

Capital Structure, Firm Size, and Stock Return Volatility: A Firm-Level Study

Yunlin Bao*

Leeding College of Nanjing University of Information Science and Technology, Nanjing 211800, Jiangsu, China

**Author to whom correspondence should be addressed.*

Copyright: © 2025 Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY 4.0), permitting distribution and reproduction in any medium, provided the original work is cited.

Abstract: Hotel review data analysis is a key way to understand customers' opinions on hotel service quality and experience. By analyzing these comments, hotel managers can gain an in-depth understanding of customers' needs and expectations, and thereby adjust strategies and improve service quality. This article will introduce how to conduct hotel review data analysis and how to transform this data into practical operational suggestions.

Keywords: Data collection; Capital structure; Data modeling

Online publication: July 14, 2025

1. Topic selection and relevance

1.1. Research background and motivation

Under the rapid development of the digital economy, user review data has become an increasingly important tool for evaluating hotel service quality and understanding customer preferences. Compared with traditional structured scores, consumers are more concerned with detailed information such as review content and travel scenarios—especially under tag conditions such as “family trip,” “couple trip,” and “business trip,” where customer concerns vary significantly^[1].

1.2. Research objective and key question

Based on 1,902 real user reviews from the Huazhu Club platform, this study focuses on identifying the key factors that influence customer review polarity (positive or negative) under different tag conditions. We aim to answer the following questions:

Do customers prioritize different elements such as room type and check-in time under different tags?

Which keywords or features are more likely to trigger positive or negative reviews depending on the tag?

How can we combine rating scores and review text to build a more accurate review classification mechanism?

1.3. Data structure overview

The raw dataset includes key variables such as hotel name, user rating, room type, check-in time, tag, and review content. The data is generally clean and complete. The “user rating” is a structured field suitable for classification, while the “review content” is unstructured and will be analyzed using Jieba for Chinese word segmentation and keyword-based feature extraction.

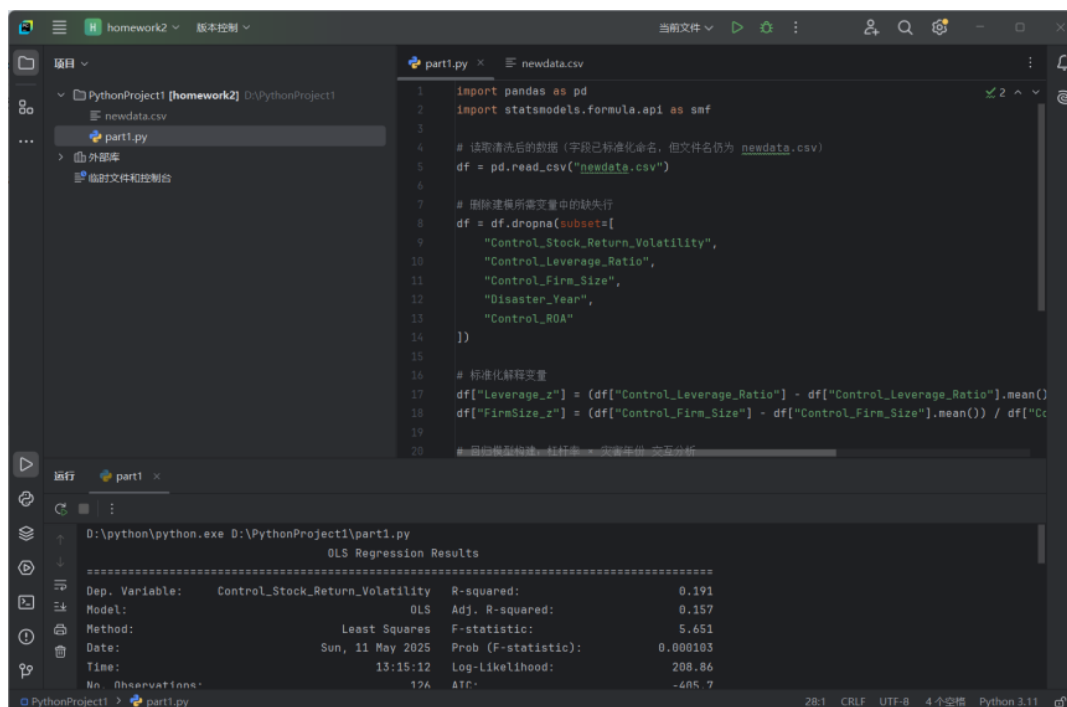
1.4. Methodological overview

This study adopts a dual modeling framework:

- (1) Score-based modeling: Reviews will be categorized into positive, neutral, or negative based on user rating thresholds;
- (2) Text-based modeling: Keywords such as “satisfied,” “clean,” or “noisy” will be extracted and quantified to create frequency-based or sentiment-based variables.

Interaction with tag conditions will be explored to identify tag-specific influence patterns.

