Jiarui (Jerri) Zhou, Zhizun (Skye) Li

Professor Ying Becker

Fin 285A-1

Dec 8, 2021

Using Machine Learning Algorithms to Estimate VaR of S&P 500

Part I: Objective of the research

With efficient market hypothesis(EMH)[1], we believe that stock prices follow a random walk. That is, all the available information and expectations are already fully reflected in the stock prices, and future price movements would be driven by unforeseen events, thus are unpredictable. On the other hand, however, many researchers believe the volatility of one stock carries on. Poon and Granger (2003)[2] summarized 93 studies of volatility forecasting ,and concluded that financial market volatility is forecastable. They stated that this conclusion does not violate EMH since "accurate volatility forecast is not in conflict with underlying asset and option prices being correct".

---

[1] Norton. "A Random Walk Down Wall Street." Accessed Jan. 22, 2021.

[2] Ser-Huang Poon, & Granger, C. W. J. (2003). Forecasting volatility in financial markets: A review. *Journal of Economic Literature, 41*(2), 478. Retrieved from https://www.proquest.com/scholarly-journals/forecasting-volatility-financial-markets-review/docview/213188445/se-2?accountid=9703

In this project, we use machine learning models to predict volatility of S&P500, and compare predicting results with traditional prediction methods. Specifically, we build and compare 6 machine learning algorithms using daily trading information of S&P500 from 1950 to 2012 and choose *Lasso Regression* model as our best model to predict moving standard deviations for S&P500 from 2018 to 2021, which represent stock's volatility. For models built on sample standard deviations, we try the Moving Average method (MA), the Exponentially Weighted Moving Average method ($EWMA$), and fixed standard deviation as a benchmark to estimate volatility in the same period. With estimated volatility, we can calculate Value At Risk(VaR) using historical data.

To evaluate the performance of the above volatility forecasting methods, we backtest 95% VaR with real historical data and count each underestimation of VaR as an exception to get violation ratio (VR). To take overestimation into consideration as well, we also plot rolling mean of VR overtime to see if our risk forecasting method is being too conservative. The backtest results suggest that $EWMA$is the best model of this experiment, whereas our machine learning model fails to outperform traditional forecasting methods.

Part II: Brief Literature Review

Though volatility is not the same as risk, we can interpret it as uncertainty, which is an important factor to consider when making investment decisions or creating portfolios. Investors have their level of risk to bear, and a decent prediction of price volatility can be a good starting point to assess investment risks.

VaR is a widely used market risk measurement tool nowadays. Ever since Basel Accord I in 1988, Banks and trading houses must set aside enough capital based on the calculation of VaR to prevent Bank failures, which were particularly prominent during the 1980s[3]. Therefore, an effective forecasting of volatility to manage risk becomes a necessity, since it helps to generate a reasonable VaR calculation.

Recently, using machine learning algorithms to predict risks in financial markets has become a popular topic in academics. Xiong, Nichol et al. (2015)[4] applied deep learning and neural network models to predict S&P500 volatility and resulted in a mean absolute percentage error of 24.2% that outperformed *linear Ridge/Lasso* and autoregressive $GARCH$ benchmarks by at least 31%. Their research showed strong promise for better predicting stock behavior via machine learning. Actually, they incorporated Google domestic trends as indicators of the public mood and macroeconomic factors to help their algorithm interpret the potential reason in the outside world for changes in stock prices, which didn't really seem to be a fair game for traditional forecasting methods using only trading information. With only trading data, Sullivan[5] used 1000 largest and most liquid single stocks for individual US stocks over the past 18 years. He trained a neural network model with several daily transformations data including Open, High, Low, Close, Volume, and compared between complex models with more *LSTM* layers and simpler models, and found that the extended network outperforms during training, but the simpler network outperforms during testing.

---

[3] Fadi Zaher, How Basel 1 Affected Banks, https://www.investopedia.com/articles/07/baselcapitalaccord.asp#the-purpose-of-basel-i

[4] Xiong, Ruoxuan, Eric P. Nichols, and Yuan Shen. "Deep learning stock volatility with google domestic trends." *arXiv preprint arXiv:1512.04916* (2015).

