Department of Mathematical Sciences

Project IV - (MATH4072)

Equity Fund Profitability and Sustainability Modelling

Using Multiple Response Regression

Author:
Raul Unnithan

Supervisor:
Ric Crossman

April 24, 2025

Acknowledgement

Write an acknowledgement to Ric...

Declaration

This piece of work is a result of my own work and I have complied with the Department's guidance on multiple submission and on the use of AI tools. Material from the work of others not involved in the project has been acknowledged, quotations and paraphrases suitably indicated, and all uses of AI tools have been declared.

Contents

Chapter 1 Introduction

1.1 Background

1.1.1 Multiple Response Regression

Single-response regression models one response against a set of predictor variables. This method can be extended to multiple predictors, but what if we need to model more than one response? One approach is to run several independent single-response models. However, when responses are correlated, the off-diagonal elements of the response-covariance matrix are non-zero, which this approach ignores, leading to suboptimal predictions. Multiple-response regression (MRR) addresses this issue, thereby improving predictions.

However, selecting the most relevant predictors remains a challenge, and each MRR model does this differently. This report has examined and applied different MRR models to an equity fund dataset to determine which of these fits it the best.

1.1.2 Equity Funds

An equity fund is a pooled investment that puts money mainly in stocks listed on major exchanges.[1] They provide investors with several benefits, such as diversification and potential improved returns.[1] However, equity funds also come with risks associated with stock market volatility and losses.[1] Equity funds can be categorised according to factors such as their investment style and portfolio focus, which is how they are represented in this data.[1]

1.2 Equity Fund Dataset

1.2.1 Overview

The equity fund dataset analysed in this report contains information about a diverse range of funds capturing different equity fund characteristics, including, for example, the type of equity. Modelling equity fund data is important as it can indicate which funds are worth investing in and which are not. Traditional financial models prioritise profitability above all else.[2] However, in this report, there are two responses, with moderate correlation, modelled. These responses are return on equity (ROE) and sustainability score. A noteworthy response correlation is important here because, otherwise, one model for each response could be built.

1.2.2 Responses Modelled

ROE is a metric of an equity fund's profitability and how efficiently it generates those profits.[3] It is calculated by dividing net income by shareholders' equity.[3] ROE is typically expressed as a percentage, so this value is multiplied by a hundred. In the context of an equity fund, net income is not derived from sales minus the cost of goods sold as it would be for a company.[4] Instead, an equity fund's net income comes primarily from earnings such as interest, after subtracting operating expenses like

distribution costs.[4] The higher the ROE, the more efficiently an equity fund is generating income and growth from its portfolio of equity investments.[5]

The sustainability score measures an equity fund's sustainability performance and how effectively it mitigates environmental, social, and governance (ESG) risks across its portfolio. Each equity fund's sustainability score is its Morningstar Sustainability Rating, which directly measures these ESG factors.[6] The Morningstar Sustainability Rating is calculated through a multi-step process. First, a fund qualifies for a rating only if at least 67% of its assets have an assigned ESG score.[6] Each company within the fund's portfolio is then scored, typically on a scale ranging up to 100.[6] The fund's portfolio score is adjusted by subtracting points for any ESG-related negative events, such as environmental incidents.[6] Finally, the equity fund's overall sustainability score is then compiled as weighted averages over the past 12 months, giving more emphasis to recent performance. Lower sustainability scores indicate better ESG performance of an equity fund.

1.2.3 Motivation

Modelling ROE and sustainability scores in particular together is important as it gives a more long-term view of each equity fund's risk versus return potential. Sustainability has become more relevant as investors move to prioritise sustainable portfolios. A key driver of this trend is the growing concern over climate change and global warming, which have significant long-term social and economic implications. In a 2021 letter to other CEOs, Larry Fink, the CEO of BlackRock, wrote that 81% of a globally representative selection of BlackRock's sustainable indexes outperformed their parent benchmarks.[7] This finding emphasises the importance of considering sustainability metrics alongside profitability measures when evaluating financial assets, and this includes equity funds.

1.3 Small-Scale Dataset

Every MRR model in this report generated predictions for the two responses. These predictions were evaluated using the average normalised root mean square error, which is the root mean square error divided by the standard deviation averaged across each response (see subsection 3.2.1 for more).

Each MRR model was also applied to a small-scale dataset with Exam Scores to demonstrate explicitly how these models make predictions:

X_1: Hours Studied	X_2: Time Spent on Papers	Y_1: Math Scores	Y_2: Science Scores
5	2	78	80
7	3	85	79
8	4	88	88
3	1	65	70
10	5	92	74

Table 1.1: Study Time vs. Exam Scores

The calculations involving this dataset used precise values throughout, but where written, the intermediary and final results were rounded to 2 decimal places for brevity unless explicitly stated otherwise. The Exam Scores dataset was used for this purpose because its values are straightforward, and the responses have a moderate correlation of 0.526 (3dp), indicating that it requires MRR (see A.2.1 for the full calculation of this).

1.4 Report Notation

In this report, certain common terminology was used. First, the identity matrix was denoted throughout as $\mathbf{I}$ with dimensions subscripted where necessary. Second, there was the correlation and covariance between the responses, which, as stated, MRR considers unlike single-response regression. This consideration is essential for every MRR model as it is what qualifies them as multiple-response.

The correlation between the responses, referred to as the response intercorrelation, was defined as a matrix, $\mathbf{R}$, using the formula: $\mathbf{R} = \mathbf{D}^{-1}\mathbf{\Sigma_Y}\mathbf{D}^{-1}$, where $\text{Cov}(\mathbf{Y}) = \mathbf{\Sigma_Y}$ denotes the covariance matrix of the response variables, or the response covariance matrix, and $\mathbf{D}$ is a diagonal matrix of the standard deviations of the response variables.[8] Each element of $\mathbf{R}$ is computed as:

$$R_{ij} = \frac{(\mathbf{\Sigma_Y})_{ij}}{\sigma_i \sigma_j},$$

where $(\mathbf{\Sigma_Y})_{ij}$ is the covariance between response variables i and j, and σ_i, σ_j are their respective standard deviations.[8] This relation ultimately means that if an MRR model considers the off-diagonal elements in the response covariance matrix, it considers the response intercorrelation and vice versa.

1.5 Report Outline

The rest of this report is broken down into chapters corresponding to the exploratory data analysis, the MRR models and a conclusion comparing all of the models' performances.

Chapter 2 covered an Exploratory Data Analysis of the equity fund dataset. The master dataset included equity and bond funds with significant missing data. This chapter focused on cleaning and refining the dataset, defining the predictor variables, applying imputation techniques to address missing data and identifying relationships between the two responses: ROE and sustainability score.

Chapter 3 introduced Multiple Response Linear Regression. It delved into its underlying theory and explained how it can be combined with Multiple Analysis of Variance and stepwise selection to select the most accurate models. These models were then evaluated on the equity fund dataset.

Chapter 4 progressed to shrinkage-based regression methods. It looked at such methods which are appropriate for MRR, including Multivariate Regression with Covariance Estimation and Reduced Rank Ridge Regression. These methods were then evaluated, and their performance was assessed to determine whether they offer improvements over the MRLR models.

Chapter 5 introduced non-linear modelling through Random Forests, including their adaptation to the multiple-response context with Covariance Regression with Random Forests. Covariance Regression with Random Forests was then applied to the dataset.

Chapter 6 continued the non-linear modelling approach by examining XGBoost. This chapter also outlined Cholesky Decomposition, which enabled XGBoost to be adapted for MRR. Cholesky-Decomposition XGBoost was then evaluated on the equity fund dataset.

Finally, Chapter 7 gave a whole comparison of the performance of all the modelling approaches applied. It identified the most effective method for predicting ROE and sustainability scores, reflected on the findings, and discussed potential directions for future work.

Chapter 2 Exploratory Data Analysis

This chapter provided an outline of the process of cleaning the equity fund dataset, from its source to explaining the predictors, to examining correlation and multi-collinearity, to imputing missing values and testing individual and joint underlying distributions.

2.1 Initial Data Collection and Preprocessing

The dataset was obtained from Kaggle and simplified for this analysis. This report focused on analysing equity funds, but the master dataset contained a mixture of equity and bond funds. The first step was to filter out the bond funds and keep the equities. This filtering was done simply using Excel's filter features. A text filter was used to select the relevant rows within the `category` column. Then, there was an issue with the columns because there were 130 dependent variables in this intermediary dataset. Many variables were for bonds only, so these were removed. Some variables, such as a fund's `isin` number, were irrelevant for analysis, further reducing the number of dependent variables to 29. Another critical factor for the chosen independent variables was to be able to look into how they would affect the dependent variables, so the aim was to have 10-20 variables in the model because this is a manageable amount. Also, a lot of the remaining variables served the same purpose, which would lead to multicollinearity. For example, `management_fees` are a significant component of `ongoing_cost`, so the `management_fees` column was removed. The final step in preprocessing was to remove the rows corresponding to inverse equity funds as they perform the opposite of their underlying benchmark.

2.2 Variable Explanation

`category` gives the type of each equity fund. The cleaned equity fund dataset contained 91 unique equity types consisting of country equities, such as `"Italy Equity"`, and sector equities, such as `"Sector Equity Precious Metals"`. Sector equity funds invest in firms in a specific sector or industry, such as technology, regardless of the companies' country of operation. The focus here is on sector-specific trends and prospects. Country equity funds, in contrast, invest in firms incorporated in or with extensive operations in a specific country, regardless of the industry or sector. The focus in these equity funds is on the political and economic climate of the specific country.

A `rating` is given by analysts assigned to the equity fund, and this takes values {1, 2, 3, 4, 5}. In category 1, funds have excellent performance and high returns relative to their risk of investment. Category 2 includes funds with a solid performance and risk versus return performance, though not as good as category 1. Category 3 indicates funds that are fairly valued and perform in line with market averages, and investors are advised to hold their positions. Category 4 represents funds that underperform their peers. Lastly, category 5 represents funds that underperform over the long term and have high risks and should, therefore, be withdrawn from.
A `risk_rating` gives the market volatility of the equity.[9] This also takes the same values as `rating` but measures the risk of investing in an equity fund. Category 1 funds are less volatile and more stable and thus are suitable for conservative investors. Category 2 funds are moderately more volatile but still quite safe. Category 3 funds provide a medium volatility, balancing risk and potential returns.

Category 4 funds are more volatile, with a greater possibility of returns as well as a greater extent of exposure to market volatility. Lastly, Category 5 investments are subject to high risk and volatility and should be entertained only by investors with a high tolerance for risk.

`rating`, and `risk_rating` are ordinal variables as they exhibit an order by definition. `category` is a categorical variable because it represents different non-numeric groupings, here these are countries and sectors, that do not have a natural order.[10] The rest of the variables are continuous and numerical.

`equity_style_score` is the difference between two investment strategies: growth and value stocks.[11] Growth stocks are expected to bring a lot of capital due to strong growth in the underlying company.[12] Value stocks are either underrated or ignored by the market, and they could gain value eventually.[12] Hence, a lower `equity_style_score` means the stock is more of a value stock, and a higher `equity_style_score` indicates more of a growth stock. `equity_size_score` measures the performance of said equity relative to the company's value.[13] `price_prospective_earnings` gives the ratio between a company's share price and earnings per share. `price_cash_flow_ratio` is a multiple that compares a company's market value to the amount of capital it generates during a select period.[13]

A dividend is the money companies may pay you if you own their shares, and the dividend yield is the dividend paid as a percentage of the share price. The `dividend_yield_factor` is the dividend yield, except it is used to compare the different equities, considering variations in dividend yield performance. `historical_earnings_growth` measures how a stock's earnings per share has grown in the last 5 years. `sales_growth` is the increase in sales of a company's products/ services over time. `asset_cash` is the proportion of a fund's assets held in cash. `holdings_n_stock` is the number of different stock holdings each equity fund holds. A stock holding is the number of other companies in which the equity fund has invested by holding their stocks. `ongoing_cost` is the cost of the daily running of a business.[14] `fund_size` is the sum of all the assets in the fund.

2.3 Exploratory Data Analysis

Now that the dataset has been cleaned and the variables specified, the next step was to perform data analysis in R and find underlying relationships between the variables. To fully represent the dataset, it is best to use actual values rather than simulate any missing values. So, this initial analysis was done with only rows with no missing values.

2.3.1 Underlying Distributions

QQ plots indicate if a variable is normally distributed. The ROE and sustainability scores' QQ plots are shown below:

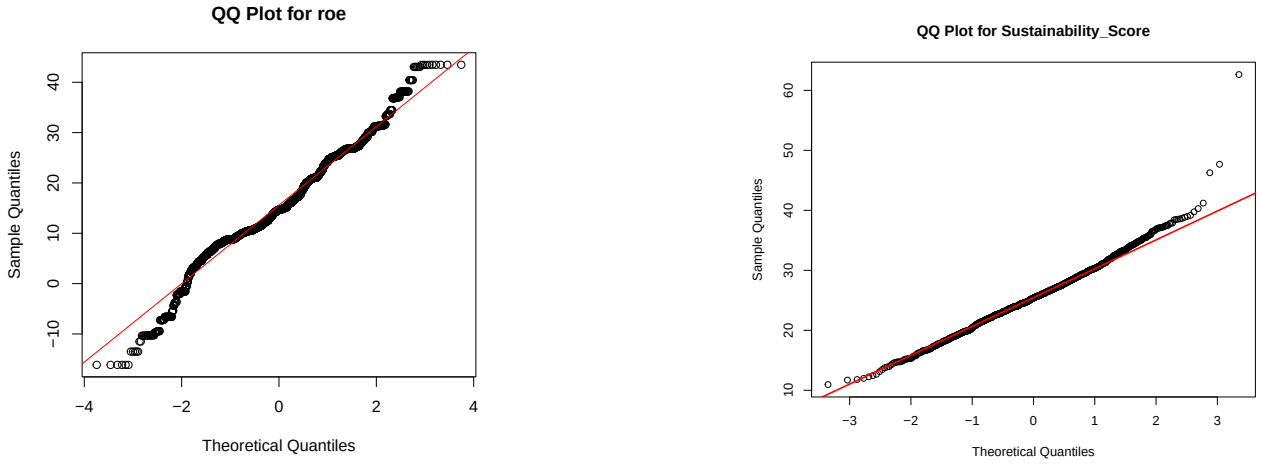

Figure 2.1: QQ Plots for ROE and Sustainability Scores

From the QQ Plots above, there is strong evidence that ROE and sustainability score are normally distributed because the points on the QQ plot approximate a straight line.

2.3.2 Correlation and Multi-Collinearity

It was also essential to examine the correlation within the dependent variables to determine whether any should be removed. High correlation implied multicollinearity, so this was checked before proceeding. After creating the correlation matrix for the dependent variables, there were two high correlation values. One was `0.8099`, which was the correlation between `price_prospective_earnings` and `price_cash_flow_ratio`. Based on the initial definitions of these variables, the latter is a more relevant factor in profitability and sustainability scores, so it is kept. The other high correlation was `-0.8029` between `dividend_yield_factor` and `equity_style_score`. Again, based on the definitions of these variables, the former is a more relevant factor in profitability and sustainability scores, so it was kept. This resulted in 12 predictor variables.

2.3.3 Imputing Missing Values

Although high correlation implies multicollinearity, the opposite is not necessarily true because multicollinearity is not just about pairwise relationships. Multicollinearity arises from linear dependencies among all independent variables. For instance, if Variable A is a linear combination of Variables B and C, say $A = 2B + C$, multicollinearity can occur even if there is no high pairwise correlation 0.7.[15] Therefore, methods such as the Variance Inflation Factor (VIF) are used to detect multicollinearity beyond correlation.[16]

The dataset should also be complete before carrying out a VIF analysis. This involves analysing how the NA values are spread and how often they occur in each row. It was found that 90% of the rows had two NA entries or less, and because the dataset is large, only these rows were used in further analysis. There were three types of missing data here corresponding to the numerical, categorical and ordinal variables that needed to be imputed.

Random forest imputation was used to fill in the missing categorical and ordinal data for several reasons. Although it is computationally more expensive than using methods such as the mode to fill in missing data, it deals well with variables with no association. The Spearman's Rank test confirmed this relationship between the ordinal variables, rating and risk rating. It returned an association of -0.0728, indicating there is sufficient evidence to suggest a lack of association between them.