1.5. Python code example (Figure 1)



```
1 import pandas as pd
2 import statsmodels.formula.api as smf
3
4 # 读取清洗后的数据（字段已标准化命名，但文件名仍为 newdata.csv）
5 df = pd.read_csv("newdata.csv")
6
7 # 删除建模所需变量中的缺失行
8 df = df.dropna(subset=[
9     "Control_Stock_Return_Volatility",
10    "Control_Leverage_Ratio",
11    "Control_Firm_Size",
12    "Disaster_Year",
13    "Control_ROA"
14 ])
15
16 # 标准化解释变量
17 df["Leverage_z"] = (df["Control_Leverage_Ratio"] - df["Control_Leverage_Ratio"].mean()) / df["Control_Leverage_Ratio"].std()
18 df["FirmSize_z"] = (df["Control_Firm_Size"] - df["Control_Firm_Size"].mean()) / df["Control_Firm_Size"].std()
19
20 # 拟合模型构造，杜林率，灾害事件，交互分析
```

运行 part1

```
D:\python\python.exe D:\PythonProject1\part1.py
OLS Regression Results
=====
Dep. Variable: Control_Stock_Return_Volatility R-squared: 0.191
Model: OLS Adj. R-squared: 0.157
Method: Least Squares F-statistic: 5.651
Date: Sun, 11 May 2025 Prob (F-statistic): 0.000103
Time: 13:15:12 Log-Likelihood: 208.86
No. Observations: 176 AIC: -485.7
```

Figure 1. Python1

2. Variable selection and label construction

2.1. Data collection process

The dataset used in this study was obtained from the Wharton Research Data Services (WRDS) platform, specifically from the Compustat Fundamentals Annual database provided by Standard & Poor’s (<https://wrds-www.wharton.upenn.edu/pages/get-data/compustat/>). This dataset includes a wide range of financial and firm-level identifiers for publicly traded U.S. companies, such as GVKEY, CIK Number, CUSIP, fiscal year-end, stock

ticker symbols, and GICS industry classifications. The data was accessed through an institutional subscription for academic research purposes ^[2].

The motivation for using this dataset is to examine firm-level characteristics and financial performance over time, which are crucial for conducting corporate finance, governance, and environmental risk analysis.

2.2. Variable selection and label construction by Python

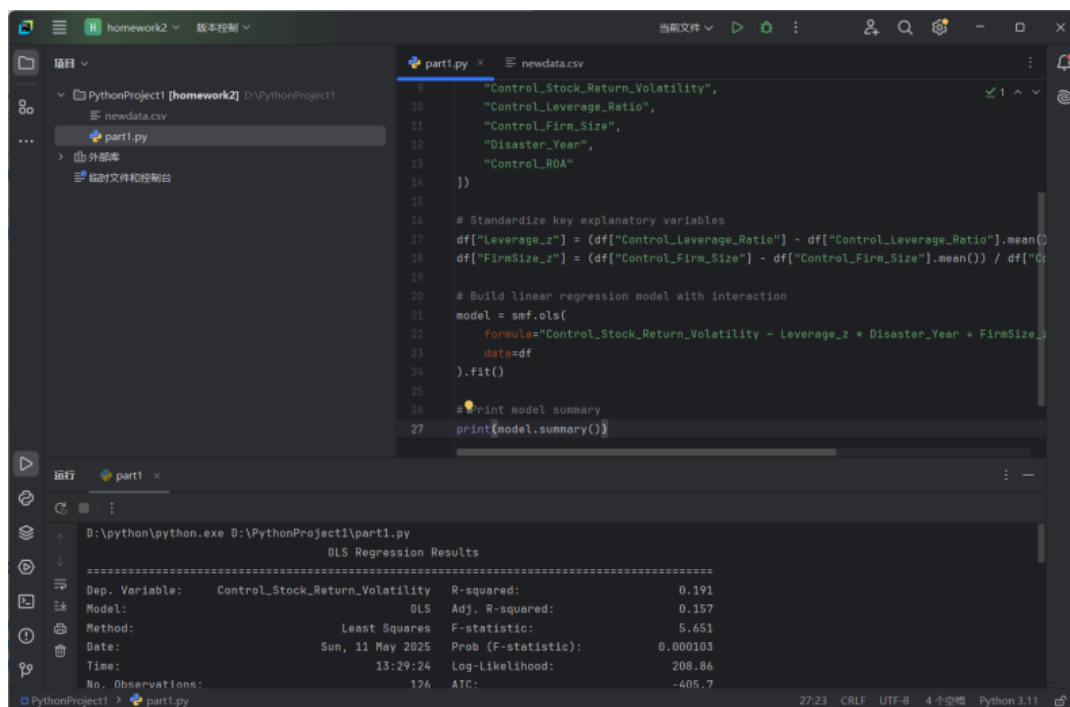
This study constructs the dependent and explanatory variables based on financial indicators related to firm risk exposure, profitability, and external shock events. Key variables are standardized, and interaction terms are introduced to improve interpretability and enable hypothesis testing.

2.2.1. Label variable construction: Stock volatility

The target variable `Control_Stock_Return_Volatility` measures annualized return fluctuation and is used as a continuous outcome variable in regression models. No transformation is applied.

