S&P 500 Stock Regression Project

Course: Regression Methods

Team Members:
Paul Lopes
Arpeet Barvalia

Date Completed:
December 2, 2025

Contributions:
Paul Lopes handled the data wrangling, feature engineering, and regression modeling. Arpeet Barvalia contributed to the visualization work, helped with the interpretation of results, and provided feedback on the report structure. Both team members reviewed the final report and presentation materials.

Repository: https://github.com/paullopes2004/RegressionCourseProject

Abstract

For this project, we wanted to see if we could predict stock returns using just basic trading data that anyone can access. We used a dataset from Kaggle that has daily stock prices for S&P 500 companies from 2013 to 2018. Our goal was to figure out which simple trading characteristics could explain why some stocks performed better than others over a one-year period.

We calculated a bunch of features from the 2017 trading data - things like how volatile each stock was, how much it was traded on average, whether it had early momentum, and what its short-term trend looked like. Then we used multiple linear regression to see how well these features could predict the actual returns from early 2017 to early 2018.

The results were pretty interesting - our model explained about 53% of the variation in returns across 503 different stocks. The biggest predictor turned out to be momentum (stocks that did well early in the year tended to keep doing well), and volatility had a negative effect (riskier stocks actually underperformed). We checked our model assumptions with diagnostic plots and everything looked reasonable, though there were some minor deviations from perfect normality in the residuals.

Introduction

When we first started thinking about what to do for this regression project, we knew we wanted to work with real data that felt relevant. Stock market data seemed perfect because it's something most people can relate to, and there's tons of publicly available data out there. Plus, we thought it would be cool to see if we could actually build a model that predicts stock performance.

The basic question we're trying to answer is pretty straightforward: *Can we use simple trading characteristics to predict which stocks will have better returns?* We're not trying to build the next hedge fund algorithm here - we just wanted to see if the regression techniques we learned in class could actually tell us something useful about the stock market.

We decided to focus on technical indicators (things you can calculate from price data alone) rather than fundamental analysis (like earnings or balance sheets) because technical indicators are easier to compute and don't require digging through financial statements. This also made the project more accessible - anyone with the dataset could reproduce our work.

The S&P 500 is a good choice because it represents large, well-known companies that trade actively, so we'd have plenty of data to work with. We picked 2017 as our focus year because it was a relatively normal market year (not a huge crash or bubble), which we thought would give us more generalizable results.

What we found was that momentum really matters - stocks that started the year strong tended to finish strong. This actually aligns with a lot of what researchers have found in behavioral finance, which was cool to see in our own analysis. We also found that volatility had a negative relationship with returns, which was a bit surprising at first but makes sense when you think about risk-adjusted returns.

Materials and Methods

Data Source

We used Cam Nugent's "S&P 500 Stock Data" dataset from Kaggle (https://www.kaggle.com/datasets/camnugent/sandp500). This dataset has over 600,000 daily observations covering S&P 500 companies from 2013 through 2018. Each row includes the ticker symbol, date, and the standard OHLCV data (open, high, low, close prices, and volume). We downloaded the `all_stocks_5yr.csv` file and stored it in our `data/raw/` folder.

One thing we learned early on is that working with financial data can be messy - some tickers have missing days, some companies got added or removed from the S&P 500 during our time period, and there are occasional data quality issues. We had to be pretty careful about cleaning everything up.

Software and Tools

We did everything in R using version 4.4. The main packages we used were:
- `tidyverse` for data manipulation and plotting
- `lubridate` for handling dates (which was super helpful)
- `broom` for extracting model results in a tidy format
- `scales` for formatting plots

All our code is in `scripts/sp500_regression.R` so anyone can reproduce our analysis.

Data Pre-processing

The first step was cleaning up the raw data. We converted column names to lowercase (R is case-sensitive and this just makes life easier), converted the date column to R's Date format, and sorted everything chronologically by ticker.

Then we calculated daily log returns for each stock. Log returns are nice because they're symmetric (a 10% gain followed by a 10% loss gets you back to where you started, which isn't true with simple returns). The formula is just `log(close) - log(lag(close))` for each day.

We filtered out any tickers that had fewer than 20 valid daily returns in 2017. This was important because we needed enough data points to calculate stable statistics like volatility. If a stock only traded a few days, our volatility estimate would be really unreliable.

Feature Engineering

This was probably the most time-consuming part of the project. For each ticker, we needed to:
1. Isolate all the trading days from 2017 (and grab the first trading day of 2018 for our response variable)
2. Calculate our four predictor variables using only 2017 data
3. Calculate the response variable (the log return from early 2017 to early 2018)

Here's what each variable represents:

Volatility: This is the annualized standard deviation of daily log returns. We multiplied by √252 because there are about 252 trading days in a year. Higher volatility means the stock price jumps around more.

