Money Talks

André Flores
up201907001@fe.up.pt

Diogo Faria
up201907014@fe.up.pt

January 6, 2024

1 Introduction & Context

Having a savings account isn't enough. Saving money is important, but it's only part of the story. Smart savers start by building sufficient emergency savings. But after building three to six months of easy-to-access savings, investing in the financial markets offers many potential advantages [5].

Investing is an effective way to put your money to work and generate passive income. Smart investing may allow your money to outpace inflation and increase values [5].

There is a risk-return trade-off in investing. Investing in stocks has the potential to provide higher returns but such investments are more volatile than others [5].

2 Problem Description

Smart investing requires financial literacy beyond most people's. What do the various indicators of financial performance mean? Do they even impact a company's stock price? How does one make sound financial investments?

Currently, the soundest way to invest is to hire a fiduciary financial advisor or to invest in ETFs or mutual funds, which are diverse pools of securities. Both solutions diminish potential investment returns, as they usually take a percentage of the money they earn you. Another problem is that they don't increase people's financial literacy, functioning like black boxes.

There is a need to make sound financial advice available to more people at a cheaper price while increasing their financial literacy so they don't have to blindly trust counsel.

3 Requirements

In terms of functional requirements, we came up with the following items:

- A user can search for a NASDAQ public company and go to a new page with a prediction about its stock price;
- A user can see past predictions ranked on the home page or a specific page;
- A user can see a company's prediction explanation in natural language.

Regarding non-functional requirements, we had:

- The prediction generation happens in a reasonable time;
- The website's UI is intuitive;
- The information is presented appealingly.

4 Final Solution

In this section, we'll provide details about our final implementation.

4.1 Architecture

We developed our frontend using Vue with Typescript, which connects to our Python backend via FastAPI. In order to make a prediction, we searched a company using the FMP API [1] to obtain its stock ticker and identification number, then the Macrotrends website [2] to obtain historical stock data, and the EDGAR API [4] for historical financial data. Lastly, after running the data through our created prediction model, we use the GPT-3.5 Turbo API [3] to generate an explanation using natural language. All of this can be seen in fig. 1.

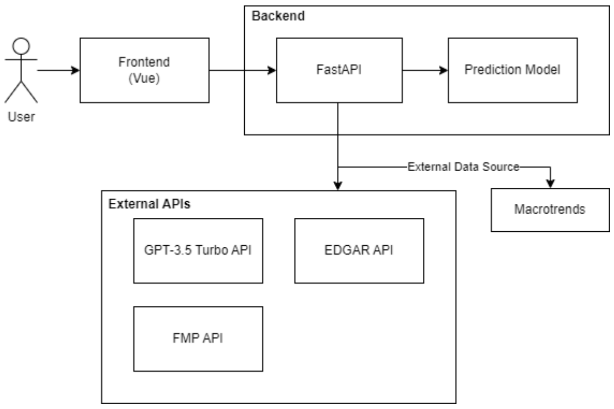


Figure 1: *Architecture*

4.2 Design

In order to design our website, we started by creating mock-ups for two pages, as shown in fig. 2, which, after recommendations from a focus group filled with possible stakeholders and further revision from us ended up as shown in fig. 3.

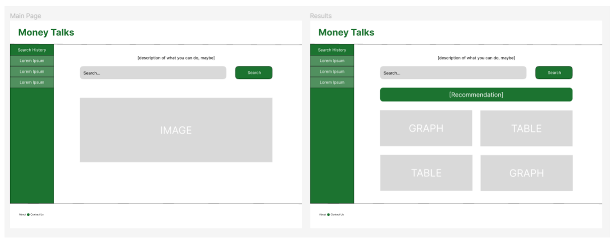


Figure 2: *Initial Mock-ups*

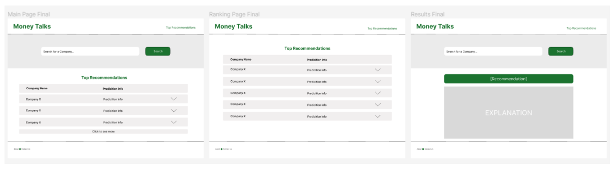


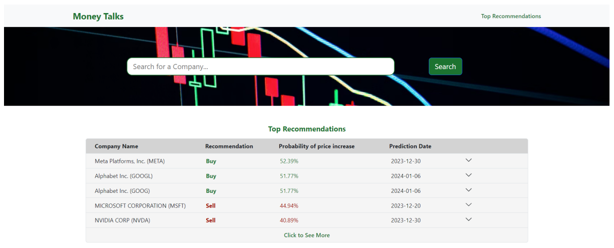
Figure 3: *Final Mock-ups*

4.3 Functionalities

In terms of functionalities, we have a main one, which is the user's ability to search for a specific company and obtain a recommendation on whether to buy or not into the company, and another one which is being able to view previously done searches, ordered by their results, in either the home page, in a limited number, or in the recommendation page which shows them all.