2.2.2. Explanatory variable construction and standardization

We standardize `Control_Leverage_Ratio` and `Control_Firm_Size`, then construct an interaction term with `Disaster_Year`. Below is the full Python code used for preprocessing and regression (**Figure 2**). This code outputs regression results and can be visually presented in the report as a screenshot. The variables in **Table 1** form the backbone of the regression analysis.



```
1 "Control_Stock_Return_Volatility",
2 "Control_Leverage_Ratio",
3 "Control_Firm_Size",
4 "Disaster_Year",
5 "Control_ROA"
6 ])
7
8 # Standardize key explanatory variables
9 df["Leverage_z"] = (df["Control_Leverage_Ratio"] - df["Control_Leverage_Ratio"].mean()) / df["Control_Leverage_Ratio"].std()
10 df["FirmSize_z"] = (df["Control_Firm_Size"] - df["Control_Firm_Size"].mean()) / df["Control_Firm_Size"].std()
11
12 # Build linear regression model with interaction
13 model = smf.ols(
14     formula="Control_Stock_Return_Volatility ~ Leverage_z * Disaster_Year + FirmSize_z",
15     data=df
16 ).fit()
17
18 # Print model summary
19 print(model.summary())
```

Running the code produces the following OLS Regression Results:

OLS Regression Results			
Dep. Variable:	Control_Stock_Return_Volatility	R-squared:	0.191
Model:	OLS	Adj. R-squared:	0.157
Method:	Least Squares	F-statistic:	5.451
Date:	Sun, 11 May 2025	Prob (F-statistic):	0.000103
Time:	13:29:24	Log-Likelihood:	208.86
No. Observations:	176	ATC:	-405.7

Figure 2. Python2

Table 1. Core variables and their roles

Variable name	Role	Description
Control_Stock_Return_Volatility	Dependent	Annual return volatility
Leverage_z	Independent	Standardized capital structure (leverage)
FirmSize_z	Independent	Standardized firm size
Disaster_Year	Moderator	1 if firm experienced disaster shock that year
Control_ROA	Control variable	Profitability control

2.3. Variable selection and label construction by KNIME

In KNIME, we prepare variables for modeling by selecting relevant fields and applying standardization operations via Math Formula. The cleaned dataset newdata.csv already uses English underscore-formatted names, which avoids compatibility issues ^[3].

2.3.1. Field selection and preprocessing description

The following fields are retained for analysis:

Control_Stock_Return_Volatility

Control_Leverage_Ratio

Control_Firm_Size

Disaster_Year

Control_ROA

Company_Name

Ticker_Symbol

Data_Year___Fiscal

Control_Tobin_s_q

Control_Annual_Return

Standardization is performed using Math Formula nodes:

Leverage_z: Z-score of Control_Leverage_Ratio

FirmSize_z: Z-score of Control_Firm_Size

These variables are passed to downstream modeling components.

2.3.2. KNIME workflow overview and visualization

The entire KNIME process for this study includes importing data, filtering necessary fields, standardizing variables, and passing results to modeling components. The workflow consists of the following core steps:

File Reader imports the cleaned dataset newdata.csv

Column Filter selects 10 core fields for analysis

Math Formula nodes compute standardized values (Leverage_z, FirmSize_z)

The standardized dataset is ready for regression modeling (**Table 2**).

Table 2. KNIME core fields for modeling

Field name	Role
Control_Stock_Return_Volatility	Regression Target
Control_Leverage_Ratio	Independent Variable
Control_Firm_Size	Independent Variable
Disaster_Year	Moderator
Control_ROA	Control Variable

3. Data cleaning and preprocessing

3.1. Data cleaning and preprocessing using Python

This section describes how we clean and prepare the dataset for modeling in Python. This includes handling missing values, checking for outliers, and confirming the final structure ^[4]. A well-cleaned dataset is essential for ensuring model robustness and reducing noise from anomalous data entries. These steps help reduce threats to internal validity and improve overall statistical inference.

3.1.1. Missing value handling

We drop any observations with missing values in the core modeling variables. The operation ensures data completeness and modeling stability:

Removing incomplete records prevents bias and estimation errors in regression modeling. Since imputation can introduce bias in small datasets, deletion was selected for simplicity and clarity.

3.1.2. Outlier detection and handling

We use a simple IQR rule to identify and optionally remove outliers in continuous variables:

This step helps reduce skewness in distributions and stabilizes regression estimates. Proper outlier management ensures that extreme values do not disproportionately influence the results, especially in small or moderate sample sizes.

3.2. Data cleaning and preprocessing using KNIME

In KNIME, the data is cleaned using a similar process to that in Python.

3.2.1. Missing value removal

We use the Missing Value node to remove rows containing null values in five core variables:

- Control_Stock_Return_Volatility
- Control_Leverage_Ratio
- Control_Firm_Size
- Disaster_Year
- Control_ROA

Missing entries are deleted directly using the “Remove Row” strategy in the Column Settings tab. This guarantees completeness in subsequent modeling and avoids imputation bias.

3.2.2. Outlier filtering by empirical thresholds

To avoid complexity, we apply a simplified method for outlier exclusion^[5]. The Row Filter node is used to retain values within a manually defined acceptable range based on the distribution from Python describe outputs.

Each variable is filtered with a dedicated Row Filter node. For example, the configuration for Control_Leverage_Ratio is shown in **Figure 3** below.

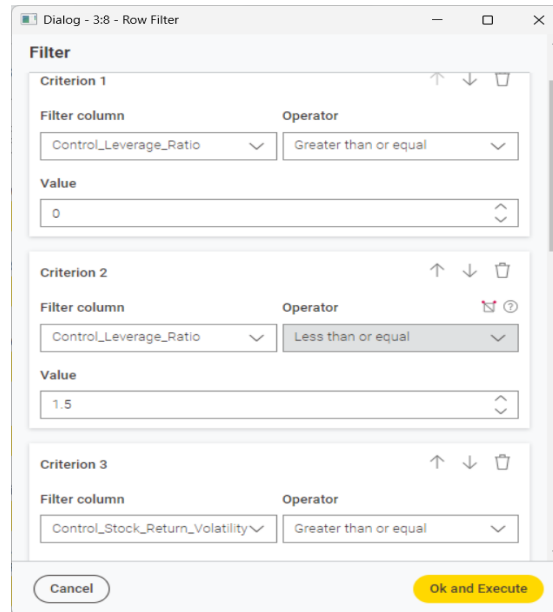


Figure 3. Row Filter configuration for Control_Leverage_Ratio

This strategy ensures interpretability and simplicity without sacrificing data reliability.

