# Forecasting Volatility for Portfolio Optimization to Create Trading Algorithms

Tracy Strickel

# Stakeholder: Credit Suisse

Credit Suisse wants to form a 'robo-advisor' division.

A robo-advisor is a type of algorithm that automatically optimizes and rebalances investment portfolios on the behalf of retail investors.

Robo-advisors globally manage approximately $600 billion of client assets.  Credit Suisse believes that expanding into this field could provide them with a much-needed source of revenue.

# Problem Statement: Forecasting Volatility

Credit Suisse believes that forecasting volatility is highly relevant to the performance of their robo-advisors.  Volatility has profound effects within risk management, portfolio optimization, and asset selection.

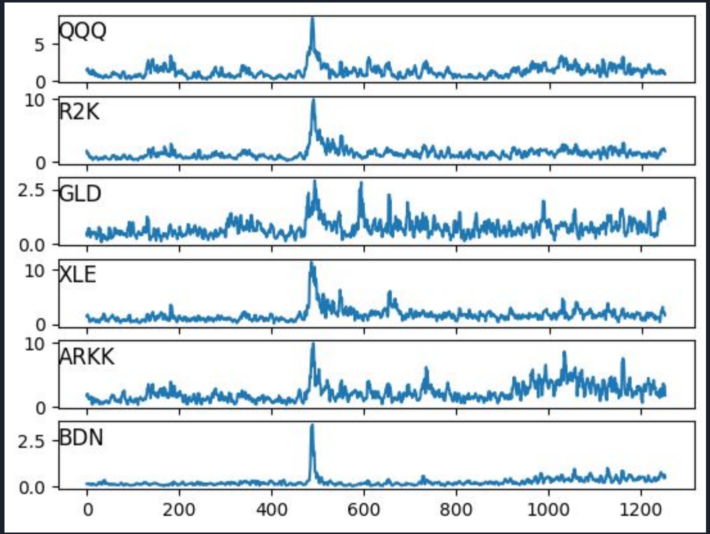
Due to its heteroscedasticity and tendency to cluster, volatility is relatively easy to forecast compared to other metrics such as asset prices, which cannot be estimated naively or reliably from historical data.

# Pt. 1: Forecasting Volatility
# Data Collection

Historical price data was collected from Yahoo Finance for the past five years for six exchange-traded funds (ETFs) across multiple asset classes.  ETFs are financial assets whose valuation is based on a basket of underlying assets and are commonly included in financial portfolios.

- ARKK - ARK Innovation
- GLD - SPDR Gold Shares
- BND - Vanguard Total Bond Market
- PFIX - Simplify Interest Rate Hedge
- QQQ - Invesco QQQ Trust Shares
- XLE - Energy Select SPDR Fund

# Pt. 1: Forecasting Volatility




Calculated Volatilities

Daily returns were calculated for each ETF by taking the percent change vs. the previous trading day. Returns were then log-normalized, which is standard practice and enables them to be summed easily later.

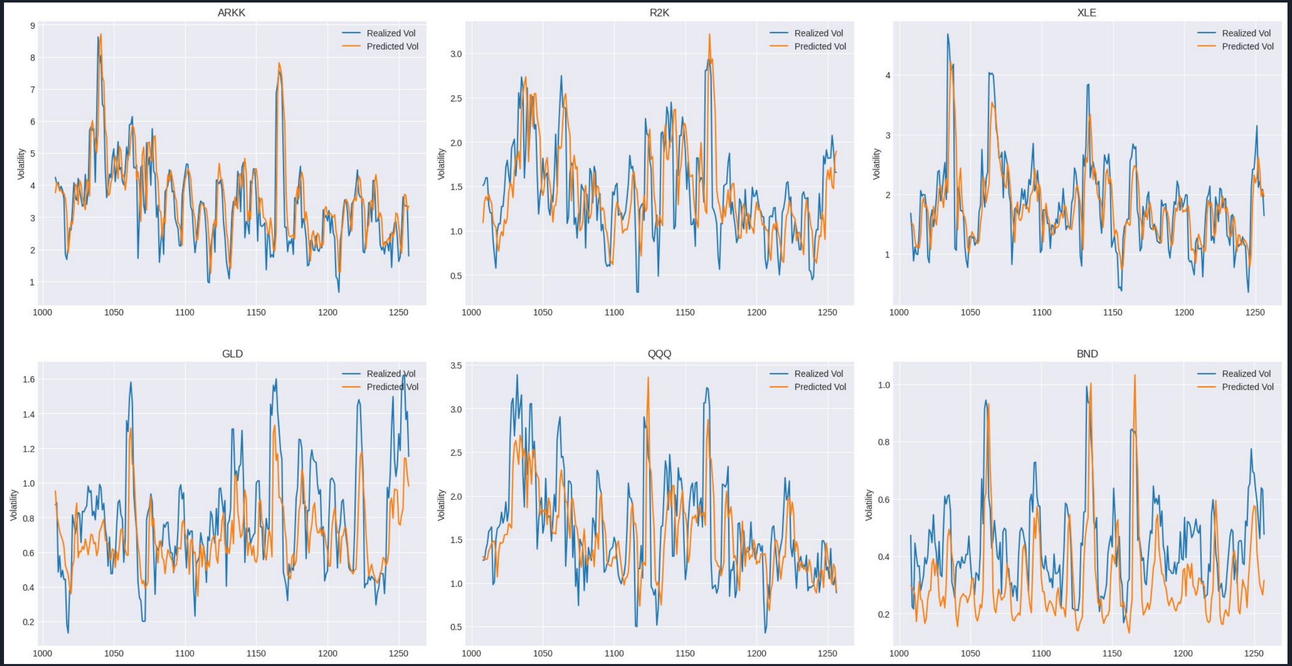
Volatility was calculated by taking the standard deviation of the return rolling average over a period of five days.