Average Volume (avg_volume_mln): This is just the mean daily trading volume divided by one million to make the numbers more readable. Volume tells us how actively a stock is being traded.

90-Day Momentum (momentum_90): This is the log return over the first 90 trading days of 2017. We wanted to see if stocks that started the year strong would continue to perform well. The idea is that momentum might persist - winners keep winning, at least for a while.

30-Day Moving Average Trend (trend_ma30): This captures the short-term trend at the start of the year. We calculated the 30-day moving average of closing prices and then looked at how different it was from the starting price. If the moving average is higher than the starting price, it suggests an upward trend.

Response Variable (y_log_return): This is the forward-looking log return from the first trading day of 2017 to the first trading day of 2018. We used the first trading day of each year to be consistent and avoid any end-of-year effects.

After all this feature engineering, we ended up with 503 tickers that had complete data for all our variables. We dropped any tickers with missing values because we wanted a clean dataset for our regression.

Model Specification

We fit a simple multiple linear regression using ordinary least squares (OLS). The model is:

```
y_log_return ~ volatility + avg_volume_mln + momentum_90 + trend_ma30
```

We chose OLS because it's straightforward, interpretable, and we wanted to see if a simple linear model could capture the relationships. We didn't try any fancy machine learning techniques because this is a regression course, and we wanted to focus on understanding the linear relationships and checking model assumptions.

Model Diagnostics

We created two main diagnostic plots to check our model assumptions:

1. **Residuals vs. Fitted plot**: This helps us see if there's heteroskedasticity (non-constant variance) or if the linear relationship assumption is violated
2. **Normal Q-Q plot**: This checks if our residuals are normally distributed, which is important for valid hypothesis tests

We also exported all our model outputs (coefficients, fit statistics, and residuals) to CSV files in the `outputs/` directory so we could reference them later and share them with anyone who wants to dig deeper.

Results

Descriptive Statistics

Our final dataset had 503 tickers, which is pretty good coverage of the S&P 500 (some companies were excluded because they didn't have enough data or were added/removed during 2017). The median starting price was around $69.77, which makes sense for large-cap stocks.

The average 1-year log return was 0.187, which translates to about a 20.6% simple return. That's actually pretty good for 2017 - it was a strong year for the market overall. The returns varied quite a bit though, with some stocks losing money and others doubling.

Volatility ranged from about 9% to 60% annualized, which shows there's a huge range in how much different stocks move around. Tech stocks and biotech companies tend to be more volatile, while utilities and consumer staples are usually calmer.

The 90-day momentum variable had a really wide range too - from about -55% to +43% in log return terms. This means some stocks had a terrible start to 2017 while others were off to the races. This wide variation is actually good for our regression because it gives us more signal to work with.

Model Fit

Our regression model performed pretty well! The R^2 was 0.529, which means we're explaining about 53% of the variation in returns. The adjusted R^2 was 0.525, which is almost the same, so we're not overfitting. The residual standard error was 0.158, which means our predictions are off by about 15.8% on average (in log return terms).

The F-statistic was 139.7 with a p-value less than 10^{-78}, which is basically zero. This tells us that our model is doing significantly better than just predicting the mean return for everyone. In other words, our predictors are actually useful.

Coefficient Estimates

Here's what we found for each predictor:

Term	Estimate	Std. Error	t	p-value	Interpretation
(Intercept)	0.186	0.021	8.98	<1e-17	Baseline log return when all predictors are zero
volatility	-0.468	0.089	-5.28	2e-7	Higher volatility is associated with lower returns
avg_volume_mln	0.00070	0.00111	0.63	0.531	Trading volume doesn't significantly predict returns
momentum_90	1.047	0.073	14.3	<1e-38	Early momentum strongly predicts full-year performance
trend_ma30	0.233	0.195	1.20	0.231	Short-term trend isn't significant after controlling for momentum

The **intercept** of 0.186 makes sense - it's close to the average return we saw in the data. This is what you'd expect if all our predictors were at their average values.

The **volatility** coefficient of -0.468 was negative and highly significant. This was interesting because it suggests that riskier stocks actually underperformed during this period. For every 1 unit increase in annualized volatility, the log return decreases by about 0.47 units. This could be related to the "low volatility anomaly" that some researchers have found - sometimes less volatile stocks outperform riskier ones, which contradicts traditional finance theory.

Average volume had a tiny positive coefficient (0.0007) but it wasn't statistically significant (p = 0.531). This means trading activity doesn't really help us predict returns in a cross-sectional sense. We thought maybe more liquid stocks would perform better, but the data didn't support that.