Random forest imputation also works for categorical and ordinal data through surrogate splits, and outliers influence them less as they use an ensemble of decision trees.[17] Random forest imputations also do not require prior assumptions of the underlying distribution, unlike methods such as linear model imputation, which requires underlying distribution normality, making them flexible.[17] The random forest model was trained on rows with no NA values, and was set to make sure it is predicting categorical and ordinal values rather than numerical ones. Ultimately, the model learned the relationships between the target variable and the other columns in the dataset.

Linear model imputation was used to fill in missing numerical features because it uses the relationships between variables rather than relying on simpler approaches such as mean imputation. This method assumes that each response variable is drawn from an underlying normal distribution, and this is supported by the Q-Q plots shown in Figure 2.1.[18]

In the linear model, the ordinal predictors, which have integer values from 1 to 5, were used as numerical predictors as they have roughly equal spacing among levels.[19] The categorical predictor, `category`, was frequency encoded to prevent it from imposing a hierarchy in the linear model.

2.3.4 Multicollinearity

VIF analysis was employed using a standard multiple response linear regression model of the responses ROE and sustainability score against the features, where the ordinal and categorical predictors were encoded as described previously. Following this, a dummy response variable was added for VIF calculation, and the VIF values were derived from this new model. If any predictors had VIF values greater than 10, they would have been removed due to multicollinearity. In this case, however, all VIF values were below 5. See Figure A.1. Therefore, none of the predictors exhibited multicollinearity.

Finally, this left a cleaned equity fund dataset with 1248 rows, each with individual equity fund information. In addition, there were 14 columns, with 2 response variables and 12 predictors.

2.3.5 Multivariate Normality

Many of the MRR models in this report relied on the assumption of multivariate normality of the response variables. This assumption was evaluated using Mardia's test, which assesses multivariate skewness and kurtosis.[8] The null and alternative hypotheses of Mardia's test here were that the responses were and were not multivariate normally distributed, respectively.

In general, for Mardia's test, if either the kurtosis or skewness statistic deviates significantly from their expected values under multivariate normality, the null hypothesis is rejected.[8]

Mardia's test returned p-values of 0.0732 for skewness and 0.0862 for kurtosis, meaning that in both cases, we failed to reject the null hypothesis of multivariate normality at a 5% significance level. Therefore, there was sufficient evidence to suggest a multivariate normal assumption for the responses (see A.2.2 for more on how these p-values were derived).

However, the relatively low p-values suggested that the assumption may not hold perfectly, and this mild deviation from normality helped explain differences in performance between models that require distributional assumptions and those that do not.

2.4 Response Intercorrelation

As stated in the introduction, considering response intercorrelation is what separate MRR from single-response regression. If the dataset consisted of response variables which shared no correlation, one model could be built for each response, and hence, MRR would not be needed.

Before proceeding with each MRR model, it was worthwhile to briefly examine the relationship between the two response variables: ROE and sustainability score. Figure 2.2 shows a scatter plot of these responses, with a fitted regression line included to highlight their correlation:

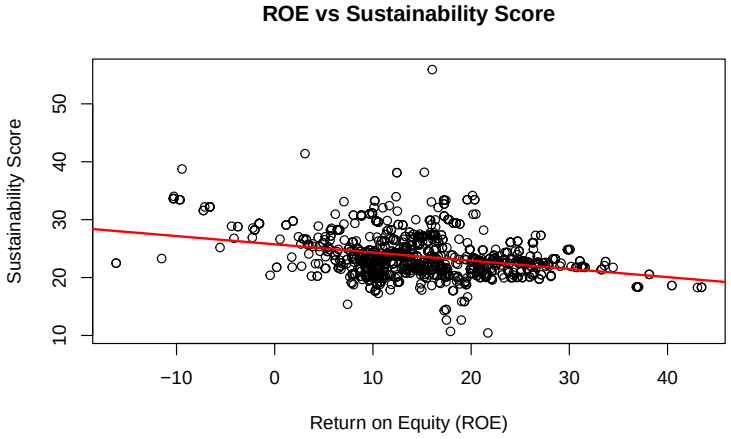

Figure 2.2: Scatter Plot of Return on Equity vs. Sustainability Score

The regression line above shows a slight negative slope, indicating a negative correlation between return on equity and sustainability score. Using the `cor` function in R gives a precise response correlation of `-0.313` (to 3sf). This means that, in general, as ROE increases, the sustainability score decreases.

Here were the full empirical covariance and correlation matrices between the responses (to 3sf):

$$\mathbf{R} = \begin{pmatrix} 1.00 & -0.313 \\ -0.313 & 1.00 \end{pmatrix}, \quad \mathbf{\Sigma_Y} = \begin{pmatrix} 62.0 & -8.79 \\ -8.79 & 12.7 \end{pmatrix}.$$

The negative correlation and covariance might be surprising given the current trends of moving toward a more sustainable climate. However, a lower sustainability score means an equity fund is a more sustainable option, so this does follow the expected trend.

Chapter 3 Linear Regression

This chapter introduces the methodology of the first MRR model, multiple response linear regression, by starting with single-response linear regression. Stepwise selection methods are then introduced, followed by an explanation of how they can be extended to the multiple response case. Multiple analysis of variance is also covered, as it plays a key role in determining which predictors are included in the multiple response models. The accompanying code will then be explained, and the models in this chapter are evaluated using the average normalised root mean square error.

3.1 Theory

3.1.1 Single-Response Linear Regression

Before delving into multiple-response linear regression, let us start with single-response linear regression (SRLR). SRLR models the relationship between a single-response variable and a set of predictor variables. The general form of this model, with the dimensions subscripted, is:

$$\mathbf{Y}_{(n\times 1)} = \mathbf{X}_{(n\times p)}\boldsymbol{\beta}_{(p\times 1)} + \boldsymbol{\epsilon}_{(n\times 1)}, \tag{3.1}$$

where n is the number of observations, p the number of predictors, including the column of ones, $\mathbf{Y}$ the response variable vector, $\mathbf{X}$ the design matrix, $\boldsymbol{\beta}$ the coefficient vector and $\boldsymbol{\epsilon}$ is the error variable vector.

Including a column of ones in $\mathbf{X}$ adds an intercept term to the model. This represents the expected response when all predictors are zero.

When modelling any data, the aim is to minimise the residuals. A residual is the difference between predicted and actual values. Reducing this ensures the model is as accurate as possible, which is done by finding the most optimal coefficient vector, $\boldsymbol{\beta}$. This optimal coefficient vector, also known as the ordinary least squares estimator (OLS), $\hat{\boldsymbol{\beta}}$, is found by minimising the sum of squared residuals. It can also be defined using $\mathbf{X}$ and $\mathbf{Y}$:

$$\hat{\boldsymbol{\beta}}_{\text{OLS}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}.$$

The model's accuracy can then be evaluated using metrics such as the R-squared value, which measures the proportion of variance in the dependent variable explained by the independent variables.

SRLR comes with a few underlying assumptions, which relate to the properties of its error terms. They are linearity, homoscedasticity, normality, and independence of its variables. Linearity means there is a linear relationship between the dependent variables and the independent variables: $E(Y_i|X_i) = \beta_0 + \beta_1 x_i$. The assumption of normality refers to the distribution of residuals, which are assumed to follow a normal distribution with mean zero and constant variance, σ. Homoscedasticity means we expect to see an even spread of residual values and that we can assume the model operates well for all values of the predictor variable. Finally, independence implies that the error covariance matrix is diagonal, and this is given by:

$$\text{Cov}(\boldsymbol{\epsilon}) = \sigma^2\mathbf{I}_n,$$

where σ^2 is the true error variance.

Since the true error variance, σ^2, is unknown, it is typically estimated from the residuals, $\hat{\boldsymbol{\epsilon}} = \mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}$ is the OLS estimator. An unbiased estimator of σ^2 is given by:

$$\hat{\sigma}^2 = \frac{1}{n-p}\hat{\boldsymbol{\epsilon}}^\top \hat{\boldsymbol{\epsilon}},$$

where n is the number of observations and p is the number of predictors. Therefore, the error covariance matrix can be estimated.

If we extended this method to model m responses using m independent single-response regression models, we would be ignoring the covariance between responses, leading to sub-optimal predictions when responses are correlated. See Equation 3.4 for a visual of this.

However, SRLR can be extended to the multiple-response setting while incorporating this structure. This extended approach is known as multiple-response linear regression.

3.1.2 Multiple Response Linear Regression

Multiple-response linear regression (MRLR) is the most basic model in MRR. Its general form is:

$$\mathbf{Y}_{(n\times m)} = \mathbf{X}_{(n\times p)}\,\mathbf{B}_{(p\times m)} + \mathbf{E}_{(n\times m)}, \tag{3.2}$$

with

$$E(e_i) = 0 \quad \text{and} \quad \text{Cov}(e_i, e_k) = \sigma_{ik}\mathbf{I} \quad i,k = 1,2,\ldots,m,$$

where n is the number of observations, m is the number of responses, and p is the number of predictors, including the column of ones, $\mathbf{Y}$ the response variable matrix, $\mathbf{X}$ the design matrix, $\mathbf{B}$ the coefficient matrix and $\mathbf{E}$ is the error variable matrix.

Including a column of ones in $\mathbf{X}$ allows the model to estimate an intercept for each response variable. These intercepts appear as the first row of the coefficient matrix $\mathbf{B}$. Again, this represents the expected value of each response when all predictors are zero.

The first key difference between MRLR and SRLR lies in the structures of the responses, coefficients and errors. These are now matrices rather than vectors, reflecting that multiple response variables are modelled jointly.

As an illustrative example, suppose $n = 4$, $p = 3$, and $m = 2$. Then the regression equation becomes:

$$\mathbf{Y}_{(4\times 2)} = \mathbf{X}_{(4\times 3)}\,\mathbf{B}_{(3\times 2)} + \mathbf{E}_{(4\times 2)}.$$

This expands to:

$$\begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \\ y_{31} & y_{32} \\ y_{41} & y_{42} \end{bmatrix}_{(4\times 2)} = \begin{bmatrix} 1 & x_{11} & x_{12} \\ 1 & x_{21} & x_{22} \\ 1 & x_{31} & x_{32} \\ 1 & x_{41} & x_{42} \end{bmatrix}_{(4\times 3)} \cdot \begin{bmatrix} b_{01} & b_{02} \\ b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}_{(3\times 2)} + \begin{bmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \\ e_{31} & e_{32} \\ e_{41} & e_{42} \end{bmatrix}_{(4\times 2)}.$$

Each row in $\mathbf{Y}$ corresponds to a single observation (i.e., a data point), and each column represents one of the m response variables. The coefficient matrix $\mathbf{B}$ contains a separate column of regression coefficients for each response, and the error matrix $\mathbf{E}$ captures the errors associated with each response for every observation.

The second key difference between SRLR and MRLR lies in the structure of the error terms. MRLR allows for correlation between errors across different response variables within the same observation. This correlation is captured by the error covariance matrix, $\boldsymbol{\Sigma}_{\mathbf{E}} = \{\sigma_{ik}\}$, where σ_{ik} denotes the covariance between the error terms e_i and e_k associated with response variables i and k within the

same observation.[20]

A sample version of the error covariance matrix can be estimated using a formula analogous to the unbiased estimator of σ^2 in SRLR:

$$\hat{\mathbf{\Sigma}}_{\mathbf{E}} = \frac{1}{n-p}\mathbf{E}^\top\mathbf{E},$$

where n is the number of observations and p is the number of predictors.

For illustration, consider again the case where $n = 4$, $p = 3$ and $m = 2$:

$$\hat{\mathbf{\Sigma}}_{\mathbf{E}} = \frac{1}{n-p}\mathbf{E}^\top\mathbf{E} = \begin{bmatrix} e_{11} & e_{21} & e_{31} & e_{41} \\ e_{12} & e_{22} & e_{32} & e_{42} \end{bmatrix} \begin{bmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \\ e_{31} & e_{32} \\ e_{41} & e_{42} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^4 e_{i1}^2 & \sum_{i=1}^4 e_{i1}e_{i2} \\ \sum_{i=1}^4 e_{i1}e_{i2} & \sum_{i=1}^4 e_{i2}^2 \end{bmatrix}. \tag{3.3}$$

If we used 2 independent single-response models, the off-diagonal elements of $\hat{\mathbf{\Sigma}}_{\mathbf{E}}$ would be 0. In this case, $\hat{\mathbf{\Sigma}}_{\mathbf{E}}$ instead becomes:

$$\hat{\mathbf{\Sigma}}_{\mathbf{E}} = \begin{bmatrix} \sum_{i=1}^4 e_{i1}^2 & 0 \\ 0 & \sum_{i=1}^4 e_{i2}^2 \end{bmatrix}, \tag{3.4}$$

which may lead to suboptimal predictions in the presence of correlated responses.

This error covariance structure is important as it qualifies MRLR as an MRR model. It does this because it is used to compute the sample response covariance matrix, $\hat{\mathbf{\Sigma}}_{\mathbf{Y}}$, as follows:

$$\begin{aligned} \hat{\mathbf{\Sigma}}_{\mathbf{Y}} &= \text{Cov}(\mathbf{XB} + \mathbf{E}) \\ &= \text{Cov}(\mathbf{XB}) + \text{Cov}(\mathbf{E}) \\ &= \mathbf{B}^\top\text{Cov}(\mathbf{X})\mathbf{B} + \text{Cov}(\mathbf{E}) \\ &= \mathbf{B}^\top\hat{\mathbf{\Sigma}}_{\mathbf{X}}\mathbf{B} + \hat{\mathbf{\Sigma}}_{\mathbf{E}}, \end{aligned} \tag{3.5}$$

where the linearity property of covariance justifies the first expansion (see A.2.3 for the simplification of $\text{Cov}(\mathbf{XB})$ to $\mathbf{B}^\top\text{Cov}(\mathbf{X})\mathbf{B}$).

Equation 3.5 shows that error interdependence remains, even after accounting for the predictors, and is captured directly by the off-diagonal elements of $\hat{\mathbf{\Sigma}}_{\mathbf{E}}$. Since $\hat{\mathbf{\Sigma}}_{\mathbf{E}}$ appears explicitly in the expression for the response covariance matrix $\hat{\mathbf{\Sigma}}_{\mathbf{Y}}$, modelling this structure is essential to capturing the covariance of the responses when modelling.

Therefore, a non-zero value of the off-diagonal element of $\mathbf{\Sigma}_{\mathbf{E}}$, σ_{ik}, implies that responses i and k tend to vary together, suggesting interdependence between them.

MRLR comes with a few underlying assumptions, which can be extended from the single-response case. Linearity comes from the formula for MRLR, which, when expanded, is:

$$\begin{aligned} Y_1 &= b_{01} + b_{11}x_1 + \cdots + b_{r1}x_r + e_1, \\ Y_2 &= b_{02} + b_{12}x_1 + \cdots + b_{r2}x_r + e_2, \\ &\vdots \\ Y_m &= b_{0m} + b_{1m}x_1 + \cdots + b_{rm}x_r + e_m. \end{aligned}$$

These rows are also independent, another MRLR property from SRLR.

Normality also holds here, but it differs slightly. If we look at using several independent single-response models, each error term is independent and identically normally distributed, with $\epsilon_i \sim \mathcal{N}(0, \mathbf{I})$. However, in MRLR, the error terms, $\mathbf{E}$, share the same distribution. They are assumed to have a multivariate normal distribution with constant variance: $\mathbf{E} \sim \mathcal{N}_m(\mathbf{0}, \mathbf{\Sigma}_{\mathbf{E}})$, where $\mathbf{\Sigma}_{\mathbf{E}}$ is the covariance matrix of errors. The error terms associated with different responses may be correlated.[20]

In spite of the consideration of the correlation between errors, we continue to assume that obser-

vations (i.e., rows of $\mathbf{Y}$) are uncorrelated.[20] To formalise this, let the vector of responses for the ith observation be denoted by:

$$\mathbf{Y}_{(i)} = \mathbf{X}\mathbf{B}_{(i)} + \mathbf{E}_{(i)}, \quad i = 1, 2, \ldots, m,$$

with $\text{Cov}(\mathbf{E}_{(i)}) = \sigma_{ii}\mathbf{I}$ as expected, but the errors associated with different responses within the same trial may be correlated.[20] Each response variable i is modelled independently, but all responses share the same set of predictors.

