Predict2Optimize / WIDS Final Report

- **Krish Kumar Pal**

1. Introduction

Financial markets produce a large amount of data every day, but using this data to make reliable predictions is very difficult. Asset prices move due to many factors, and daily returns are highly noisy. Due to this, models that look good in theory often fail when they are tested properly over time. The project focuses on understanding this gap between theory and real-world behavior.

The main aim of this project is to understand how portfolio optimization behaves when prediction errors, rebalancing, and transaction costs are taken into account. Simple baseline models were studied first to understand how difficult financial prediction actually is. Walk-forward evaluation was used instead of random train-test splits to avoid look-ahead bias and unrealistic results.

The project starts with basic financial data analysis and return computation, then moves to baseline prediction models. After that, portfolio optimization methods such as equal-weight and Markowitz portfolios are studied. Finally, all the components are combined into a single backtesting framework where portfolios are run over time under realistic assumptions.

This report summarizes the theory learned, experiments performed, mistakes made during implementation, and the insights gained while working through the Predict2Optimize / WIDS program.

2. Financial Basics (Week 1)

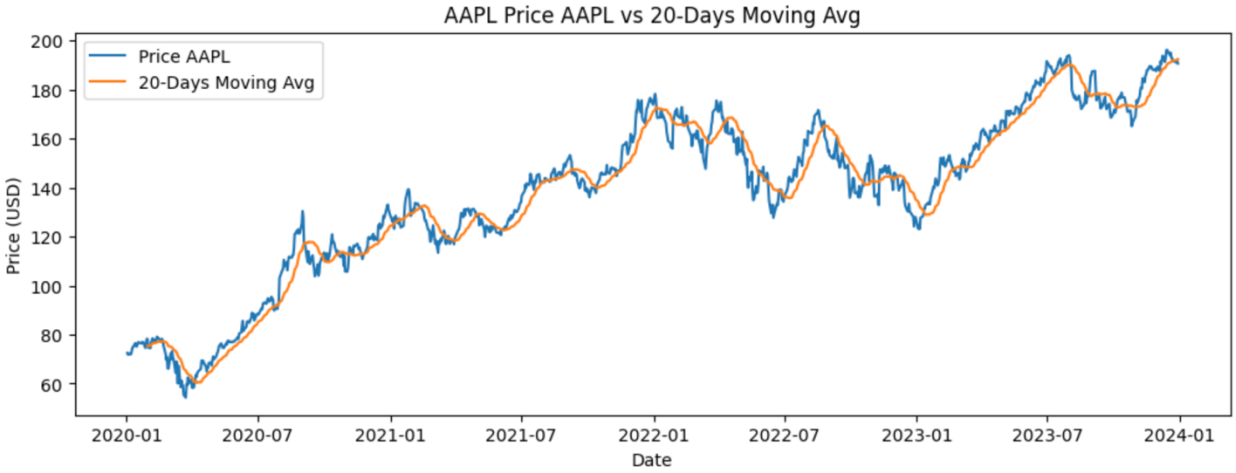
The first part of the project focused on understanding financial data and basic properties of asset returns. Historical price data was collected using Yahoo Finance through the yfinance library. Adjusted Close prices were used instead of raw close prices to account for stock splits and dividends.

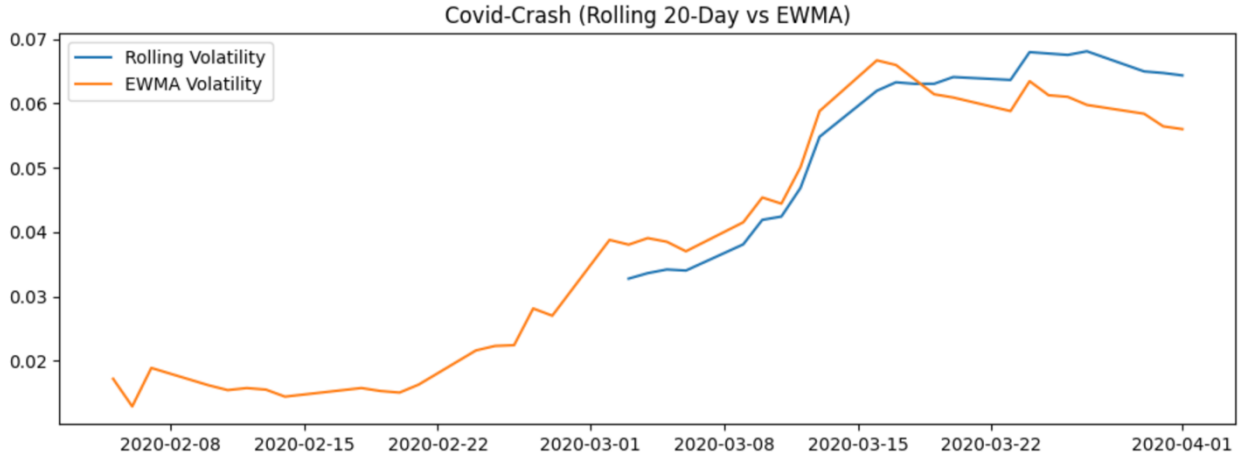
The assets used across different weeks included large-cap US stocks like AAPL, MSFT, GOOG, AMZN, and TSLA. Three different time horizons were studied: a long-term period from 2015 to 2024, and a medium-term period from 2020 to 2024, and yearly short-term period too which included volatile market phases such as the COVID period.