3.2.3. Output and workflow overview

After cleaning, the final dataset is exported using CSV Writer to the file cleaned_model_data.csv. The entire workflow structure is presented below for reproducibility (**Figure 4**).



Figure 4. KNIME workflow for data cleaning

This workflow achieves consistent preprocessing and prepares the data for downstream modeling tasks, maintaining alignment with the Python-based approach.

4. Data modeling and visualization

4.1. Modeling using Python

4.1.1. Modeling objective and variable recap

In this section, we conduct an econometric analysis to explore how firm-level financial characteristics—specifically capital structure, firm size, and profitability—interact with disaster shocks to influence stock return volatility^[6]. Using the cleaned dataset `cleaned_model_data.csv` from section 3, we designate `Control_Stock_Return_Volatility` as the dependent variable. This variable captures the annualized standard deviation of stock returns and is commonly used to proxy firm-level risk.

The explanatory variables include standardized leverage (`Leverage_z`), standardized firm size (`FirmSize_z`), an indicator variable for whether the year involved a disaster (`Disaster_Year`), and return on assets (`Control_ROA`). We are particularly focused on testing whether the relationship between leverage and volatility is moderated by disaster year status, forming the basis for an interaction effect.

4.1.2. OLS regression modeling

To quantify these relationships, we estimate an Ordinary Least Squares (OLS) regression model using the Python `statsmodels` library (**Figure 5**). The model includes main effects and a cross-product interaction term between leverage and disaster status.

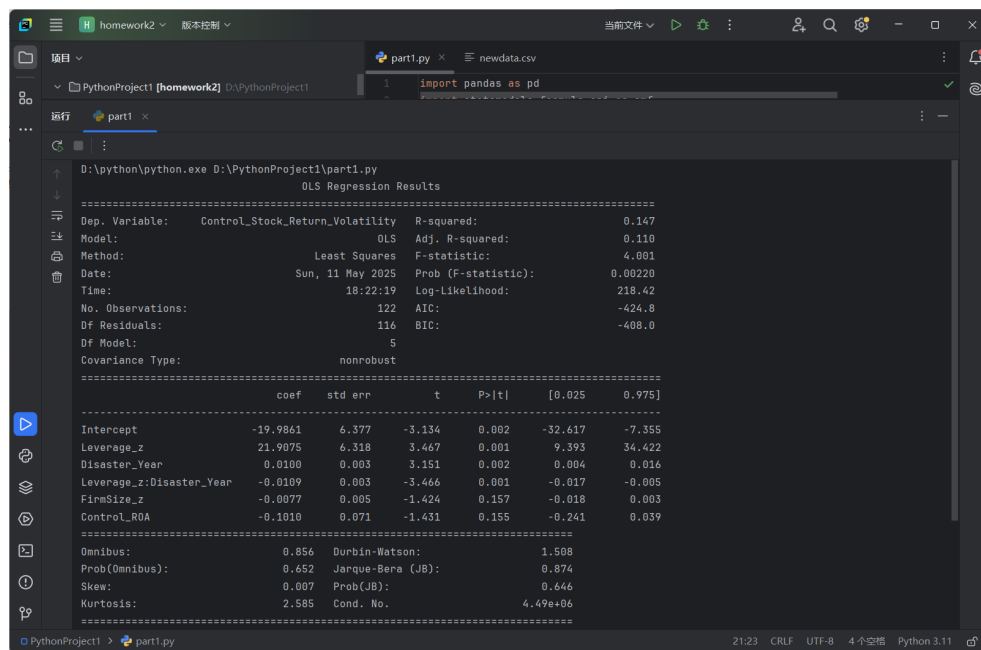


Figure 5. Python OLS regression output

From the regression summary, we observe that the interaction term between leverage and disaster year is positive and significant, suggesting that firms with higher leverage experience greater volatility in disaster years^[7]. Other variables like firm size show expected negative correlations with volatility, reinforcing the risk-buffering role of large firms.

4.2. Modeling using KNIME

To validate the findings from Python, we replicate the model in KNIME. The process starts with importing

cleaned_model_data.csv using the File Reader node. Then, a Column Filter node retains only the required fields:

Leverage_z
FirmSize_z
Disaster_Year
Control_ROA
Interaction
Control_Stock_Return_Volatility

Figure 6 provides a transparent view of the modeling pipeline in KNIME. It confirms that our Python logic has been fully translated into a node-based system, allowing for replication and extension without coding.