The predictor matrix $\mathbf{X}$ must also have full column rank to ensure that the least squares estimator of $\mathbf{B}$ is well-defined.[20] Full rank ensures that the least squares estimator, $\hat{\mathbf{B}}_{(i)}$, exists and is unique for each response.[20] Under the assumptions of MRLR, where responses are estimated independently, OLS, which derives $\hat{\mathbf{B}}_{(i)}$, minimises the residual sum of squares (RSS) separately for each response in MRLR as shown here:

$$\text{RSS}_{(i)} = \|\mathbf{Y}_{(i)} - \mathbf{X}\mathbf{B}_{(i)}\|_2^2.$$

This logic for the OLS estimate is just like single-response but applied to each response variable separately:

$$\hat{\mathbf{B}}_{(i)} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}_{(i)}.$$

Collecting these univariate least squares estimates gives:

$$\hat{\mathbf{B}} = \begin{bmatrix}\hat{\mathbf{B}}_{(1)} & \hat{\mathbf{B}}_{(2)} & \cdots & \hat{\mathbf{B}}_{(m)}\end{bmatrix} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top \begin{bmatrix}\mathbf{Y}_{(1)} & \mathbf{Y}_{(2)} & \cdots & \mathbf{Y}_{(m)}\end{bmatrix},$$

or equivalently:

$$\hat{\mathbf{B}}_{\text{OLS}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}.$$

Note the similarity of this formula to the OLS estimator in SRLR.

Using the least squares estimate, $\hat{\mathbf{B}}_{\text{OLS}}$, the matrices of the predicted values and residuals can be made.[20] The predicted values are calculated as:

$$\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}} = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}.$$

And, the residuals are given by:

$$\hat{\mathbf{E}} = \mathbf{Y} - \hat{\mathbf{Y}} = [\mathbf{I} - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top]\mathbf{Y}.$$

The predicted values and residuals can ultimately be used to evaluate MRLR's performance, which is what we did on the equity fund dataset. See Section 3.2.1 for more on how these criteria were used.

3.1.3 Stepwise Selection Methods

Now that we have established MRLR and how it qualifies as an MRR model, we needed to consider which predictors to use. Selecting the most significant predictors is crucial in evaluating model fit. There are always multiple ways to construct a model in a given situation, and we want to be able to judge which of those models is best. Here, we used selection methods.

These stepwise selection methods include forward, backward and bidirectional stepwise selection. These stepwise selection methods were used because they are more computationally efficient than best subset selection, as they consider a smaller set of models.

Forward stepwise selection starts with a model having no predictors and adds the predictor that produces the largest improvement in model fit. Model fit is usually determined using a selection criterion, for example, the Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), or adjusted R^2. At each step, the variable that contributes most significantly to this criterion is added.

Backward stepwise selection begins with the full model as the initial model and successively removes

the predictor that makes the smallest contribution to the model according to the same criterion.

Bi-directional stepwise selection incorporates both forward and backward stepwise selection, with inclusion or exclusion of predictors at every step based on their impact on the model fit as a whole.

Another change from best subset selection is that all these stepwise selection methods consider the statistical impact of dropping variables that were considered previously. [21]

When using all of these selection methods in single-response regression, the "best" model is chosen for every number of predictors, and then the single best model is picked. This "best" criterion is given to the model with the smallest residual sum of squares, or largest R^2. The single "best" model is then picked using a cross-validated prediction error, C_p (or the AIC), BIC, or the adjusted R^2 value.

However, this changes for MRR as the response covariance matrix needs to be considered. Therefore, we needed to re-evaluate our selection criteria. This report used the multiple analysis of variance in conjunction with the average normalised root mean square error.

3.1.4 Analysis of Variance

ANOVA primarily analyses the differences among group means and their associated variances. It helps determine whether there is a significant difference between the means of different groups by comparing the variance within groups to the variance between groups.

It can also be used to evaluate the predictor contribution to response variation. It does this by partitioning total response variance, or the total sum of squares (SST), into 2 components: the sum of squares of regression (SSR) and the sum of squares of error (SSE). The SSR is the variation in responses accounted for by the predictors. The SSE is the variation in responses not accounted for by the predictors. From their setup, these are all related via:

$$\text{SST} = \text{SSR} + \text{SSE}.$$

This decomposition is crucial in evaluating how much the predictors explain response variation. ANOVA results are displayed as a table which generates an F statistic, $F = \frac{\text{MSR}}{\text{MSE}}$, where MSR = SSR/df and MSE = SSE/df. This, in turn, generates a p-value and a low p-value (e.g. $p < 0.05$), and hence, the predictors significantly impact response variation.

Sequential ANOVA is a type of ANOVA which decomposes the regression component of an ANOVA table by considering a sequence of nested models. This approach allows us to assess how the unexplained variation in the response changes as predictors are added to the model one at a time.

Here is how it works: first consider a sequence of m nested models $M_1 \subset M_2 \subset \cdots \subset M_m$, where each model M_j contains all predictors from M_{j-1} plus one additional predictor. Each model M_j is represented as:

$$M_j : \mathbb{E}[Y \mid X] = b_1 + b_2 X_2 + \cdots + b_{p_j} X_{p_j},$$

and the next model adds an extra predictor:

$$M_{j+1} : \mathbb{E}[Y \mid X] = b_1 + b_2 X_2 + \cdots + b_{p_j} X_{p_j} + b_{p_{j+1}} X_{p_{j+1}}.$$

Sequential ANOVA was useful in this report as it can be extended to the multivariate setting to evaluate which predictors are most appropriate for inclusion in MRLR.

3.1.5 Multiple Analysis of Variance

Multivariate Analysis of Variance (MANOVA) extends ANOVA to multiple responses. It comes with the assumption that the responses need to have a multivariate normal distribution, like MRLR. [22]

MANOVA tests if the means of several dependent variables differ across independent variables. It also compares groups on a set of dependent variables simultaneously rather than conducting separate

ANOVAs for each dependent variable.[23] This report uses MANOVA to assess the impact of the predictors across the joint variation in responses. It assesses this using variance-covariance matrices instead of scalar variance values.

The total response variation, or the total sum of squares and cross products (SSCP) matrix, $\mathbf{T}$, is partitioned into 2 matrices, like ANOVA. The first is the explained SSCP matrix, $\mathbf{H}$, representing the response variation explained by the predictor variable(s).[24] The second is the unexplained SSCP matrix, $\mathbf{W}$, representing the response variation not accounted for by the predictors.[24] By their set-up, $\mathbf{W}$ and $\mathbf{H}$ relate through the total SSCP matrix: $\mathbf{T} = \mathbf{H} + \mathbf{W}$.

MANOVA is suitable for MRR because the total SSCP matrix, $\mathbf{T}$, is an unscaled version of the response covariance matrix, i.e.:

$$\mathbf{T} = (n-1)\mathbf{\Sigma_Y},$$

where n is the number of observations. This relation means MANOVA accounts for response intercorrelation by directly considering response covariances, which is ignored when using several independent univariate ANOVAs. Before proceeding with these values, let us look at how they are computed.

From single-response ANOVA, $\mathbf{H}$ and $\mathbf{W}$ and $\mathbf{T}$ are extensions of SSE, SSR and SST, respectively. The unexplained SSCP matrix $\mathbf{W}$ is calculated as follows:

$$\mathbf{W} = (\mathbf{Y} - \hat{\mathbf{Y}})^\top (\mathbf{Y} - \hat{\mathbf{Y}}),$$

where $\mathbf{Y}$ is the $n \times m$ matrix of observed response values, and $\hat{\mathbf{Y}}$ is the $n \times m$ matrix of predicted response values from the model. $\hat{\mathbf{Y}}$ is computed using $\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}}$, where $\hat{\mathbf{B}}$ is the MRLR OLS estimator. The explained SSCP matrix $\mathbf{H}$ is calculated as follows:

$$\mathbf{H} = (\hat{\mathbf{Y}} - \bar{\mathbf{Y}})^\top (\hat{\mathbf{Y}} - \bar{\mathbf{Y}}),$$

where $\bar{\mathbf{Y}}$ is the overall mean response vector, $n \times 1$ given by $\bar{\mathbf{Y}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{Y}_i$.
The total SSCP matrix $\mathbf{T}$ is calculated as follows::

$$\mathbf{T} = (\mathbf{Y} - \bar{\mathbf{Y}})^\top (\mathbf{Y} - \bar{\mathbf{Y}}).$$

To test significance, MANOVA has multiple tests it can use. This report uses the Wilks' Lambda (Λ) test, which measures the proportion of variance not explained by the independent variables:

$$\Lambda = \frac{|\mathbf{W}|}{|\mathbf{T}|},$$

where $|\mathbf{W}|$ and $|\mathbf{T}|$ are the determinants of the unexplained and total SSCP matrices, respectively.[24] A small Λ (close to 0) indicates the predictors explain a large proportion of response variation, and a large Λ (close to 1) indicates the predictors explain a small proportion of response variation.

The Wilks' Lambda value is useful because it can be transformed into an F-statistic:

$$F = \frac{(1-\Lambda)/m}{\Lambda/(N-m-1)}$$

where: m is the number of dependent variables and N is the total sample size. The null hypothesis in MANOVA is that all group means for the dependent variables are equal. So, a significant F-test ($p < 0.05$) suggests at least one of the dependent variables significantly differs across groups.[23]. However, this report uses the difference in Wilks' Lambda values as specified by Sequential MANOVA.

Sequential MANOVA extends sequential ANOVA by testing predictors one at a time in a nested sequence of models. It works the same as sequential ANOVA, but at each step, we compute the Wilks'

Lambda (Λ_j) value for each model:

$$\Lambda_1 \geq \Lambda_2 \geq \cdots \geq \Lambda_m,$$

where the difference: $\Lambda_j - \Lambda_{j+1}$ is the additional variance explained by the newly added predictor. A significant drop in Λ indicates that the new predictor significantly improves the model. [23]

Another advantage of using Wilks' Lambda is that it prevents the inflation of Type I errors. If multiple dependent variables are analysed separately using ANOVA, the probability of finding at least one false positive increases. Since MANOVA jointly examines multiple dependent variables, it controls for the increased risk of false positives by adjusting for intercorrelations among the responses. [23] This provides a more accurate assessment of significant predictors.

3.2 Application

3.2.1 Model Evaluation and Performance

Multiple models were developed and evaluated throughout this report. Their performance was assessed using the average normalised root mean square error (ANRMSE).

While the root mean square error (RMSE) measures model error for each response variable separately, it retains the original units, making direct comparisons difficult. In contrast, the normalised RMSE (NRMSE) scales the error by the standard deviation of each response, returning a unitless and comparable measure of model performance. Since two response variables were considered, the NRMSE was computed for each and then averaged to obtain a single summary statistic, the ANRMSE.

Here is the formula for the ANRMSE:

$$\text{ANRMSE} = \frac{1}{m}\sum_{j=1}^{m}\left(\frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_{ij}-\hat{y}_{ij})^2}}{s_j}\right), \tag{3.6}$$

where m is the number of responses, n is the number of observations, y_{ij} is the observed value for response j, observation i, $\hat{y}_{ij}$ is the predicted value and s_j is the sample standard deviation of the jth response given by:

$$s_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_{ij}-\bar{y}_j)^2}. \tag{3.7}$$

The ANRMSE gives the proportion of natural variability in the responses not accounted for by the model. An excellent model has a 0-0.5 ANRMSE, a good model has a 0.5-0.6 ANRMSE, a satisfactory model has a 0.6-0.7 ANRMSE, and an unsatisfactory model has a > 0.7 ANRMSE. [25]

3.2.2 Standard Multiple Linear Regression

Before proceeding with the analysis, we verified that the equity fund dataset followed the necessary assumptions from the Sequential MANOVA Stepwise Selection and MRLR. Most of the assumptions from MRLR and MANOVA were all catered for in the exploratory data analysis through methods such as Mardia's Test for Skewness and Kurtosis. However, the assumption of a full rank predictor matrix, $\mathbf{X}$, had yet to be checked. This was confirmed using QR-decomposition, which showed that the rank of $\mathbf{X}$ equalled the number of predictors.

The first model evaluated on the equity fund dataset was an MRLR using all of the available predictors. The residuals from the model were used to calculate the ANRMSE, which was `0.7606`. This high value indicated that this model did not perform well on the equity fund dataset. This makes sense because the model was not built to understand the underlying complexity of the equity fund dataset, so it underfits the data. Stepwise selection will consider this more, and it had been adapted

to the multiple response case using Sequential MANOVA.

3.2.3 Code Explanation

All of these selection methods had to be manually coded because the `stepAIC` command, which performs stepwise selection in R, only applies to single-response regression models.

The selection process begins with different initial models depending on the chosen method. In forward selection, the model starts with no predictors. However, in backward and bidirectional selection, it starts with all available predictors included.

At each iteration, the algorithm evaluates the effect of adding or removing each candidate predictor using the functions `add_predictors` and `remove_predictors`, respectively. The criterion for inclusion or exclusion is based on the change in Wilks' Lambda (Λ), computed as:

$$\Delta\Lambda = \Lambda_{\text{previous}} - \Lambda_{\text{new}},$$

where $\Lambda_{\text{previous}}$ is the Wilks' Lambda from the current model, and Λ_{new} is the value after including or excluding a given predictor.

The predictor associated with the largest reduction in Wilks' Lambda is selected or removed, depending on the chosen selection method. If no predictor reduces Wilks' Lambda, so $\Delta\Lambda < 0$ for all candidates, the model remains unchanged for that step.

A key consideration in this procedure that had not been discussed thus far was the risk of overfitting. Just as in single-response regression, where the residual sum of squares always decreases with the inclusion of more variables, Wilks' Lambda in MRR also tends to decrease monotonically as predictors are added. If used alone, this could result in models that overfit the data.

To address this, the algorithm includes an additional evaluation step using the ANRMSE, which is computed via k-fold cross-validation (CV) using the `calculate_nrmse` function. A predictor is only accepted into or removed from the model if its inclusion or exclusion improves Wilks' Lambda and reduces ANRMSE. This ensures that model selection favours variables that improve performance on unseen data. The new model formed is then stored in a list named `candidate_models`.

The process continues iteratively. In bidirectional selection, the algorithm stops if two consecutive iterations fail to improve the model. In forward or backward selection, it stops once all available predictors have been evaluated without improving Wilks' Lambda and ANRMSE.

The final model is then selected as the one with the lowest ANRMSE from the `candidate_models` list. If two models have a similar performance, the simpler model is preferred due to its interpretability.

For brevity, this overall procedure was referred to as Sequential MANOVA Stepwise Selection for the remainder of this paper. The method was first illustrated using a simpler dataset of Study Time versus Exam Scores before being applied to the full equity fund dataset.

3.2.4 Small-Scale Example

Let us take a small-scale dataset evaluating correlated Mathematics and Science Scores against factors that may influence them:

X_1: Hours Studied	X_2: Time Spent on Papers	Y_1: Math Scores	Y_2: Science Scores
5	2	78	80
7	3	85	79
8	4	88	88
3	1	65	70
10	5	92	74

Table 3.1: Study Time vs. Exam Scores

Let us now walk through Sequential MANOVA Forward Stepwise Selection. The first step is evaluating each predictor one at a time to see which predictor gives the largest Wilks' lambda difference. Let us look at X_1 as a predictor to demonstrate how the Wilks' Lambda difference is computed.

