Predicting Post-Earnings Stock Price Direction: A Quantitative Approach

Carl Nordahl

June 16, 2025

Lund University
LINC-STEM

Abstract

This report details a quantitative research project focused on predicting the short-term direction of stock price movements following corporate earnings announcements. Leveraging a diverse dataset comprising earnings surprise metrics and market-wide financial indicators, a Logistic Regression model was developed and rigorously tuned. The project showcases a comprehensive data science pipeline, from data acquisition and advanced feature engineering to model training, hyperparameter optimization, and performance evaluation using key classification metrics.

Introduction

Predicting stock price movements is a central challenge in finance, especially around significant events like corporate earnings reports. These announcements often trigger rapid short-term changes in stock values. This project focuses on forecasting the direction of a stock's price movement over the three days following its earnings release. The main goal is to determine if earnings surprise data, combined with other relevant financial and market metrics, can reliably predict these post-earnings shifts.

To achieve this, a machine learning pipeline was built using historical earnings data from kaggle, supplemented with company-specific and market-wide financial data from the YFinance API. A logistic regression model was used to make the predictions. The model was optimized using GridSearchCV, and the evaluation metrics used were accuracy, F1-score, precision, recall, and the Area Under the Receiver Operating Characteristic Curve (AUC).

Theory

Predicting the movement of a stock's price is extremely difficult, as it involves predicting, or guessing, things that are in the future. In the short term this becomes especially difficult, since there are an endless amount of volatile, moving parts that affect a stock's price. One of the more fundamental of these is the EPS – earnings per share. It is fundamental because it relates a the company's earnings, which in the long term is what the company is worth, to the stock price, which decides the value of the company.

Companies release their earnings report each quarter. The report gives current and future investors insight into the companies' earnings and overall financial health. Naturally, an earnings report can have a big impact on the pricing of that company's stock. More specifically, an earnings report that contradicts investor expectations can have a big impact on the stock price. Since people and investors don't like uncertainty, a great effort is put in to estimate the outcomes of these earnings reports. These estimates are made by various analysts and can give an indication of the expectations of the market on a specific company. The difference between the estimated and reported EPS is called the EPS surprise.

Recent years have seen a surge in people using machine learning models to try and predict market movements, with varying success. These models are designed for handling large amounts of data, and seeing patterns in that data. In the financial world, one use case is to predict the movement of stock prices. A common machine learning model used for classification tasks is a logistic regression model. The model fits a function that can take a large amount of dependent or independent variables as input, and outputs a probability between zero and one for a certain event – like a stock's price rising. The logistic regression model is easy and efficient to implement and train, although there are setbacks. The major setback is that it assumes linearity between variables and the log-odds of the event it is predicting:

$$\log\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = \beta_o + \beta_1 X_1 + ... + \beta_n X_n.$$

This can result in a logistic regression model failing to identify some more complex relations between input and output variables. One way to overcome this is to manually transform the features before training the model, to help it uncover nonlinear relationships. In the case of predicting whether a stock price moves up or down, a logistic regression is a fine choice, because of it's simplicity to use and implement.

Project Goal

The goal of this project is to construct a logistic regression model that can successfully predict the three-day direction (up or down) of the stock price following the release of an earnings report. In order to do this, a functioning data pipeline will be built and various evaluation metrics will be used.

Method

Data Acquisition and Pre-processing

The earnings report data and the stock price data following the reports were downloaded from Kaggle. The relevant content for the project was: Company name and ticker, date of earnings report release, estimated and reported EPS, and price data for the days immediately following the report. The EPS surprise information was then calculated using:

$$\text{Surprise} = \frac{\text{Reported} - \text{Estimated}}{|\text{Estimated}|}.$$

The rest of the data were downloaded using the YFinance API in Python. This included both stock specific and market wide relevant information. The stock specific data included: the momentum of the stock price and the stock return volatility. The momentum was defined as the stock's ten day return, using the day before the earnings report as the final day. The return volatility was calculated by taking the standard deviation of the stock's return in the 20 days before the day before the earnings report release. The first market metric used was the market return the day before earnings report release. It was fetched using the ^GSPC ticker. The second market metric used was the VIX-index, a volatility index based on the S&P 500. This was fetched from YFinance using the ^VIX ticker.

