For Stock Market

1. **Understanding the Role of News Sentiment in Stock Markets:** Sentiment analysis refers to the process of determining the emotional tone or sentiment of a text. In the context of stock markets, sentiment analysis helps gauge the positive, negative, or neutral sentiment of news articles related to stocks. By understanding the sentiment behind such news, investors can uncover potential market trends and sentiment-driven price movements.
2. **Collecting Stock Price News Data:** To begin, we need a reliable

Alpha Vantage or Yahoo Finance. These APIs offer access to a wide range of news articles related to specific stocks. In this blog, we will use the Alpha Vantage API to fetch the news data.

3. **Preparing the Data for Sentiment Analysis:** Once we have collected the news data, we need to preprocess it before feeding it into our ML models. Key preprocessing steps include:

- Text cleaning: Removing unnecessary characters, punctuation, and special symbols.
- Tokenization: Breaking down the text into individual words or tokens.
- Stop-word removal: Removing common words that do not carry significant sentiment.
- Lemmatization or stemming: Reducing words to their root form for better analysis.

4. **Building a Sentiment Analysis Model:** In this section, we will build an ML model to classify news sentiment. We can employ various ML algorithms, such as Naive Bayes, Support Vector Machines (SVM), or even deep learning models like Recurrent Neural Networks (RNNs) or Transformers. For this blog, we will use a simple yet effective algorithm called Multinomial Naive Bayes, which is well-suited for text classification tasks.

5. **Training and Evaluating the Model:** To train the sentiment analysis model, we need labeled data, where each news article is tagged with its corresponding sentiment (positive, negative, or neutral). This labeled data serves as the ground truth for training the

performance on the testing set using evaluation metrics such as accuracy, precision, recall, and F1-score.

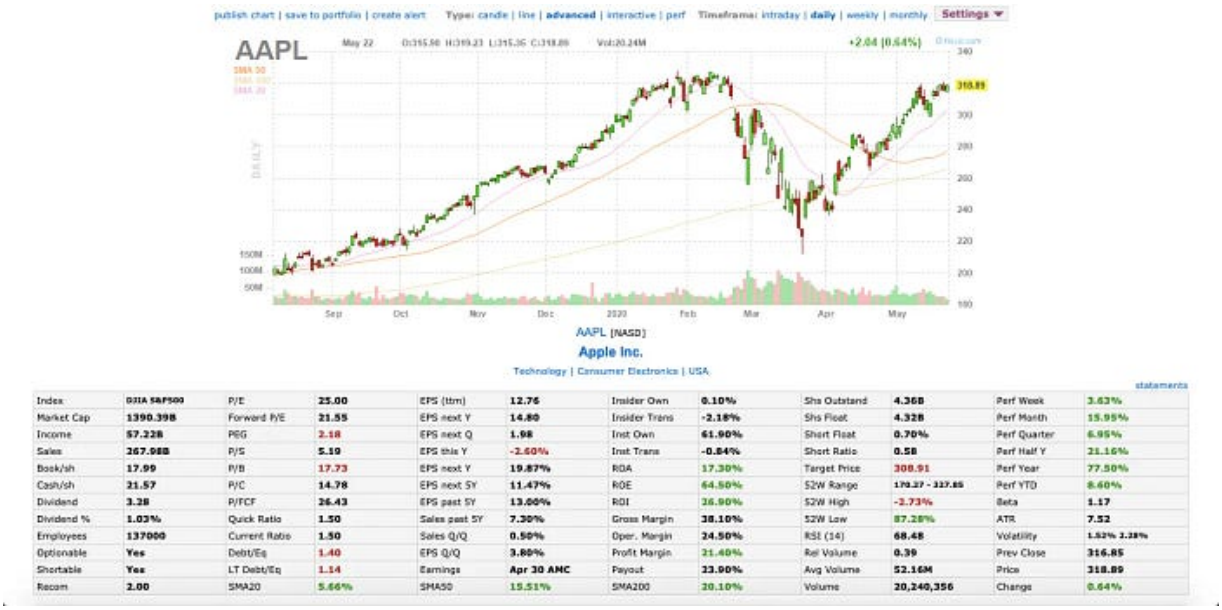
6. **Predicting Sentiment for New News Articles:** Once our model is trained and evaluated, we can utilize it to predict sentiment for new, unseen news articles. We apply the same preprocessing steps to these articles and then feed them into the trained model to obtain sentiment predictions.

7. **Visualizing Sentiment Results:** To gain better insights and interpret the sentiment analysis results, we can create visualizations. Python offers various visualization libraries such as Matplotlib or Plotly. We can generate visualizations like bar charts or word clouds to showcase sentiment distribution or highlight key sentiment-related terms in news articles

8. **Incorporating Sentiment Analysis in Trading Strategies:** Sentiment analysis can serve as a valuable component of trading strategies. By incorporating sentiment predictions into trading models, investors can make more informed decisions. For instance, positive sentiment may trigger a bullish stance, while negative sentiment may indicate a bearish outlook.

9. **Conclusion and Future Directions:** Analyzing stock price news sentiment using ML models in Python empowers investors to leverage the power of data-driven decision-making. With further advancements in ML algorithms and techniques, such as deep learning and transformers, we can expect even more accurate

their chances of success in the stock market.



