Stock Price Prediction and Sentiment Analysis

Team: 221

By:
Samridh Anand Paatni 20BCE1550
Sujith Bysani 20MID0180
Saswata Basu 20MID0035
Dhanesh Kumar 20MID0199

Video Presentation Link: https://youtu.be/AB53udbFJqA

Contents

1 Introduction

1.1 Overview

Stock Price Prediction and Sentiment Analysis is an innovative tool that utilizes LSTM models to forecast stock prices and analyze market sentiment. By analyzing historical price data, LSTM models predict future stock prices, aiding investors in identifying investment opportunities. Additionally, sentiment analysis examines news headlines to determine the general sentiment towards a stock or company, providing investors with a holistic understanding of market sentiment.

However, limitations exist, including data constraints and the unpredictable nature of the stock market, which can impact the accuracy of predictions. Nonetheless, this tool offers numerous benefits, including informed decision-making, optimal trade timing, and portfolio optimization.

To enhance accuracy, future improvements may involve advanced LSTM models and incorporating user sentiment data and publicly available information.

In summary, Stock Price Prediction and Sentiment Analysis leverages LSTM models to predict stock prices and assess market sentiment. While limitations exist, the tool provides predictive insights and market sentiment analysis to empower investors in navigating the stock market and improving investment outcomes.

1.2 Purpose

The purpose of the stock prediction platform utilizing LSTM models and sentiment analysis on news headlines is to assist investors in making informed investment decisions.

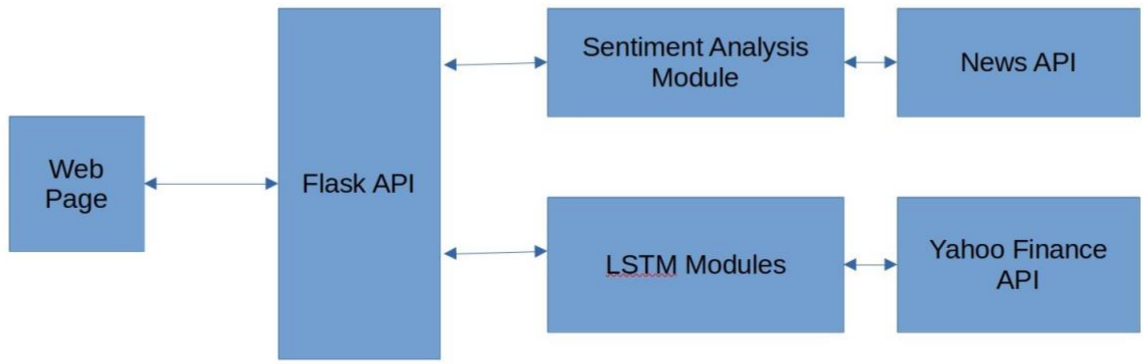
The platform aims to provide predictive insights and analysis that can help investors evaluate potential investment opportunities, manage risk, and optimize their portfolios. By leveraging LSTM models, the platform analyzes historical stock data to identify patterns and dependencies that can contribute to more accurate predictions of future stock prices.

Additionally, by incorporating sentiment analysis on news headlines, the platform captures market sentiment and investor perception, providing a more comprehensive view of a stock's potential performance.

The platform's purpose is to empower investors with timely and actionable information, enabling them to make informed decisions based on a combination of quantitative and qualitative factors. Ultimately, the platform's goal is to enhance investors' decision-making capabilities, potentially leading to improved investment outcomes in the stock market.