After acquiring all the data, the rows containing NaN values were removed, to prepare the data for further engineering.

Feature Engineering

To reduce the noise in the surprise data, it was log-scaled. For negative surprises, the absolute value was log-scaled and the sign was later added. To help the model uncover non-linear relationships, interaction features and polynomial features were added. The interaction features were created by multiplying some of the features together, for example: surprise scaled x momentum. The polynomial features added were simply squares of the original data.

The final derived feature added was surprise direction, represented simply by a 1 for up, or a 0 for down.

Feature Importance and Selection

To decide which data should be used to train the model, the P- and F-values were calculated using the f_classif method from the sklearn.feature_selection package.

Table 1: F- and p-values.

Feature	F value	p value
Surprise scaled	$2.99 \cdot 10^{2}$	$6.79 \cdot 10^{-65}$
Surprise Direction	$2.80 \cdot 10^{2}$	$4.98 \cdot 10^{-61}$
Surprise_scaled_x_Volatility	$1.08 \cdot 10^{2}$	$4.81 \cdot 10^{-25}$
Market Return	$2.66 \cdot 10^{1}$	$2.54 \cdot 10^{-7}$
Surprise	$2.46 \cdot 10^{1}$	$7.32 \cdot 10^{-7}$
Volatility_x_Market_Return	$2.24 \cdot 10^{1}$	$2.33 \cdot 10^{-6}$
Relative Surprise	$1.97 \cdot 10^{1}$	$9.49 \cdot 10^{-6}$
Reported EPS	$1.44 \cdot 10^{1}$	$1.48 \cdot 10^{-4}$
Market_Return_sq	$1.17 \cdot 10^{1}$	$6.46 \cdot 10^{-4}$
Momentum	$3.10 \cdot 10^{0}$	$7.85 \cdot 10^{-2}$
Surprise_scaled_x_Market_Return	$1.86 \cdot 10^{0}$	$1.72 \cdot 10^{-1}$
Volatility_sq	$1.83 \cdot 10^{0}$	$1.76 \cdot 10^{-1}$
Volatility_x_Momentum	$1.76 \cdot 10^{0}$	$1.84 \cdot 10^{-1}$
Volatility	$1.10 \cdot 10^{0}$	$2.95 \cdot 10^{-1}$
EPS Estimate	$9.84 \cdot 10^{-1}$	$3.21 \cdot 10^{-1}$
Surprise_scaled_sq	$5.14 \cdot 10^{-1}$	$4.73 \cdot 10^{-1}$
Momentum_sq	$1.83 \cdot 10^{-1}$	$6.69 \cdot 10^{-1}$
Market Volatility	$1.56 \cdot 10^{-1}$	$6.93 \cdot 10^{-1}$
Surprise_scaled_x_Momentum	$4.29 \cdot 10^{-3}$	$9.48 \cdot 10^{-1}$

Above is a table summarizing the results from the testing. A low p-value and high F-value generally represents statistical significance. 0.05 is usually the bar for what's considered a good p-value. Using the results from testing a final set of eight features were selected for training. The selected features were: Surprise Scaled, Surprise Direction, Surprise Scaled x Volatility, Market Return, Volatility x Market Return, Relative Surprise, Reported EPS, and squared Market Return.

Model Development

After cleaning and preparing the data the model was developed. The data was split into 75% training

and 25% testing, and the numbers were scaled using StandardScaler from sklearn. To determine the optimal parameter values, a grid search method was implemented, testing all possible combinations from a given set of parameters and optimizing for accuracy. The resulting model was later used on the test data, in order to review it's performance.

Results

The model achieved an accuracy of 64.3%. The table below shows some other key metrics to determine model performance. All of precision, recall and f1-score were a bit higher for the up-class than they were for the down-class. But overall, the results were even.