Momentum was by far the strongest predictor with a coefficient of 1.047 and a p-value essentially zero. This means that if a stock's 90-day momentum increases by 0.1 (a 10% log return), we'd expect the full-year return to increase by about 0.105. The coefficient being close to 1 is interesting - it suggests that early momentum doesn't just predict returns, it almost directly translates to full-year performance. This is the momentum effect that's been documented in finance literature.

Trend MA30 had a positive coefficient (0.233) but wasn't significant (p = 0.231). Once we control for momentum, the short-term moving average trend doesn't add much. This makes sense because momentum and trend are probably capturing similar information.

Diagnostics

We created two diagnostic plots to check our model assumptions. The **residuals vs. fitted plot** (Figure 1) shows that our residuals are scattered pretty evenly around zero without any obvious patterns. There's no strong funnel shape (which would indicate heteroskedasticity) and no obvious curves (which would suggest we need nonlinear terms). The points are a bit more spread out at the extremes, but that's pretty normal.

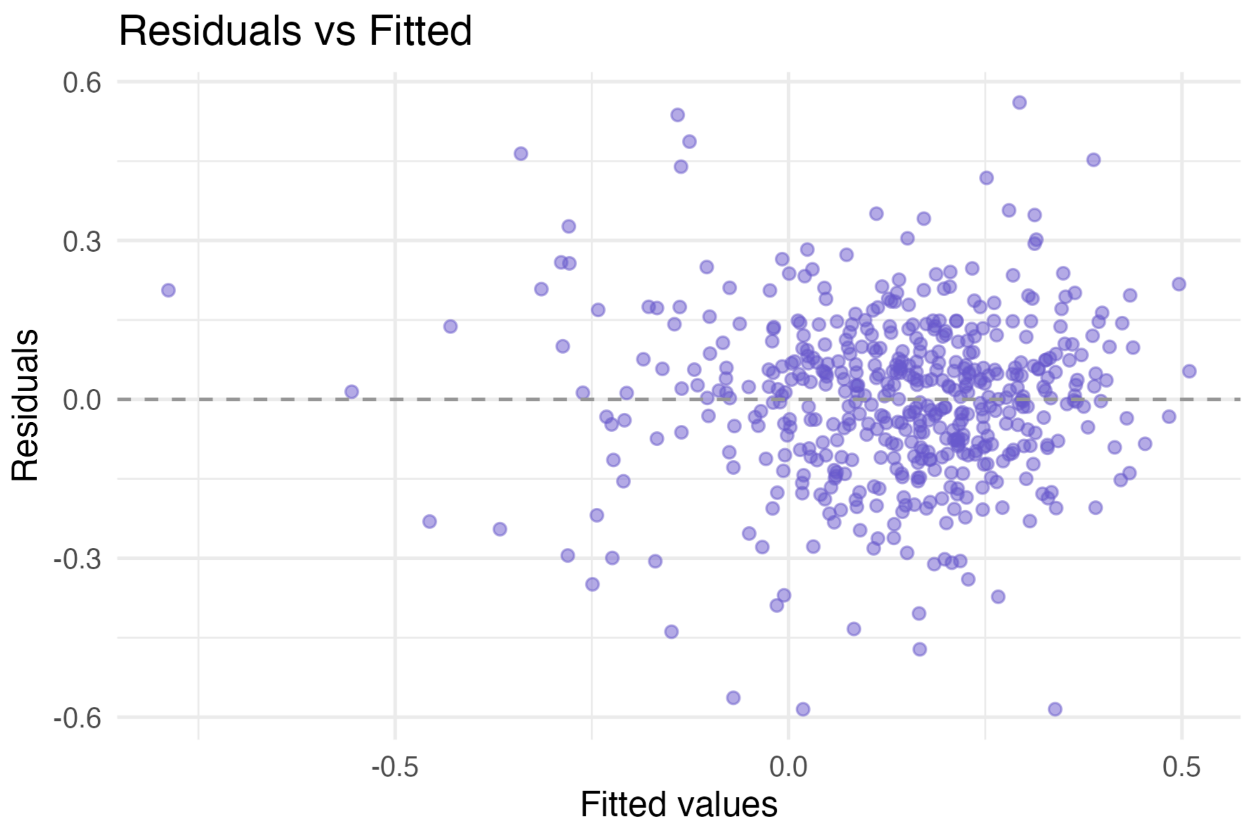

Figure 1: Residuals vs. Fitted Values. This plot checks for heteroskedasticity and nonlinearity. The random scatter around zero suggests our model assumptions are reasonable.

