Portfolio Management & Construction Tool

By Andrei G. Sipos, Hao Zheng, Nicholas D. Zambrotta, Roxy Zhang

1.0 Abstract

Build a **Portfolio Construction & Management** Tool using sentiment analysis and machine learning models. Given a sample $1 million dollars to allocate, our tool selected the 10 most likely stocks to outperform and forecasted a portfolio which reached a return of 19.89% or $198,921.46 upon successful optimization.

2.0 Executive Summary

Asset Managers have always charged a steep premium for building a portfolio of stocks deemed to outperform the markets and actively managing it in order to generate alpha. Furthermore, the general lack of liquidity and transparency within the hedge fund world made it difficult to exit positions as well as mimic an actively managed portfolio. Moreover, recent advances in natural language processing (NLP), machine learning as well as an increased focus on data driven decisions can be leveraged to automate the portfolio construction, active management, forecasting and optimization of the portfolio thus helping asset managers as well as investors to better understand the market sentiment, identify potential investment opportunities, manage downside risk, and monitor the impact of rapidly changing events within financial markets. As such, as part of this project, we will build a one stop shop **Portfolio Construction & Management** Tool that will: ***i.) Identify and Construct*** *a portfolio of the 10 most popular stocks based on sentiment,* ***ii.) Build a classifier*** *that will identify whether a new comment/post online represents a "Buy", "Strong Buy", "Sell", "Strong Sell" or "Hold" for each specific stock in order to actively manage the portfolio,* ***iii.) Predict the price*** *of each stock in the portfolio leveraging machine learning techniques and* ***iv.) Optimize*** *the weights of the portfolio in order to obtain a portfolio of stocks that maximized return while minimizing risk in order to deliver increased value for investors and asset managers alike.* Finally, we will compute all of the above steps for a sample $1,000,000 portfolio and observe the potential to outperform the market through our **Portfolio Construction & Management** Tool.

3.0 Data

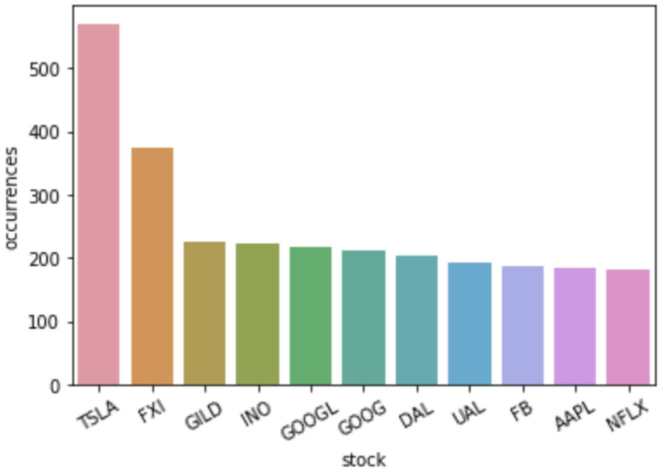
The data for sentimental analysis and forecasting investment labels is from Kaggle, containing Daily Financial News for 6000+ Stocks with 843,062 unique article headlines and 6,193 unique stocks with a time span of 12 years (2009-2020).

The data for future stock price prediction is extracted from 2012-2022 using Yahoo API, which allows for analysis of stock performance over time (e.g., open and adjusted close price).

4.0 Methodology

4.1 Sampling

Before processing the data, we examined the frequency of stock mentions across the raw dataset (N = 843,062). In the visualization below, the top 10 most frequently mentioned stocks are plotted in the bar plot, with TSLA (Tesla Motors), FXI (iShares China ETF), and GILD (Gilead Sciences) as the top three stocks by mentions.



Plot 1. Frequency of Stock Mentions

Using the raw data, we selected analysts' articles from the year 2020 in an effort to i.) use the most recent and updated news and ii.) decrease our sample size for processing time and memory limitations. By selecting data only from 2020, the raw data sample size became N = 106,964 observations. Next, we took a random sample of 50,000 observations from the 2020 sample of 106,964. We used this random sample of analysts' stock articles to predict stock performance using sentiment analysis, and Random Forest and XGBoost models, as discussed in detail below.