Table 2: Classification report.

index	precision	recall	f1-score	support
down	0.641	0.576	0.607	552
up	0.645	0.705	0.674	603
macro avg	0.643	0.640	0.640	1155
weighted avg	0.643	0.643	0.642	1155

Figure 1 shows a confusion matrix of the models predictions.

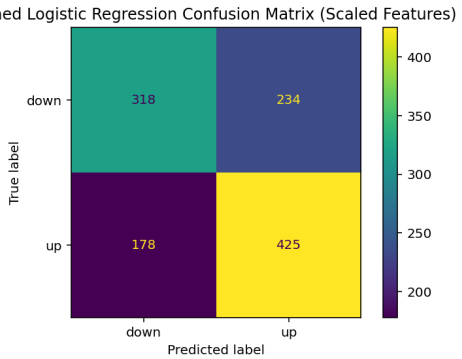

Figure 1: Confusion matrix.

Figure 2 shows the final coefficient values used by the model. See theory for reminder of how these coefficients affect the model.

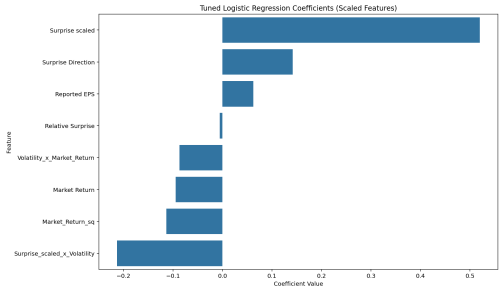

Figure 2: Coefficients used by the model.

The features with a positive (negative) coefficient have a positive (negative) correlation with the log-probability of the stock price rising.

Analysis

The model's accuracy of 64.3% indicate a moderate level of predictive power in determining the short-term direction of stock price movements following earnings reports.

Looking at the classification report, the model performs slightly better on the "up" class (precision: 0.645, recall: 0.705) compared to the "down" class (precision: 0.641, recall: 0.576). This imbalance implies that the model is more likely to correctly identify rising stock prices than falling ones. This could be due to market asymmetry where positive surprises tend to have more consistent price effects, or due to an imbalance in the dataset itself.

The confusion matrix further illustrates that the model tends to misclassify some of the "down" cases as "up", which could be bettered by further tuning or class-weight balancing. Feature importance, represented by the coefficients in the logistic model, indicates that features like EPS surprise (scaled), surprise direction, and market return played major roles in influencing predictions. Interaction features such as "Surprise Scaled x Volatility" also demonstrated strong significance, validating the inclusion of engineered features to capture non-linear patterns.

Conclusion

This project demonstrated the implementation of a logistic regression model to predict the direction of stock price movement three days after corporate earnings announcements. A combination of earnings surprise data, market-wide indicators, and some feature engineering was used, which led to the model achieving a 64.3% test accuracy.

This result is notable given the inherent unpredictability of short-term stock movements. It shows that quantitative features—particularly earnings surprises and volatility-adjusted indicators—carry predictive value. The use of feature scaling, interaction terms, and statistical selection methods contributed to the overall model performance.

However, the model is relatively simple and linear. It cannot capture many of the complex non-linear dynamics that exist in real market behavior. Logistic regression is a good starting point, but it leaves room for future improvement using more advanced machine learning models.

One possible problem with this research is the case of market efficiency. How fast will the EPS surprise information be embedded in the stock price? If the effect is instant than it might not be possible to use the correlation to make monetary gains. Though this report didn't explore this, is would be an interesting path for further exploration on the subject. A possible route to verify the model's real world usability would be to change it to predict the returns between one and three days after the earnings release. If the model can still make meaningful predictions, then there is some very real value to be found in further developing and implementing it in a real trading strategy.

In conclusion, the research confirms that post-earnings stock direction can be partially predicted using quantitative methods. The findings support the practical value of integrating financial metrics with data science tools, while also emphasizing the limitations of simplistic models in a highly dynamic environment.