Prices were first converted into returns, since prices themselves are non-stationary, difficult to model. Both simple returns and log returns were computed. Log returns were mainly used for analysis because they have better statistical properties, easier to work with in time-series models.

rolling volatility was computed using the standard deviation of returns over different window sizes. Rolling mean and variance analysis was also performed to study how return distributions change over time. During periods of market stress, volatility was observed to increase sharply, while calm market periods showed relatively stable behavior.

From this analysis, it was observed that asset prices are non-stationary, while returns are approximately stationary over short horizons. Another important observation was volatility changes, where high-volatility periods tend to be followed by more high-volatility periods rather than returning immediately to normal levels


AAPL Price AAPL vs 20-Days Moving Avg
Price AAPL
20-Days Moving Avg
Price (USD)
200
180
160
140
120
100
80
60
2020-01
2020-07
2021-01
2021-07
2022-01
2022-07
2023-01
2023-07
2024-01
Date


Covid-Crash (Rolling 20-Day vs EWMA)
Rolling Volatility
EWMA Volatility
0.07
0.06
0.05
0.04
0.03
0.02
2020-02-08
2020-02-15
2020-02-22
2020-03-01
2020-03-08
2020-03-15
2020-03-22
2020-04-01

3. Baseline Prediction Models and Time-Series Evaluation (Week 2)

The goal of Week 2 was to understand how difficult financial return prediction actually is, before applying any complex models. Financial returns are known to be noisy, so it is important to first establish strong baseline models. These baselines help determine whether a model is learning any real signal or just fitting noise. Three baseline prediction models were implemented and compared.

The first model was a **zero-return predictor**, which always predicts the next day return to be zero. Although this model is extremely simple, it serves as an important check because daily stock returns typically have a mean close to zero.

The second model was a **rolling mean predictor**, where the next-day return is predicted using the average of recent past returns. Introduces basic time dependence without using any machine learning or optimization techniques.

The third model was an **Ordinary Least Squares (OLS) regression model**. Several return-based features were used here, including today's return, yesterday's return, 20-day rolling mean, 20-day rolling volatility, and 5-day momentum etc.

To evaluate these models correctly, **walk-forward (time-series) validation** was used instead of random train-test splits. At each step, models were trained only on past data and evaluated on future data, ensuring that no look-ahead bias was there. Mean Squared Error (MSE) was used as the primary evaluation metric, with RMSE considered too. The choice between MSE and RMSE did not affect model comparisons.

The results showed that all three models performed similarly. The zero-return predictor achieved an error close to that of the rolling mean and OLS models. This indicates that more complex models did not significantly outperform naïve baselines in this setting. These results highlight an important insight that adding more features or using more complex models does not guarantee better performance.

	Model	MSE	RMSE
0	Zero Predictor	0.000347	0.018617
1	Rolling Mean	0.000360	0.018985
2	OLS	0.000350	0.018704

4. Portfolio Optimization Methods (Week 4)

In Week 4, the focus shifted from predicting returns to constructing portfolios using optimization techniques. It becomes quite complex, involving math and models. The goal was to study how portfolio weights change when expected returns and risk are combined through optimization.

The simplest portfolio studied was the **equal-weight (1/N) portfolio**, where each asset is assigned the same weight. Serves as a strong and stable baseline. Despite its simplicity, it often performs well.

Mean–variance (Markowitz) portfolio optimization was implemented. In this framework, portfolio weights are chosen by balancing expected return and risk, where risk is measured using the covariance matrix of returns.