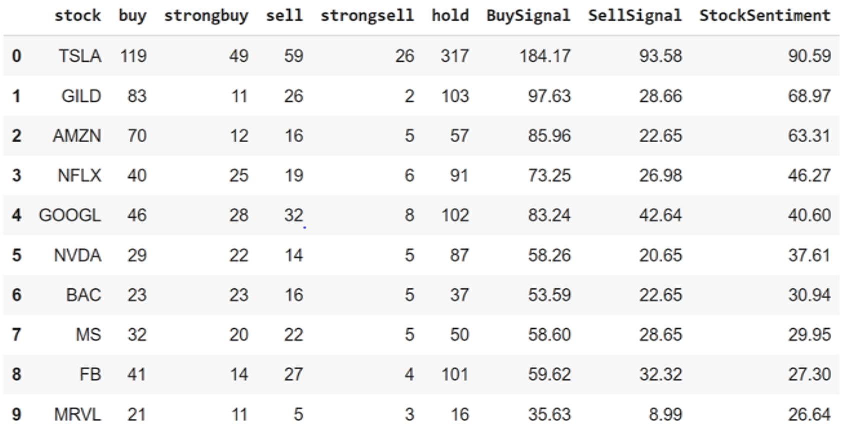
4.2 Portfolio Construction through Sentiment Analysis

In order to construct a portfolio of the top 10 stocks most likely to outperform their peers within their industry, we conducted sentiment analysis. More specifically, we investigated how positive or negative comments are on each stock, translated the numbers into buy and sell labels and finally used a heuristics algorithm to score the likelihood of each stock to outperform in the future.

The first step was to perform several data cleaning techniques on the selected text and remove all stop words since they do not provide any value as well as stem all words such that we only train our models using the roots of the words and avoid long training times as well as overfitting. Once we prepped the data, we applied Vader Sentiment in order to get a score from -1 to 1 that would represent the positivity/negativity of that respective comment towards that respective stock.

Second, we use a heuristics algorithm to determine whether each comment associated with each stock represents a buy or a sell, which we later use in computing the stock sentiment score for each stock. Any comment with a Vader Sentiment score above 0.5 will be labeled as a "strong buy", comments with a score between 0 and 0.5 will be labeled "buy", scores below -0.5 will be labeled as a "strong sell", any comment with a score between 0 and -0.5 will be labeled "sell", while any comment with a Vader Sentiment of 0 will be labeled as "hold" since this text is neither positive or negative. Furthermore, we aggregate the labels at the stock level in order to obtain an impression on the capacity of each stock to severely outperform or underperform its peers. In order to do so, we assign a value of 1.33 to each comment labeled as a "strong buy", a value of 1 to each comment labeled as a "buy", a value of -1.33 to each comment labeled as a "strong sell", a value of -1 to each comment labeled as a "sell" and a finally a value of 0 to each comment labeled as a "hold". Finally, we compute a BuySignal, SellSignal and composite Stock Sentiment for each stock by aggregating the values assigned earlier for each stock. Aggregating all buys will result in the BuySignal, aggregating all sells will result in the SellSignal, while combining the 2 indices will result in our Stock Sentiment score for each equity.

Lastly, we construct our portfolio by selecting the top 10 stocks with the highest Stock Sentiment as shown below.



	stock	buy	strongbuy	sell	strongsell	hold	BuySignal	SellSignal	StockSentiment
0	TSLA	119	49	59	26	317	184.17	93.58	90.59
1	GILD	83	11	26	2	103	97.63	28.66	68.97
2	AMZN	70	12	16	5	57	85.96	22.65	63.31
3	NFLX	40	25	19	6	91	73.25	26.98	46.27
4	GOOGL	46	28	32	8	102	83.24	42.64	40.60
5	NVDA	29	22	14	5	87	58.26	20.65	37.61
6	BAC	23	23	16	5	37	53.59	22.65	30.94
7	MS	32	20	22	5	50	58.60	28.65	29.95
8	FB	41	14	27	4	101	59.62	32.32	27.30
9	MRVL	21	11	5	3	16	35.63	8.99	26.64

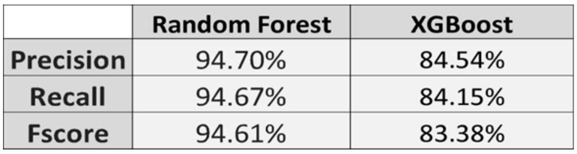
Form 1. Top 10 Stocks with the Highest Stock Sentiment

4.3 Forecasting Buy/Sell Labels

Next, we would like to build a formal model that helps us predict the “Strong Buy”, “Buy”, “Hold”, “Sell”, “Strong Sell” label for a new post/text and ensures robustness. This will help the portfolio manager to incorporate new data into the portfolio and actively manage the existing portfolio that was constructed in the previous step. In order to do so, we would need to pre-process all of the text within a new post and further apply a classification model to associate the text with one of the above defined labels.

