Greenwich.HR Project

Team Gannett Peak: Congda Xu, Binqi Shen, Matthew Ko, Isaac Choi

I. Introduction

As a leading labor market intelligence company, Greenwich.HR holds real-time and accurate job market data that can generate immense valuable and actionable business insights. Our project centers on uncovering the inherent relationship between companies' hiring behaviors and their stock performances. Using Greenwich's exclusive job market data, we developed machine learning models that are predictive of future stock performance. Additionally, we formulated a trading strategy that re-balances stock selection in an investment portfolio every month. In the end, our portfolio return is compared against the S&P 500 index performance to validate model effectiveness and accuracy.

II. Data Manipulation

Stock Data

We scraped our stock price data from an online API source called AlphaVantage. It offers reliable financial metrics including stock prices, sales, revenues, etc. Among these metrics, we decided to use Closing Price as our response variable because it is best suited for our project and will allow us to compare the stock return with the S&P 500 index in the later sections. With the scraped data, we did some data cleaning and computed the close price as well as the stock return in each month for each company ticker.

Greenwich's Proprietary Job Market Data

We initially did a screening of Greenwich's job dataset. With their proprietary software, Greenwich scrapes data from over 1.5 million sources and 800,000 organizations. This makes it inevitable that there were numerous text encoding and misalignment issues in the untouched files. Therefore the first thing we did was to perform various data cleaning and exploratory analysis.

After doing a preliminary screening of the dataset, we filtered out jobs that were posted after January 31st, 2020 to avoid the COVID-19 impact on the job and the stock market. Then, we reformulated the dataset to extract key features from Greenwich's job data for stock predictions. In our assumption, we treated each company's data in each month as a single observation. More specifically, we aggregated all job-related and stock-related features in the monthly interval for each company ticker. This allowed us to join the job metrics and stock metrics to one master table based on the company 'ticker' column and the 'month' column, enabling us to make inferences on how well each company is performing and to realize the immense value of Greenwich's data in the modeling stage.

Feature Introduction

With Greenwich's proprietary data, we extracted 9 features as the predictors for our model, including 'Cbsa_Pop_Percentile(20s)_min', 'Cbsa_Pop_Percentile(20s)_max', 'New_Cbsa', 'Salary', 'average_posting_duration', 'Job_posting', 'new_posting', 'impCount', and 'intCount'. In addition, we chose 'stock price' as our response variable in our regression model and chose 'Positive' to be our response variables in our classification model. More detailed descriptions of each are explained in *Appendix I.*

III. Model Selection and Implementation

Our main idea was to build 2 models, a classification model and a regression model. The first was used to predict stock trends by classifying return as positive or negative, while the second was used to predict the exact stock price. Upon exploring several models, we decided to use Random Forest Classifier as our first screening condition and the Vector Auto Regression (VAR) model as our second screening condition in the stock selection procedure. We aimed to select the stocks that are predicted to have positive returns by the classification model and predicted to have high returns in the VAR model.

Stock Trend Prediction: Random Forest Classifier

The classification model was built around company based data, focusing on how the company was performing and whether the company will generate positive or negative returns for a given month. We decided to utilize the random forest model in our classification predictions. This model does not place any weight on time nor on company's past performances, but rather focuses on how they are acting based on Greenwich data in order to predict if they will perform well in the future.

The Random Forest model performed at around 57%-65% accuracy depending on the number of months lagging chosen. This can be attributed to the difficulty in predicting stock performance without looking toward the stock market for information. However, with a Random Forest model, we were able to interpret the importance of various features. Among the utilized features, the change in job postings, average posting duration, and average change in salary were of relatively high importance (see *Appendix II*).

Stock Return Prediction: Vector Auto Regression Model

We started with regression models including Multiple Linear Regression and Random Forest Regressor, but we realized that one common disadvantage of both of the models is that they don't take into account the fact that each company is a unique presence and has a distinct stock price pattern. Another key drawback of Random Forest Regressor is that it cannot extrapolate data outside the range of the current data. For example, if the largest stock return in the training dataset is 10%, then the highest predicted stock

return in the test set will never exceed this value no matter how high it actually is, making this model not suitable for predicting stock returns.

Considering the special nature of stock patterns, we decided to use time series models to predict monthly stock returns. The time series model we decided to use is Vector Auto Regression (VAR). This model is good for dealing with multivariate time series since it considers not only the historical values but also the dependency on other variables when predicting the monthly stock price. Therefore, we decided to build individual VAR models for each individual stock. In order to fit time series models, we transformed our data tables by converting monthly percentage changes to monthly values.

