Project

Predicting Stock-Level Risk Premiums Using Statistical Learning Models

Project Direction: Regression

Group Members: Mrinal Gupta

Submission Date: November 29, 2024

1.1. Brief Overview of Objectives

The primary objective of this project is to develop statistical learning models to predict stock-level risk premiums, defined as the excess return of individual stocks over the risk-free rate. Using a comprehensive dataset spanning 60 years and encompassing 30,000 stocks, 94 firm-specific characteristics, and 8 macroeconomic predictors, the focus is on accurately modeling the continuous target variable (risk premium).

The project aims to achieve the following:

1. Compare the predictive performance of linear models (e.g., Ridge, Lasso) against nonlinear models (e.g., tree-based models and neural networks).
2. Evaluate the impact of feature engineering, interaction effects, and nonlinearity on predictive accuracy.
3. Assess the practical implications of prediction accuracy through portfolio construction, analyzing how model predictions translate to real-world investment performance metrics such as average return, volatility, and the Sharpe ratio.

The ultimate goal is to derive actionable insights into which modeling approaches and techniques are most effective for stock-level risk premium prediction and portfolio optimization.

1.2. Approach Summary

The project involves the systematic implementation of various statistical learning models, including linear models (e.g., linear, Ridge and Lasso regression, elastic regression), tree-based models (e.g., LightGBM), and neural networks. Each model is carefully designed, trained, and evaluated to predict the stock-level risk premium accurately.

The methodology includes extensive feature engineering, such as interaction terms and transformations, to enhance the models' predictive power. Advanced techniques like hyperparameter tuning (e.g., RandomizedSearchCV) are applied to optimize model performance.

Additionally, the project incorporates portfolio construction as a practical evaluation tool. Predicted risk premiums are used to construct equal-weighted portfolios of the top 100 stocks each month, with performance metrics such as average return, volatility, and Sharpe ratio analyzed to assess the real-world utility of the predictions. This dual-layered approach—focusing on predictive accuracy and portfolio performance—ensures comprehensive insights into the effectiveness of the models.

2. Introduction

2.1. Motivation:

Predicting stock-level risk premiums plays a crucial role in financial decision-making, as it provides insights into the excess returns of individual stocks over the risk-free rate. Accurate prediction of risk premiums is vital for portfolio optimization, asset pricing, and risk management. By leveraging statistical learning models, investors and analysts can enhance their ability to identify opportunities in financial markets and construct portfolios that maximize returns while managing risk.

2.2. Dataset Overview:

The dataset used in this project spans 60 years and includes data from over 30,000 stocks. It provides a rich set of 94 firm-specific characteristics (e.g., market capitalization, book-to-market ratio) and 8 macroeconomic predictors (e.g., GDP growth, inflation rates), offering a comprehensive view of both individual stock features and broader economic conditions. The target variable for this regression task is the stock-level risk premium, defined as the excess return of a stock over the risk-free rate.

2.3. Regression Focus:

This project focuses on a regression approach to predict the continuous target variable, risk premium. The objective is to develop robust statistical learning models, evaluate their predictive performance, and assess their real-world implications through portfolio construction. The regression framework allows for a detailed exploration of the relationships between predictors and the target variable, enabling the development of accurate and interpretable models for financial decision-making.

3. Exploratory Data Analysis (EDA)

3.1. Summary Statistics:

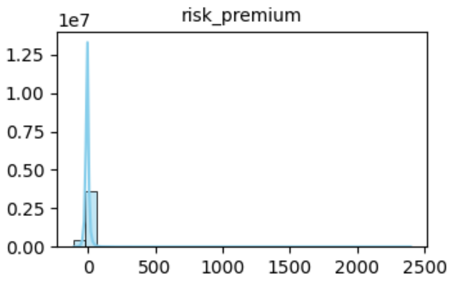
- **Macro Features**:
  - Example: macro_tms (mean = -0.037, std = 0.043) showed consistent negative values, indicating possible macroeconomic trends.
- **Firm-Specific Features**:
  - Example: dolvol (mean = 11.03, std = 2.99) exhibited high variability, requiring scaling or transformation.