First, we will once again pre-process the data as follows: remove all stop words, stem each word to obtain its root and finally vectorize all stemmed stock text into unigrams/bigrams. Once vectorized, we will reduce the dimensionality of our problem in order to focus on the most important features, preserve them and ensure our Google Colab environment has sufficient computing power to train our model accordingly. The final step of our data preparation will be to divide our data into test and train data frames as well as run a One-Hot Encoding to ensure our categorical variables are factored properly into our models.

Second, we will be running 2 models in parallel for our classification problem, both known to perform classification very well: Random Forest and XGBoost. We will run both models with the default parameters, observe their precision, recall and f-score and finally make a determination on which one to use for our task based on performance. Once we select a model, we will perform a hyper parameter tuning to ensure we extract every bit of performance and maximize our precision/recall metrics. We selected the **Random Forest** model to use for our Buy/Sell label prediction as it outperformed the XGBoost model across all the measures (Precision, Recall, Fscore). Once we tuned the hyper parameters we obtained the following as our best performing input variables: *criterion = 'gini', max_depth = None, n_estimators = 100.*



	Random Forest	XGBoost
Precision	94.70%	84.54%
Recall	94.67%	84.15%
Fscore	94.61%	83.38%

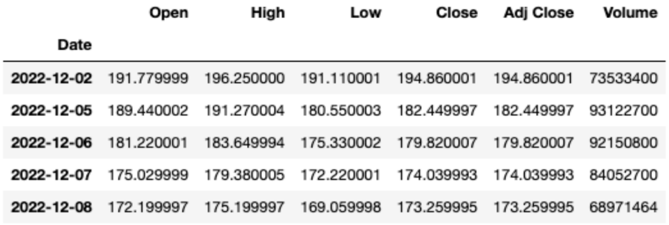
Form 2. Scores of Label Prediction Models

Now that we have a portfolio of stocks and a way to iterate on any new information on the said stocks, we will look into predicting their value in the future as well as their impact on the overall portfolio performance.

4.4 Forecasting Stock Prices

In this part, we will try to predict the stock price in the following half a year using the top 10 stocks selected in the previous steps. Time series analysis comprises methods for analyzing time series data in order to extract meaningful statistics and other characteristics of the data. A time series is a series of data points indexed in time order and it is used to predict the future based on previously observed values. Time series are very frequently plotted via line charts. Time series are used in statistics, weather forecasting, stock price prediction, pattern recognition, earthquake prediction, etc. Time series forecasting is the use of a model to predict future values based on previously observed values.[1] In this part, we will also tackle with different topics concerning data science in stock price predictions, like data extraction and cleaning, data processing, as well as the creation of machine learning models to make predictions.

The first step is to extract stock price data. Yahoo Finance is one of the widely used platforms that provides stock market data. We can easily download the dataset from their website and can access it directly from a Python program with *yfinance* library. By setting the start date and end date, we can easily get the data between the time range. The date of the earliest historic data of these top ten stocks we can extract is 2012-06-01. Therefore, we will extract historic stock price data between 2012-06-01 and today (12/08/2022 as the project finalized) to train the price prediction models. We can get the daily open, high, low, close, adjusted close price, and transaction volume data.

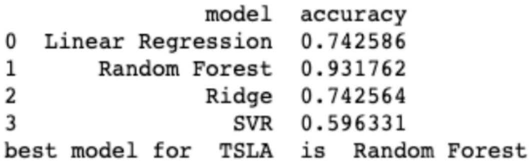


	Open	High	Low	Close	Adj Close	Volume
Date						
2022-12-02	191.779999	196.250000	191.110001	194.860001	194.860001	73533400
2022-12-05	189.440002	191.270004	180.550003	182.449997	182.449997	93122700
2022-12-06	181.220001	183.649994	175.330002	179.820007	179.820007	92150800
2022-12-07	175.029999	179.380005	172.220001	174.039993	174.039993	84052700
2022-12-08	172.199997	175.199997	169.059998	173.259995	173.259995	68971464

Form 3. Example Historical Stock Data of Tesla

The second step is data processing. Due to the high correlation, We have to add some features to the dataset. The HL_PCT calculates the difference percentage between the highest price and the lowest for each day, and the PCT_change is the difference percentage between the open and the close price for each day. We will use these two features and feature 'Adj Close', and feature 'Volume' to train our model. We will also scale the data between -1 and 1 in order to put all columns in the data set in the same range. Before we train the model, we need to split the data into the train set and the test set with cross-validation, so we can use the test set to evaluate the accuracy and compare the model's performances.