The first step is computing the total, explained and hence unexplained SSCP matrices. This starts by first computing the means to get $\bar{\mathbf{Y}} = [\bar{Y_1}, \bar{Y_2}]^\top$:

$$\bar{Y_1} = \frac{78+85+88+65+92}{5} = 81.6, \quad \bar{Y_2} = \frac{80+79+88+70+74}{5} = 78.2.$$

Next, compute the deviations from the mean as follows:

$Y_{1,i} - \bar{Y_1}$	$Y_{2,i} - \bar{Y_2}$
-3.6	1.8
3.4	0.8
6.4	9.8
-16.6	-8.2
10.4	-4.2

The total SSCP matrix, $\mathbf{T}$, is then given by:

$$\mathbf{T} = \begin{bmatrix} \sum_i (Y_{1,i} - \bar{Y_1})^2 & \sum_i (Y_{1,i} - \bar{Y_1})(Y_{2,i} - \bar{Y_2}) \\ \sum_i (Y_{1,i} - \bar{Y_1})(Y_{2,i} - \bar{Y_2}) & \sum_i (Y_{2,i} - \bar{Y_2})^2 \end{bmatrix},$$

where i goes from 1 to 5, which is the number of observations. Now, compute each term:

$$\sum_i (Y_{1,i} - \bar{Y_1})^2 = 449.2, \quad \sum_i (Y_{2,i} - \bar{Y_2})^2 = 184.8, \quad \sum_i (Y_{1,i} - \bar{Y_1})(Y_{2,i} - \bar{Y_2}) = 634.$$

Thus,

$$\mathbf{T} = \begin{bmatrix} 449.2 & 634 \\ 634 & 184.8 \end{bmatrix}.$$

The next step is to compute the unexplained SSCP Matrix, $\mathbf{W}$, which uses the prediction from MRLR: $\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}}$. Here, we want to compute the OLS estimator for MRR, $\hat{\mathbf{B}}$. This is done like in single-response, by simply using $\hat{\mathbf{B}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$. In our case:

$$\mathbf{Y} = \begin{bmatrix} 78 & 85 & 88 & 65 & 92 \\ 80 & 79 & 88 & 70 & 74 \end{bmatrix}^\top \quad \text{and} \quad \mathbf{X} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 5 & 7 & 8 & 3 & 10 \end{bmatrix}^\top.$$

Remember, only X_1 is being used here because a model with 1 predictor is being evaluated first. Plugging these values into the formula for the OLS estimator, $\hat{\mathbf{B}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$, gives:

$$\hat{\mathbf{B}} = \begin{bmatrix} 56.47 & 72.23 \\ 3.81 & 0.90 \end{bmatrix}.$$

Therefore, the predicted values are: $\hat{Y}_{1,i} = 56.47 + 3.81 X_{1,i}$ and $\hat{Y}_{2,i} = 72.23 + 0.90 X_{1,i}$. Now, we can compute $\mathbf{W} = (\mathbf{Y} - \hat{\mathbf{Y}})^\top (\mathbf{Y} - \hat{\mathbf{Y}})$ to 4sf:

$$\mathbf{W} = \begin{bmatrix} 25.73 & 50.86 \\ 50.86 & 160.9 \end{bmatrix}.$$

Next, we compute the Wilks' Lambda value using the determinants of these matrices:

$$\Lambda = \frac{|\mathbf{W}|}{|\mathbf{T}|} = \frac{1553.08}{60090.2} = 0.0258,$$

This value is very low, which means the model including X_1 is explaining a large proportion of the variation in the responses, which is ideal.

Next, the difference between this and the previous Wilks' Lambda value is computed. The previous case is just when we have the column of ones in our design matrix, $\mathbf{X}$. This means the model only estimates the mean of each response, so no actual predictors are included to explain differences in $\mathbf{Y}$. Hence, the predicted values are the column means. As a result, the explained SSCP matrix, $\mathbf{H}$, is 0. Therefore, the unexplained SSCP matrix, $\mathbf{W}$ equals the total explained SSCP matrix, $\mathbf{T}$, and $\Lambda = 1$.

Computing the differences gives: 1-0.0258 = 0.9742. This is compared to the Wilks' Lambda Difference from using only X_2, and the predictor with the largest difference is added to the model.

To calculate the ANRMSE, say for the model with only the X_1 predictor, start by computing the predictions. For MRLR, the predictions are computed using $\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}}$ where $\mathbf{X}$ and $\hat{\mathbf{B}}$ have been set out previously for this model:

$$\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}} = \begin{bmatrix} 1 & 5 \\ 1 & 7 \\ 1 & 8 \\ 1 & 3 \\ 1 & 10 \end{bmatrix} \begin{bmatrix} 56.47 & 72.23 \\ 3.81 & 0.90 \end{bmatrix} = \begin{bmatrix} 75.52 & 76.73 \\ 83.14 & 78.53 \\ 86.95 & 79.43 \\ 67.90 & 74.93 \\ 94.57 & 81.23 \end{bmatrix}.$$

From here, the residuals can be computed as:

$$\mathbf{Y} - \hat{\mathbf{Y}} = \begin{bmatrix} 78 & 80 \\ 85 & 79 \\ 88 & 88 \\ 65 & 70 \\ 92 & 74 \end{bmatrix} - \begin{bmatrix} 75.52 & 76.73 \\ 83.14 & 78.53 \\ 86.95 & 79.43 \\ 67.90 & 74.93 \\ 94.57 & 81.23 \end{bmatrix} = \begin{bmatrix} 2.49 & 3.25 \\ 1.88 & 0.44 \\ 1.07 & 8.53 \\ -2.89 & -4.95 \\ -2.55 & -7.27 \end{bmatrix}$$

This gives the standard deviation of each response as: $(10.60, 6.80)$ after plugging this into Equation 3.7 with $n = 5$ and $m = 2$. Finally, plugging all these values into Equation 3.6. gives an NRMSE of each response variable as (to 3dp):

$$\begin{bmatrix} 0.214 \\ 0.835 \end{bmatrix}.$$

This gives an overall ANRMSE of 0.524 (3dp) in this example.

3.2.5 Applying Stepwise Selection

This method was then applied to the equity fund dataset with roe and sustainability_score as the responses, and MANOVA Stepwise Selection was applied, with `category` frequency encoded.

Both Forward and bi-directional Stepwise Selection led to the same model with all the predictors kept. This model was the sum of all the predictors in the dataset and, hence, had the same ANRMSE.

Backward elimination returned a different model to the other selection methods. The difference was that it did not include `risk_rating`. This gave a higher ANRMSE than the previous model with `0.7924`. It makes sense for the selection methods to potentially produce a different model because each selection method has different starting points and directions, which can lead to other paths through the model and, ultimately, a different set of selected predictors.

3.2.6 Extending and Evaluating Stepwise Selection

Although there were already a large number of predictors in this dataset, extending the model to consider non-linear and interaction terms improved its performance. To include these, all that had to be done was to make a new function which generated the new predictors. When generating the non-linear terms, the model improved as shown by the ANRMSE reducing to `0.6893`. Using the interaction terms further improved, and the ANRMSE was reduced to `0.5613`, a noteworthy improvement.

One counterargument to introducing these terms was that these models overfit the data, but the implementation of CV prevented this. Another potential issue was that collecting this extra information could be complex. However, all this data was already available.
The table below provides an overview of all the selection models evaluated in this chapter.

Model	**ANRMSE**
Full Model - MRLR using all the Predictors	0.7606
Forward Stepwise Selection - Sum of all the Predictors	0.7606
Backward Stepwise Selection - One predictor removed	0.7924
Bidirectional Stepwise Selection - Sum of the Predictors	0.7606
Bidirectional Stepwise Selection with Interaction Terms	0.5613
Bidirectional Stepwise Selection with Non-Linear Terms	0.6893

Table 3.2: Selection Methods and their ANRMSE Values

See A.2 for a graphical illustration of Model 2: Forward Stepwise Selection in action.

Stepwise selection comes with multiple benefits. Firstly, it is easy to implement. Although manual code had to be made in this case, it proved to be easy to test on the dataset. Stepwise selection reduces the work of model building to the click of a button and often yields simple results.[26] Also, stepwise selection can sift through lots of predictors to identify potential relationships, mainly when dealing with high-dimensional datasets.

However, they come with a few drawbacks. Firstly, although CV can be employed, they are prone to over-fitting because they can find idiosyncrasies in the population that do not exist.[26] Also, the stepwise selection method produces models that are highly sensitive to slight variations in the data, causing inconsistent results. Another issue of this method is inflated effect sizes, so even when the model identifies valid predictors, it overestimates their effect sizes.[26] This means that the impact of significant predictors on the response variables, ROE and sustainability score, will be inflated.

Shrinkage Methods were the next type of regression model evaluated in this report. None of the stepwise selection drawbacks apply to shrinkage methods. Shrinkage Methods address them by penalising the model coefficients, which prevents over-fitting and reduces sensitivity to outliers.

Chapter 4 Shrinkage Methods

This chapter delves into Shrinkage Methods, focusing on Multivariate Regression with Covariance Estimation and Reduced Rank Ridge Regression, which extend Lasso and Ridge Regression. First, it introduces the theory behind Lasso and Ridge Regression and then extends them to multiple responses. Next, it delves into the theory of Multivariate Regression with Covariance Estimation and Reduced Rank Ridge Regression, and finally applies these concepts to the equity fund dataset.

4.1 Theory

The methods given in Chapter 3 were discrete model search methods, where the best model is selected based on information criteria. The next type of model this report will consider is shrinkage methods, which are a continuous model search approach. In these methods, the entire model is trained, but the estimated regression coefficients are reduced in magnitude and, in some cases, even set to zero. This is achieved by minimising a loss function that combines model fit with a penalty on coefficient size.

4.1.1 Ridge Regression

Ridge regression shrinks estimated regression coefficients towards zero by minimising the following penalised loss function:

$$\mathcal{L}(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2, \tag{4.1}$$

where $||\cdot||_2$ is the ℓ_2-norm, $\lambda \geq 0$ is a regularisation parameter, and all the variables follow on from SRLR; see Equation 3.1. The tuning parameter is crucial as it controls the strength of the penalty, balancing the trade-off between model fit and coefficient magnitude. Therefore, as λ increases, the penalty term gets larger, so in order to minimise the entire loss function, the regression coefficients get smaller.

Ridge regression minimises the cost function, and this can be shown by differentiating the cost function with respect to $\boldsymbol{\beta}$. This gives the ridge regression closed-form coefficient estimator as:

$$\hat{\boldsymbol{\beta}}^{\text{Ridge}} = \left(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}_p\right)^{-1}\mathbf{X}^\top\mathbf{Y}. \tag{4.2}$$

See A.2.4 for the full derivation of the closed-form estimator.

Ridge regression is particularly useful when the predictors exhibit high multicollinearity or when the number of predictors p is close to or exceeds the number of observations n. In such cases, the matrix $\mathbf{X}^\top\mathbf{X}$ becomes singular, making the OLS estimator unstable or undefined. The ridge penalty term $\lambda\|\boldsymbol{\beta}\|_2^2$ regularises the inversion by ensuring that $\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}$ is always invertible, leading to more stable coefficient estimates. This regularisation reduces the variance of the model at the expense of a small increase in bias, which generally improves model prediction accuracy when there is predictor multicollinearity or limited sample size.

Ridge regression is also faster than model-search based methods, such as stepwise selection in Subsection 3.1.3, as it only trains one model.

Ridge regression estimates are obtained by solving the least squares problem subject to a constraint

on the sum of squared coefficients: $\|\boldsymbol{\beta}\|_2^2 = \sum_{j=1}^{p} \beta_j^2 \leq c$, where p is the number of predictors and c is a positive constant. This constraint defines an ℓ_2-ball, which in the two-dimensional case, so $p = 2$, corresponds to a filled circle and provides a geometric interpretation of the ridge penalty. See the right-hand graphic in Figure 4.1 for this interpretation:

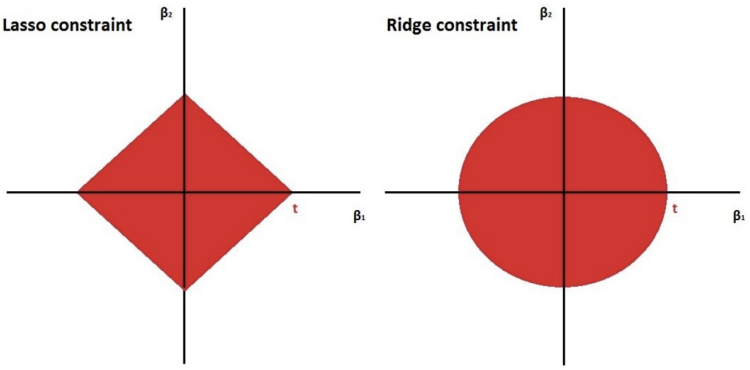

Figure 4.1: Left: the Lasso constraint $|\beta_1| + |\beta_2| \leq c$, Right: the Ridge constraint $\beta_1^2 + \beta_2^2 \leq c$.

In higher dimensions, so as p increases, the ℓ_2-norm forms a hypersphere, which is a higher-dimensional generalisation of the circle.

As the constraint region is circular, the ridge regression solution typically lies within the interior of the ball rather than on the axes. As a result, although ridge regression shrinks coefficients toward zero, it does not set any of them exactly to zero, even those corresponding to irrelevant predictors. This lack of sparsity is a key limitation, as ridge regression cannot perform feature selection. Ideally, a simpler model would exclude uninformative variables entirely. Lasso regression does exactly this.

4.1.2 Lasso Regression

Lasso regression removes irrelevant features by shrinking their coefficients exactly to zero. Instead of penalising the squared magnitude of the coefficients like ridge, lasso penalises their absolute values. It shares the same aim of minimising a loss function as Ridge but with a different penalty term:

$$\mathcal{L}(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1, \tag{4.3}$$

where $||\cdot||_1$ is the ℓ_1-norm and the other variables follow from Equation 4.1.

Unlike ridge regression, Lasso has no closed-form solution because the ℓ_1-norm penalty, $\|\boldsymbol{\beta}\|_1$, is not differentiable at zero, making the optimisation problem non-smooth and requiring iterative algorithms such as coordinate descent to solve it.

Like ridge regression, lasso introduces bias in exchange for reduced variance, which can improve prediction accuracy. Lasso is particularly useful in high dimensions, as it is scalable and can perform automatic feature selection by shrinking some coefficients exactly to zero. Both ridge and lasso can also mitigate multicollinearity. However, for the equity fund dataset used in this report, such issues were largely addressed during preprocessing. See Chapter 2.

As with ridge regression, this optimisation problem can be viewed as least squares subject to a different constraint, the ℓ_1-norm: $\|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j| \leq c$. Geometrically, in the two-dimensional case, when there are two predictors, this constraint defines a diamond-shaped region in the parameter space. The corners of this diamond lie on the coordinate axes, encouraging sparsity by increasing the chance that the solution lies exactly on one of them. This is why lasso can shrink some coefficients exactly to zero. See the left-hand graphic in Figure 4.1 for an illustration of this constraint.

In higher dimensions, the ℓ_1-norm forms a cross-polytope. This is a higher-dimensional generalisation of the diamond, which continues to promote sparsity by encouraging axis-aligned solutions.

These regularisation methods, developed for single-response models, can also be extended to MRR.

4.1.3 Shrinkage Methods for MRR

Ridge Regression extends to MRR by adding a Frobenius norm penalty to the coefficient matrix $\mathbf{B}$:

$$\min_{\mathbf{B}} \frac{1}{2}\|\mathbf{Y} - \mathbf{XB}\|_F^2 + \lambda\|\mathbf{B}\|_F^2,$$

where $\lambda \geq 0$ is the regularisation parameter, $\|\mathbf{B}\|_F^2 = \sum_{j=1}^{q}\sum_{i=1}^{m} b_{ij}^2$ denotes the Frobenius norm penalty, and the matrices $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{B}$ are defined as in Equation 3.2.[27]

In single-response ridge regression, the penalty is given by the squared ℓ_2-norm of the coefficient vector. See Equation 4.1. However, since the standard ℓ_2-norm applies only to vectors, it cannot directly penalise all the entries in the coefficient matrix. The Frobenius norm replaces it because it extends the ℓ_2-norm to matrices, summing the squared entries across all rows and columns in $\mathbf{B}$.

The closed-form solution for $\hat{\mathbf{B}}^{\text{Ridge}}$ in multiple response ridge regression is:

$$\hat{\mathbf{B}}^{\text{Ridge}} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}_m)^{-1}\mathbf{X}^\top\mathbf{Y}, \tag{4.4}$$

which follows from single-response ridge regression. See Equation 4.2.