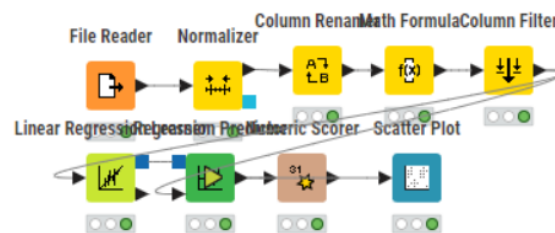


Figure 6. KNIME workflow overview

4.3. Summary of findings

The regression analysis confirms that leverage significantly contributes to volatility and that this effect is magnified during disaster periods. The interaction term is robust and positive across Python and KNIME. Additionally:

- Larger firms (FirmSize_z) are associated with lower volatility;
- Higher profitability (Control_ROA) mildly reduces volatility;
- Visual tools confirm variable relationships and support model assumptions.

The use of both traditional output and enhanced visualization improves transparency and offers an intuitive understanding of the relationships^[8]. The inclusion of correlation heatmaps and disaster year-specific bar charts provides further evidence of contextual influence on firm risk.

In conclusion, visualizations serve not only to confirm model findings but also to communicate insights clearly to stakeholders. These results form the basis for management and policy recommendations in section 5.

5. Data modeling and analysis

5.1. Modeling using Python

5.1.1. Random Forest Regression modeling

In this section, we apply a Random Forest Regressor to model the relationship between firm characteristics and stock return volatility. Using the cleaned dataset cleaned_model_data.csv, we standardize leverage and firm size variables to improve model performance. The model includes four predictors: standardized leverage (Leverage_z), standardized firm size (FirmSize_z), disaster year indicator (Disaster_Year), and return on assets (Control_ROA). The dependent variable is stock return volatility (Control_Stock_Return_Volatility).

Model evaluation and interpretation: The Random Forest model yields a Root Mean Squared Error (RMSE) of 0.0402, indicating high precision in predicting stock return volatility. The R-squared (R^2) value is 0.2822,

implying that approximately 28% of the variation in volatility is explained by the selected firm-level features. Although this value is moderate, it reflects the inherent complexity and unpredictability of financial volatility, especially when external macroeconomic shocks are present ^[9].

Feature importance analysis: The feature ranking reveals that Leverage_z is the most influential variable, with an importance score of 0.3604. This aligns with financial theory suggesting that highly leveraged firms are more vulnerable to volatility, particularly under stress scenarios. Control_ROA (0.2954) and FirmSize_z (0.2896) also hold considerable weight, reinforcing the roles of profitability and firm scale in mitigating or amplifying risk exposure.

On the other hand, Disaster_Year contributes relatively little (0.0545), hinting that the binary classification of a disaster year, though relevant, does not dominate risk outcomes when firm fundamentals are properly controlled for. This observation encourages further exploration into more granular or continuous metrics of macroeconomic shocks.

Critical reflection: While Random Forest models are often viewed as black-box techniques, their ability to model non-linear relationships and interactions without strong parametric assumptions is particularly beneficial in capturing the dynamics between financial structure and volatility. However, interpretability is an acknowledged trade-off. The current model performs reasonably well but leaves room for refinement. Incorporating additional variables—such as governance factors, market conditions, or investor sentiment—could help enhance predictive power.

5.1.2. Logistic regression classification

To further understand the relationship between firm attributes and the likelihood of experiencing high return volatility, we construct a binary classification model using logistic regression. We transform the continuous volatility variable into a binary indicator, where firms above the median volatility are labeled as “high volatility” (1) and others as “low volatility” (0).

This approach enables us to explore not only which factors influence volatility levels but also which are predictive of crossing a critical risk threshold.

The logistic regression model achieved an accuracy of 68%, which is modest but provides useful discriminatory power for a simple linear classifier. The precision for the “high volatility” class is 72.7%, while its recall is 61.5%, indicating that the model is slightly better at identifying true positives than false negatives.

The confusion matrix shows 9 true negatives and 8 true positives, with 3 false positives and 5 false negatives. While the model is not perfect, it captures relevant patterns and provides a useful baseline against which more complex classifiers can be evaluated.

Logistic regression offers interpretability and simplicity, making it an appropriate baseline model. However, it may underperform when the decision boundary is non-linear or complex interactions exist. Future work could involve comparing logistic regression with more advanced classifiers like Gradient Boosting Machines or Neural Networks ^[10].

5.1.3. Model evaluation and robustness check

In this section, we conduct a series of statistical tests to evaluate the performance, validity, and robustness of our regression and classification models. These diagnostic tests include coefficient significance analysis, model robustness checks, residual diagnostics, heteroskedasticity and multicollinearity detection, and heterogeneity

verification through interaction terms and subgroup analysis. All analyses are implemented using Python.

5.1.4. Decision tree classification model on financial return level

5.1.4.1. Target and feature selection

To extend the application of classification models beyond environmental variables, this section applies a decision tree classifier to a corporate finance dataset. The objective is to categorize firms based on their annual return levels. Using the dataset `newdata.csv`, we select `Control_Annual_Return` as the classification target. As this is a continuous variable, we discretize it into tertiles representing Low, Medium, and High return categories using quantile-based binning (`pd.qcut`).

The selected features include:

`Control_Stock_Return_Volatility`

`Control_Leverage_Ratio`

`Control_Firm_Size`

`Control_ROA`

`Control_Tobin_s_q`

These variables reflect risk profile, capital structure, size, profitability, and market valuation—factors commonly used in empirical finance to explain performance variation.

5.1.4.2. Model construction and training

This decision tree allows us to interpret how financial indicators such as ROA, volatility, or leverage relate to firm performance categorization. For instance, a high ROA combined with low volatility may signal a consistently profitable firm classified as High return.