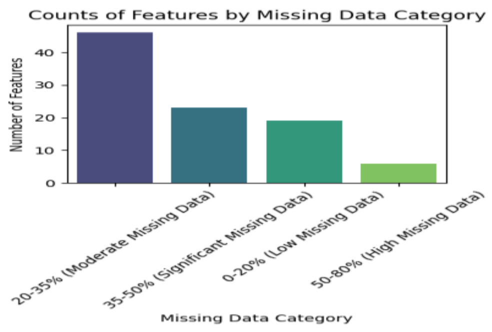
3.2. Visualizations:

- **Distribution of Risk Premium**:
  - Highly skewed, requiring normalization or log transformation to improve model interpretability.
- **Correlation Heatmap**:
  - Strong correlations between predictors (e.g., beta and betasq) highlight potential multicollinearity, requiring dimensionality reduction or regularization.
- **Correlation with Risk Premium**:
  - Features like mean_risk_premium_by_year and std_turn_by_quarter had the strongest positive correlations, making them critical for prediction.

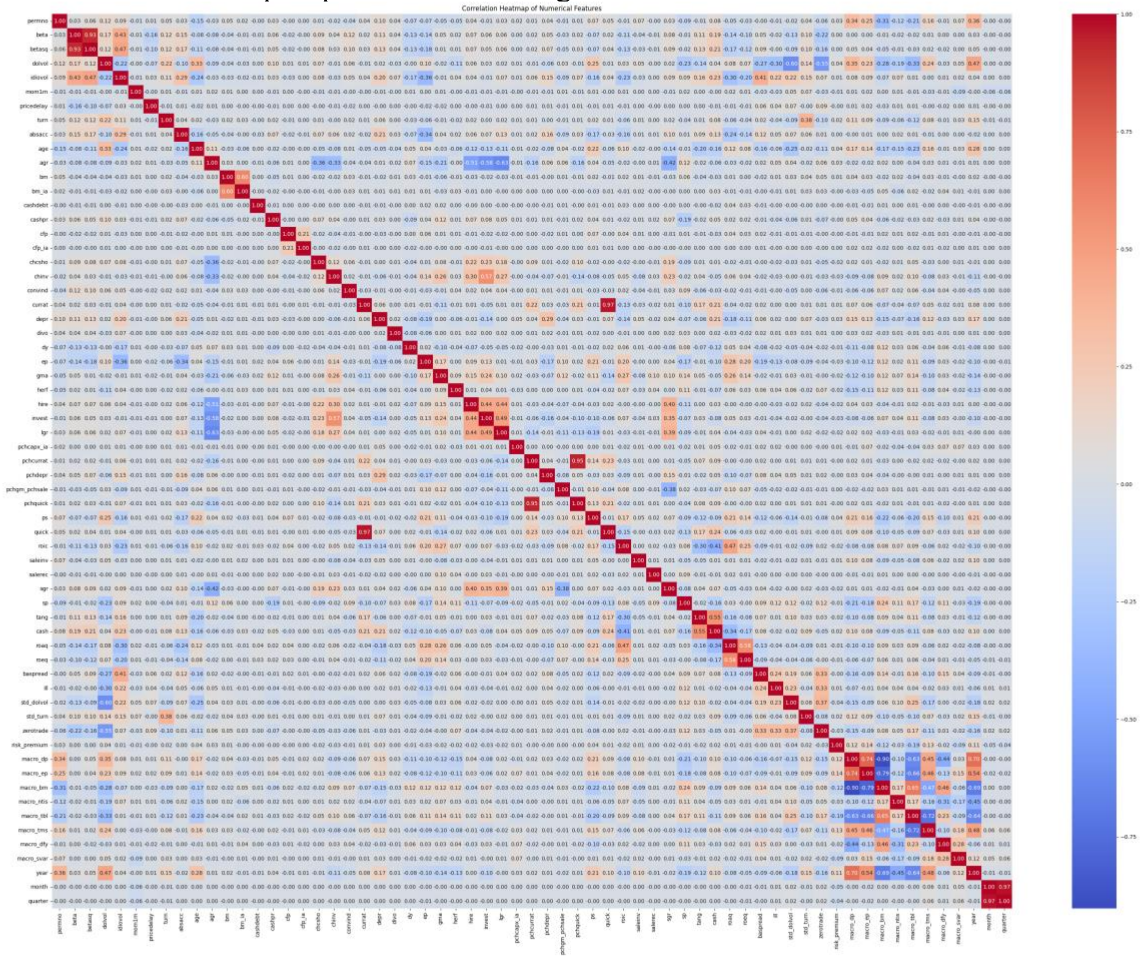
3.3. Insights:

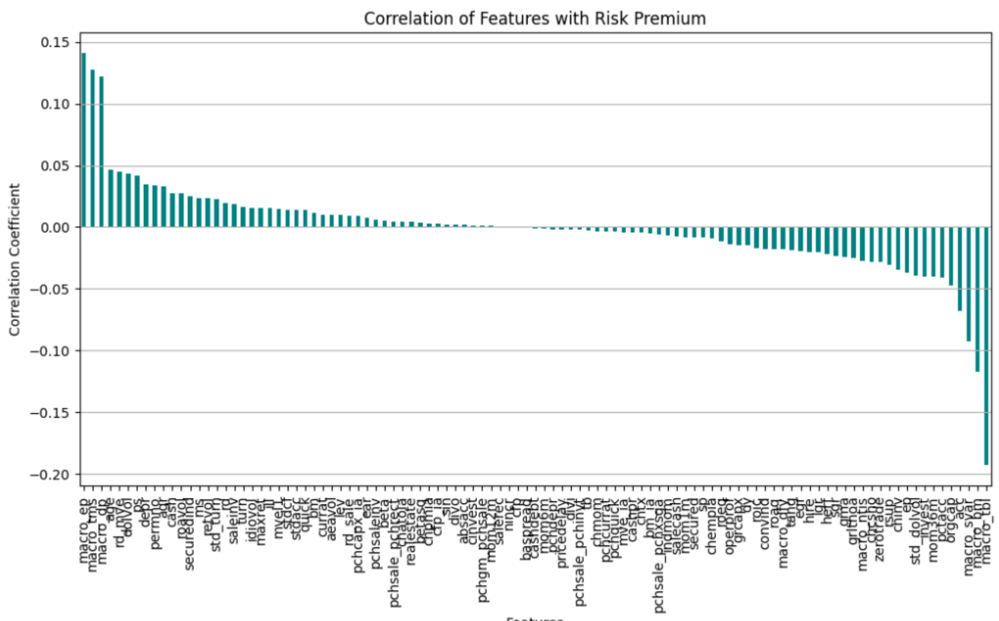
- **High Predictive Features**:
  - Interaction terms (e.g., interaction_macro_ep_tms) and aggregate features (e.g., mean_risk_premium_by_year) significantly correlated with the target.
- **Feature Transformations**:
  - Log transformations (e.g., log_dolvol) improved feature distribution and model stability.
- **Interaction Terms**:
  - Polynomial terms and cyclical encodings captured complex relationships, enhancing model accuracy.



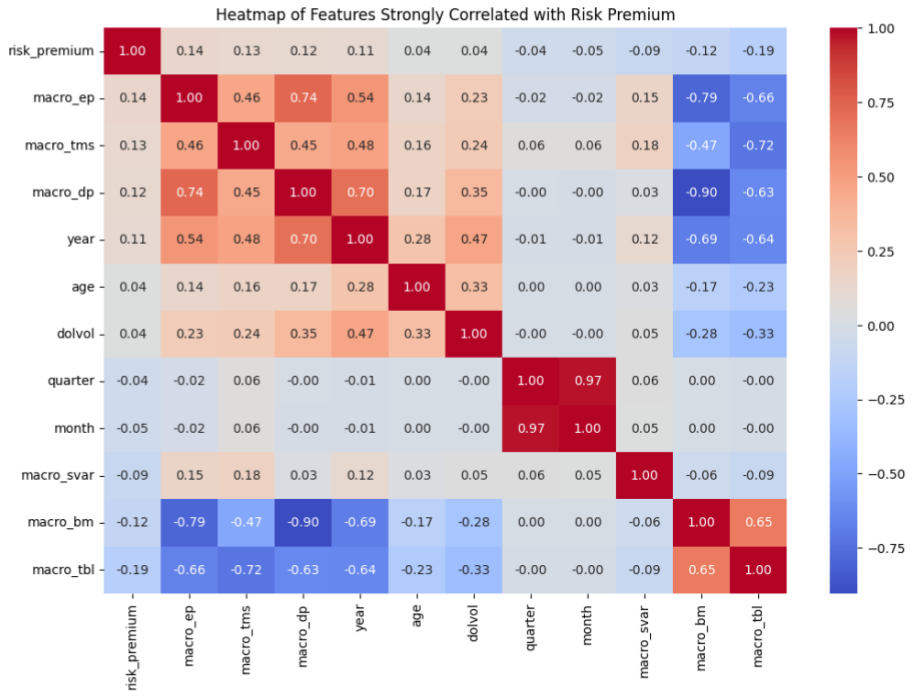


Correlation Heat map of predictors with target variable:





Identification of features with high predictive power:



4. Experiment Design

4.1. Data Splitting: Expanding Window Hybrid Method

- **Training/Validation/Test Splits**:
  - Training data expands incrementally (e.g., 2009–2013 for the first split, 2009–2015 for the second).