# Pt. 1: Forecasting Volatility
# Convolutional Neural Network

Data was split into an 80% train-test split, with the test set representing 250 trading days. This data was then min-max scaled and fed into a convolutional neural network, with one feature as volatility over a preceding time window and target as daily volatility.

- Convolutional layer, 64 filters, kernel size 2
- MaxPooling1D layer, pool size 2
- Flatten layer
- Dense layer (50) and Dense layer (1)
- Adam optimizer, mean squared error loss

# Pt. 1: Forecasting Volatility Convolutional Neural Network




RMSE:

- ARKK - 1.862
- GLD - 0.383
- R2K - 0.728
- QQQ - 0.772
- XLE - 0.952
- BND - 0.251

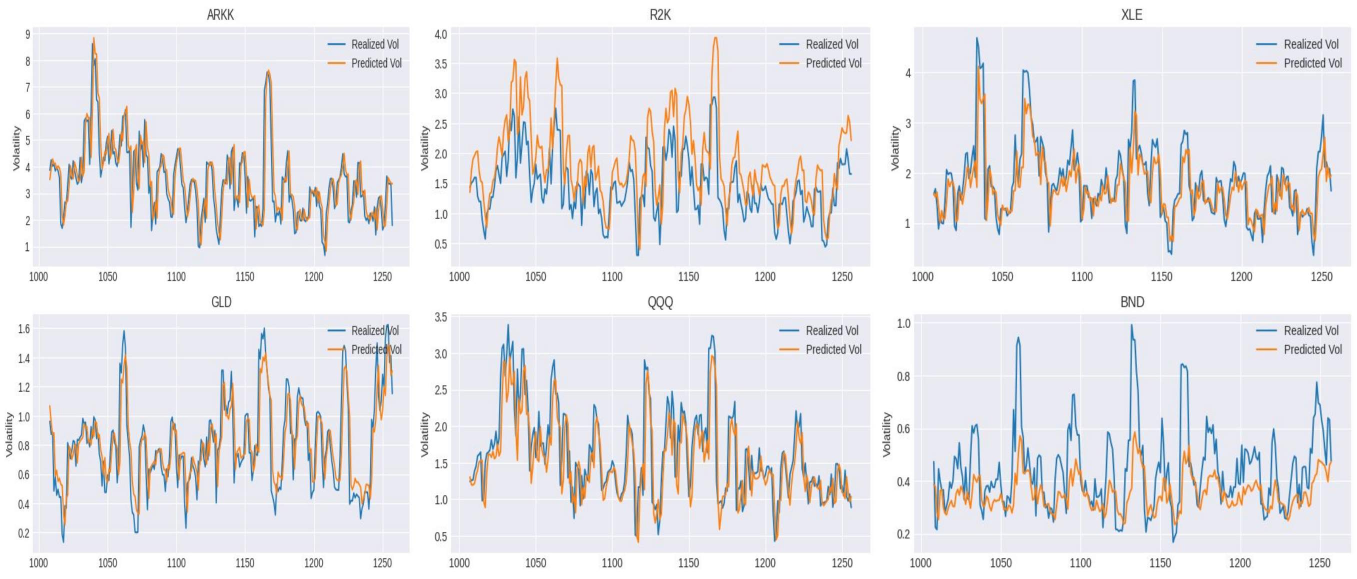
Predicted volatility (orange) vs. true volatility (blue), over test set

# Pt. 1: Forecasting Volatility
# Long Short Term Memory Network

The same data was then fed to a long short term memory network. This type of network was chosen due to its memory cell, which allows it to retain long-term information from data.

- LSTM layer (100)
- LSTM layer (100)
- Dense layer (25) and Dense layer (1)
- Adam optimizer, mean squared error loss

# Pt. 1: Forecasting Volatility
# Long Short Term Memory Network




RMSE:

- ARKK - 0.874
- GLD - 0.179
- R2K - 0.606
- QQQ - 0.397
- XLE - 0.486
- BND - 0.126

# Pt. 2: Building the Robo-Advisor Data Collection

Next, I built Credit Suisse's robo-advisor, using predicted next-day volatility to try to gain advantage.  This requires data for calculating market performance and the riskless rate (the rate of return of a theoretically riskless asset).