[5] Jason C. Sullivan, *Stock Price Volatility Prediction with Long Short- Term Memory Neural Networks,* Department of Computer Science, Stanford University

For non-machine learning techniques, we use MA and $EWMA$ as basic changing volatility methods to estimate VaR using rolling windows. Galdi and Tamayo(2007)[6] used the one-day VaR bounds violation test to compare the efficiency of the $GARCH$ and $EWMA$ models. They suggested that VaR calculated by $EWMA$suffered a smaller number of violations than that calculated by $GARCH$ for a window size of 1500 observations.

In this project, we compare machine learning methods and non-machine learning techniques to predict stock prices volatility, represented by standard deviations. Specifically, we'd like to see, when providing almost the same amount of data from daily trading information published, whether or not machine learning techniques can outperform classic estimation methods.

Part III: Data and Methodology

We gathered data from yahoo finance using the Python package *yfinance*[7], and the data set contains S&P500 ticker data from 1950-01-03 to 2021-12-04. The original data frame includes six columns: *Open, High, Low, Close, Adj Close* and *Volume*, and 18099 rows.

We identified predictors to predict volatility for the Machine Learning method, which will further predict VaR. The predictor variables include *Daily Spread* (High-Low of the previous day), *Daily Return* (the percentage change of adjusted closing price between today and yesterday), *five-day closing price* (rolling mean of 5 days' adjusted closing price), *30 days closing price* (rolling mean of 30days' adjusted closing price), *five-day volatility* (rolling

[6] Galdi, Fernando Caio, and Leonel Molero Pereira. "Value at Risk (VaR) using volatility forecasting models: EWMA, GARCH and Stochastic Volatility." BBR-Brazilian Business Review 4.1 (2007): 74-95.
[7] See https://pypi.org/project/yfinance/ for detail information

standard deviation of 5 days' adjusted closing price), *30 days volatility* (rolling standard deviation of 30 days' adjusted closing price), *day of the week* (as in dummies form) and *month of the year* (as in dummies form). We also identified predict variables - 2-year volatility by comparing 0.5 years, one year, 1.5 years, two years, 2.5 years, and 3 years rolling window using backtesting method, 2-year rolling window performs the best among all.

Since the data before the 2008 financial crisis may have different behavior compared to a post-financial crisis, we divide the data frame into training (1950 - 2012), validation (2013 - 2017), testing (2018 - 2021), using the validation set to test model parameters, then use the best fine-tuned model we have to predict the volatility for 2018 - 2021.

For non-machine learning techniques, we choose Simple Moving Average (MA) and Exponential Weighted Moving Average ($EWMA$), both classic techniques, to forecast volatility.

We use formula (1) and formula (2) to calculate rolling standard deviation for stock price at time t. We also calculate the fixed standard deviation of stock prices from 2014 to 2021 as a benchmark.

$$MA: \quad \widehat{\sigma}_t^2 = \frac{1}{m} \sum_{j=0}^{m-1} (R_{t-j} - E(R_t))^2 \qquad \text{formula(1)}$$

$$EWMA: \widehat{\sigma}_t^2 = (1 - \lambda)(R_{t-1}^2 + \lambda R_{t-2}^2 + \lambda^2 R_{t-3}^2 + \ldots + \lambda^{m-1} R_{t-m}^2) + \lambda^m \sigma_{t-m}^2 \text{formula(2)}$$

With moving standard deviations, we can calculate the return rate (R^*) of $(1 - p)\%$ VaR each day with formula (3), where $percentile(p)$ is the critical value of the historical stock prices distribution based on percentile method, and μ is the mean of stock prices during the estimation period.

$$R_t^* = \mu + percentile(p)\sigma_t \quad \text{formula(3)}$$

The purpose of calculating VaR for financial activities is to prepare for extreme loss. To interpret, we can think of $(1-p)\%$ VaR as the maximum amount of loss for $(1-p)\%$ of situations. In this case, if we count one exception for each actual return ratio (R) smaller than R^*, then a good estimation of risk should end up with the total number of exceptions accounting for around 5% of the estimation period, while the closer to 5% the better this forecasting.