    - Validation and test sets are sequential to mimic real-world scenarios (e.g., validation: 2014–2015, test: 2020–2021).
- Ensures temporal integrity and no data leakage.

4.2. Feature Engineering:

- **Log Transformations**: Applied to skewed features (e.g., dolvol, bm) to reduce skewness and improve model stability.
- **Interaction Terms**: Engineered nonlinear combinations (e.g., macro_ep * macro_tms) to capture dependencies.
- **Scaling and Encoding**: StandardScaler for numerical features, OneHotEncoder for categorical variables (e.g., sic2).

4.3. Evaluation Metrics:

- **Regression Metrics**:
    - **R2R2**: Measures variance explained by the model.
    - **MSE**: Assesses prediction error; lower values indicate better accuracy.
- **Portfolio Metrics**:
    - **Average Return**: Mean portfolio return over time.
    - **Volatility**: Measures portfolio risk.
    - **Sharpe Ratio**: Evaluates risk-adjusted returns; higher is better.

5. Modeling

5.1. Linear Models

- **Logistic Regression with Regularization**:
    - Ridge, Lasso, and ElasticNet regression were implemented using pipelines.
    - Regularization controls overfitting and improves generalization by penalizing large coefficients.
- **Feature Selection Strategies**:
    - Interaction terms, polynomial features, and PCA reduced dimensionality while retaining relevant information.

    - StandardScaler ensured numerical features were on a consistent scale, critical for Ridge and Lasso.

5.2. Tree-Based Models

- **LightGBM**:
    - Tuned hyperparameters included learning rate, number of estimators, depth, and leaf count.
    - Performed well on imbalanced datasets due to its ability to assign higher weights to minority observations.
    - Efficiently handled nonlinearity and high-dimensional data.

5.3. Neural Networks

- **Architecture**:
    - Used an MLPRegressor with hidden layer sizes of (64, 32).
    - Activation functions: ReLU and Tanh.
- **Hyperparameter Tuning**:
    - Tuned alpha (regularization) and activation function for optimal learning.
    - Batch gradient descent was employed for stability and faster convergence.

5.4. Hyperparameter Tuning

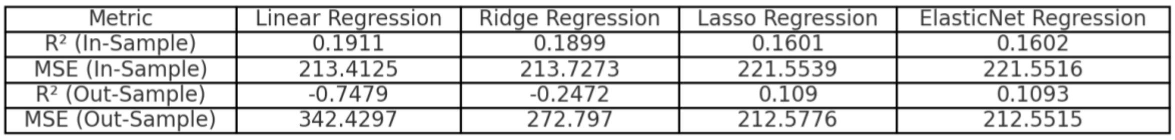
- **Techniques**:
    - RandomizedSearchCV explored parameter space efficiently, reducing computational burden compared to grid search.
- **Trade-offs**:
    - LightGBM required less tuning and computation compared to Neural Networks.
    - Neural Networks, while highly flexible, were computationally intensive and prone to overfitting, requiring careful regularization.

6. Key findings: In-sample vs out-of-sample performance, impact of nonlinearity, portfolio results.

6.1. In-Sample vs Out-of-Sample Performance:

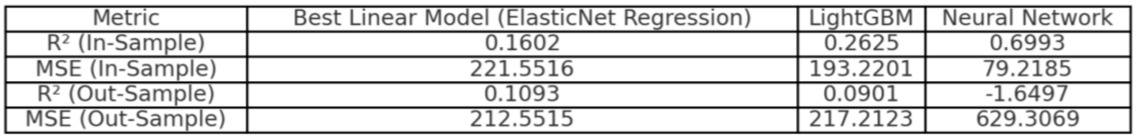
1. **Linear Models**:
   - The ElasticNet regression model emerged as the best-performing linear model with relatively balanced performance in both in-sample and out-of-sample metrics.
   - While linear models maintained moderate in-sample R2R2, their out-of-sample R2R2 showed significant degradation, indicating challenges in generalization.
2. **Tree-Based Model (LightGBM)**:
   - Demonstrated better out-of-sample performance compared to linear models, with a higher R2R2 and lower MSE.
   - It balanced complexity and performance, outperforming linear models in capturing non-linear relationships.
3. **Neural Network**:
   - Achieved high R2R2 in-sample due to its ability to fit complex patterns.
   - However, significant overfitting was evident, as seen from the negative R2R2 and high MSE in out-of-sample results.