- Market data: SP500 Index Fund (Yahoo Finance)
- Riskless rate: 1-Year US Treasury Historical Price (New York Public Library Bloomberg terminal)

# Pt. 2: Building the Robo-Advisor Portfolio Optimization

**Formula and Calculation of Sharpe Ratio**

In its simplest form,

$$Sharpe\ Ratio = \frac{R_p - R_f}{\sigma_p}$$

**where:**

$R_p$ = return of portfolio

$R_f$ = risk-free rate

$\sigma_p$ = standard deviation of the portfolio's excess return

I used the Sharpe Ratio method for portfolio optimization, which seeks to maximize return vs. risk. The robo-advisor will re-optimize the portfolio asset 'weights', or percentages, near the end of each trading day by buying or selling assets, using predicted volatility for the next day. I calculated the riskless rate for each day from 1-year US treasury bond yields.

# Pt. 2: Building the Robo-Advisor
# Portfolio Optimization Sub-Calculations

Daily risk free rate of return:

- Log of $1 + r_t^f$ divided by 365

Expected return

- $r_a = r_f + \beta_a(r_m - r_f)$
  - $r_a$ = Expected return
  - $r_f$ = Risk free rate of return
  - $\beta$ = The asset beta

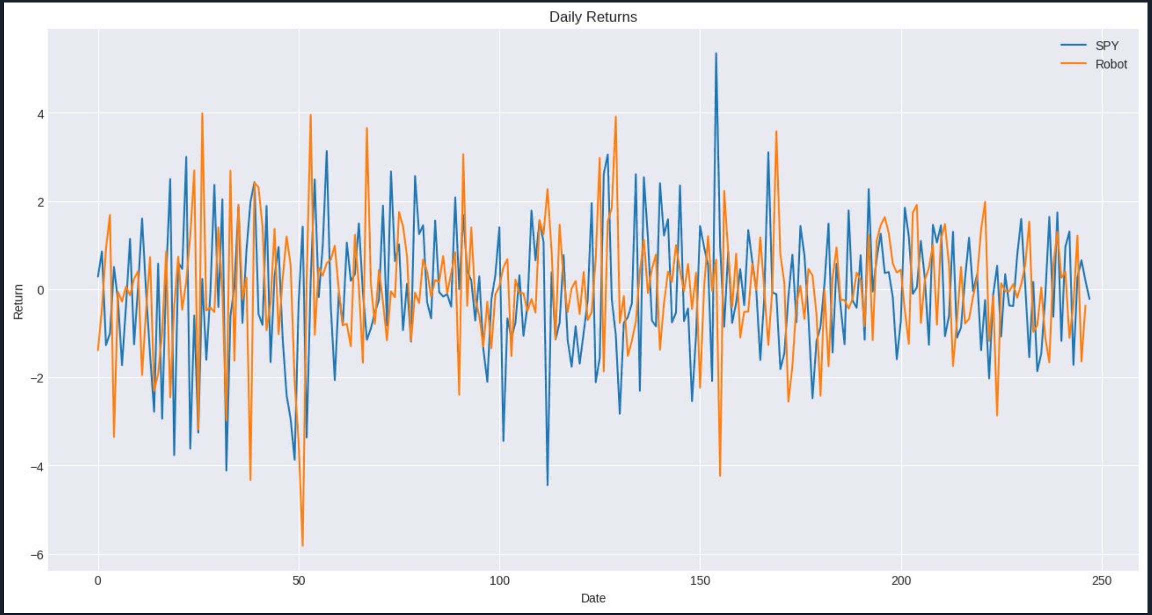
Beta

- $\beta$ = $(Covariance(R_e, R_m))/(VarianceR_m)$
- where
  - $R_e$ = Asset return
  - $R_m$ = Market return
  - Covariance between asset and market
  - Variance of market

Because my deep learning algorithms calculated daily asset volatility, I had to convert these to one portfolio volatility. I did this using the formula below. To the left are the asset expected return and asset beta formulas used.

$$\sigma^2_{port} = \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i \mathrm{cov}(i,j)\omega_j = \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i \sigma_{i,j}\omega_j$$

# Results: Trading Robot




The robo-advisor gave returns of 2.4% vs. SPY performance of approximately -11% over

the same time period.

# Further Steps

Validate the math - this is a toy constructed for learning purposes and there may be errors.

Change the time window - rebalancing a portfolio every day is very uncommon, although not unheard of.

Change the comparison metric - here I use the SP500 as basis for comparison due to time constraints, but the portfolio should be compared to an equivalent portfolio with constant weights.

# Thank you!

Contact:

- tracystrickel@gmail.com
- https://github.com/hypermoderndragon
- https://www.linkedin.com/in/tracy-strickel/