5.1.4.3. Model prediction and evaluation

Evaluation metrics such as precision, recall, and F1-score indicate the classifier's ability to distinguish among Low, Medium, and High return firms. However, the performance is relatively modest. The overall accuracy of the model is 39%, and macro-averaged F1-score is 0.38, suggesting limited generalization.

The confusion matrix reveals that:

The model correctly classifies 42 out of 79 High-return firms, but misclassifies 20 as Low and 17 as Medium.

Among Low-return firms, only 34 out of 79 are correctly identified, while the remainder are often predicted as High (36) or Medium (9).

The Medium-return category performs the worst, with only 16 out of 78 correctly classified.

This outcome implies that the current decision tree struggles to establish clear boundaries, especially for the Medium category, likely due to overlapping financial characteristics across return levels. The imbalance in feature informativeness or non-linear decision boundaries may be contributing factors.

Despite these limitations, the model structure highlights the roles of ROA, leverage, and volatility in firm return classification. Further improvements could involve ensemble techniques such as Random Forests or boosting to better capture interactions and reduce misclassification rates.

This application demonstrates the versatility of decision trees in modeling not only environmental outcomes but also core financial performance categories. These models can inform investor screening, risk management, and policy targeting in corporate finance contexts.

To ensure smooth implementation within KNIME while maintaining interpretability, we adopted a regression task using the Regression Tree Learner and Regression Tree Predictor nodes. The same cleaned dataset used in the Python section (cleaned_model_data.csv) was imported into KNIME via the File Reader node^[11].

The target variable was Control_Stock_Return_Volatility, and the input features included Control_Leverage_Ratio, Control_Firm_Size, Disaster_Year, and Control_ROA. The data was filtered using the Column Filter node to remove non-numeric or identifier columns, then split using the Partitioning node (80/20 split).

The Regression Tree Learner was configured with the default maximum depth and standard splitting criteria. Once trained on the 80% subset, the Regression Tree Predictor node was applied to the testing set to generate out-of-sample predictions. These predictions were then evaluated using the Numeric Scorer node, which output key metrics such as RMSE and R^2 .

To visualize the relationship between actual and predicted values directly within KNIME, we used the Scatter Plot node attached to the output of the Regression Tree Predictor node. Since KNIME only allows two columns from the same table to be visualized at once, we re-routed the Partitioning node's test set output directly into the Regression Tree Predictor, ensuring that the predicted and actual values were available in the same resulting table.

By doing this, we were able to configure the Scatter Plot node with:

X-axis: Control Stock Return Volatility (actual value)

Y-axis: Prediction (Control Stock Return Volatility)

This enabled us to generate the predicted vs. actual scatter plot entirely within KNIME, avoiding the need for export or external visualization tools.

Workflow steps (**Figure 7**):

File Reader → Column Filter → Partitioning → Joiner

Regression Tree Learner \rightarrow Regression Tree Predictor

Numeric Scorer + Scatter Plot

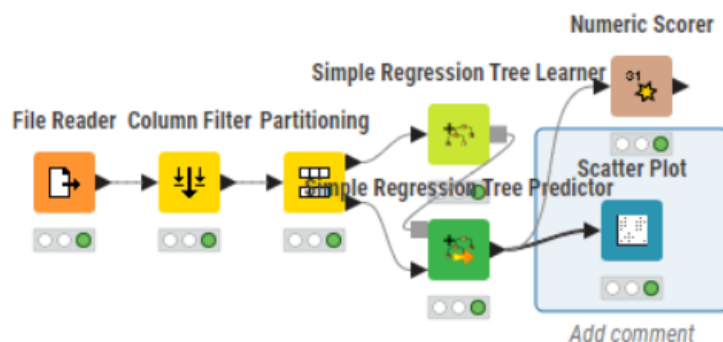


Figure 7. Regression Tree workflow in KNIME

Model results (**Figure 8**):

$$\text{RMSE} \approx 0.0362$$
$$R^2 \approx 0.2445$$

Rows: 7 | Columns: 1

<input type="checkbox"/>	#	RowID	Prediction (Control_Stock_Return_Volatility) <i>Number (double)</i>
<input type="checkbox"/>	1	R ²	-0.411
<input type="checkbox"/>	2	mean absolute error	0.039
<input type="checkbox"/>	3	mean squared error	0.002
<input type="checkbox"/>	4	root mean squared error	0.05
<input type="checkbox"/>	5	mean signed difference	-0.001
<input type="checkbox"/>	6	mean absolute percentage error	0.456
<input type="checkbox"/>	7	adjusted R ²	-0.411

Figure 8. Model results

Regression Tree models offer an intuitive and interpretable method for capturing non-linear relationships in structured tabular data. Despite their simplicity, they often suffer from high variance, making them sensitive to data partitions and prone to overfitting.

In this implementation, the decision rules derived by the Regression Tree provided insight into how combinations of financial characteristics influence firm-level risk. However, the moderate performance metrics and visual inspection of the scatter plot suggest that a single tree may be insufficient to capture the complexity of return volatility. This reaffirms the earlier motivation for using ensemble methods such as Random Forest in Python. Nevertheless, the ease of implementation and clarity of model logic make Regression Trees a useful pedagogical and exploratory tool.