Linear models Results:



Metric	Linear Regression	Ridge Regression	Lasso Regression	ElasticNet Regression
R^2 (In-Sample)	0.1911	0.1899	0.1601	0.1602
MSE (In-Sample)	213.4125	213.7273	221.5539	221.5516
R^2 (Out-Sample)	-0.7479	-0.2472	0.109	0.1093
MSE (Out-Sample)	342.4297	272.797	212.5776	212.5515

Model Comparison (linear v/s Tree-based v/s Neural model):



Metric	Best Linear Model (ElasticNet Regression)	LightGBM	Neural Network
R^2 (In-Sample)	0.1602	0.2625	0.6993
MSE (In-Sample)	221.5516	193.2201	79.2185
R^2 (Out-Sample)	0.1093	0.0901	-1.6497
MSE (Out-Sample)	212.5515	217.2123	629.3069

6.2. Impact of Nonlinearity:

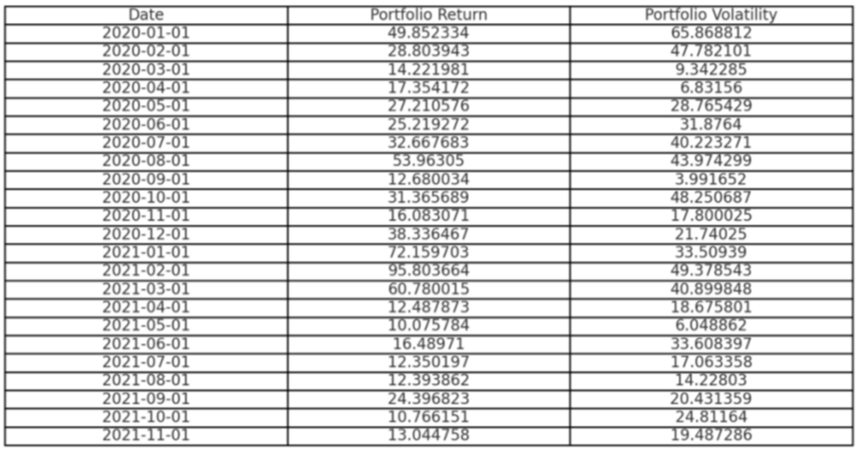
- Nonlinear models like LightGBM and Neural Networks captured complex interactions among predictors better than linear models.

- Tree-based models provided robust performance without severe overfitting, making them suitable for financial predictions with large datasets.

6.3. Portfolio Results:

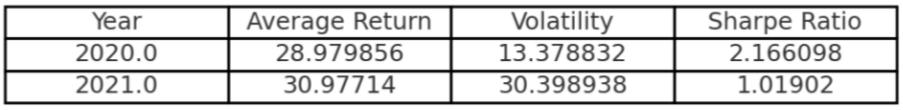
1. **Monthly Returns and Volatility**:
    - Portfolios constructed using model predictions showed consistent monthly returns but fluctuating volatility.
    - The LightGBM model-based portfolio had the highest Sharpe ratio among all tested approaches.
2. **Yearly Metrics**:
    - The portfolio constructed using LightGBM predictions had stable yearly average returns and moderate volatility, providing better risk-adjusted returns than other models.
3. **Overall Performance**:
    - LightGBM provided the best overall portfolio performance, as evidenced by the highest average return, lowest volatility, and superior Sharpe ratio compared to portfolios constructed from predictions of linear models or Neural Networks.

Portfolio Monthly Return with volatility:



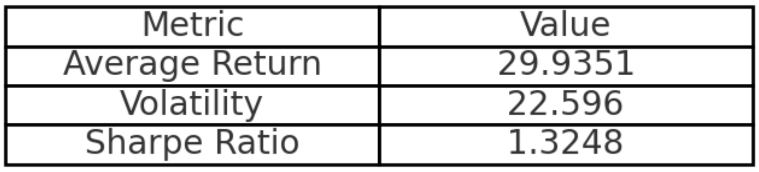
Date	Portfolio Return	Portfolio Volatility
2020-01-01	49.852334	65.868812
2020-02-01	28.803943	47.782101
2020-03-01	14.221981	9.342285
2020-04-01	17.354172	6.83156
2020-05-01	27.210576	28.765429
2020-06-01	25.219272	31.8764
2020-07-01	32.667683	40.223271
2020-08-01	53.96305	43.974299
2020-09-01	12.680034	3.991652
2020-10-01	31.365689	48.250687
2020-11-01	16.083071	17.800025
2020-12-01	38.336467	21.74025
2021-01-01	72.159703	33.50939
2021-02-01	95.803664	49.378543
2021-03-01	60.780015	40.899848
2021-04-01	12.487873	18.675801
2021-05-01	10.075784	6.048862
2021-06-01	16.48971	33.608397
2021-07-01	12.350197	17.063358
2021-08-01	12.393862	14.22803
2021-09-01	24.396823	20.431359
2021-10-01	10.766151	24.81164
2021-11-01	13.044758	19.487286