4.4 Implementation

Our website, taking inspiration from the previously shown mock-ups, is comprised of three pages shown in fig. 4, fig. 5, and fig. 6.

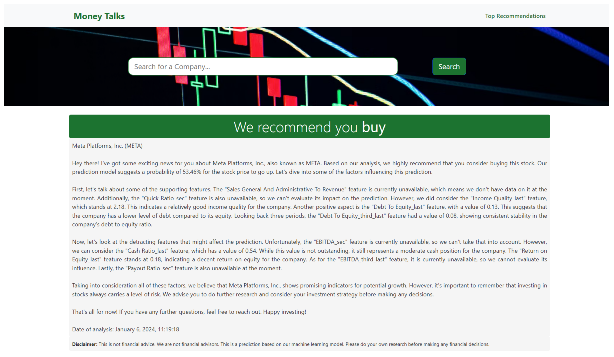


Top Recommendations

Company Name	Recommendation	Probability of price increase	Prediction Date
Meta Platforms, Inc. (META)	Buy	52.39%	2023-12-30
Alphabet Inc. (GOOGL)	Buy	51.77%	2024-01-06
Alphabet Inc. (GOOG)	Buy	51.77%	2024-01-06
MICROSOFT CORPORATION (MSFT)	Sell	44.94%	2023-12-20
NVIDIA CORP (NVDA)	Sell	40.89%	2023-12-30

Click to See More

Figure 4: *Home Page*



We recommend you **buy**

Meta Platforms, Inc. (META)

Hey there! I've got some exciting news for you about Meta Platforms, Inc., also known as META. Based on our analysis, we highly recommend that you consider buying this stock. Our prediction model suggests a probability of 53.46% for the stock price to go up. Let's dive into some of the factors influencing this prediction.

First, let's talk about some of the supporting features. The "Sales General And Administrative To Revenue" feature is currently unavailable, which means we don't have data on it at the moment. Additionally, the "Quick Ratio_sec" feature is also unavailable, so we can't evaluate its impact on the prediction. However, we did consider the "Income Quality_last" feature, which stands at 2.18. This indicates a relatively good income quality for the company. Another positive aspect is the "Debt To Equity_last" feature, with a value of 0.13. This suggests that the company has a lower level of debt compared to its equity. Looking back three periods, the "Debt To Equity_third_last" feature had a value of 0.08, showing consistent stability in the company's debt to equity ratio.

Now, let's look at the detracting features that might affect the prediction. Unfortunately, the "EBITDA_sec" feature is currently unavailable, so we can't take that into account. However, we can consider the "Cash Ratio_last" feature, which has a value of 0.54. While this value is not outstanding, it still represents a moderate cash position for the company. The "Return on Equity_last" feature stands at 0.18, indicating a decent return on equity for the company. As for the "EBITDA_third_last" feature, it is currently unavailable, so we cannot evaluate its influence. Lastly, the "Payout Ratio_sec" feature is also unavailable at the moment.

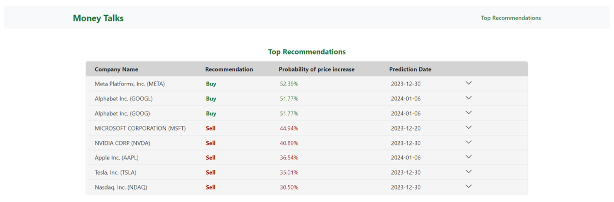
Taking into consideration all of these factors, we believe that Meta Platforms, Inc., shows promising indicators for potential growth. However, it's important to remember that investing in stocks always carries a level of risk. We advise you to do further research and consider your investment strategy before making any decisions.

That's all for now! If you have any further questions, feel free to reach out. Happy investing!

Date of analysis: January 6, 2024, 11:19:18

Disclaimer: This is not financial advice. We are not financial advisors. This is a prediction based on our machine learning model. Please do your own research before making any financial decisions.