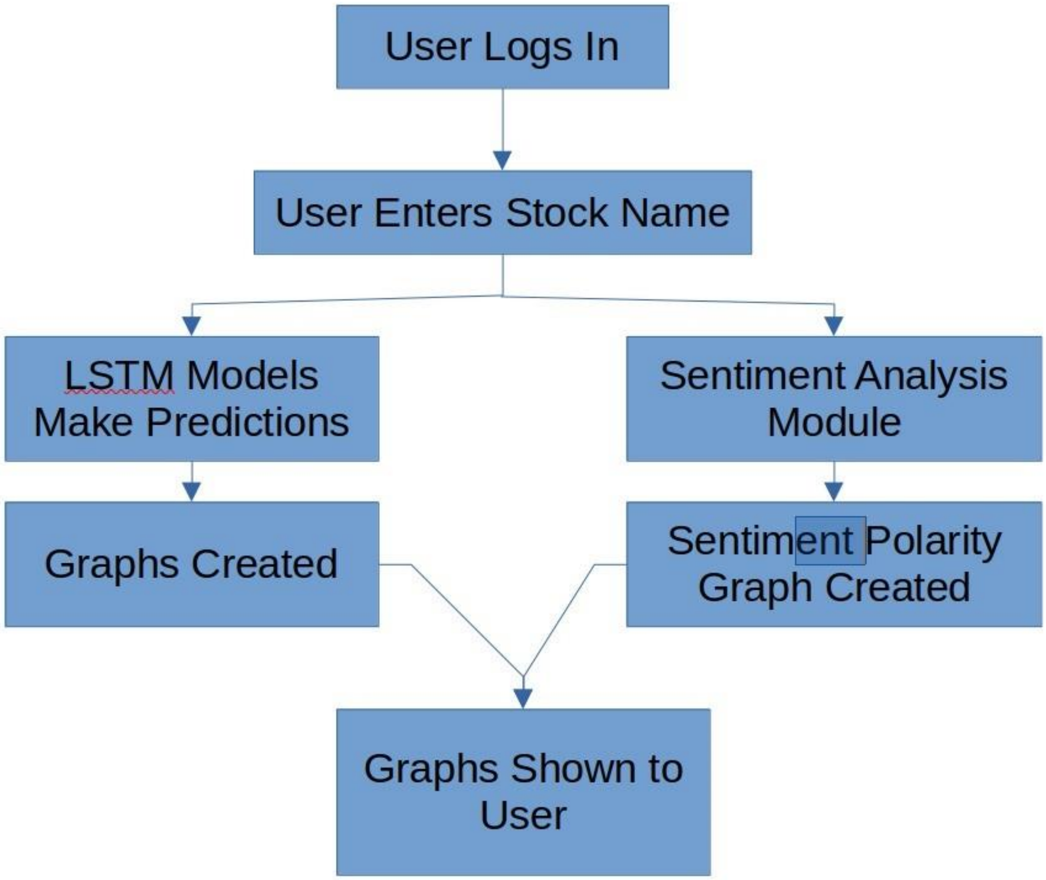
2 Theoretical Analysis

We have created two models, one which uses RNNs to predict the future price and one which uses sentiment