Then we need to train models. For each stock, we are going to try different machine-learning models, including linear regression, random forest, ridge, and support vector regression. Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data; ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where the independent variables are highly correlated; random forest regression is a supervised learning algorithm that uses ensemble learning method for regression; support vector machine regression is a nonparametric regression relying on kernel functions. Different models have different performance on different data, therefore, picking the best model with the highest prediction accuracy for each stock data is important. For example, for the Tesla stock, the best model is the random forest with an accuracy of 0.93.



```
               model  accuracy
0  Linear Regression  0.742586
1      Random Forest  0.931762
2              Ridge  0.742564
3                SVR  0.596331
best model for  TSLA  is  Random Forest
```

Form 4. Price Forecast Model Performance

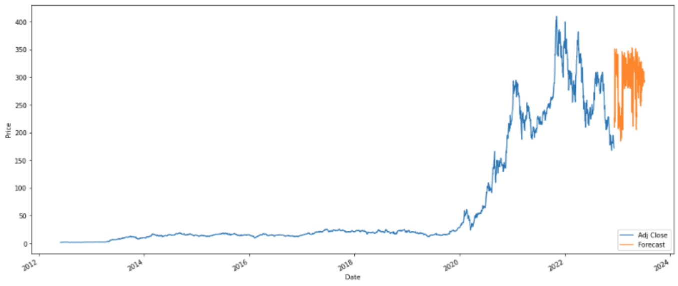
4.5 Portfolio Optimization

Portfolio optimization is the process of selecting the best portfolio, out of the set of all portfolios being considered, according to some objective. The objective typically maximizes factors such as expected return, and minimizes costs like financial risk. [2] Finding the right methods for portfolio optimization is an important part of the work done by investment banks and asset management firms.

We conducted portfolio optimization based on the 10 stocks we chose from previous steps using three methods: Mean Variance Optimization, Hierarchical Risk Parity (HRP), and Mean Conditional Value at Risk (mCVAR). [3] Mean Variance Optimization is one of the early methods, it works by assuming investors are risk-averse. Specifically, it selects a set of assets that are least correlated (i.e., different from each other) and that generates the highest returns. The HRP method works by finding subclusters of similar assets based on returns and constructing a hierarchy from these clusters to generate weights for each asset, and it is not as sensitive to outliers as mean variance optimization is. The mCVAR works by measuring the worst-case scenarios for each asset in the portfolio, which is represented here by losing the most money. The worst-case loss for each asset is then used to calculate weights to be used for allocation for each asset. In each method, we manually set the upper limit of the weight of each stock to 20% in order to let the weight spread out more evenly. Output from mCVAR is selected for final results since the weights are the most balanced.

5.0 Results

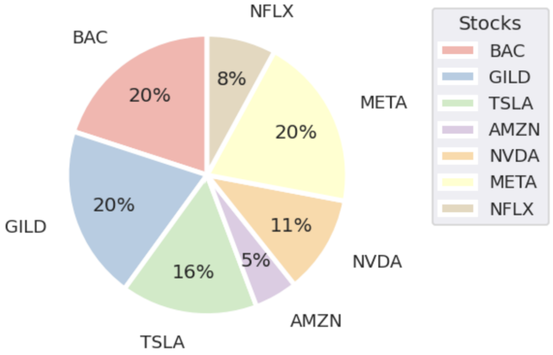
For each stock, we can use the best model to make future price predictions. Our model predicts from December 2022 to July 2023, the trend for Tesla will be like this plot, and the volatility during the prediction duration is about 23%.



```
the volatility percent of  TSLA  during the prediction period is  22.84 %.
```

Plot 2. Future Price Forecast Trend of Tesla

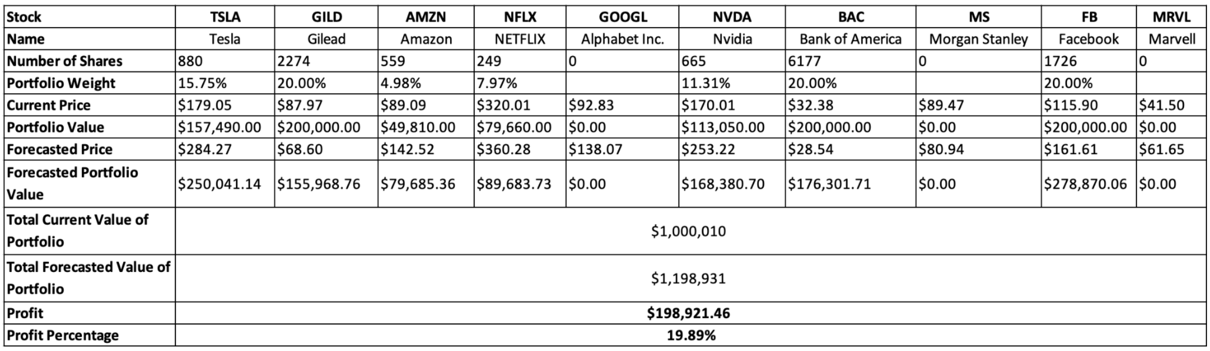
The pie chart below shows the weights of the stocks in the optimized portfolio. There are 7 stocks with weights over 0: Gilead, Bank of America and Facebook take the majority of the weight by 20% percent each, Tesla takes 16%, Nividia takes 11%, NETFLIX takes 8%, and Amazon takes 5%. Percentages are rounded for the clarity of the chart, while the actual percentages can be found in the conclusion.