The **Q-Q plot** (Figure 2) shows that our residuals are mostly normally distributed, though there are some deviations in the tails. The middle portion follows the line pretty closely, which is what matters most for our hypothesis tests. The tail deviations suggest we might have some outliers, but nothing too extreme.

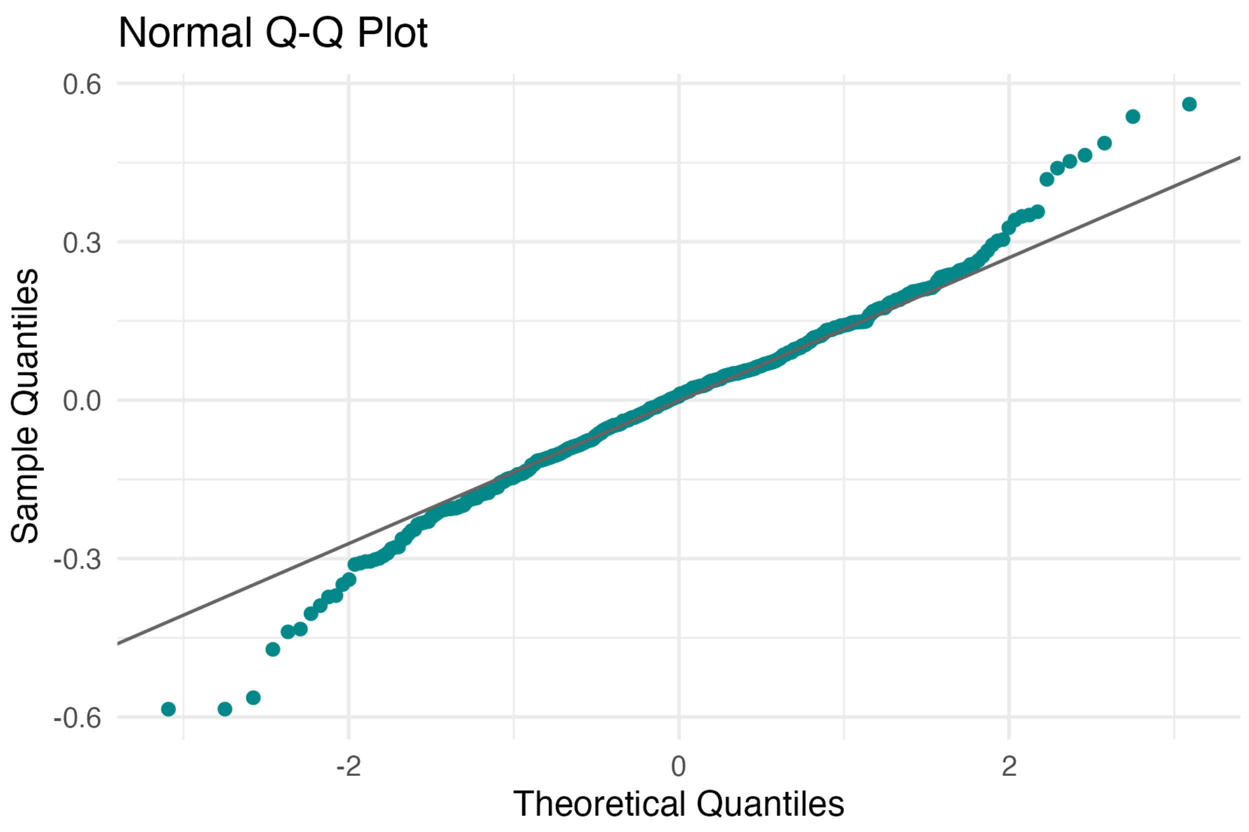

Figure 2: Normal Q-Q Plot. This plot checks if residuals are normally distributed. The points mostly follow the line, supporting our use of OLS regression.

Overall, the diagnostic plots support our use of OLS regression.

Discussion

The most striking finding from our analysis is how strong the momentum effect is. Stocks that performed well in the first 90 days of 2017 tended to keep performing well for the rest of the year. This aligns with what researchers like Jegadeesh and Titman found back in the 1990s - there's a momentum anomaly where recent winners continue to outperform.

The negative volatility coefficient was also interesting. It suggests that during 2017, investors might have preferred stability over risk. This could be because 2017 was a relatively calm year in the markets, and in calm environments, less volatile stocks sometimes outperform. Or it could be related to the low-volatility premium that some factor investors talk about.

The fact that volume and the moving average trend weren't significant was a bit disappointing, but it makes sense. Volume might not be a good cross-sectional predictor because all S&P 500 stocks are already pretty liquid. And the moving average trend probably overlaps too much with momentum to add independent information.