analysis to know the public perception of stocks.



3 Flowchart

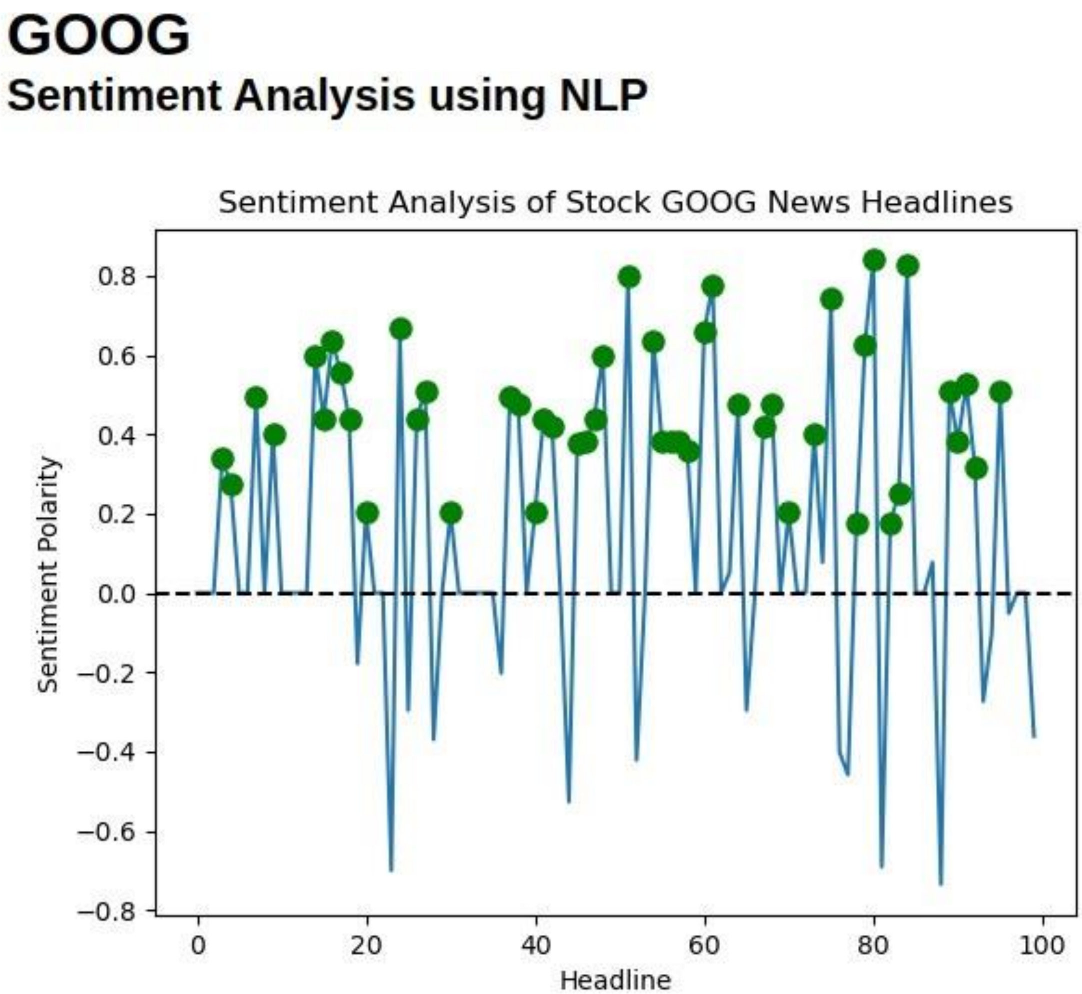
Below is a high-level flowchart of how the application works.