Part IV: Analysis and Estimate (results and discussion)

We use and compare 6 different Machine Learning models, *Linear Regression, Ridge, Lasso, Regression Tree, Random Forest, XGBoost Regression*. The table below list R^2 and MSE results for the testing set for comparison. *Linear Regression*, the simplest model, has an R^2 of 0.1172 and MSE of 12888.23. *Ridge*, with very similar results to *Linear Regression*, has an R^2 of 0.1172 and MSE of 12888.24. *Lasso* has an R^2 of 0.122602 and MSE of 12808.83. *Decision Tree* and *Random Forest* both have a -1.22 R^2 score and 32000+ MSE score. Finally, *XGBoost Regression* provides an R^2 score of -0.3379 and an MSE score of 19531.54. *Lasso Regression* performs the best among all the models, with the lowest MSE and the highest R^2. We will use the *Lasso* model to compareMA, $EWMA$, and fixed variance models. (The parameter tuning table is listed in the appendix - model selection)

Mode	Test set R^2	Test set MSE

Linear Regression	0.117163	12888.23
Ridge Regression	0.117162	12888.24
Lasso Regression	0.122602	12808.83
Decision Tree Regression	-1.2231	32454.25
Random Forest Regression	-1.2088	32245.55
XGBoost Regression	-0.3379	19531.54

Using our best mode *Lasso*, we predict the standard deviation of stock prices each day between 2018 and 2021, then use formula (3) to calculate R^* of 95% VaR at a daily level. Similarly, R^* is calculated using standard deviation estimates with MA, $EWMA$, and fixed standard deviation as comparison.

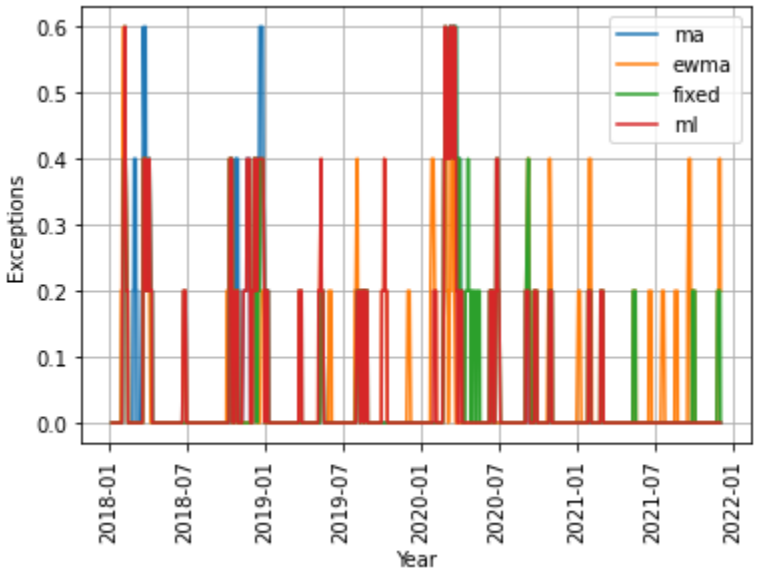
To compare the estimation results, we backtest using historical daily returns and count each time of violation return with that daily arithmetic return smaller than R^* for 95% VaR. Then, the violation ratio (VR) is calculated using formula (4), with p being the percentage level of VaR and T the length of the entire test set. The result is summarized in the table below.

$$VR = \frac{Total\ Number\ of\ Exceptions}{p \times T} \qquad \text{formula (4)}$$

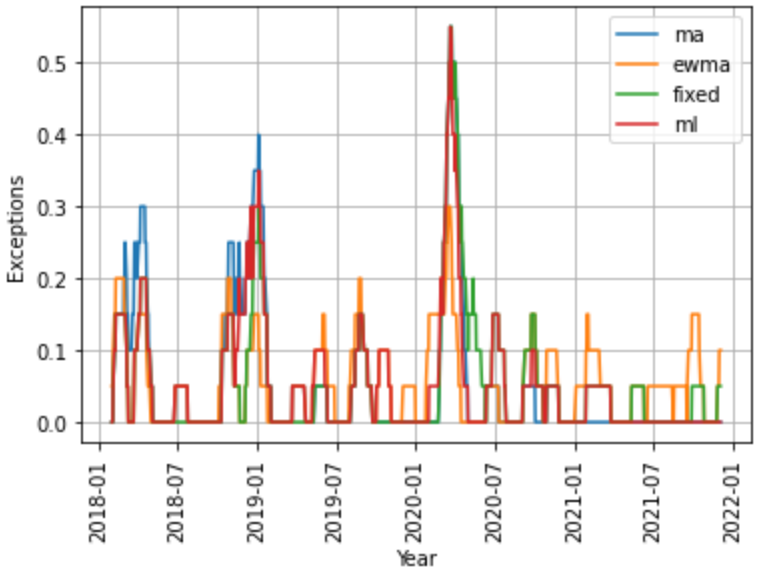
Estimation Method	Violation Ratio
MA	1.01