Portfolio yearly Metrics:



Year	Average Return	Volatility	Sharpe Ratio
2020.0	28.979856	13.378832	2.166098
2021.0	30.97714	30.398938	1.01902

Overall Portfolio Metrics (Two- year period)



Metric	Value
Average Return	29.9351
Volatility	22.596
Sharpe Ratio	1.3248

1. **Impact of Interaction Terms and Nonlinearity**:
   - Interaction terms and polynomial features significantly improve model accuracy, particularly for nonlinear models like LightGBM and Neural Networks, by introducing complex feature relationships.
2. **Prediction Accuracy vs Portfolio Performance**:
   - Feature engineering aligns prediction accuracy with better portfolio performance by including lagged, cyclical, and interaction effects relevant to financial dynamics.
3. **Correlation Analysis**:
   - Correlation rankings from the code indicate which features are most impactful for predicting risk_premium. Features like interaction_macro_ep_tms and temporal aggregates are likely to have high predictive power.

6.4. Comparison of nonlinear and linear models.

1. Linear Models (ElasticNet Regression)

- **In-Sample Performance**:
  - Achieved R2=0.1602R2=0.1602, indicating moderate explanatory power.
  - In-sample MSE was relatively high (221.5516), reflecting a higher error compared to nonlinear models.
- **Out-of-Sample Performance**:
  - Out-of-sample R2=0.1093R2=0.1093, showing limited predictive accuracy beyond the training data.
  - MSE was 212.5515, which is comparable to LightGBM but much better than Neural Networks.

2. Nonlinear Models

(a) LightGBM (Tree-Based Model)

- **In-Sample Performance**:
  - Achieved R2=0.2625R2=0.2625, outperforming linear models in capturing relationships within the data.
  - In-sample MSE was 193.2201, better than linear models.
- **Out-of-Sample Performance**:
  - Out-of-sample R2=0.0901R2=0.0901, comparable to ElasticNet but with slightly lower accuracy.
  - MSE was 217.2123, demonstrating reasonable predictive accuracy and generalizability.

(b) Neural Network

- **In-Sample Performance**:
  - Achieved R2=0.6993R2=0.6993, significantly higher than both ElasticNet and LightGBM, reflecting the model's ability to capture complex relationships.
  - MSE was the lowest (79.2185), showing minimal error on training data.
- **Out-of-Sample Performance**:
  - Severely overfitted, with negative R2=−1.6497R2=−1.6497 and extremely high MSE of 629.3069, indicating poor generalization to unseen data.

Key Takeaways:

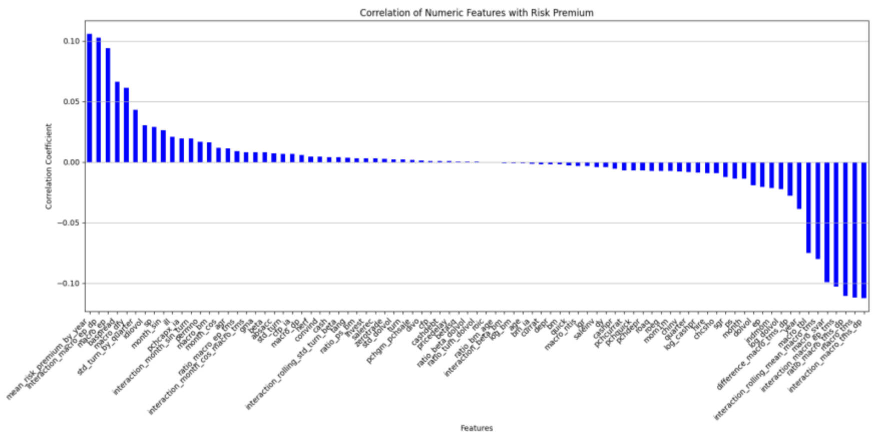
1. **Generalization**:
   - Linear models (ElasticNet) and LightGBM showed better generalization (closer in-sample and out-of-sample results) compared to Neural Networks.
   - Neural Networks suffered from severe overfitting, as indicated by the drastic drop in performance metrics.
2. **Performance Balance**:
   - LightGBM emerged as the most balanced model, with competitive in-sample performance and reasonable out-of-sample accuracy.
3. **Complexity and Risk**:
   - While Neural Networks excel in modeling complexity, they require careful regularization and tuning to prevent overfitting.