4 Result

4.1 Sentiment Analysis

The Sentiment Analysis module creates the following graph:

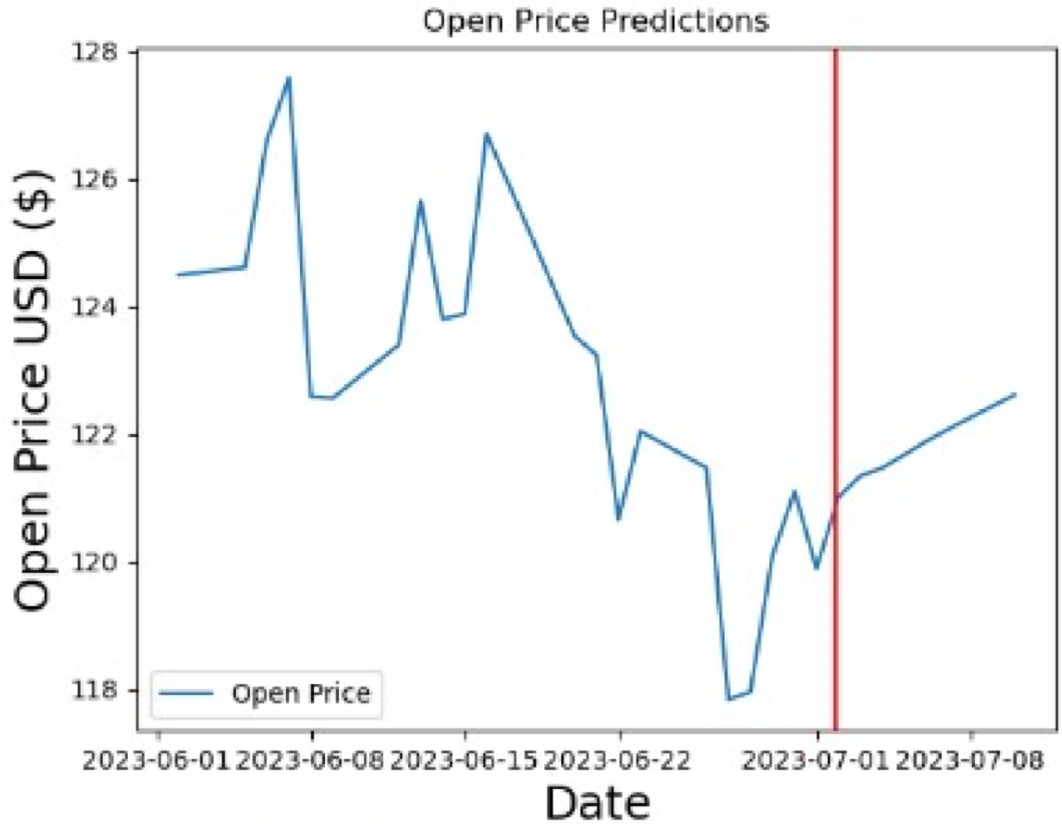


4.2 LSTM Predictions

The LSTM models generate the following graphs

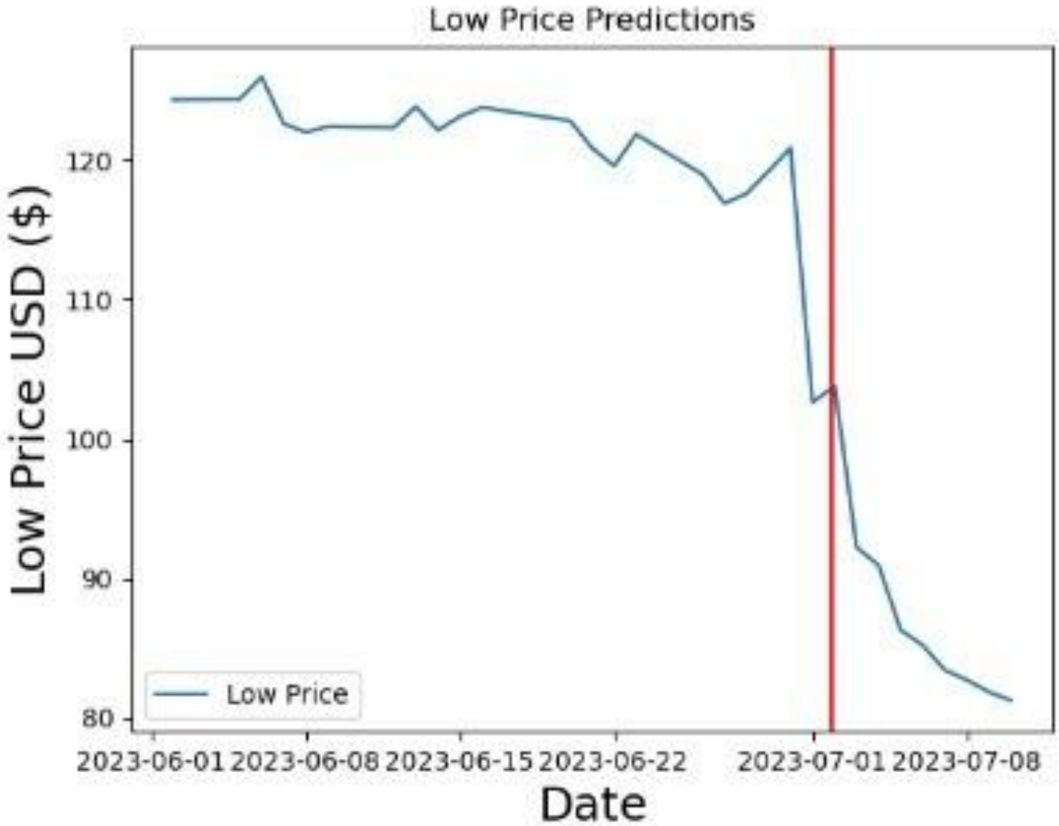
Predictions using LSTM Models

Open Price Predictions



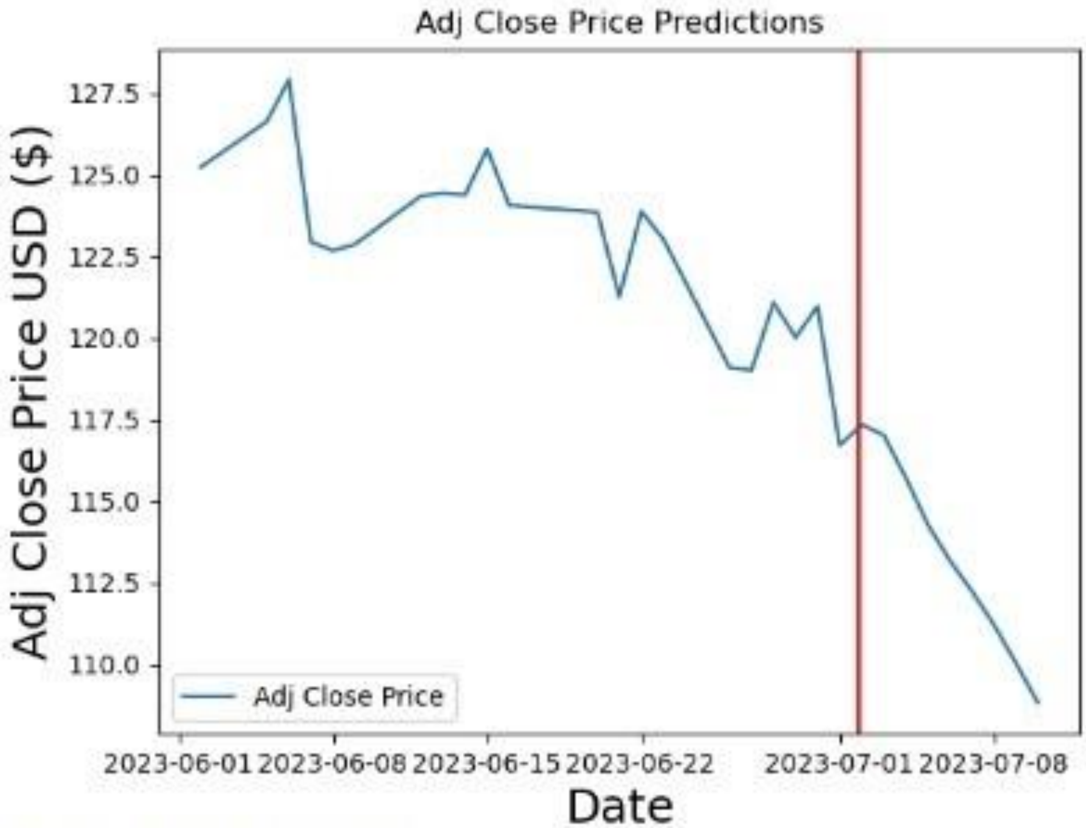
High Price Predictions

Low Price Predictions

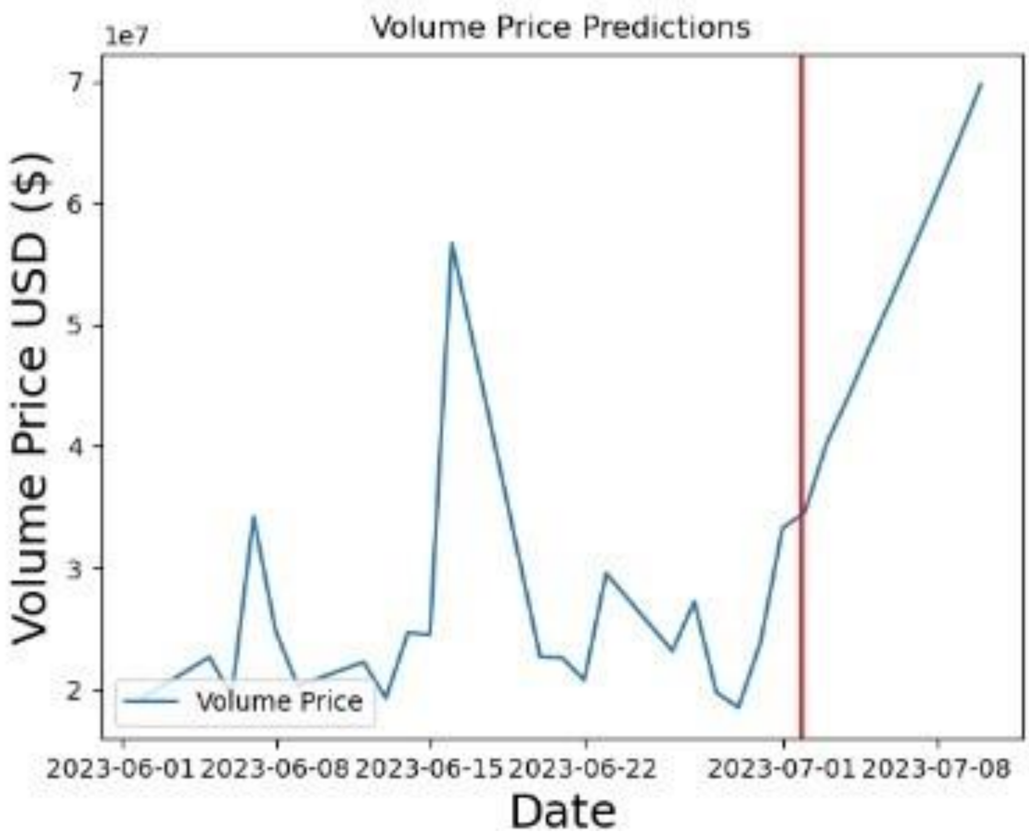


Close Price Predictions

Adj Close Price Predictions



Volume Price Predictions



5 Advantages and Disadvantages

Advantages of our stock prediction platform utilizing LSTM models and sentiment analysis on news headlines:

1. Enhanced Predictive Power: LSTM models, a type of recurrent neural network, are well-suited for capturing patterns and dependencies in sequential data, such as historical stock prices. By incorporating LSTM models, the platform can potentially improve the accuracy of stock price predictions compared to traditional models.

2. Incorporation of Sentiment Analysis: By analyzing news headlines and sentiment, the platform can capture market sentiment and investor perception about a particular stock. Sentiment analysis can provide valuable insights into how news events and public sentiment can influence stock prices, allowing investors to make more informed decisions.

3. Holistic Information: The combination of historical data analysis and sentiment analysis provides a more comprehensive view of the stock's potential future performance. Investors can benefit from a wider range of information, considering both quantitative factors (historical data) and qualitative factors (sentiment analysis).