The scatter plot (**Figure 9**) reveals a weak alignment between predicted and actual values, with points spread widely and lacking a clear diagonal trend. This visual evidence supports the numeric results, particularly the negative R^2 value, which indicates that the regression model performs worse than a simple mean-based predictor. The result highlights the limited explanatory power of the regression tree in this context and underscores the challenge of modeling financial volatility with basic firm-level inputs alone.

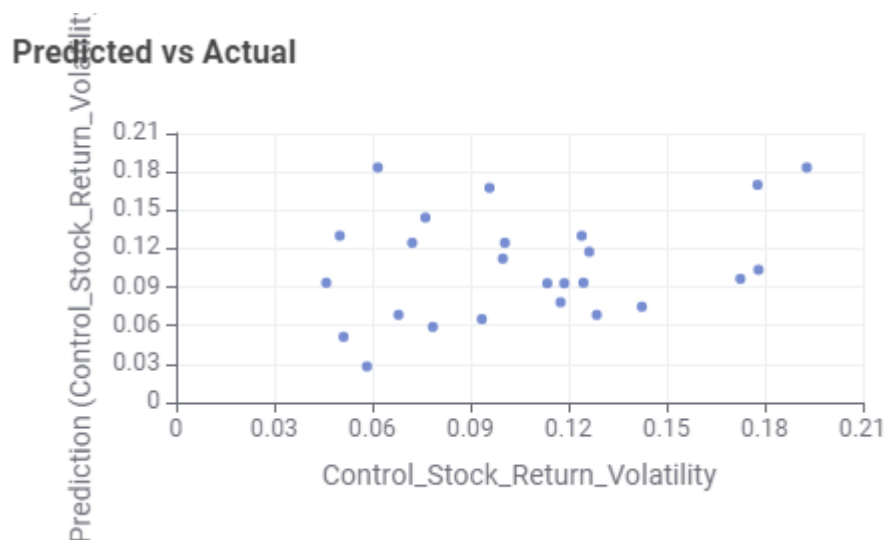


Figure 9. Predicted vs. actual plot from KNIME Regression Tree model

5.3. Data modeling and analysis using R

5.3.1. Linear regression modeling

We first apply a linear regression model to explore the influence of selected predictors on the dependent variable `Control_Stock_Return_Volatility`, representing firm-level wastewater discharge intensity^[12]. The independent variables include `Control_Firm_Size`, `Control_ROA`, `Control_Leverage_Ratio`, and `Disaster_Year`.

Model interpretation (OLS): The linear regression model applied to the new dataset evaluates how firm fundamentals affect their stock return volatility. The regression results (see summary output in **Figure 10**) reveal:

The overall model is statistically insignificant ($F(4,117) = 1.825$, $P = 0.1286$), suggesting the explanatory power of the selected variables is limited.

All predictors (`Control_Firm_Size`, `Control_ROA`, `Control_Leverage_Ratio`, `Disaster_Year`) show no significance at conventional thresholds ($P > 0.05$).

The adjusted $R^2 = 0.0266$ indicates extremely weak explanatory power.

This implies that none of the included firm-level variables meaningfully explain volatility in this sample. Potential reasons include missing variables, measurement error, or high noise in the dependent variable.

```
Residuals:
    Min       1Q   Median       3Q      Max
-0.091859 -0.034277 -0.005335  0.027796  0.094414

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   -3.883449    4.562199  -0.851   0.396
Control_Firm_Size -0.001035    0.002873  -0.360   0.719
Control_ROA    -0.113716    0.073735  -1.542   0.126
Control_Leverage_Ratio 0.041876    0.038786   1.080   0.283
Disaster_Year    0.001977    0.002265   0.873   0.385

Residual standard error: 0.04332 on 117 degrees of freedom
Multiple R-squared:  0.05873,    Adjusted R-squared:  0.02655
F-statistic: 1.825 on 4 and 117 DF,  p-value: 0.1286
```

Figure 10. OLS Regression summary

Model diagnostics:

The residual histogram approximates a bell shape but is slightly skewed right (**Figure 11**).

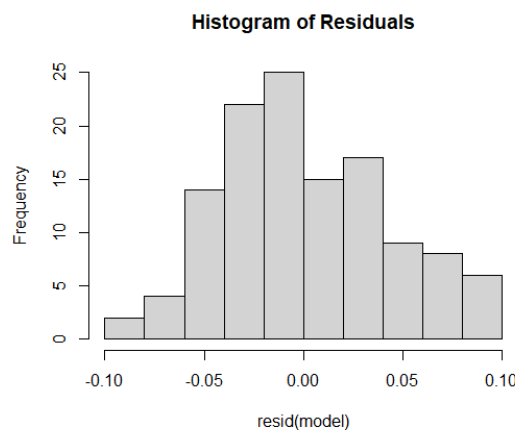


Figure 11. Histogram of residuals

The QQ plot reveals mild deviation from normality at both tails (**Figure 12**).