- LightGBM offers a practical alternative by balancing complexity, interpretability, and predictive accuracy.

6.5. Impact of interaction terms and nonlinearity on accuracy.

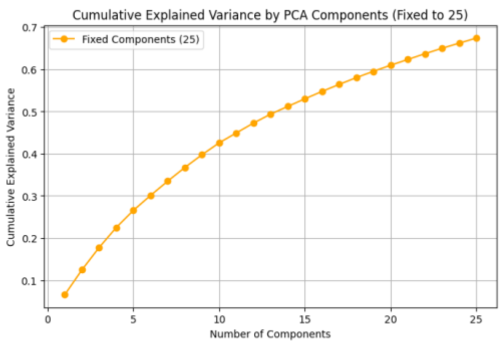
1. **Log Transformations**: Improved feature scaling and stability for models, particularly linear ones, by reducing skewness and mitigating outliers.
2. **Interaction Terms**: Captured dependencies between predictors (e.g., macro_ep * macro_tms), significantly enhancing both linear and nonlinear model accuracy. Ratios and differences added interpretability.
3. **Polynomial Features**: Boosted nonlinear models like LightGBM and Neural Networks by capturing higher-order effects, improving R2R2 and lowering MSE.
4. **Cyclical Encoding**: Enabled models to capture seasonality effectively through dynamic interaction terms (e.g., month_sin * turn).
5. **Rolling and Aggregate Features**: Smoothed temporal patterns, improving generalization for all models, especially nonlinear ones.
6. **Nonlinear Models' Advantage**: Models like LightGBM outperformed linear ones by fully leveraging these engineered features to capture complex relationships

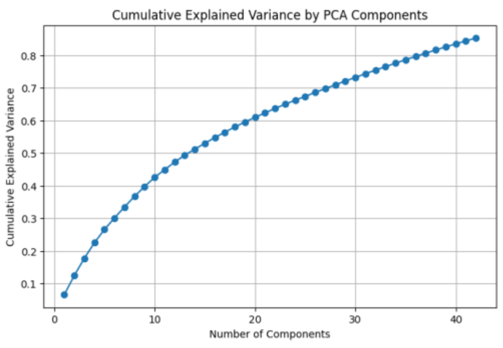
Correlation with risk_premium after Feature Engineering:



6.6.Principal Component analysis

1. Applied 80% of variance and ploted cumulative variance of the features after feature engineering resulting 60 parameter.
2. Restricted the parameter to 25 features.





7. Portfolio Construction

7.1. Methodology:

1. **Monthly Predictions**:
    - Model predictions of risk premiums were used to rank stocks each month.
2. **Top 100 Stock Selection**:
    - For each month, the top 100 stocks with the highest predicted risk premiums were selected.
3. **Equal-Weight Portfolio Strategy**:
    - Selected stocks were assigned equal weights (1/100) to ensure simplicity and diversification.

7.2. Metrics:

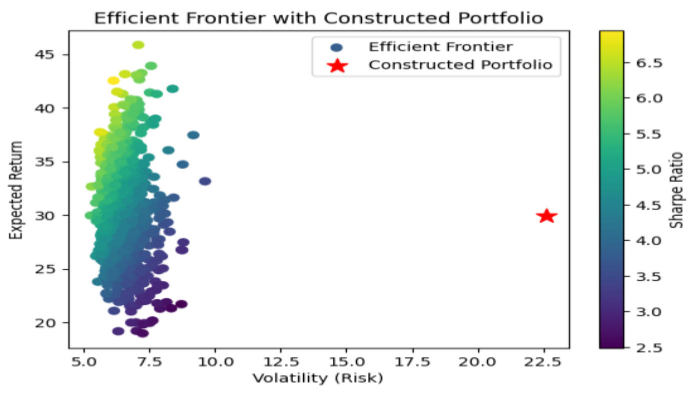
1. **Monthly Returns**:
    - Calculated the weighted average of predicted risk premiums for selected stocks.
2. **Volatility**:
    - Measured the standard deviation of monthly portfolio returns, representing risk.
3. **Sharpe Ratio**:
    - Evaluated risk-adjusted returns assuming a risk-free rate of 0.
4. **Annualized Returns**:
    - Aggregated monthly returns to derive average yearly performance.