$EWMA$	1.09
Fixed Variance	1.01
Machine Learning	1.01

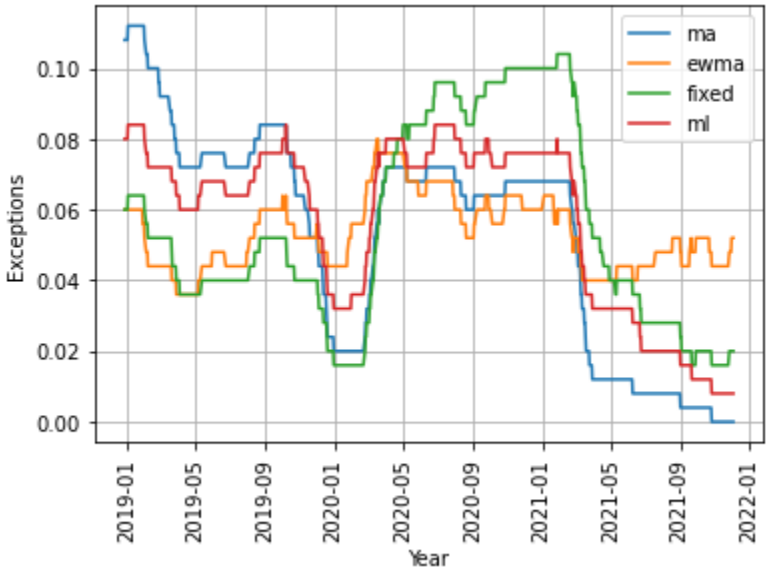
To visualize the change of percentage of violation during the test period, we also draw the plot below, with rolling window sizes of one week (5 days), one month (20 days), one year (250 days), and two years (500 days).



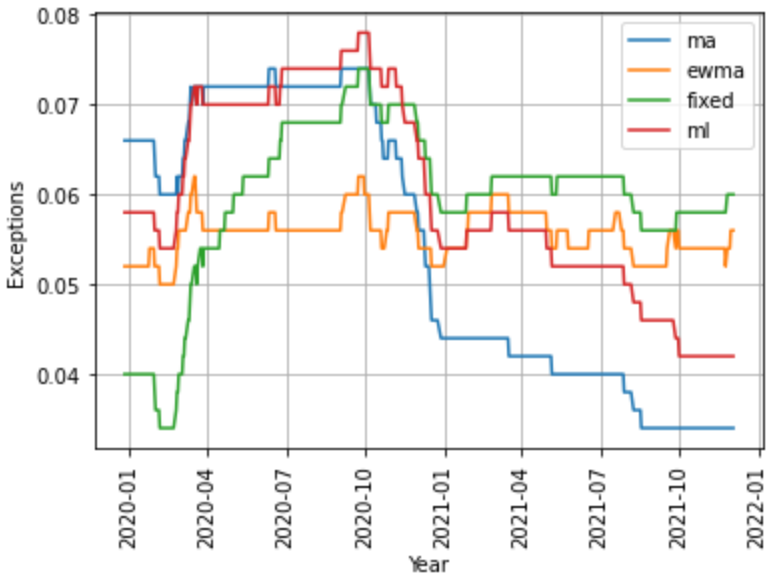
Window size = 5 days



Window Size = 20 days



Window Size = 250 days



Window Size = 500 days

Simply based on the violation ratio, $EWMA$ may seem to be the worst since it is the furthest from 1. However, VR only considers loss worse than VaR as a violation but fails to reflect situations where you overestimate the risk. A good risk estimate should generate a bar that neither overestimates or underestimates the potential volatility. With all the above considerations, we believe that $EWMA$ performs the best for test set, given it have a VR close to 1 and generates 95% VaR close to 5% overtime.