Similarly, Lasso Regression can also be extended to MRR using the ℓ_1-norm penalty:

$$\min_{\mathbf{B}} \frac{1}{2}\|\mathbf{Y} - \mathbf{XB}\|_2^2 + \lambda\|\mathbf{B}\|_1.$$

As in single-response lasso regression, the ℓ_1-penalty shrinks some elements of $\mathbf{B}$ to exactly zero, enabling variable selection in the regression model.

Despite the flexibility of multivariate ridge and lasso, these methods do not account for the intercorrelation between response variables, which is critical in MRR. However, there are extensions of them that address this limitation. Two such approaches are reduced rank ridge regression and multivariate regression with covariance estimation.

4.1.4 Reduced Rank Ridge Regression

In MRLR, predictions are generated using $\mathbf{Y} = \mathbf{X}\hat{\mathbf{B}}_{\text{OLS}}$ where $\hat{\mathbf{B}}_{\text{OLS}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}$.

Reduced rank ridge regression (RRRR) modifies the estimation of the coefficient matrix using multiple response ridge regression (see equation 4.1.3) and a rank constraint. It aims to solve the following optimisation problem:

$$\hat{\mathbf{B}}(\lambda, r) = \arg\min_{\mathbf{B}:\text{rank}(\mathbf{B})\leq r} \|\mathbf{Y} - \mathbf{XB}\|_F^2 + \lambda\|\mathbf{B}\|_F^2, \tag{4.5}$$

where $r \leq \min\{p, m\}$ controls the rank of $\mathbf{B}$ and $\lambda \geq 0$ is the ridge parameter.
Note: m is the number of responses and p is the number of predictors.

Although the unconstrained ridge solution $\hat{\mathbf{B}}^{\text{Ridge}}$ is typically full rank, which is confirmed on the equity fund dataset via QR-decomposition (see Subsection 3.2.2), a rank constraint remains useful. It encourages dimensionality reduction and reflects the assumption that the responses are caused by a small number of latent variables.[27] This results in a model that not only captures the underlying low-dimensional structure of the data but is also interpretable.

Before proceeding, assume that $\mathbf{X}$ and $\mathbf{Y}$ have been mean-centred, so each column has zero mean. As a result, the number of predictors, p, excludes the intercept term, and the column of ones is omitted from the design matrix. This assumption will be further justified later in this section.

The first step involves transforming this problem into a standard Reduced Rank Regression problem on an augmented dataset which incorporates the ridge penalty.[27] This augmentation involves adding artificial rows to the original predictor and response matrices.

The augmented least squares problem is equivalent to applying the closed-form ridge estimator from Equation 4.4, when no rank constraint is applied. Augmentation is used instead in RRRR because it allows the RRRR problem to be solved more efficiently using Singular Value Decomposition (SVD), rather than solving the regularised optimisation problem directly.

Augmentation is also performed separately for each fixed λ, as the optimal value is not known in advance and is selected via CV in this report.

Here is augmentation written mathematically:

$$\mathbf{X}^*_{(n+p)\times p} = \begin{pmatrix} \mathbf{X} \\ \sqrt{\lambda}\mathbf{I} \end{pmatrix}, \quad \mathbf{Y}^*_{(n+p)\times m} = \begin{pmatrix} \mathbf{Y} \\ \mathbf{0} \end{pmatrix}.$$

The augmented data leads to the following rank-constrained optimisation problem:

$$\hat{\mathbf{B}}(\lambda, r) = \underset{\text{rank}(\mathbf{B})\leq r}{\arg\min} \ \|\mathbf{Y}^* - \mathbf{X}^*\mathbf{B}\|_F^2,$$

where $\mathbf{B}$ can be estimated using OLS. See A.2.5 for the full derivation of this.

This problem can be simplified further due to the orthogonality of the OLS estimate. This orthogonal projection property means the squared error loss function can be decomposed into 2 parts:

$$\|\mathbf{Y}^* - \mathbf{X}^*\mathbf{B}\|_F^2 = \|\mathbf{Y}^* - \hat{\mathbf{Y}}^*_R\|_F^2 + \|\hat{\mathbf{Y}}^*_R - \mathbf{X}^*\mathbf{B}\|_F^2,$$

where $\hat{\mathbf{Y}}^*_R = \mathbf{X}^*\hat{\mathbf{B}}^*_R$ denotes the Ridge Regression estimate.[27] The first term of this equation does not involve $\mathbf{B}$ so the objective function can be further simplified into:

$$\hat{\mathbf{B}}(\lambda, r) = \arg \min_{\text{rank}(\mathbf{B})\leq r} ||\hat{\mathbf{Y}}^*_R - \mathbf{X}^*\mathbf{B}||_F^2. \tag{4.6}$$

The next step involves decomposing the matrix $\hat{\mathbf{Y}}^*_R$ This starts by taking its SVD:

$$\hat{\mathbf{Y}}^*_R = \sum_{i=1}^{\tau} d_i \mathbf{u}_i \mathbf{v}_i^\top,$$

where the d_i's denote the singular values, $\mathbf{u}_i \in \mathbb{R}^{n\times 1}$ and $\mathbf{v}_i \in \mathbb{R}^{m\times 1}$ denote the left and right singular vectors of $\hat{\mathbf{Y}}^*_R$, respectively.[27]

This step is important because RRRR captures response intercorrelation through $\mathbf{v}_i$. This can be proven by how the sample response covariance, $\mathbf{\Sigma_Y}$, is defined:

$$\mathbf{\Sigma_Y} = \frac{1}{n-1}(\mathbf{Y} - \bar{\mathbf{Y}})^\top(\mathbf{Y} - \bar{\mathbf{Y}}) = \frac{1}{n-1}\mathbf{Y}^\top\mathbf{Y}.$$

The last equality holds because we have mean-centred $\mathbf{Y}$. See A.2.5 for how $\mathbf{\Sigma_Y}$ and $\mathbf{Y}^\top\mathbf{Y}$ would relate without mean centring.

This is important because the SVD of $\mathbf{Y}$ in matrix form is: $\mathbf{Y} = \mathbf{UDV}^\top$. Now, compute $\mathbf{Y}^\top\mathbf{Y}$ using its SVD form:

$$\begin{aligned} \mathbf{Y}^\top\mathbf{Y} &= \mathbf{VD}^\top\mathbf{U}^\top\mathbf{UDV}^\top \\ &= \mathbf{VD}^2\mathbf{V}^\top, \end{aligned}$$

where the final equality comes because $\mathbf{U}$ is orthonormal, meaning $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$, and $\mathbf{D} = \mathbf{D}^\top$.[28] This means $\mathbf{V}$ clearly diagonalises $\mathbf{Y}^\top\mathbf{Y}$. Therefore, it directly represents the principal directions of

response covariance. Thus, the columns of $\mathbf{V}$, $\mathbf{v}_i$, capture response intercorrelation.

Then, the Eckhart-Young Theorem is used to provide the best rank r approximation to $\hat{\mathbf{Y}}_R^*$ in the Frobenius norm as follows:

$$\hat{\mathbf{Y}}_r^* = \sum_{i=1}^{r} d_i \mathbf{u}_i \mathbf{v}_i^\top .^{27}$$

The next step to RRRR is to define the projection matrix, $\mathbf{P}_r$, which projects the predictions onto an r-dimensional subspace: $\mathbf{P}_r = \sum_{i=1}^{r} \mathbf{v}_i \mathbf{v}_i^\top$.

Let $\hat{\mathbf{B}}(\lambda, r) = \hat{\mathbf{B}}_R^* \mathbf{P}_r$ and clearly, $\text{rank}(\hat{\mathbf{B}}(\lambda, r)) \leq r$, as $\text{rank}(\mathbf{P}_r) = r$.[27] Substituting this into the fitted values, $\mathbf{X}^* \hat{\mathbf{B}}(\lambda, r)$, we obtain predictions corresponding to the rank-r solution in Equation 4.6, and thereby to the original problem in Equation 4.5:

$$\mathbf{X}^* \hat{\mathbf{B}}(\lambda, r) = \mathbf{X}^* \hat{\mathbf{B}}_R^* \mathbf{P}_r = \left(\sum_{i=1}^{\tau} d_i \mathbf{u}_i \mathbf{v}_i^\top \right) \left(\sum_{j=1}^{r} \mathbf{v}_j \mathbf{v}_j^\top \right) = \sum_{i=1}^{r} d_i \mathbf{u}_i \mathbf{v}_i^\top = \hat{\mathbf{Y}}_r^* .^{27}$$

See A.2.5 for the third simplification.

This has shown that the proposed solution $\hat{\mathbf{B}}(\lambda, r) = \hat{\mathbf{B}}_R^* \mathbf{P}_r$ is the minimiser of 4.6 and hence the RRRR optimisation problem.[27] Writing it explicitly in terms of $\mathbf{X}, \mathbf{Y}, \lambda$, and r gives the following:

$$\hat{\mathbf{B}}(\lambda, r) = \hat{\mathbf{B}}_R^* \mathbf{P}_r = (\mathbf{X}^{*\top} \mathbf{X}^*)^{-1} \mathbf{X}^{*\top} \mathbf{Y} \mathbf{P}_r = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{Y} \mathbf{P}_r .^{27}$$

Therefore, $\hat{\mathbf{Y}}(\lambda, r) = \mathbf{X} \hat{\mathbf{B}}(\lambda, r) = \mathbf{X} \left(\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I} \right)^{-1} \mathbf{X}^\top \mathbf{Y} \mathbf{P}_r = \hat{\mathbf{Y}}_\lambda \mathbf{P}_r$.

From this estimate, the role of the ridge closed-form solution becomes clear: the full-rank ridge estimator $\hat{\mathbf{B}}_R^*$ is first computed, and then its fitted values $\hat{\mathbf{Y}}_\lambda = \mathbf{X}^* \hat{\mathbf{B}}_R^*$ are projected onto a lower-dimensional subspace. The rank-r constraint then arises from this projection, which maps the fitted values from the original m-dimensional response space to an r-dimensional subspace spanned by the top r right singular vectors.

While RRRR accounts for response covariance by applying SVD to a low-rank approximation of the fitted response matrix, an alternative approach is to model the inverse error covariance matrix, which captures conditional dependencies between responses, directly. This idea underlies Multivariate Regression with Covariance Estimation, which jointly estimates both the regression coefficients and the inverse error covariance structure.

4.1.5 Multivariate Regression with Covariance Estimation

Multivariate Regression with Covariance Estimation (MRCE) builds upon MRLR by jointly estimating the regression coefficient matrix $\mathbf{B}$ and the inverse error covariance matrix $\mathbf{\Omega} = {\mathbf{\Sigma_E}}^{-1}$.

Before solving this problem, we again assume that $\mathbf{X}$ and $\mathbf{Y}$ have been mean-centred. Unlike in RRRR, this is for ease of notation rather than a necessity.

Starting with the familiar MRLR model:

$$\mathbf{Y}_{(n \times m)} = \mathbf{X}_{(n \times p)} \mathbf{B}_{(p \times m)} + \mathbf{E}_{(n \times m)},$$

where p no longer includes the column of ones.

Each row $\mathbf{y}_i \in \mathbb{R}^{1 \times m}$ of $\mathbf{Y}$ follows the conditional distribution: $\mathbf{y}_i | \mathbf{x}_i \sim \mathcal{N}_m(\mathbf{x}_i \mathbf{B}, \mathbf{\Sigma_E})$, from the multivariate normality of the errors. Therefore, the probability density function for the i^{th} row, i.e. a single observation, is:

$$p(\mathbf{y}_i \mid \mathbf{x}_i) = \frac{1}{(2\pi)^{m/2} |\mathbf{\Sigma_E}|^{1/2}} \exp\left(-\frac{1}{2} (\mathbf{y}_i - \mathbf{x}_i \mathbf{B})^\top {\mathbf{\Sigma_E}}^{-1} (\mathbf{y}_i - \mathbf{x}_i \mathbf{B}) \right). \tag{4.7}$$

Equation 4.7 means that, in MRLR, the negative log-likelihood function of $(\mathbf{B}, \mathbf{\Omega})$ can be expressed up to a constant as:

$$g(\mathbf{B}, \mathbf{\Omega}) = \operatorname{tr}\left[\frac{1}{n}(\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})\mathbf{\Omega}\right] - \log|\mathbf{\Omega}|.^{29}$$

See Section A.2.6 for the full derivation.

The MRCE estimation for $\mathbf{B}$ extends this by adding 2 penalties to the negative log-likelihood function g to construct a sparse estimator of $\mathbf{B}$ depending on $\mathbf{\Omega}$:

$$(\hat{\mathbf{B}}, \hat{\mathbf{\Omega}}) = \arg\min_{\mathbf{B},\mathbf{\Omega}} \left\{ g(\mathbf{B}, \mathbf{\Omega}) + \lambda_1 \sum_{j' \neq j} |\omega_{j'j}| + \lambda_2 \sum_{j=1}^{p} \sum_{k=1}^{q} |b_{jk}| \right\}, \tag{4.8}$$

where $\omega_{j'j}$ are the elements of $\mathbf{\Omega}$, b_{jk} are the elements of $\mathbf{B}$, λ_1 controls sparsity in $\mathbf{\Omega}$ and λ_2 for $\mathbf{B}$.[29] A sparse estimator of $\mathbf{B}$ forces many coefficients to zero, improving feature selection and efficiency.

Equation 4.8 can be solved in 2 ways: by solving it for $\hat{\mathbf{\Omega}}$ with $\mathbf{B}$ fixed at a chosen $\mathbf{B}_0$ and for $\hat{\mathbf{B}}$ with $\mathbf{\Omega}$ fixed at a chosen $\mathbf{\Omega}_0$:

1. This is the formula for $\hat{\mathbf{B}}$, fixing $\hat{\mathbf{\Omega}}$ as $\mathbf{\Omega}_0$, therefore the $\mathbf{\Omega}$ terms in Equation 4.8 can be removed from the objective function:

$$\hat{\mathbf{B}}(\mathbf{\Omega}_0) = \arg\min_{\mathbf{B}} \left\{ \operatorname{tr}\left[\frac{1}{n}(\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})\mathbf{\Omega}_0\right] + \lambda_2 \sum_{j,k} |b_{jk}| \right\}. \tag{4.9}$$

2. This is the formula for $\mathbf{\Omega}$, fixing $\mathbf{B}$ as $\mathbf{B}_0$, therefore the $\mathbf{B}$ terms in Equation 4.8 can be removed from the objective function:

$$\hat{\mathbf{\Omega}}(\mathbf{B}_0) = \arg\min_{\mathbf{\Omega}} \left\{ \operatorname{tr}\left(\frac{1}{n}(\mathbf{Y} - \mathbf{XB}_0)^\top(\mathbf{Y} - \mathbf{XB}_0)\mathbf{\Omega}\right) - \log|\mathbf{\Omega}| + \lambda_1 \sum_{j' \neq j} |\omega_{j'j}| \right\}. \tag{4.10}$$

Before delving further into this problem, let us discuss how this model qualifies as MRR and, consequently, how it accounts for response intercorrelation.

Since MRCE estimates both $\mathbf{B}$ and $\mathbf{\Omega}$ simultaneously, the coefficient matrix is optimised considering response covariance. This reasoning follows from the explanation for why MRLR qualifies as an MRR. See Equation 3.5 for how $\mathbf{\Sigma_Y}$, $\mathbf{B}$ and $\mathbf{\Sigma_E}$ relate.

The error covariance matrix $\mathbf{\Sigma_E}$ describes marginal dependencies between response variables, whereas the inverse error covariance matrix $\mathbf{\Omega} = \mathbf{\Sigma_E}^{-1}$ describes conditional dependencies.[30] This means that if $\Omega_{ij} = 0$, then responses Y_i and Y_j are conditionally independent given all other responses and, if $\Omega_{ij} \neq 0$, then responses Y_i and Y_j have a dependency not accounted for by the predictors.[30] Thus, MRCE does not assume independent residuals but instead models how responses are related even after accounting for predictors.

In contrast, MRLR assumes that $\mathbf{\Sigma_E}$ is unstructured but fixed during estimation, and does not account for conditional residual dependencies when estimating $\mathbf{B}$.