Figure 5: *Prediction Page*



Top Recommendations

Company Name	Recommendation	Probability of price increase	Prediction Date
Meta Platforms, Inc. (META)	Buy	52.39%	2023-12-30
Alphabet Inc. (GOOGL)	Buy	51.77%	2024-01-06
Alphabet Inc. (GOOG)	Buy	51.77%	2024-01-06
MICROSOFT CORPORATION (MSFT)	Sell	44.94%	2023-12-20
NVIDIA CORP (NVDA)	Sell	40.89%	2023-12-30
Apple Inc. (AAPL)	Sell	36.54%	2024-01-06
Tesla, Inc. (TSLA)	Sell	35.01%	2023-12-30
Nasdaq, Inc. (NDAQ)	Sell	30.50%	2023-12-30

Figure 6: *Recommendations Page*

In the home and prediction pages, we are able to search for a company, which is sent to our backend through FastAPI. We then search the company's name with FMP API so that we can obtain the company's ticker, i.e., "Apple" search would give us the "AAPL" ticker. This ticker can then be used in both Macrotrends and the EDGAR API to obtain historical stock data and the financial data a company must report at certain intervals, respectively.

We then process the obtained data in order to be usable by our classification model, from which we can extract not only the actual prediction ("buy" or "sell") but also the probability of it being "buy" of the model and which features of the model contributed the most towards the result, so we can explain why this prediction was made. Lastly, we send all this information using GPT-3.5 Turbo API in order to obtain an explanation in natural language that can be presented to the user and is easily understandable to those with little to no knowledge about this matter and cache the full prediction to be able to be shown at a later date.

In order to develop and train our classification model, we went through several steps:

1. We obtained **training data** from 600 different companies using the previously mentioned means, which were all then processed to calculate 48 different metrics, such as EBIT, Quick Ratio, and Return on Equity, among many others. Besides this, we also saved the mean of all daily stock price differences from the 90 days previous to the time the supposed prediction was made. To determine whether our model should predict "buy" or "sell", we calculated the median of the stock in the 90 days after the prediction was to be made. If the difference between the medians of 90 days after and 90 days before the prediction was positive the model should return "buy", else it should return "sell".

2. The data obtained from the previous step was then randomly split into train and test data at a 75% and 25% distribution, respectively. At every step, the model was evaluated against the test data.

3. Then, we created our **base model** using the Python XGBoost library, which contains an efficient implementation of gradient-boosted trees algorithm, though we tested with a few more that ended up performing worse.

4. To improve our base model, we started by **scaling the data**, followed by **feature selection**.

5. Then, we did **hyperparameter tuning** to try and understand the best parameters to use for the model.

6. Using the best parameters and features, we create a new model with the scaled data, which we save for later use.

In the top recommendations page and on the table on the home page, we show previous recommendations, including when they were given to know if the information is too old, in order for a user to check their best options. If this was to continue in development, instead of locally caching the recommendations, we would've created a database that would contain every single user's search, so that we could, at all times, present the most updated recommendations.

5 Evaluation of the result

As mentioned before, we worked to improve our classification model, and, in the end, we achieved the metrics shown in table 1 and an accuracy of 0.72.

	Precision	Recall	F1-Score	Support
Sell	0.76	0.82	0.79	988
Buy	0.65	0.56	0.60	580

Table 1: *Classification model's metrics*

We believe our model performs well, especially when it comes to predicting a "sell" decision, with high values in precision, recall, and f1-score. Though in the "buy" prediction, our values aren't as high, they're still reasonably good with the precision and f1-score being above 0.60. Overall, with an accuracy of 0.72, our predictions will be right pretty much three-quarters of the time, which we believe is a good achievement.

Since our model looks at three-month scenarios when performing the analysis of buying or selling, we are unable to confirm whether our predictions are correct or not since we don't have a prediction from three months ago, our older prediction being from only two weeks ago.

6 Tests

We provide acceptance tests in appendix A. These tests cover the features available to the user.

Listing 1 covers the search feature. It encompasses both search for a company by its name or stock ticker. There is also a case for searching for a company not in our data sources.

Listing 2 covers the ranking features. It encompasses the various ways a user can access and use the ranking feature.

7 Discussion

We may have bit off more than we can chew.

For our project to function, we need various financial data sources, which are not easy to find for free. Both FMP and OpenAI provide useful and easy-to-use APIs. However, EDGAR's financial data only includes raw financial values, we had to calculate key financial metrics ourselves. This presented a challenge as EDGAR often does not provide all the required information for a given metric at the same point in time, which leaves us with a lot of null values to handle. Macrotrends does not provide an API. It only makes stock price history available as a CSV, which makes us have to download and parse a company's entire stock even if we only need a part of it. Using OpenAI's LLM to generate accurate text based on our data required a good amount of prompt engineering.