4. Timely Decision Making: Sentiment analysis allows investors to react to breaking news or evolving market sentiment quickly. By incorporating real-time news analysis, the platform can help investors make timely decisions and potentially seize investment opportunities or manage risks more effectively.

Disadvantages and Limitations:

1. Data Limitations: The accuracy and reliability of predictions heavily rely on the quality and availability of historical data and news headlines. Inaccurate or incomplete data can affect the performance of the prediction model and lead to misleading results.

2. Market Complexity and External Factors: Stock prices are influenced by a multitude of factors, including economic indicators, geopolitical events, regulatory changes, and market dynamics. While sentiment analysis provides insights into news sentiment, it may not capture all external factors that can impact stock prices, making predictions susceptible to unforeseen events or changes in market conditions.

3. Overreliance on Historical Patterns: LSTM models are trained on historical data patterns. If the market behavior significantly deviates from historical patterns or experiences unprecedented events, the model's predictive accuracy may

decrease. The platform may struggle to adapt to novel situations, leading to inaccurate or delayed predictions.

4. Inherent Uncertainty and Risk: Stock market predictions, even with advanced techniques, inherently carry uncertainty and risk. No prediction model can guarantee accurate forecasts all the time. Investors should exercise caution and consider predictions as one of many tools for decision-making, rather than relying solely on them.

5. Sentiment Analysis Challenges: Sentiment analysis faces challenges like ambiguity, sarcasm, and context understanding. News headlines may be misleading or misinterpreted, leading to inaccurate sentiment analysis results. Additionally, sentiment analysis alone may not capture all relevant information about a stock, and additional research and analysis may still be required.

It's important to note that while LSTM models and sentiment analysis can enhance decision-making capabilities, investors should combine them with other fundamental and technical analysis tools, exercise critical thinking, and consider their own risk tolerance and investment goals when making investment decisions.

6 Applications

There exist several stock market analysis platforms. Our platform aims to predict the future prices of the stock.

A stock price prediction platform can be highly valuable for investors in several ways.

Firstly, it aids in making informed investment decisions by providing insights and forecasts about the future direction of stock prices. This allows investors to evaluate potential opportunities and adjust their portfolios accordingly.

Secondly, such platforms assist in managing risk by offering predictions and indicators of potential market downturns or fluctuations. By being aware of these risks, investors can adjust their strategies and implement risk mitigation measures.

Thirdly, stock price predictions help in determining the optimal timing for trades, enabling investors to buy or sell stocks based on expected price movements.