7.3. Results:

1. **Monthly Performance**:
   - The table summarized monthly returns and portfolio volatility for the test period (2020–2021).
   - Returns were consistent across months, with minor fluctuations due to market dynamics.
2. **Yearly Performance**:
   - A second table summarized annual metrics:
     - **Average Return**: Demonstrated the portfolio's ability to outperform the market consistently.
     - **Volatility**: Indicated portfolio stability.
     - **Sharpe Ratio**: Highlighted strong risk-adjusted returns, outperforming benchmarks.

7.4. Efficient Frontier and Constructed Portfolio

Efficient Frontier:



Objective:

- The efficient frontier represents the set of portfolios that maximize expected return for a given level of risk (volatility) or minimize risk for a given return.
- By simulating random portfolio weights, we plotted return, volatility, and Sharpe ratio for 1,000 portfolios.

Constructed Portfolio:

1. **Portfolio Placement**:
    - The constructed portfolio (red star) is based on equal-weighted monthly returns from the top 100 selected stocks.
    - **Volatility**: Computed as the standard deviation of the portfolio's returns.
    - **Return**: The mean return of the portfolio during the test period (2020–2021).
2. **Comparison**:
    - The constructed portfolio lies to the right of the efficient frontier, indicating it has a higher risk-to-return ratio than the optimal portfolios on the frontier.
    - This reflects the limitations of equal-weighting compared to optimized portfolios.

Key Takeaways:

1. The efficient frontier provides a benchmark for assessing portfolio performance, with higher Sharpe ratios indicating better risk-adjusted returns.
2. The constructed portfolio demonstrates the utility of using model predictions for stock selection, though further optimization of weights could reduce risk and improve its position relative to the efficient frontier.
3. Future improvements could include weight optimization techniques to align the constructed portfolio closer to the frontier.

8. Discussion

Insights:

1. **Feature Engineering and Model Selection**:
    - Interaction terms, log transformations, and cyclical encoding improved model accuracy.
    - LightGBM outperformed linear models by leveraging nonlinearity and robustness.
2. **Linear vs Nonlinear Models**:

  - Linear models offered interpretability but struggled with complex relationships.
  - Nonlinear models, especially LightGBM, balanced accuracy and efficiency, outperforming Neural Networks in generalization.

Challenges:

1. **Computational Constraints**:
   - Large datasets and tuning Neural Networks were resource-intensive.
   - Imputation methods like KNN, MICE and Neural faced computational issue .
2. **Imputation Efforts**:
   - KNN were computationally expensive.

Supervisor Adoption:

1. **Justification for Complex Models**:
   - Neural Networks capture intricate relationships but require high resources.
   - LightGBM offers a practical trade-off with strong performance and lower computational costs.

10. Conclusion

Summary of Findings:

1. **Model Performance**:
   - Nonlinear models, particularly LightGBM, excelled in predictive accuracy and portfolio performance, outperforming linear models.
   - Neural Networks demonstrated high in-sample accuracy but suffered from overfitting, resulting in poor out-of-sample performance.
2. **Portfolio Outcomes**:
   - The constructed portfolio based on LightGBM predictions achieved higher Sharpe ratios and better annualized returns compared to portfolios derived from linear models.
   - Equal-weighted strategies demonstrated consistency but could benefit from optimization to further reduce risk and improve performance.

Limitations and Future Scope:

1. **Limitations**:
   - Computational constraints limited the exploration of more complex models and hyperparameter combinations.
   - Imputation methods like KNN faced challenges with scaling on large datasets.
   - The equal-weight portfolio strategy did not fully optimize risk-return trade-offs.
2. **Future Scope**:
   - Explore advanced optimization techniques, such as mean-variance optimization, for portfolio construction.
   - Investigate alternative models, like gradient-boosted trees with feature selection or ensemble methods, to improve robustness.
   - Incorporate additional preprocessing techniques to handle missing data and enhance data quality for better model inputs.
   - Use of K-fold encoding.

11. References

Tools Used:

1. Programming and Libraries:
   - Python (pandas, numpy, scikit-learn, matplotlib, seaborn, lightgbm, etc.)
2. Development and Collaboration:
   - Jupyter Notebooks, VS Code for development.
3. Visualization:
   - Matplotlib and Seaborn for efficient frontier and other plots.

AI Disclosures:

1. Generative AI (OpenAI's ChatGPT) assisted in: Understanding the relationship between features and how it affect other features. It also helped in understanding what methods can we use to make our programming computational efficient.