Part V: Summary

Looking back at the stock market history, a financial crisis in 2007 and 2008 greatly affected stock prices during that period. Most recently, the outbreak of Covid-19 at the end of 2019 also brought more uncertainty to the stock prices. We tried to build a more robust model by exposing the algorithm to the financial crisis during the training session and test if it would help

the model “learn” to predict in the future. Unfortunately, it looks impossible to predict unforeseen chaos using only past trading information since the outbreak of an event like a pandemic would not be depicted by past transaction data at all. Indeed, one interesting finding with this experiment is that the so-called “fancier” models (like *Random Forest a*nd *XGBoost*, compared with simple *Linear Regression*) that tend to have higher predictive power performed worse than simpler models like *Lasso.* This result conforms to the little experiment conducted by Sullivan with Neural Network as mentioned in Part II. One possible reason is that though volatility of stock prices carries on to some degree, purely counting on past trading information to forecast future risk can be hard. In this case, more sophisticated algorithms are more likely to overfit by learning noise in the past. Further analysis can use different train / validation/ test splits to see if our explanation is right.

However, as students from MSBA who spent the whole past year learning machine learning, we still believe that better designed models and smarter predictor-selection could help estimate future risk in the financial market in a much better way. For example, by including information outside the market like public moods, Xiong, Nichol et al. could build ensemble models that better predict volatility in stock prices. A wild guess: if machines, after learning from the past, could widely gather and faster digest news, will they “foresee” changes in risk from seemingly unrelated traces?

Reference

1. Norton. "A Random Walk Down Wall Street." Accessed Jan. 22, 2021.
2. Ser-Huang Poon, & Granger, C. W. J. (2003). Forecasting volatility in financial markets: A review. Journal of Economic Literature, 41(2), 478. Retrieved from https://www.proquest.com/scholarly-journals/forecasting-volatility-financial-markets-review/docview/213188445/se-2?accountid=9703
3. Fadi Zaher, How Basel 1 Affected Banks, https://www.investopedia.com/articles/07/baselcapitalaccord.asp#the-purpose-of-basel-i
4. Xiong, Ruoxuan, Eric P. Nichols, and Yuan Shen. "Deep learning stock volatility with google domestic trends." arXiv preprint arXiv:1512.04916 (2015).
5. Jason C. Sullivan, Stock Price Volatility Prediction with Long Short- Term Memory Neural Networks, Department of Computer Science, Stanford University
6. Galdi, Fernando Caio, and Leonel Molero Pereira. "Value at Risk (VaR) using volatility forecasting models: EWMA, GARCH and Stochastic Volatility." BBR-Brazilian Business Review 4.1 (2007): 74-95.

Appendix - Model selection

To fine-tune hyperparameters in each machine learning algorithm, we perform grid search for *Ridge Regression, Lasso Regression, Decision Tree Regression, Random Forest Regression* and *XGBoost Regression*. Each time, we train the model using training set with different combination of hyperparameters, then compare predicting result using validation set with metrics R^2 and MSE. Results of best[8] grid search models are summarized in the table below.

Model	parameters	train_R2	val_R2	val_MSE
Linear Regression[9]		0.75	-0.96	2978.17
Lasso[10]	alpha: 0.001	0.75	-0.94	2956.47
Ridge[11]	alpha: 0.001	0.75	-0.94	2956.47
Decision Tree[12]	{'ccp_alpha': 1.0, 'max_depth': 1, 'max_features': 15}	0.77	-0.20	1820.77

[8] The lowest R2 and MSE. For results with the same R2 and MSE, we choose the model with the highest cost penalty to avoid the overfitting problem.
[9] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
[10] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html
[11] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html
[12] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

Random Forest[13]	{'ccp_alpha': 1.0, 'max_depth': 1, 'max_features': 15, 'n_estimators': 1}	0.77	-0.18	1795.02
XGBoost[14]	{'alpha': 0.05, 'learning_rate': 0.6, 'max_depth': 5}	0.995003	-0.31195	1997.155

[13] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
[14] https://xgboost.readthedocs.io/en/latest/index.html