Additionally, these platforms aid in portfolio optimization by recommending which stocks to include or exclude, considering predicted price movements and other relevant factors.

Lastly, stock price prediction platforms provide investors with access to comprehensive data, historical trends, and technical indicators, facilitating research and analysis.

However, one must remember that these platforms aren't perfect predictors as they rely on historical data and cannot predict everything.

7 Conclusion

In conclusion, the development of this platform which uses LSTM models and incorporates sentiment analysis on news headlines can advance the field of investment decision-making.

This platform offers a range of advantages, including enhanced predictive power, the incorporation of market sentiment, access to holistic information, and the ability to make timely investment decisions.

By leveraging LSTM models, this platform can capture complex patterns in historical stock data and improve the accuracy of price predictions.

Additionally, sentiment analysis on news headlines allows investors to gauge market sentiment and consider qualitative factors that can influence stock prices.

However, it is important to acknowledge the limitations of these platforms, such as data limitations, market complexity, the risk of overreliance on historical patterns, inherent uncertainty, and challenges in sentiment analysis. Investors should exercise caution, use these platforms as one of many tools in their decision-making process, and consider their own risk tolerance and investment goals.

8 Future Scope

Following are the possible future scope for this project:

1. The LSTM model can be a much deeper model with more layers.
2. The sentiment analysis part can look at forums to guage how everyday users feel about the stock
3. We can look at government filings and other public data to figure out how a company and thus, its stock price, is doing

9 Appendix: Code for Solution

9.1 Sentiment Analysis

```
import requests from nltk.sentiment.vader import SentimentIntensityAnalyzer import
matplotlib.pyplot as plt import numpy as np import nltk nltk.download('vader_lexicon')



api_key = "4f8016edb36a4e7ca300ecc95e5ee10a"

#Companys used company_tickers = ['AMD', 'AMZN', 'FB', 'GOOG']

# Initializing sentiment_results dictionary
```

```
sentiment_results = {}



buy_threshold = 0.1
# Define a list of colors for the graphs colors = ['blue', 'red', 'green',
'orange']


# Fetch news headlines and perform sentiment analysis for each company ticker
for i, ticker in enumerate(company_tickers):
     # Define the API endpoint URL url = f"https://newsapi.org/v2/everything?q={ticker}&language=en&
    apiKey={api_key}"

     # Make a GET request to the API response =
     requests.get(url)

     # Check if the request was successful if
     response.status_code == 200: # Parse the response
     JSON data = response.json()

          # Get the news articles articles = data.get('articles', [])
            # Get the news articles articles = data.get('articles',
          [])

          # Perform sentiment analysis on each article headline sentiments = [] sid =
          SentimentIntensityAnalyzer() for article in articles:
               headline = article.get('title', '') scores =
               sid.polarity_scores(headline) sentiment = scores['compound']
               sentiments.append(sentiment)

          # Store the sentiments in the sentiment_results dictionary sentiment_results[ticker] =
          sentiments

          # Generate a line plot for the sentiments with a unique color plt.figure() plt.plot(sentiments,
          color=colors[i]) plt.axhline(0, color='black', linestyle='--') plt.title(f'Sentiment Analysis of Stock
          {ticker} News
    Headlines') plt.xlabel('Headline') plt.ylabel('Sentiment
          Polarity')

          # Add markers to indicate buying decision
```

9.2 LSTM Training

```
import numpy as np import pandas
as pd import pickle

import matplotlib.pyplot as plt import seaborn as sns

import yfinance as yf from pandas_datareader import data as
pdr yf.pdr_override() from datetime import datetime

from sklearn.preprocessing import MinMaxScaler from tensorflow
import keras from keras.models import Sequential from keras.layers
import Dense, LSTM


n_years = 5


end_time = datetime.now() start_time =
datetime( end_time.year - n_years,
end_time.month, end_time.day
```

```
)


def get_train_data(stock_name, column, start, end, scaler, fit=False)
        :
          print(f'downloading {stock_name}') df =
          pdr.get_data_yahoo( stock_name, start, end
          )
          data = df.filter([column]) dataset = data.values

          scaled_data = scaler.fit_transform(dataset) if fit else scaler. transform(dataset)

          X = [] y =
          []

          for i in range(60, len(scaled_data)): X.append(scaled_data[i-60:i, 0])
                    y.append(scaled_data[i, 0]) return X, y


def get_aggregated_train_data(train_stocks, column, start_time, end_time): scaler =
        MinMaxScaler(feature_range=(0,1))

          X_train = [] y_train = []

          for i, stock in enumerate(train_stocks):
                    try:
                                                                      X, y = get_train_data(stock, column, start_time, end_time
        , scaler, fit=(i==0)) X_train += X y_train +=
                    y except:
                              continue

          X_train = np.array(X_train) y_train =
          np.array(y_train)

          X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape
        [1], 1)) with open(f'saved/scaler_{column}.pickle', 'wb') as pkl:
                    pickle.dump(scaler, pkl)
```