We used MAE(Mean Absolute Error) and MAPE(Mean Absolute Percentage Error) to evaluate our VAR model prediction on over 1400 stocks, and we can get a MAE around 6 and a MAPE around 18%.

Lagging Period Selection

For many companies, hiring behaviors are “leading” indicators of “lagged” stock market responses, and the information we derive from job posting data might not immediately be reflected in stock prices. Therefore, we tested our model on 6 different lagging relationships (ranging from 0 month to 5 month) between our x and y variables to find out the most appropriate lagging period. The resulting metrics calculated for the 6 lagging possibilities are displayed in *Appendix III*. After we compared the results of different lagging periods in both of our models, we picked 2 months lagging as the optimal one.

Train-Test Split

We splitted our model-ready data into training and testing datasets. The testing period we selected was from August 2019 to January 2020, which are the last 6 months before COVID, and the training period is all months before August 2019 for each company. We filtered out companies that have less than 12 months of historical data in our model to prevent bias introduction. But these companies can be added to the models after they meet this 12-month benchmark. We used this untouched testing dataset to test our model performance and trading techniques in the next section.

IV. Trading Strategy Development

Our trading strategy has two parts: one is the stock selection strategy to decide portfolio components for each month and the other is the weight allocation strategy which determines the weight of each stock in the portfolio.

Stock Selection

Our stock selection strategy starts with picking the top 10 stocks for each month, which are determined by the following criteria:

1) According to classification prediction, pick the stocks that have positive return for at least one of the next two months.
2) According to regression prediction, pick the stocks whose predicted return is in the range of 2% ~ 20%. The upper bound is set to be 20% because those stocks that are predicted to have a return over 20% involve large model risks and require additional fundamental research. The lower bound is set to be 2% because the average monthly return of S&P 500 is lower than 2% and we want to select the stocks that have the potential to outperform S&P 500.
3) We rank the rest of the stocks by their Sharpe ratio, which is a financial indicator that takes into account both the return and risk, and our logic here is to filter those stocks that can maximize returns while minimizing volatility.
4) We exclude those stocks that are in the Dow Jones index, because the client wants us to select those stocks that are not “familiar big names”.

After finding the top ten stocks for each month based on the stock selection strategy above, we add those that are not in the original portfolio to the portfolio. We then drop the stocks in the portfolio which are predicted to have negative returns in the next two consecutive months. And we finally drop the stocks in the portfolio which have lost money in the past two consecutive months.

Weight Allocation

After we get the portfolio components of each month, we also need to establish a weight allocation strategy to decide what percentage of amount each stock should take in the portfolio.

We started out with trying to implement the tangency portfolio weight optimization of CAPM model, which is widely used in the financial industry, however, we discovered some issues that made this strategy unsuitable for our project. We implemented this strategy using a python package called pypfopt, and we found that a lot of the weights of the stocks were assigned to negative or zero, which is not ideal in our case because the client did not want shorting and we did not want to drop stocks that were selected by our model. Thus, we came up with two possible plans to avoid this issue:

Plan A: Buy an equal amount of shares for each stock in the portfolio. The logic behind this method is to give an equal chance to all stocks we selected from model prediction. Whereas, one potential problem is that those stocks whose stock price are higher will be assigned with larger weight.

Plan B: Assign weight according to the Sharpe Ratio of stocks. The weight of each stock is its Sharpe Ratio over the sum of all portfolio stocks’ Sharpe Ratios. As mentioned above, Sharpe Ratio is an indicator that balances both return and risk, hence

those stocks with higher Sharpe Ratio are expected to perform better than others. Therefore, we hope that if we assign more weight to the stocks that have larger Sharpe Ratios. This way, we can achieve better and more stable portfolio performance.

Performance Evaluation

To test the performance of our stock selection and weight allocation strategy, we ran our strategy over the prediction result of our Random Forest Classification model and Vector Autoregression model under the assumption of 2 month lagging. The testing period is from August 2019 to January 2020, as introduced in prior sections.

After we simulated the strategies, we used the actual stock price during that period to calculate our portfolio return. The result showed that for Plan A, we can achieve a HPR(Holding Period Return) of 16.6%, and for Plan B we can achieve a HPR of 11.5%, both outperforming the S&P 500's 8.2% HPR during the same period.