Plot 3. Stock Weight Allocation of the Optimised Portfolio

6.0 Conclusion

Form 1 shows the optimized number of shares and allocation of the stocks using mCVAR method. The stocks with non-zero portfolio values and the suggested invested amounts are: Gilead $155,968.76, Bank of America $176,301.71, Facebook $278,870.06, Tesla $250,041.14, Nividia $168,380.70, NETFLIX $89,683.73, and Amazon $79,685.36. These companies spread out into different industries such as healthcare, finance, social media and automotive. Suppose we have $1,000,000 to allocate, our profit after two quarters will be $198,921.46.



Stock	TSLA	GILD	AMZN	NFLX	GOOGL	NVDA	BAC	MS	FB	MRVL
Name	Tesla	Gilead	Amazon	NETFLIX	Alphabet Inc.	Nvidia	Bank of America	Morgan Stanley	Facebook	Marvell
Number of Shares	880	2274	559	249	0	665	6177	0	1726	0
Portfolio Weight	15.75%	20.00%	4.98%	7.97%		11.31%	20.00%		20.00%	
Current Price	$179.05	$87.97	$89.09	$320.01	$92.83	$170.01	$32.38	$89.47	$115.90	$41.50
Portfolio Value	$157,490.00	$200,000.00	$49,810.00	$79,660.00	$0.00	$113,050.00	$200,000.00	$0.00	$200,000.00	$0.00
Forecasted Price	$284.27	$68.60	$142.52	$360.28	$138.07	$253.22	$28.54	$80.94	$161.61	$61.65
Forecasted Portfolio Value	$250,041.14	$155,968.76	$79,685.36	$89,683.73	$0.00	$168,380.70	$176,301.71	$0.00	$278,870.06	$0.00
Total Current Value of Portfolio	$1,000,010									
Total Forecasted Value of Portfolio	$1,198,931									
Profit	**$198,921.46**									
Profit Percentage	**19.89%**									

Form 5. Forecasted Portfolio and Profit

From this point, we combined the methods of natural language processing and machine learning to construct and optimize the investment portfolio. The forecasted price of the stocks selected and the profit percentage of the portfolio is also provided for reference.

7.0 Discussion

In reflection, our modeling was successful in achieving our ultimate goal of using natural language processing techniques to inform stock exchange decisions. However, our methodology and results can be improved upon, particularly when it comes to sampling methods, modeling, and hyperparameter tuning.

As aforementioned, we used 2020 data from a longitudinal dataset that spanned from 2009 to 2020. By selecting analysts' articles by 2020, we introduce a degree of bias to our methodology. For example, by excluding data from the year 2019, we could be missing out on analyst data that informs the long-term performance of certain stocks. However, we made this decision based on our computational limitations, and if given the resources, would have used a multi-year sample.

Additionally, we examined the performance of Random Forest and XGBoost models to predict the performance of the top 10 stocks. However, these models do possess limitations, so applying other models to our methodology, like principle component analysis (potentially a hazard because we are losing important identifiable information) and neural network models for feature selection.

Finally, we would explore the degree to which adjusting the hyperparameters of our models would impact the precision of our methodology. More specifically, we suspect that tuning accounts for the number of trees and verbosity may provide additional information that informs our stock exchange recommendations. However, too much information, especially information that does not provide significant predictive power, may negatively affect our modeling.

8.0 Appendix

8.1 Disclaimer

This project is intended to be used and must be used for informational purposes only. It is important to do your own research before making any investment based on your own personal circumstances. Independent financial advice should be taken from professionals.

8.2 References

1. Wikipedia contributors. (2022b, December 8). Time series. Wikipedia. https://en.wikipedia.org/wiki/Time_series
2. Wikipedia contributors. (2022, August 2). Portfolio optimization. Wikipedia. https://en.wikipedia.org/wiki/Portfolio_optimization
3. Pierre, S. (2021, October 6). An Introduction to Portfolio Optimization in Python. Built In. https://builtin.com/data-science/portfolio-optimization-python