The **Global Minimum Variance Portfolio (GMVP)** was also studied. This portfolio ignores expected returns entirely and focuses only on minimizing portfolio variance.

During experimentation, it became clear that mean return estimates are highly noisy and unstable. Small changes in estimated returns often led to large and unrealistic changes in portfolio weights. This makes standard Markowitz optimization sensitive and difficult to use in practice.

To address this issue, a **robust version of Markowitz optimization** was implemented. Uncertainty in expected returns is penalized by removing extreme portfolio weights. This produces smoother and more stable allocations over the time.

Robust optimization provided a practical way to reduce sensitivity to noisy return values and is further used in the final backtesting stage.

$$
\begin{aligned}
\max_{w} \quad & \mu^\top w - \lambda w^\top \Sigma w \\
\text{s.t.} \quad & \mathbf{1}^\top w = 1 && \text{(fully invested)} \\
& w \geq \mathbf{0} && \text{(long only)} \\
& ||w||_1 \leq \gamma && \text{(leverage limit)} \\
\forall i \quad & |w_i| \leq u && \text{(max position limit)}
\end{aligned}
$$

5. Backtesting Framework and Final Strategy (Week 5)

In Week 5, all components developed in earlier weeks are combined into a single backtesting framework. The goal is to study what actually happens when portfolio optimization strategy is run over time.

A **walk-forward backtesting framework** was implemented. At each time step, only past data was used to make decisions to avoid look-ahead bias, and portfolio performance was evaluated on future returns.

At each rebalancing date, expected returns and the covariance matrix were estimated using a rolling historical window. These estimates were then passed to the optimization routine to compute the portfolio weights. Between this, portfolio weights were held constant.

Rebalancing was performed at a **monthly frequency**, rather than daily. Portfolio turnover was computed as the sum of absolute changes in portfolio weights at each step. Transaction costs (being proportional to this turnover) was evaluated and then deducted from portfolio wealth.

Portfolio wealth was updated over time using a full reinvestment assumption, where gains and losses are reinvested back into the portfolio.

```
Equal Weights Strategy:
Performance summary
--------------------
Total return:     24.92%
Avg daily return: 0.0009
Daily volatility: 0.0068

Robust Markowitz Strategy:
Performance summary
--------------------
Total return:     24.72%
Avg daily return: 0.0009
Daily volatility: 0.0069
```

Two strategies were evaluated using this framework:

1. An equal-weight (1/N) baseline portfolio
2. A robust Markowitz portfolio using convex optimization

By running both strategies through the same backtesting process, their performance, stability, and sensitivity were comparable. This framework allowed for evaluation of returns, portfolio turnover, weight stability etc.

This stage of the project highlighted that many strategies which appear complex in theory can perform poorly at practical settings.

Portfolio wealth over time for the equal-weight (1/N) portfolio and the robust Markowitz portfolio. Both strategies exhibit similar performance, with the robust strategy closely tracking the baseline.

7. Errors Faced and Key Learnings

During the project from Week 2 to Week 5, several mistakes were made. These errors were important in understanding how financial models behave in practice and why many ideas fail when implemented over time.

Week 2 – Prediction Errors

- Initially expected regression models to outperform simple baselines.
- Found that zero-return and rolling-mean predictors performed almost equally well compared to OLS.
- This showed that short-term financial returns are extremely noisy.
- Adding more features or complexity does not guarantee better prediction.

Learning:

Financial markets have a very low signal-to-noise ratio. Strong baselines are necessary before trusting any predictive model.

Week 4 – Optimization Issues

- Mean–variance optimization was very sensitive to small changes in mean estimates.
- Small estimation errors led to large and unstable portfolio weight changes.

Learning:

Mean estimates are unreliable. Robust optimization helps improve stability.

Week 5 – Backtesting and Implementation Errors

- Initial backtests produced unrealistic results such as complete wealth collapse.
- Errors were mainly due to incorrect handling of transaction costs and time indexing.
- Some plots initially causing look-ahead bias.

Fixes Applied:

- Correct walk-forward backtesting logic
- Proper turnover-based transaction cost modeling
- Time-consistent estimation and visualization
- Realistic wealth update rules

Overall Learning

- Most mistakes were not mathematical, but related to time, uncertainty, and evaluation.
- Robust and stable strategies are more valuable due to their focus smoother increase rather than explosive returns.