V. Potential Risks

Although the data provided was comprehensive in many ways, the time range spanned approximately 4 years for our model. A longer time period would benefit the model by allowing the models to potentially pick up various seasonal trends. Additionally, the models chosen for this project are representative of their performance on this dataset, and their performance on new data is unknown. Moreover, the weight allocation strategies we currently use are experimental and require further research.

VI. Conclusion

Utilizing Greenwich hiring data as monthly time series data for each company, we implemented two models to aid in choosing stocks for a portfolio. We implemented a Random Forest Classifier model to predict the stock trend and a Vector Autoregression model to predict stock price. Based on this, we could predict the stock performance of individual companies with their job posting data. We also discovered that 2 months lagging is the optimal lagging period to link our job and stock data. Finally, we came up with a stock picking strategy and two weight allocation strategies. All the testing results we got outperformed the S&P 500 return during the testing period, which indicated that our strategy was effective.

Future research potentials for this project include attempting to improve model performance using more complex time series models such as Long Short Term Memory networks. Additionally, we would suggest our client consult with experts about more possible input features and information that could be included in our current model. This may potentially increase variables' predicting power. Besides, more research could be done on the possible weight allocation strategies in order to make our entire trading strategy more stable and profitable.

VII. Appendix

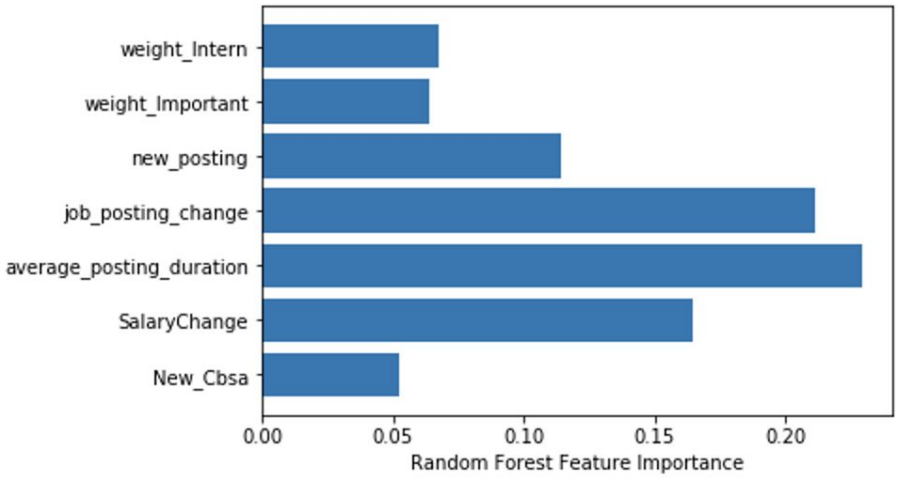
Appendix I: Feature Description

Feature	Description
X Features (Predictor Variables)	
Cbsa_Pop_Percentile(20s)_min	Least populated cbsa (core-based statistical area) where company has a job posting
Cbsa_Pop_Percentile(20s)_max	Most populated cbsa (core-based statistical area) where company has a job posting
New_Cbsa	Total number of postings to new CBSAs Note: This feature extracts geographical information from zip code level information
Salary	Average salary of job listings for each company in each month
average_posting_duration	Average cumulative posting days for all active jobs for each company in each month
Job_posting	Number of active job postings for each company in each month
new_posting	Number of new job postings for each company in each month
impCount	Total number of important roles postings Note: This feature allows us to track the open positions a company is hiring for in regards to roles such as VP, supervisors, directors, chief officers, and those roles requiring more than 10 years of experience. Selected through regular text expressions of desired titles and tags for jobs of interest.
intCount	Total number of entry-level roles postings Note: This feature allows us to track the open positions a company is hiring for in regards to roles such as interns, entry-level positions, as well as those where training is needed or training is provided. Selected through regular text expressions of desired titles and tags for jobs of interest.
Y Features (Response Variable)	

Stock Price (Regression)	Stock close price of each month
Positive (Classification)	1 if positive return for given month, 0 otherwise

Note: Each of the features above is calculated for *each company* in a *monthly interval Additionally, the features of important roles, intern roles, and salary were utilized on a percent change basis for the classification model.*

Appendix II: Relative Feature Importance from Random Forest Model



Appendix III: MAE and MAPE Calculated for 6 Lagging Possibilities

Lagging	MAE	MAPE
0	6.11092	18.8887
1	5.92207	18.9751
2	5.82751	18.6476
3	6.06731	18.9849
4	7.38485	24.3747
5	6.94149	22.2253