Index	DJIA S&P500	P/E	25.00	EPS (ttm)	12.76	Insider Own	0.10%	Shs Outstand	4.36B	Perf Week	3.63%
Market Cap	1390.39B	Forward P/E	21.55	EPS next Y	14.80	Insider Trans	-2.18%	Shs Float	4.32B	Perf Month	15.95%
Income	57.22B	PEG	2.18	EPS next Q	1.98	Inst Own	61.90%	Short Float	0.70%	Perf Quarter	6.95%
Sales	267.98B	P/S	5.19	EPS this Y	-2.60%	Inst Trans	-0.84%	Short Ratio	0.58	Perf Half Y	21.16%
Book/sh	17.99	P/B	17.73	EPS next Y	19.87%	ROA	17.30%	Target Price	308.91	Perf Year	77.50%
Cash/sh	21.57	P/C	14.78	EPS next 5Y	11.47%	ROE	64.50%	52W Range	170.27 - 337.85	Perf YTD	8.60%
Dividend	3.28	P/FCF	26.43	EPS past 5Y	13.00%	ROI	26.90%	52W High	-2.73%	Beta	1.17
Dividend %	1.03%	Quick Ratio	1.50	Sales past 5Y	7.30%	Gross Margin	38.10%	52W Low	87.28%	ATR	7.52
Employees	137000	Current Ratio	1.50	Sales Q/Q	0.50%	Oper. Margin	24.50%	RSI (14)	68.48	Volatility	1.52% 3.28%
Optionable	Yes	Debt/Eq	1.40	EPS Q/Q	3.80%	Profit Margin	21.40%	Rel Volume	0.39	Prev Close	316.85
Shortable	Yes	LT Debt/Eq	1.14	Earnings	Apr 30 AMC	Payout	23.90%	Avg Volume	52.16M	Price	318.89
Recom	2.00	SMA20	5.66%	SMA50	15.51%	SMA200	20.10%	Volume	20,240,356	Change	0.64%

Fundamental Ratios and Interactive Charts for AAPL Stock

Finally, the data is ready to be manipulated and viewed in an appealing manner. For each ticker in the inputted list, a new DataFrame will be created that includes its headlines and their respective scores. Last, a final data frame will be created that includes each ticker's mean sentiment value over all the recent news parsed.

Copy

```
# Import libraries
import pandas as pd
from bs4 import BeautifulSoup
import matplotlib.pyplot as plt
from urllib.request import urlopen, Request
from nltk.sentiment.vader import SentimentIntensityAnalyzer

# Parameters
```

```
# Get Data
finwiz_url = 'https://finviz.com/quote.ashx?t='
news_tables = {}

for ticker in tickers:
    url = finwiz_url + ticker
    req = Request(url=url,headers={'user-agent': 'my-app/0.0.1'})
    resp = urlopen(req)
    html = BeautifulSoup(resp, features="lxml")
    news_table = html.find(id='news-table')
    news_tables[ticker] = news_table

try:
    for ticker in tickers:
        df = news_tables[ticker]
        df_tr = df.findAll('tr')

        print ('\n')
        print ('Recent News Headlines for {}: '.format(ticker))

        for i, table_row in enumerate(df_tr):
            a_text = table_row.a.text
            td_text = table_row.td.text
            td_text = td_text.strip()
            print(a_text,'(',td_text,')')
            if i == n-1:
                break
except KeyError:
    pass

# Iterate through the news
parsed_news = []
for file_name, news_table in news_tables.items():
    for x in news_table.findAll('tr'):
        text = x.a.get_text()
        date_scrape = x.td.text.split()

        if len(date_scrape) == 1:
            time = date_scrape[0]
```

```
            date = date_scrape[0]
            time = date_scrape[1]

        ticker = file_name.split('_')[0]

        parsed_news.append([ticker, date, time, text])

# Sentiment Analysis
analyzer = SentimentIntensityAnalyzer()

columns = ['Ticker', 'Date', 'Time', 'Headline']
news = pd.DataFrame(parsed_news, columns=columns)
scores = news['Headline'].apply(analyzer.polarity_scores).tolist()

df_scores = pd.DataFrame(scores)
news = news.join(df_scores, rsuffix='_right')


# View Data
news['Date'] = pd.to_datetime(news.Date).dt.date

unique_ticker = news['Ticker'].unique().tolist()
news_dict = {name: news.loc[news['Ticker'] == name] for name in unique_t

values = []
for ticker in tickers:
    dataframe = news_dict[ticker]
    dataframe = dataframe.set_index('Ticker')
    dataframe = dataframe.drop(columns = ['Headline'])
    print ('\n')
    print (dataframe.head())

    mean = round(dataframe['compound'].mean(), 2)
    values.append(mean)

df = pd.DataFrame(list(zip(tickers, values)), columns =['Ticker', 'Mean S
df = df.set_index('Ticker')
df = df.sort_values('Mean Sentiment', ascending=False)
print ('\n')
print (df)
```

#stock-analysis #news-sentiment #machine-learning #artificial-intelligence #python