Equations 4.9 and 4.10 are each solved using a distinct approach. Equation 4.9 is solved by the following algorithm. Let us call it Algorithm 1. This algorithm comes from Rothman et al. (2010) and needs some set-up: given $\mathbf{\Omega}$ and an initial value $\hat{\mathbf{B}}^{(0)}$, let $\mathbf{S} = \mathbf{X}^\top\mathbf{X}$ and $\mathbf{H} = \mathbf{X}^\top\mathbf{Y}\mathbf{\Omega}$.

1. For the m-th iteration, set $\hat{\mathbf{B}}^{(m)} \leftarrow \hat{\mathbf{B}}^{(m-1)}$. Visit all entries of $\hat{\mathbf{B}}^{(m)}$ in some sequence, and for each entry (r, c), update $\hat{b}_{rc}^{(m)}$ by minimising the objective function with respect to that single

entry, keeping all other coefficients fixed:

$$\hat{b}_{rc}^{(m)} \leftarrow \text{sign}\left(\hat{b}_{rc}^{(m)} + \frac{h_{rc} - u_{rc}}{s_{rr}\omega_{cc}}\right)\left(\left|\hat{b}_{rc}^{(m)} + \frac{h_{rc} - u_{rc}}{s_{rr}\omega_{cc}}\right| - \frac{n\lambda_2}{s_{rr}\omega_{cc}}\right)_+,$$

where $u_{rc} = \sum_{j=1}^{p}\sum_{k=1}^{q}\hat{b}_{jk}^{(m)}s_{rj}\omega_{kc}$ and $(a)_+$ means taking $\max(a, 0)$.

2. If $\sum_{j,k}\left|\hat{b}_{jk}^{(m)} - \hat{b}_{jk}^{(m-1)}\right| < \epsilon\sum_{j,k}\left|\hat{b}_{jk}^{\text{Ridge}}\right|$, then stop, otherwise go to the prior step. Here: $\hat{\mathbf{B}}^{\text{Ridge}}$ is the ridge closed-form estimator of the coefficient matrix. See Equation 4.4 for this estimator.

Note: ϵ is a convergence tolerance, specifying how small the change in $\hat{\mathbf{B}}$ must be between iterations for the algorithm to be considered converged.

Equation 4.10 is solved using the graphical lasso (glasso) algorithm due to its speed.[29] Glasso is, in fact, widely used for estimating sparse inverse covariance matrices in high-dimensional cases.

The glasso algorithm, from Friedman et al. (2008), is, for MRCE, as follows:

1. Initialise $\mathbf{\Omega}$ with: $\mathbf{\Omega}^{(0)} = (\mathbf{\Sigma}_{\mathbf{E}}^{(m)} + \lambda_1\mathbf{I})^{-1}$, where

$$\mathbf{\Sigma}_{\mathbf{E}}^{(m)} = \frac{1}{n}(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(m)})^\top(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(m)}),$$

is the initial error covariance matrix.

2. For the k-th iteration and for each $j = 1, 2, \ldots, q$ in $\mathbf{\Omega}^{(k)}$, solve the graphical lasso regression problem using the previous iteration $\mathbf{\Omega}^{(k-1)}$:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{2}\|\mathbf{\Omega}_{-j,j}^{(k-1)} - \mathbf{\Omega}_{-j,-j}^{(k-1)}\boldsymbol{\beta}\|_2^2 + \lambda_1\|\boldsymbol{\beta}\|_1,$$

where $\mathbf{\Omega}_{-j,-j}^{(k-1)}$ is the $q-1$ vector containing all elements in the j-th column except for the diagonal element $\Omega_{jj}^{(k-1)}$ and $\boldsymbol{\beta}$ is an estimate of $\mathbf{\Omega}_{-j,j}^{(k)}$ by solving the lasso regression problem using $\mathbf{\Omega}^{(k-1)}$.

3. Update $\mathbf{\Omega}$ by filling in the corresponding row and column using:

$$\mathbf{\Omega}_{-j,j}^{(k)} = -\mathbf{\Omega}_{-j,-j}^{(k)}\hat{\boldsymbol{\beta}}.$$

4. Repeat steps 2 and 3 until convergence, so when $||\mathbf{\Omega}^{(k)} - \mathbf{\Omega}^{(k-1)}||_\infty < \epsilon$, where ϵ is chosen.

Now, we can explicitly outline the full MRCE algorithm from Rothman et al. (2010) as follows:
For fixed values of λ_1 and λ_2, initialise $\hat{\mathbf{B}}^{(0)} = \mathbf{0}$ and $\hat{\mathbf{\Omega}}^{(0)} = \hat{\mathbf{\Omega}}(\hat{\mathbf{B}}^{(0)})$.

1. Compute $\hat{\mathbf{B}}^{(m+1)} = \hat{\mathbf{B}}(\hat{\mathbf{\Omega}}^{(m)})$ by solving Equation 4.9 using Algorithm 1.
2. Compute $\hat{\mathbf{\Omega}}^{(m+1)} = \hat{\mathbf{\Omega}}(\hat{\mathbf{B}}^{(m+1)})$ by solving Equation 4.10 using the glasso algorithm.
3. If $\sum_{j,k}\left|\hat{b}_{jk}^{(m+1)} - \hat{b}_{jk}^{(m)}\right| < \epsilon\sum_{j,k}\left|\hat{b}_{jk}^{\text{Ridge}}\right|$ then stop; otherwise, go back to Step 1.

This process uses blockwise descent to compute a local solution for 4.8 with steps 1 and 2 decreasing the objective function value.[29]

MRCE performance also depends on the regularisation parameters, which are selected via CV:

$$(\lambda_1^*, \lambda_2^*) = \arg\min_{\lambda_1,\lambda_2}\sum_{k=1}^{K}\|\mathbf{Y}^{(k)} - \mathbf{X}^{(k)}\mathbf{B}_{\lambda_1,\lambda_2}^{(-k)}\|_F^2,$$

where $\mathbf{Y}^{(k)}$ is the matrix of responses with observations in the kth fold, $\mathbf{X}^{(k)}$ is the matrix of predictors of observations in the kth fold, and $\mathbf{B}^{(-k)}_{\lambda_1,\lambda_2}$ is the estimated regression coefficient matrix computed with observations outside the kth fold, with tuning parameters λ_1 and λ_2.[29]
The next section will talk about how these 2 methods were evaluated on the equity fund dataset.

4.2 Application

4.2.1 Small-Scale Example

Let us now apply both of these shrinkage methods to a small dataset before proceeding to the equity fund dataset. We will use the familiar Mathematics and Science scores data:

X_1: Hours Studied	X_2: Time Spent on Papers	Y_1: Math Scores	Y_2: Science Scores
5	2	78	80
7	3	85	79
8	4	88	88
3	1	65	70
10	5	92	74

Table 4.1: Study Time vs. Exam Scores

Reduced Rank Ridge Regression

This example will show a rank-1 reduction of RRRR using $\lambda = 0.1$ as the shrinkage parameter.

Based on our prior justification of RRRR as an MRR model, mean centring is essential to prevent biasing from the mean structure. The initial predictor matrix, $\mathbf{X}$, and initial response matrix $\mathbf{Y}$ are:

$$\mathbf{X} = \begin{bmatrix} 5 & 7 & 8 & 3 & 10 \\ 2 & 3 & 4 & 1 & 5 \end{bmatrix}^\top, \quad \mathbf{Y} = \begin{bmatrix} 78 & 85 & 88 & 65 & 92 \\ 80 & 79 & 88 & 70 & 74 \end{bmatrix}^\top.$$

After mean-centering, i.e. $\tilde{\mathbf{X}} = \mathbf{X} - \bar{\mathbf{X}}$ and $\tilde{\mathbf{Y}} = \mathbf{Y} - \bar{\mathbf{Y}}$, the predictor and response matrices become:

$$\tilde{\mathbf{X}} = \begin{bmatrix} -1.6 & 0.4 & 1.4 & -3.6 & 3.4 \\ -1 & 0 & 1 & -2 & 2 \end{bmatrix}^\top, \quad \tilde{\mathbf{Y}} = \begin{bmatrix} -3.6 & 3.4 & 6.4 & -16.6 & 10.4 \\ 1.8 & 0.8 & 9.8 & -8.2 & -4.2 \end{bmatrix}^\top.$$

We apply reduced rank ridge regression with $\lambda = 0.01$ by first incorporating ridge regularisation, which we define as:

$$\mathbf{X}^* = \begin{bmatrix} \tilde{\mathbf{X}} \\ \sqrt{\lambda}\mathbf{I} \end{bmatrix}, \quad \mathbf{Y}^* = \begin{bmatrix} \tilde{\mathbf{Y}} \\ \mathbf{0} \end{bmatrix}.$$

Since $\lambda = 0.01$, we compute $\sqrt{\lambda} = \sqrt{0.01} = 0.1$, so the augmented matrices are:

$$\mathbf{X}^* = \begin{bmatrix} -1.6 & 0.4 & 1.4 & -3.6 & 3.4 & 0.1 & 0 \\ -1 & 0 & 1 & -2 & 2 & 0 & 0.1 \end{bmatrix}^\top, \quad \mathbf{Y}^* = \begin{bmatrix} -3.6 & 3.4 & 6.4 & -16.6 & 10.4 & 0 & 0 \\ 1.8 & 0.8 & 9.8 & -8.2 & -4.2 & 0 & 0 \end{bmatrix}^\top.$$

The Ridge penalty is incorporated into $\mathbf{X}^*$ through $\sqrt{\lambda}\mathbf{I}$. The augmented system behaves like a standard regression problem, meaning we solve for $\mathbf{B}$ using the standard OLS formula. Let us denote this as $\hat{\mathbf{B}}^*_R$ as a reminder that we are incorporating ridge regression.

This formula is: $\hat{\mathbf{B}}_R^* = (\mathbf{X}^{*\top}\mathbf{X}^*)^{-1}\mathbf{X}^{*\top}\mathbf{Y}^*$. After subbing in $\mathbf{X}^*$ and $\mathbf{Y}^*$, this gives us the augmented coefficient matrix, $\hat{\mathbf{B}}_R^*$ (to 2dp) as:

$$\hat{\mathbf{B}}_R^* = \begin{bmatrix} 7.40 & -2.28 \\ -6.18 & 5.47 \end{bmatrix}$$

Thus, the ridge predictions are (to 2dp):

$$\hat{\mathbf{Y}}_R^* = \mathbf{X}^*\hat{\mathbf{B}}_R^* = \begin{bmatrix} -5.67 & 2.96 & 4.19 & -14.29 & 12.81 & 0.74 & -0.62 \\ -1.82 & -0.91 & 2.28 & -2.73 & 3.19 & -0.23 & 0.55 \end{bmatrix}^\top$$

The next step is to perform singular value decomposition on this prediction, i.e. $\hat{\mathbf{Y}}_R^* = \mathbf{U}\mathbf{D}\mathbf{V}^\top$, with singular values (to 2dp) of 21.21 and 2.29, giving:

$$\mathbf{D} = \begin{bmatrix} 21.21 & 0 \\ 0 & 2.29 \end{bmatrix}.$$

For full derivation of $\mathbf{D}, \mathbf{U}$ and $\mathbf{V}$, see A.2.5.

Since $D_1 = 21.21 \gg D_2 = 2.29$, the best rank ($r$=1) approximation is taken on the first column of $\mathbf{D}$ and so, take the first columns of $\mathbf{U}$ and $\mathbf{V}$. Therefore, $\hat{\mathbf{Y}}_1^* \approx d_1\mathbf{u}_1\mathbf{v}_1^\top$, where (to 2dp):

$$d_1 = 21.21, \quad \mathbf{u}_1 = \begin{bmatrix} -0.28 & 0.13 & 0.22 & -0.69 & 0.62 & 0.03 & -0.02 \end{bmatrix}^\top, \quad \mathbf{v}_1 = \begin{bmatrix} 0.97 & 0.22 \end{bmatrix}^\top.$$

However, this is just an intermediary step as we still need to rescale the coefficient matrix. This is done through the projection matrix, $\mathbf{P}_r$ (to 2dp):

$$\mathbf{P}_r = \mathbf{v}_1\mathbf{v}_1^\top = \begin{bmatrix} 0.95 & 0.22 \\ 0.22 & 0.05 \end{bmatrix}$$

Finally, the reduced rank ridge regression estimator is:

$$\hat{\mathbf{B}}_r = \hat{\mathbf{B}}_R^*\mathbf{P}_r = \begin{bmatrix} 6.54 & 1.49 \\ -4.68 & -1.07 \end{bmatrix}$$

Now, we can do the final predicted values, which are $\hat{\mathbf{Y}} = \tilde{\mathbf{X}}^*\hat{\mathbf{B}}_r$, giving (to 2dp):

$$\hat{\mathbf{Y}} = \begin{bmatrix} -5.78 & 2.62 & 4.47 & -14.18 & 12.87 \\ -1.32 & 0.60 & 1.02 & -3.24 & 2.94 \end{bmatrix}^\top$$

Remember, these predictions are on the **mean centered Y**. We can remove this mean centring, but that does not affect the model fit evaluation using the ANRMSE.

Multivariate Regression with Covariance Estimation

Now, we perform an MRCE iteration on Table 4.1 with $\lambda_1 = \lambda_2 = 0.1$. Remember we want to mean centre $\mathbf{X}$ and $\mathbf{Y}$, so let us use $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$ from the RRRR calculation:

$$\tilde{\mathbf{X}} = \begin{bmatrix} -1.6 & 0.4 & 1.4 & -3.6 & 3.4 \\ -1 & 0 & 1 & -2 & 2 \end{bmatrix}^\top, \quad \tilde{\mathbf{Y}} = \begin{bmatrix} -3.6 & 3.4 & 6.4 & -16.6 & 10.4 \\ 1.8 & 0.8 & 9.8 & -8.2 & -4.2 \end{bmatrix}^\top.$$

From here on, $\mathbf{X} = \tilde{\mathbf{X}}$ and $\mathbf{Y} = \tilde{\mathbf{Y}}$. This is to make notation easier. Let us first compute $\hat{\mathbf{B}}^{\text{Ridge}}$ because that is used in determining when Algorithm 1 stops:

$$\hat{\mathbf{B}}^{\text{Ridge}} = (\mathbf{X}^\top \mathbf{X} + \lambda_2 \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{Y} = \begin{bmatrix} 5.07 & -0.77 \\ -2.19 & 2.89 \end{bmatrix}$$

For Algorithm 1, initialise $\hat{\mathbf{B}}^{(0)} = \mathbf{0}$ and $\hat{\mathbf{\Omega}}^{(0)} = \hat{\mathbf{\Omega}}(\hat{\mathbf{B}}^{(0)})$. To compute $\hat{\mathbf{\Omega}}^{(0)}$, first do the error, $\mathbf{E}^{(0)}$:

$$\mathbf{E}^{(0)} = \mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(0)} = \begin{bmatrix} -3.6 & 3.4 & 6.4 & -16.6 & 10.4 \\ 1.8 & 0.8 & 9.8 & -8.2 & -4.2 \end{bmatrix}^\top.$$

Now, compute the residual covariance matrix $\mathbf{\Sigma}_{\mathbf{E}}^{(\mathbf{0})}$:

$$\mathbf{\Sigma}_{\mathbf{E}}^{(\mathbf{0})} = \frac{1}{n} {\mathbf{E}^{(0)}}^\top \mathbf{E}^{(0)} = \begin{bmatrix} 89.84 & 30.28 \\ 30.28 & 36.96 \end{bmatrix}$$

Next, add this to a regularisation parameter $\lambda_2 = 0.1$. The regularisation parameter avoids dividing by zero when inverting ${\mathbf{\Sigma}_{\mathbf{E}}}^{(0)}$ for the precision matrix (to 2sf):

$$\mathbf{\Omega}^{(0)} = \begin{bmatrix} 0.015 & -0.013 \\ -0.013 & 0.037 \end{bmatrix}$$