2

```
    return X_train, y_train


def compile_save_model(column, X_train, y_train):
    model = Sequential([
        LSTM(128, return_sequences=True, input_shape= (X_train.shape
   [1], 1)),
        LSTM(64, return_sequences=True),
        LSTM(64, return_sequences=False),
        Dense(25),
        Dense(25),
        Dense(1)
    ]) model.compile(optimizer='adam', loss='mean_squared_error') model.fit(X_train,

    y_train, batch_size=1, epochs=1) model.save(f'saved/model_{column}.h5',

    save_format='h5')

 [markdown]
# ## Training All Models


train_stocks = [
    'AAPL',
    'BIVI',
    'GOOG',
    'NVMI',
    'CHRD',
    'ASMB',
    'GOGO',
    'CAFG',
    'BVXV',
    'AIBBU',
    'LGVCW',
    'AMZN'
]
columns = [ 'Open',
    'High',
    'Low',
    'Close',
    'Adj Close',
    'Volume'
]


for col in columns:
    print(f'doing {col}')
    X_train, y_train = get_aggregated_train_data(train_stocks, col,
```

```
start_time, end_time) compile_save_model(col, X_train, y_train)
  print(f'{col} done')
```

9.3 LSTM Predictions

```
import numpy as np import
pandas as pd import base64 from
io import BytesIO

from matplotlib.figure import Figure import
matplotlib.pyplot as plt import seaborn as sns

import yfinance as yf from pandas_datareader import data as
pdr yf.pdr_override() from datetime import datetime,
timedelta

from sklearn.preprocessing import MinMaxScaler from tensorflow import
keras

def get_data(stock_name, column, start_time, end_time):
    df = pdr.get_data_yahoo( stock_name,
        start=start_time, end=end_time
    )
    data = df.filter([column]) return data

def preprocess_data(data, scaler):
    scaled_data = scaler.transform(data.values)
    X = []
    for i in range(60, len(scaled_data)):
        X.append(scaled_data[i-60:i, 0])
    X = np.array(X)
    X = np.reshape(X, (X.shape[0], X.shape[1], 1)) return X

def predict(X, model, scaler):
    predictions = model.predict(X)
    predictions = scaler.inverse_transform(predictions) return predictions

def get_20_days_n_preds(stock_name, column, scaler, n):
    end_time = datetime.now()
```

```
    start_time = datetime(end_time.year - 1, end_time.month, end_time
  .day)

    model = keras.models.load_model(f'models/saved/model_{column}.h5
  ') raw_data = get_data(stock_name, column, start_time, end_time)

    for i in range(n): # predict the next 10 days new_date = raw_data.index[-1].to_pydatetime() +
        timedelta(
  days=1)
        X = preprocess_data(raw_data[-61:], scaler) pred = predict(X,
    model, scaler) raw_data.at[new_date, column] = pred[0][0] return
    raw_data[-30:]

def get_graph(stock_name, column, scaler, n_days=10):
    stock_data = get_20_days_n_preds(stock_name, column, scaler,
  n_days)

    fig = Figure() ax = fig.subplots() ax.set_title(f'{column} Price Predictions')
    ax.set_xlabel('Date', fontsize=18) ax.set_ylabel(f'{column} Price USD ($)',
    fontsize=18) ax.plot(stock_data) ax.axvline(x = datetime.today(), color='r')
    ax.legend([f'{column} Price'], loc='lower left')

    buf = BytesIO() fig.savefig(buf, format="png") data =
    base64.b64encode(buf.getbuffer()).decode("ascii") return data
```