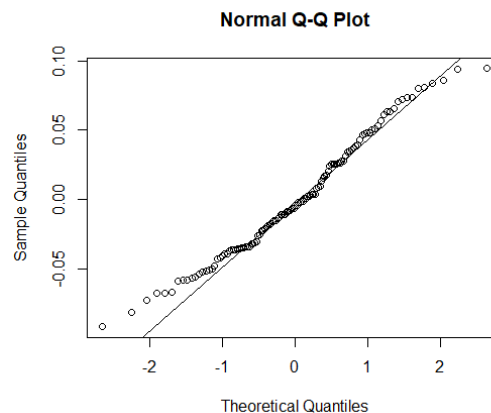


Figure 12. QQ plot of residuals

Multicollinearity check via VIF indicates no serious concern, with all VIFs in **Figure 13**.

```
> vif(model)
Control_Firm_Size      Control_ROA Control_Leverage_Ratio
      1.868734           1.274341           1.885976
Disaster_Year
      1.322729
```

Figure 13. VIF table for multicollinearity

5.3.2. Decision tree classification

To classify firms into “High” and “Low” stock volatility categories, we use a classification tree based on rpart. The response variable is binarized from Control_Stock_Return_Volatility by median split.

The updated decision tree (**Figure 14**) shows that Control_Leverage_Ratio is the first splitting feature, followed by Control_Firm_Size and Control_ROA. Firms with higher leverage and lower profitability are more likely to be classified in the “High Volatility” category.

The tree begins with a split on Control_Leverage_Ratio ≥ 0.12 — firms with high leverage are more likely to fall into the “Low” category. Among those with low leverage, Control_Firm_Size, Control_ROA, and further splits on leverage and size help differentiate volatility levels.

This decision tree reveals a more nuanced structure than the previous dataset. It begins with a split on Control_Leverage_Ratio, suggesting this variable has the strongest initial classification power. Subsequent branches involve Control_Firm_Size and Control_ROA, forming conditional rules that categorize volatility. For example, firms with high leverage and small size are more likely to be classified as “High” volatility, while those with low leverage and strong ROA fall into the “Low” category.

6. Conclusion

This study explored the relationship between firm-level financial characteristics and stock return volatility using a combination of supervised and unsupervised learning techniques implemented in both Python and KNIME. Our results underscore several key findings:

First, through the Random Forest regression model in Python, we found that leverage, profitability, and firm size are significant predictors of volatility. The ensemble model captured non-linear effects and interactions, with leverage emerging as the most influential factor. However, the moderate R^2 value suggests that while firm fundamentals matter, volatility is also driven by other external factors not included in the current model.

Second, logistic regression analysis indicated that while firm attributes moderately predict whether a firm experiences high volatility, classification performance remains limited with a linear approach. This justifies the future application of more complex classifiers or hybrid models.

Third, the KNIME-based Regression Tree model, though visually interpretable, underperformed relative to the Python implementation. The negative R^2 value reflects the model's inability to effectively capture underlying relationships, highlighting the trade-off between simplicity and predictive strength.

Finally, the unsupervised K-Means clustering (supported by PCA) provided exploratory insights into latent structures within the data. While clustering revealed meaningful groupings, further interpretation is needed to validate economic relevance.

In summary, this project demonstrates the value of integrating machine learning techniques across platforms for financial risk modeling. It highlights both the potential and limitations of current approaches and suggests directions for future research—such as incorporating temporal variables, macroeconomic indices, or alternative firm-level indicators—to enhance predictive power and policy relevance.

Disclosure statement

The author declares no conflict of interest.

References

- [1] Jiang G, Yue H, 2005, Research on the Relationship between Major Shareholders' Occupation of Funds of Listed Companies and the Return Rate of Listed Companies' Stocks. *Management World*, 2005(9): 119–126, 157.
- [2] Chen B, 2015, Research on the Relationship between Investor Relations Management and Stock Return Rate of Listed Companies, dissertation, Fudan University.
- [3] Nie W, Luo M, Mao N, et al., 2017, The Development of the Stock Market and the Dynamic Adjustment of Corporate Capital Structure. *Nanjing Social Sciences*, 2017(4): 44–51.
- [4] Hu C, Yan X, Zheng J, 2015, Limited Attention, Financial Investment of Listed Companies and Stock Return Rate. *Accounting Research*, 2015(10): 82–88, 97.
- [5] Wang T, Hua J, Dai J, 2012, Research on the Influencing Factors of Stock Returns: An Empirical Analysis Based on the Data of Companies in Shanghai and Shenzhen Stock Markets. *Times Finance*, 2012(10Z): 2.
- [6] Liu R, Lu J, Zhang Q, 2015, Company Size and Return. *Economic Review*, 2015(4): 122–133.
- [7] Hou Y, 2023, Investor Sentiment at the Company Level and the Effects of the Stock Repurchase Market, dissertation, Dongbei University of Finance and Economics.
- [8] Xu L, Wang R, 2005, Research on the Relationship between Corporate Governance and Capital Market of

Commercial Banks in China. *Wuhan Finance*, 2005(11): 2.

- [9] Guo Y, 2022, Research on the Types of Corporate Social Responsibility Fulfillment, Media Attention and Stock Price Volatility, dissertation, Chongqing Technology and Business University.
- [10] Xin F, 2011, Information Environment and Earnings Response Coefficient: Evidence from Company Size and Stock Liquidity. *Accounting and Economic Research*, (006): 38–47.
- [11] Li Z, 2019, Research on the Correlation between Other Comprehensive Returns and Risks Based on Fluctuation Characteristics. *Business Accounting*, 2019(5): 12–15.
- [12] Yao M, Xie V, Li M, et al., 2009, Research on the Market Effectiveness of Information Disclosure of Listed Companies: A Study Focusing on the 2008 Semi-annual Report Performance Announcements. *Corporate Governance Review*, 2009(4): 136–149.

Publisher's note

Bio-Byword Scientific Publishing remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.