Now compute $\hat{\mathbf{B}}^{(1)} = \hat{\mathbf{B}}(\hat{\mathbf{\Omega}}^{(0)})$ by solving Equation 4.9 using Algorithm 1. The manual calculation of this will be shown on 1 element of $\mathbf{B}^{(1)}$: $\hat{b}_{11}^{(1)}$. First compute $\mathbf{H}$ and $\mathbf{S}$:

$$\mathbf{S} = \mathbf{X}^\top \mathbf{X} = \begin{bmatrix} 29.2 & 17 \\ 17 & 10 \end{bmatrix}, \quad \mathbf{H} = \mathbf{X}^\top \mathbf{Y} \mathbf{\Omega}^{(0)} = \begin{bmatrix} 1.37 & -0.41 \\ 0.78 & -0.21 \end{bmatrix}.$$

Now set $\hat{\mathbf{B}}^{(1)} = \hat{\mathbf{B}}^{(0)}$ and update $\hat{b}_{11}^{(1)}$ with the minimiser of the objective function along its coordinate direction given by:

$$\hat{b}_{11}^{(1)} \leftarrow \text{sign}\left(\hat{b}_{11}^{(1)} + \frac{h_{11} - u_{11}}{s_{11}\omega_{11}}\right)\left(\left|\hat{b}_{11}^{(1)} + \frac{h_{11} - u_{11}}{s_{11}\omega_{11}}\right| - \frac{n\lambda_2}{s_{11}\omega_{11}}\right)_+,$$

where $u_{11} = \sum_{j=1}^{2}\sum_{k=1}^{2} \hat{b}_{jk}^{(1)} s_{1j}\omega_{k1}$. After subbing in and computation, this gives: $\hat{b}_{11} = 1.95$. Doing this on every element, we get:

$$\hat{\mathbf{B}}^{(1)} = \begin{bmatrix} 1.95 & 0 \\ 0 & 0 \end{bmatrix}.$$

The next step of Algorithm 1 involves checking for convergence of $\hat{\mathbf{B}}$ relative to $\hat{\mathbf{B}}^{\text{Ridge}}$. Here, if:

$$\sum_{j,k} \left|\hat{b}_{jk}^{(1)} - \hat{b}_{jk}^{(0)}\right| < \epsilon \sum_{j,k} \left|\hat{b}_{jk}^{\text{Ridge}}\right|,$$

then stop; otherwise, go to the prior step. The left-hand side simplifies too:

$$\sum_{j,k} \left|\hat{b}_{jk}^{(1)} - \hat{b}_{jk}^{(0)}\right| = \sum_{j,k} |\hat{b}_{jk}^{(1)}| = 1.95.$$

And the right-hand side, too:

$$\epsilon \sum_{j,k} \left| \hat{b}_{jk}^{\text{Ridge}} \right| = 10.92\epsilon.$$

Therefore, we want $1.95 < 10.92\epsilon$. Therefore $\epsilon \gtrsim 0.18$. Let us choose $\epsilon = 0.2$ so we can move on to the next phase of MRCE, estimating $\mathbf{\Omega}$ using the graphical lasso algorithm. Firstly, let us set a new initial $\mathbf{\Omega}$, using this updated $\hat{\mathbf{B}}$:

$$\mathbf{\Sigma}_{\mathbf{E}}^{(1)} = \frac{1}{n}(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(1)})^{\top}(\mathbf{Y} - \mathbf{X}\hat{\mathbf{B}}^{(1)}) = \begin{bmatrix} 25.24 & 19.97 \\ 19.97 & 36.96 \end{bmatrix}.$$

When inverting, again use a regularisation parameter $\lambda_1 = 0.1$ to prevent a zero determinant, giving:

$$\mathbf{\Omega}^{(0)} = \begin{bmatrix} 0.069 & -0.037 \\ -0.037 & 0.047 \end{bmatrix}.$$

Like for Algorithm 1, we will only apply this manually to one element. Here, it will be $\Omega_{21}^{(1)}$. Now, let us solve the graphical lasso regression problem using $\hat{\mathbf{B}}^{(1)}$. For $j = 1$, which updates the first column and row of $\mathbf{\Omega}$, we define:

$$(\mathbf{\Sigma}_{\mathbf{E}}^{(1)})_{-1,-1} = 36.96, \quad (\mathbf{\Sigma}_{\mathbf{E}}^{(1)})_{-1,1} = 19.97.$$

We compute the inverse submatrix: $\Omega_{-1,-1}^{(1)} = (\mathbf{\Sigma}_{\mathbf{E}}^{(1)})_{-1,-1}^{-1} = 0.027$ and $\Omega_{-1,1}^{(1)} = (\mathbf{\Sigma}_{\mathbf{E}}^{(1)})_{-1,1}^{-1} = 0.050$ to contribute to $\hat{\boldsymbol{\beta}}$:

$$\hat{\boldsymbol{\beta}} = -\mathbf{\Omega}_{-1,-1}^{(1)}(\mathbf{\Sigma}_{\mathbf{E}}^{(1)})_{-1,1} = -0.00135.$$

Next, $\hat{\boldsymbol{\beta}}$ undergoes soft-thresholding:

$$\hat{\boldsymbol{\beta}} = \text{sign}(\hat{\boldsymbol{\beta}}) \max(|\hat{\boldsymbol{\beta}}| - \lambda_1, 0) = 0$$

The updated element is computed as:

$$\mathbf{\Omega}_{-1,1}^{(1)} = -\mathbf{\Omega}_{-1,-1}^{(1)}\hat{\boldsymbol{\beta}} = 0.$$

This occurs for each element in $\mathbf{\Omega}^{(1)}$ and these steps continue until convergence, so when $||\mathbf{\Omega}^{(k)} - \mathbf{\Omega}^{(k-1)}||_{\infty} < \epsilon$, where ϵ is chosen for controlled estimation.

This completes one full MRCE iteration: initialising $\mathbf{B}^{(0)}$ and $\hat{\mathbf{\Omega}}^{(0)}$ and then updating to $\mathbf{B}^{(1)}$ and $\hat{\mathbf{\Omega}}^{(1)}$ after solving the penalised optimisation problem. This process continues iteratively until convergence. The final $\hat{\mathbf{B}}$, is then used to compute the predictions: $\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}}^{(n)}$. This can again have its centring removed, but it is not needed when evaluating the ANRMSE.

4.2.2 Code Explanation

The MRCE model was applied using the `MRCE` package, which jointly estimates the regression coefficient matrix and the inverse error covariance matrix of the response variables. The optimal values of the regularisation parameters λ_1 and λ_2 were selected using a 5-fold CV procedure implemented in the `fit_mrce` function. A grid of candidate values was evaluated by training and validating the model across CV folds, and the parameter pair, (λ_1, λ_2), that minimised the cross-validated ANRMSE was used for final model fitting. Test-set predictions were then computed using the fitted coefficient matrix.

RRRR was then evaluated using the `fit_rrridge` function, which applies a rank constraint to the regression coefficient matrix. A ridge-penalised estimate is first computed to stabilise the solution, followed by SVD to retain only the top singular components, enforcing a reduced rank. The resulting coefficient matrix is then used to generate test-set predictions, which are evaluated via ANRMSE.

Next, additional feature transformations were applied to try to improve the performance of these methods. A degree-two polynomial expansion was applied using `poly`, and second-order interaction terms were generated using `model.matrix`. Finally, all models are compared based on their test set ANRMSE scores as specified previously.

4.2.3 Results

These regression models were applied to the dataset to evaluate how well the predictors model the response variables: ROE and sustainability score. The performance of each model was assessed using the ANRMSE, as was the case with the other models in this report.

The baseline models show that RRRR had the smallest ANRMSE of 0.751, followed by MRCE at 0.761. This means that the reduced-rank constraint in RRRR enabled it to better capture common structure among responses, particularly where there were weak intercorrelations among them. MRCE, by contrast, involves an additional step of covariance estimation, which introduces bias where response intercorrelations are weak, as they were here, with a correlation of -0.3171 between the ROE and sustainability score. MRCE will perform better under stronger correlations among response variables.

The inclusion of polynomial and interaction feature transformations improved performance in both models. MRCE benefited most by polynomial features, whose ANRMSE fell from 0.761 to 0.678 — the lowest ANRMSE for any of the polynomial forms. RRRR also benefited, though less so, from 0.751 to 0.727, suggesting that the inclusion of polynomial expansion introduced unnecessary complexity not well used under a low-rank constraint.

Adding interaction terms improved performance further. MRCE had the lowest overall ANRMSE of 0.678, and RRRR achieved 0.723. This suggests that MRCE's covariance estimation benefits more from structured interactions between predictors, whereas RRRR can be more impacted by the added multicollinearity from such terms.

These results indicate that the performance of each model depends on the type of introduced features. Subsequently, the regression models were fitted to the data to examine how well the predictors model the response variables: ROE and sustainability score. Each model's performance was measured using ANRMSE, with the smaller value reflecting the better performance.

Generally, while MRCE had the smallest ANRMSE in the most complex feature setting, RRRR showed a more stable performance in all conditions. Thus, RRRR could be considered the more stable shrinkage method for this dataset.

Model	**ANRMSE**
Model 1: MRCE	0.7612
Model 2: RRRR	0.7510
Model 3: MRCE including Polynomial Terms	0.6781
Model 4: RRRR including Polynomial Terms	0.7276
Model 5: MRCE including Interaction Terms	0.6777
Model 6: RRRR including Interaction Terms	0.7231

Table 4.2: Comparison of ANRMSE values across shrinkage methods

The models in this chapter were also surprising because they performed as well as, or sometimes even worse than, MRLR in Chapter 3 on the equity fund dataset (see Table 3.2 for the evaluation of the Chapter 3 model). This was unexpected given that MRCE and RRRR are designed to handle multiple responses with more considerations, particularly in the presence of multicollinearity or high-dimensional predictors, meaning they should generally be more accurate in modelling.

However, the predictors used here were not highly correlated after the pre-processing steps described in Chapter 2, and the dataset is not high-dimensional ($p = 14$, $n = 1269$). As such, the benefits of shrinkage and dimensionality reduction were less pronounced. In fact, standard OLS outperformed both MRCE and RRRR, likely because it preserved the relationship between the predictors and responses without the added bias introduced by regularisation.

Another aspect that was examined was how each feature contributes to each response.

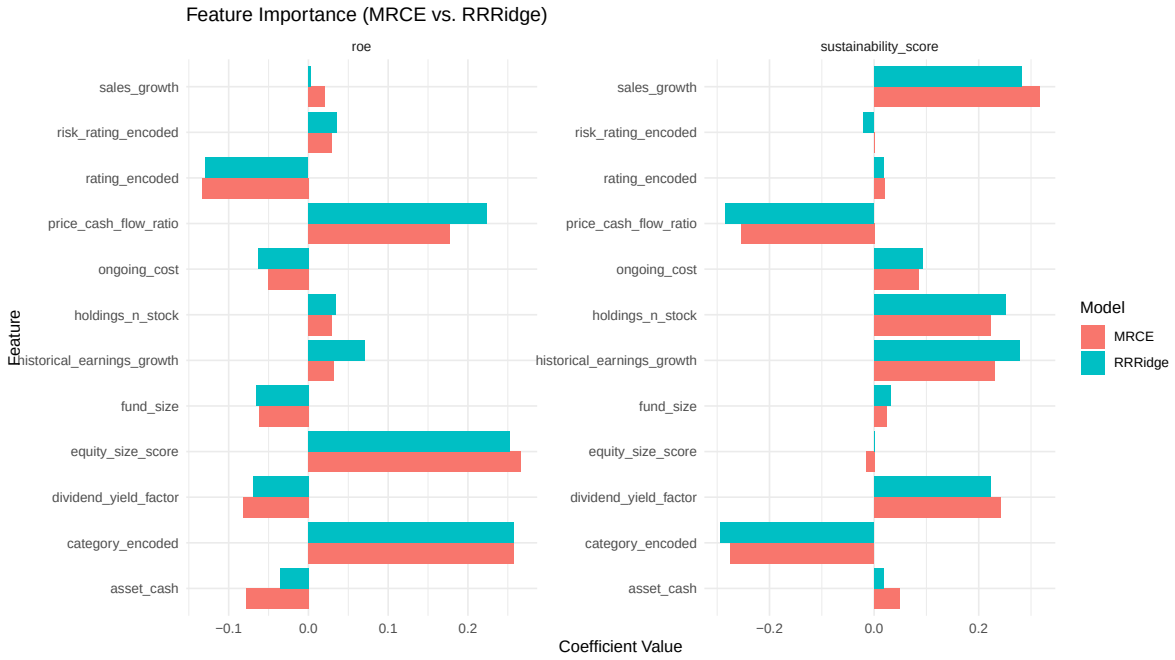

Figure 4.2: MRCE vs RRRR Feature Contributions

For `roe`, both models identify `price_cash_flow_ratio`, `category_encoded`, and `equity_size_score` as important predictors. However, MRCE applies stronger shrinkage to these variables because it estimates both the regression coefficients and the precision matrix simultaneously. As a result, `price_cash_flow_ratio` has a much smaller coefficient in MRCE, suggesting that RRRR depends on it more heavily. `sales_growth` has little influence in either model, while `asset_cash` is slightly negative in MRCE but negligible in RRRR.

For the `sustainability_score`, the differences between models are more noticeable. `sales_growth` is the strongest positive predictor in both cases, with MRCE assigning it a slightly larger coefficient. On the other hand, `historical_earnings_growth` has a strong positive effect in RRRR but is heavily shrunk in MRCE, likely due to concerns about multicollinearity, despite being addressed earlier in Chapter 2. Interestingly, `category` receives a higher coefficient in MRCE, suggesting it emphasises the type of equity fund more. `equity_size_score` is close to zero in MRCE but still contributes in RRRR, and `price_cash_flow_ratio` again has a larger absolute coefficient in RRRR.

Although shrinkage methods like MRCE and RRRR are helpful in reducing overfitting and handling high-dimensional data, their linearity-dependent nature can limit their expressiveness. Here, as seen, there will be a loss of some interaction and patterns. Random Forests mitigate the flaws by employing an ensemble of decision trees for learning non-linear interactions and patterns, eliminating the need for feature engineering. The application of an ensemble also reduces overfitting, which further renders Random Forests particularly suitable for handling complex datasets.

Chapter 5 Random Forests

This chapter delves into Random Forests. It starts with classification and regression trees, from which we can create an ensemble of decision trees. This gives rise to random forests, which improve predictive performance. The forests are then extended to MRR, and the loss function is adjusted to form Covariance Regression Random Forests, which are tested on the equity fund dataset.

5.1 Theory

The previous modelling approaches used in this project, MRLR (see Chapter 3) and MRCE and RRRR (see Chapter 4), rely on global parametric assumptions. These include properties such as the multivariate normality of the response and linear relationships between predictors and responses.. Tree-based models, however, are non-parametric, meaning they require no prior assumptions about the data. These algorithms work by partitioning the feature space into smaller regions with similar response values using a set of splitting rules.

5.1.1 Classification and Regression Trees

Random forests are tree-based models built upon classification and regression trees (CARTs) as a building block. A CART partitions the training data into groups with similar response values and then predicts the same category for all data within a given subgroup.

The CART starts with a root node containing all the objects and is then divided into "leaf" nodes by recursive binary splitting.[31] Each leaf node relates to a hyper-rectangle in the feature space, R_j, $j = \{1, \ldots, J\}$, i.e. the feature space defined by $X_1, X_2, \ldots, X_p$ is split into J distinct regions which do not overlap, as shown below:

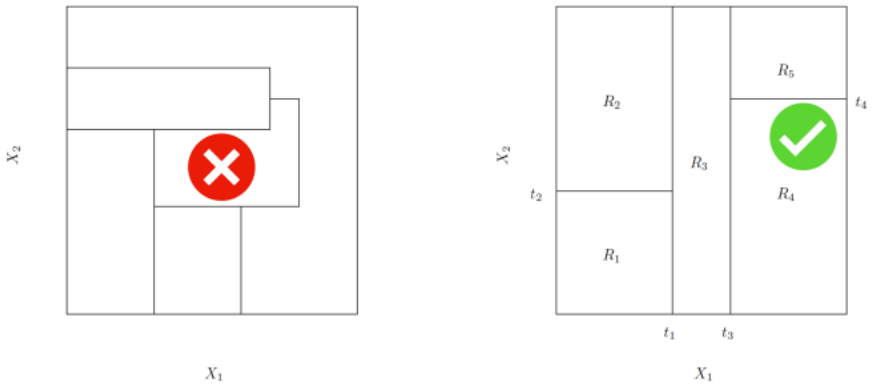

Figure 5.1: Incorrect vs. Correct Partitioning

The Greedy-Fitting Algorithm ensures each split is chosen to minimise the RSS:

$$\sum_{j=1}^{J} \sum_{i:x_i \in R_j} \left(y_i - \hat{y}_{R_j}\right)^2,$$

where $\hat{y}_{R_j}$ is the mean response for the training observations within the j-th rectangle.