The main challenge with our project is getting accurate predictions of a company's stock price. A company's key financial metrics can only be calculated at most for every quarter, as these are the most frequent filings a company has to make to SEC. Some data is only updated annually. This is at conflict with the need to predict stock prices based on current information.

There are also other elements that influence stock prices that our model does not consider. A model that involved sentiment analysis of news and social media posts from influencers about companies could provide more accurate results. An even more advanced model should include context-specific knowledge of the companies so it could judge news based on content, like predicting if a specific leadership change will lead to higher stock prices.

Lastly, our model only returns binary results: "buy" or "sell". There could be a third category, "hold", for when a stock price will remain relatively stable or when a company pays dividends to shareholders.

References

[1] *Financial Modeling Prep - Financial Data API*. Financial Modeling Prep. URL: https://financialmodelingprep.com/ (visited on 01/06/2024).

[2] *Macrotrends | The Long Term Perspective on Markets*. URL: https://www.macrotrends.net (visited on 01/06/2024).

[3] *OpenAI*. URL: https://openai.com/ (visited on 01/06/2024).

[4] *SEC.gov | EDGAR—Search and Access*. URL: https://www.sec.gov/edgar/search-and-access (visited on 01/06/2024).

[5] *Why Investing is Important | Wells Fargo Advisors*. URL: https://www.wellsfargo.com/goals-investing/why-invest/ (visited on 01/06/2024).

A Appendix - Acceptance Tests

A.1 Search Feature

```gherkin
Feature: Search for a company

    Search feature should allow users to search for a company by name or stock symbol.

    Scenario: Search for a company that is in the database by name
        Given I am on the home page
        When I search for "Alphabet"
        Then I should see a loading spinner
        Then I should see a card with the prediction for "Alphabet"
        And The card should say either buy or sell
        And The card should have text
        * The text should include the company name, "Alphabet"
        * The text should include the company stock symbol, "GOOG"
        * The text should include the prediction, "buy" or "sell", in natural language
        * The text should include the probability of the stock price going up or down
        * The text should include the financial information for the company that supports the prediction
        * The text should include the financial information for the company that goes against the prediction
        * The text should include a disclaimer so we don't get sued

    Scenario: Search for a company that is in the database by stock symbol
        Given I am on the home page
        When I search for "GOOG"
        Then I should see a loading spinner
        Then I should see a card with the prediction for "Alphabet"
        And The card should say either buy or sell
        And The card should have text
        * The text should include the company name, "Alphabet"
        * The text should include the company stock symbol, "GOOG"
        * The text should include the prediction, "buy" or "sell", in natural language
        * The text should include the probability of the stock price going up or down
        * The text should include the financial information for the company that supports the prediction
        * The text should include the financial information for the company that goes against the prediction
        * The text should include a disclaimer so we don't get sued

    Scenario: Search for a company that is not in the database
        Given I am on the home page
        When I search for "ASML"
        Then I should see a loading spinner
        Then I should see a card informing me that data for "ASML" could not be found
```

Listing 1: *Acceptance test scenarios for search feature*

A.2 Ranking Feature

```
Feature: Company ranking

    Table ranking all companies predictions have been made for.

    Scenario: See rankings table from home page
        Given I am on the home page
        Then I should see the "Top Recomendations" header
        And I should see a table with the following columns:
            | Company Name | Recomendation | Probability of price increase | Prediction Date |
        * I should see the five companies I have searched for that have the highest probability of
        ↪   stock price increase in the table
        * I should see the companies in order of probability of price increase (descending)

    Scenario: Go to rankings page from home page
        Given I am on the home page
        And I have searched for companies that returned results, for example:
            | Company Name |
            | Apple        |
            | Google       |
            | Microsoft    |
            | Meta         |
        When I click the "Top Recomendations" link (top right) or the "Click to see more" link (bottom
        ↪   of the table)
        Then I should be on the rankings page
        And I should see the "Top Recomendations" header
        * I should see a table with the following columns:
            | Company Name | Recomendation | Probability of price increase | Prediction Date |
        * I should see all the companies I have searched for in the table
        * I should see the companies in order of probability of price increase (descending)

    Scenario: Get prediction explanation from company rankings table
        Given I am on the rankings page or the home page
        When I click the expander on a company in the table
        Then I should see the explanation for the prediction that was generated when that prediction
        ↪   was made
```

Listing 2: *Acceptance test scenarios for ranking feature*