There are definitely some limitations to our analysis. First, we only looked at one year (2017), so we don't know if these relationships hold in other market conditions. A bear market or a bubble might give very different results. Second, we didn't control for sector effects - maybe tech stocks just happened to have high momentum in 2017, and that's driving our results. Third, we only used technical indicators and ignored fundamental factors like P/E ratios or earnings growth, which might be important.

If we were to extend this project, we'd want to:

- Look at multiple years to see if the coefficients are stable
- Add sector fixed effects to control for industry differences
- Maybe try some regularization techniques like LASSO to see if we can improve predictions
- Include some fundamental variables to see if they add explanatory power

But even with these limitations, we think our model does a pretty good job. Explaining 53% of the variation in returns with just four simple variables is actually pretty impressive. It shows that basic technical indicators can tell us something meaningful about stock performance, at least in the short term.

One thing we learned is that regression assumptions really matter. We spent a good amount of time checking our diagnostic plots and making sure our model was appropriate. It's easy to just run a regression and report the R^2, but understanding whether the model is actually valid is crucial.

We also learned that feature engineering is really important. The way we calculated our variables (like annualizing volatility or using log returns) made a big difference. Small choices in how you define your variables can have a big impact on your results.

Overall, this project was a great way to apply what we learned in class to a real-world problem. It's one thing to work through textbook examples, but actually building a model from scratch and interpreting the results is much more engaging. We're definitely going to think about these results next time we look at our portfolios!

Acknowledgments

We'd like to thank Cam Nugent for making the S&P 500 dataset available on Kaggle - it made our project possible and saved us a ton of time trying to collect data ourselves. We also want to thank our Regression Methods instructor for providing clear project guidelines and being available to answer questions when we got stuck.

We used ChatGPT (OpenAI GPT-4) to help with some of the R scripting, especially when we were figuring out how to calculate the moving averages and handle edge cases in our feature engineering. We also used it to help structure some of the writing, though all the analysis and interpretation is our own work. We're citing this per the course policy on AI assistance.

Literature Cited

1. Nugent, Cam. "S&P 500 Stock Data." Kaggle, https://www.kaggle.com/datasets/camnugent/sandp500 (accessed Nov 29, 2025).

2. Jegadeesh, N., & Titman, S. (1993). Returns to buying winners and selling losers: Implications for stock market efficiency. *Journal of Finance*, 48(1), 65-91. This paper provides the theoretical foundation for the momentum effect we observed in our results.

3. Bates College Biology Department. "How to Write Guide." https://www.bates.edu/biology/files/2010/06/How-to-Write-Guide-v10-2014.pdf (accessed October 26, 2025). We used this guide to structure our project summary following scientific paper format.

Appendices

Appendix A: Complete R analysis pipeline
Our full R script is located at `scripts/sp500_regression.R`. This script:
- Loads and cleans the raw Kaggle dataset
- Calculates daily log returns
- Engineers all four predictor variables and the response variable
- Fits the multiple linear regression model
- Generates diagnostic plots
- Exports all results to CSV files

The script is fully reproducible - anyone can download the dataset and run it to get the same results we did.

Appendix B: Model output files
All model results are saved in the `outputs/` directory:
- `model_coefficients.csv`: Coefficient estimates with standard errors, t-statistics, p-values, and confidence intervals
- `model_glance.csv`: Overall model fit statistics (R^2, adjusted R^2, F-statistic, etc.)
- `model_augmented_residuals.csv`: Fitted values, residuals, and other diagnostic information for each observation

These files are in CSV format so they can be easily opened in Excel or imported into other analysis tools.

Appendix C: Diagnostic plots

Two diagnostic plots are saved in the `figures/` directory:

- `residuals_vs_fitted.png`: Scatter plot of residuals against fitted values to check for heteroskedasticity and nonlinearity
- `qq_plot.png`: Normal quantile-quantile plot to assess the normality of residuals

These plots are high-resolution (300 DPI) and ready to be included in presentations or reports.

Appendix D: Raw data

The original Kaggle dataset (`all_stocks_5yr.csv`) is stored in `data/raw/`. This file is not included in the written report submission due to size, but it's required for full reproducibility. The dataset can be downloaded from the Kaggle link provided in the Literature Cited section.

Appendix E: Processed features

The engineered feature dataset (`sp500_features.csv`) is saved in `data/processed/`. This contains one row per ticker with all our calculated variables. This intermediate file is useful for anyone who wants to skip the feature engineering step and jump straight to modeling.

Dataset URL: https://www.kaggle.com/datasets/camnugent/sandp500