A regression tree is used because the responses here, ROE and sustainability score, are continuous. A regression tree assigns a value to each predictor space, R_j, i.e. the model predicts the output (i.e. which feature space) based on the average response values for all observations in that subgroup.

The Greedy fitting algorithm provides good predictions for the training data but may overfit, resulting in poor performance on the test dataset. However, there exists a method called pruning that can mitigate this issue. Pruning involves removing branches that contribute little to the tree's prediction accuracy. This is typically based on a validation set or a complexity penalty.

A common approach to pruning is the weakest link pruning algorithm, which iteratively prunes the least important branches to control the bias–variance trade-off. This algorithm introduces a non-negative tuning parameter, α, for regression trees, which minimises the following cost-complexity objective:

$$\sum_{j=1}^{|T|} \sum_{i:x_i \in R_j} (y_i - \hat{y}_{R_j})^2 + \alpha |T|,$$

where $|T|$ is the number of terminal nodes (size) of the tree T.

5.1.2 Bagging for CARTs

Although CARTs are easy to interpret and are similar to standard decision-making processes, the trees generally have high variance, i.e. small sample changes lead to significant changes in the fit, and they tend to have poor predictive accuracy. Bootstrap aggregating, also known as bagging, improves the stability and accuracy of classification and regression algorithms by averaging models. Bagging helps reduce variance and avoid over-fitting, but the model becomes more challenging to interpret.
For regression trees, all the predictions are averaged to obtain:

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^b(x),$$

where B is the total number of bootstrap samples (i.e., the number of trees trained), and $\hat{f}^b(x)$ is the prediction from the b-th tree.

Bagging for CART addresses the overfitting issue in two ways: it can grow large trees with minimal (or no) pruning, relying on the averaging effect of bagging to reduce variance, or alternatively, it can prune each tree individually.

Bagging is advantageous when there is a large amount of data, as is the case here for the equity fund dataset, because the empirical distribution will be closer to the actual underlying population's distribution. Additionally, it is the best option when the aim is to minimise the variance of a predictor.

However, bagging also brings some disadvantages. Firstly, interpretability is lost as the final estimate is not a tree. Secondly, it is an ensemble prediction, i.e. multiple trees are combined to make the final prediction.[32]. Also, bagging trees are inherently correlated, which is a problem because the more correlated the random variables are, the less the variance reduction of their average, which can undermine one of its key advantages.

5.1.3 Random Forests

Random Forests make bagged CART models more independent, improving variance reduction in their ensemble predictions. They do this by resampling observations and restricting the model to random subspaces, $\mathcal{X}' \subset \mathcal{X}$, which ensures the tree explores different parts of the data, meaning the ensembles are more diverse and hence influential.[33,34]

The Random Forests algorithm is used in the following way: a bootstrap resample $(y_i^*, x_i^*)_{i=1}^n$ is taken of the training data, like for bagging. Then, the tree is built and each time a split in a tree is considered, randomly select m predictors out of the full set of p predictors as split candidates and find the best split within those m predictors. Finally, repeat the first two steps, averaging the prediction of all the regression forest trees.

Random forests are a great option, but they are not interpretable when bagging and using random subspaces. Thus, quantifying the importance of each variable can be difficult. There are two popular approaches which quantify this importance.

One approach is to run a loop over each tree in the forest to determine the significance of each feature variable x_j, where $j = 1, \ldots, p$. Every node that splits on x_j for each tree is identified, and the improvement in the chosen loss criterion, such as accuracy or the Gini index, is calculated for each split. Each improvement is then summed for every tree in the forest, indicating the significance of x_j.

Another approach is to use out-of-bag (OOB) error estimates, which can be computed via bagging, to determine the significance of each feature x_j, where $j = 1, \ldots, p$[33]. For each tree, the OOB samples (i.e., the observations that were not included in the bootstrap sample used to train that tree) are used to evaluate predictive accuracy. To assess the importance of x_j, the entries in the j-th column of the OOB feature matrix, denoted x^{oob}, are randomly permuted, breaking the association between x_j and the rest of the variables. The modified matrix, $x^{\text{oob}*}$, is then passed through the tree, and the resulting change in predictive accuracy is computed. This decrease in accuracy caused by the permutation is averaged across all trees, providing a measure of the importance of feature x_j.

Random forests inherit the advantages of bagging and trees, and tuning is rarely needed. However, they are challenging to implement and have issues with extrapolation.

5.1.4 Multivariate Regression Trees

CARTs are limited to single-response variables, making them unsuitable for modelling relationships between multiple correlated responses. Multivariate Regression Trees (MRTs) generalise CARTs by modelling all responses simultaneously.[35]

MRTs build decision trees by recursively splitting the data along predictor variable thresholds to minimise variability across the response matrix. Each split aims to group observations with similar values across all response variables.[35] This recursive splitting means that MRTs are able to detect patterns that occur only within specific regions of the predictor space, including non-linear relationships and interactions that other models may miss.

Although MRTs can be grown until each leaf contains a single observation, this leads to overfitting. Therefore, in practice, trees are pruned using CV, typically selecting the smallest tree within one standard error of the minimum cross-validated error.[36] This is known as the 1-standard error rule.

Unlike single-response regression trees, MRTs reveal how relationships between predictors and responses vary across different regions of the predictor space.[35]

MRTs adapt the impurity criterion from CARTs to account for all response variables simultaneously, evaluating splits by minimising the total variation across the multivariate response.[36] The impurity measure in MRTs minimises this total variance, and it is the total sum of squares of the responses around the multivariate mean at each node:

$$\text{impurity} = \sum_{i=1}^{n} \sum_{j=1}^{p} (y_{ij} - \bar{y}_j)^2 ,$$

where y_{ij} represents the j-th response for the i-th sample, and $\bar{y}_j$ is the mean response at the node.[31]

Because impurity minimisation is performed over the joint space, MRTs naturally cluster correlated responses, thereby preserving their multivariate structure. However, MRTs do not explicitly model

changes in the covariance structure across predictor values.

The resulting tree structure is highly interpretable, with each node representing a split on an explanatory variable and each leaf corresponding to a distinct cluster. This makes it straightforward to visualise local interactions and the influence of predictors.

However, different visualisation tools are needed for MRTs to interpret their results. For instance, at each node of the tree, a histogram shows the distribution of each response within that node, helping to visualise the multivariate outcomes and understand the impact of splits on all responses jointly.[31]

In Figure 5.2, the left panel visualises the MRR setup: a response matrix, with two continuous response variables Y_1 and Y_2, is regressed on a set of predictors X_1, X_2, X_3, shown as distinct columns in the predictor matrix. The right-hand panel displays the fitted MRT, where each internal node represents a binary split on one of the predictors. Each terminal node contains a pair of bar plots showing the mean values of Y_1 and Y_2 for the observations within that group. This helps assess how each split affects the joint distribution of responses.

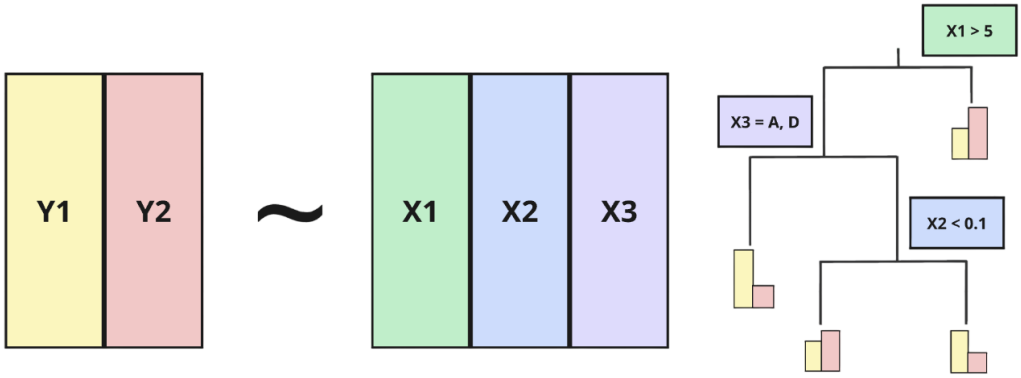

Figure 5.2: MRT Splitting.[35]

Note: if a terminal node contains fewer bars than the number of responses, this indicates that one or more response variables have a mean of zero within that group. In other words, bars are only displayed for response variables whose node-wise means are non-zero.

Although MRTs improve on CARTs in MRR, they come with some drawbacks, one of which is the inherent instability of tree-structured predictors caused partly by the split function's greedy optimisation.[31] Multivariate random forests can instead be used to solve this issue.

5.1.5 Multivariate Random Forests

Multivariate Random Forests (MRFs) extend MRTs by constructing an ensemble of MRTs through bootstrap resampling and random predictor subsampling, like for univariate random forests.[37] This ensemble process helps mitigate the instability of single-tree models, making MRFs less sensitive to small changes in the data.

However, while MRFs are a great extension of MRTs, they exhibit an important limitation in MRR. Although their loss function can be adjusted to implicitly consider response correlations, MRFs do not explicitly enforce that each split leads to a meaningful change in the response covariance structure. As a result, when responses are highly correlated, MRFs fail to consider the correlations optimally when building trees, potentially leading to suboptimal splits.[37]

5.1.6 Covariance Regression with Random Forests

Covariance Regression with Random Forests (CRRFs) extend MRFs by incorporating a splitting criterion that maximises differences in sample covariance estimates between child nodes. Unlike MRFs,

which focus solely on predicting the conditional mean, CRRFs explicitly account for response intercorrelation. They extend tree-based methods to estimate the conditional covariance matrix of a multivariate response given a set of covariates.[38] While CRRFs are designed to estimate covariance structures, they inherently compute the sample response mean at each terminal node. These mean values can be direct response estimates, allowing CRRFs to extend to MRR.

Let $\Sigma_{\mathbf{y}_i}$ be the true conditional covariance matrix of $\mathbf{y}_i$, the responses, based on covariates $\mathbf{x}_i$, and $\Sigma_{\mathbf{X}}$ the collection of all conditional covariance matrices for n observations:

$$\Sigma_{\mathbf{X}} = \{\Sigma_{\mathbf{y}_i} : i = 1, \ldots, n\}.^{38}$$

Also, let $\hat{\Sigma}_{\mathbf{y}_i}$ be the estimated conditional covariance matrix of $\mathbf{y}_i$ based on covariates $\mathbf{x}_i$, and $\hat{\Sigma}_{\mathbf{X}}$ be the collection of estimated conditional covariance matrices for n observations.[38]

First, train a random forest with the set of covariates $\mathbf{X}$ to find subgroups of observations with similar covariance matrices of $\mathbf{Y}$ using decision trees to uncover the data structures. These trees are built with a splitting criterion that will be defined later.[38] The tree-growing process follows the CART algorithm.[39] The aim of this is to obtain subgroups of observations with distinct covariance matrices. Hence, a customised splitting rule at each node will be used to increase the difference in covariance matrices between two child nodes in the tree.[40,41,42,43]

Before defining this splitting let us first set up more notation. $\mathbf{\Sigma}^L$ is the sample response covariance matrix estimate, so $\hat{\Sigma}_{\mathbf{y}_i}$ of the left node, which is as follows:

$$\mathbf{\Sigma}^L = \frac{1}{n_L - 1} \sum_{i \in t_L} (\mathbf{y}_i - \bar{\mathbf{Y}}_L)(\mathbf{y}_i - \bar{\mathbf{Y}}_L)^\top,$$

where t_L is the set of indices, or the positions within the original data, of the observations in the left node, n_L is the left node size, and:

$$\bar{\mathbf{Y}}_L = \frac{1}{n_L} \sum_{i \in t_L} \mathbf{y}_i.^{38}$$

The estimate of the sample response covariance matrix for the right node, $\mathbf{\Sigma}^R$, is calculated similarly, where t_R is the set of indices of observations in the right node and n_R is its size.
The splitting criterion of CRRF is:

$$\sqrt{n_L n_R} \times d(\mathbf{\Sigma}^L, \mathbf{\Sigma}^R), \tag{5.1}$$

where $d(\mathbf{\Sigma}^L, \mathbf{\Sigma}^R)$ is the Euclidean distance between the upper triangular part of the two matrices and computed as follows:

$$d(\mathbf{D}, \mathbf{E}) = \sqrt{\sum_{i=1}^{q} \sum_{j=i}^{q} (D_{ij} - E_{ij})^2},$$

where $\mathbf{D}_{q \times q}$ and $\mathbf{E}_{q \times q}$ are symmetric matrices.[38] The best split among those possible is the one that maximises 5.1. This split is what facilitates CRRF as an MRR model as it explicitly uses a response covariance-based splitting criteria.

The final covariance matrices are estimated using random forests. For a new observation, nearest neighbour observations are used to estimate the response mean and final covariance matrix, ensuring both the predicted value and its uncertainty are captured.[38] This set of observations is called the Bag of Observations for Prediction (BOP).

For a new observation $\mathbf{x}^*$, the set of nearest neighbour observations are formed with the oob

observations.[44,45] The BOP_{oob} for a new observation is:

$$BOP_{oob}(\mathbf{x}^*) = \bigcup_{b=1}^{B} O_b(\mathbf{x}^*),$$

where B is the tree numbers and $O_b(\mathbf{x}^*)$ is the set of oob observations in the same terminal node as $\mathbf{x}^*$ in the bth tree, and each tree is built with a selected random sub-sample.[38]
After training the random forest with the new split for a new observation $\mathbf{x}^*$, we form $BOP_{oob}(\mathbf{x}^*)$, which is key for response prediction.[38] The response mean and covariance matrix are then estimated using the sample mean and covariance matrix of the observations in $BOP_{oob}(\mathbf{x}^*)$, respectively:

$$\hat{\mathbf{Y}}^* = \frac{1}{|BOP_{oob}(\mathbf{x}^*)|} \sum_{i \in BOP_{oob}(\mathbf{x}^*)} y_i,$$

$$\hat{\mathbf{\Sigma}}^*_{\mathbf{Y}} = \frac{1}{|BOP_{oob}(\mathbf{x}^*)| - 1} \sum_{i \in BOP_{oob}(\mathbf{x}^*)} (y_i - \hat{\mathbf{Y}}^*)(y_i - \hat{\mathbf{Y}}^*)^\top,$$

where y_i is an observation from $BOP_{oob}(\mathbf{x}^*)$, $\hat{\mathbf{Y}}^*$ is the predicted response mean and $\hat{\mathbf{\Sigma}}^*_{\mathbf{Y}}$ represents the estimated conditional covariance matrix of the responses given $\mathbf{x}^*$.

5.2 Application

5.2.1 Small-Scale Example

We will work with this familiar dataset again:

X_1: Hours Studied	X_2: Time Spent on Papers	Y_1: Math Scores	Y_2: Science Scores
5	2	78	80
7	3	85	79
8	4	88	88
3	1	65	70
10	5	92	74

Table 5.1: Study Time vs. Exam Scores

Each tree is trained on a bootstrap sample, which is obtained by randomly drawing observations with replacements from the original dataset. For the provided dataset, a possible bootstrap sample could be, for example, rows 1,1,2,4 and 5. Here, you can see that row 1 has been repeated, and row 3 has been left out. Another possible bootstrap sample is just getting each row once, so rows 1,2,3,4 and 5. This corresponds to:

$$\mathbf{X} = \begin{bmatrix} (5, 2, 78, 80) \\ (7, 3, 85, 79) \\ (8, 4, 88, 88) \\ (3, 1, 65, 70) \\ (10, 5, 92, 74) \end{bmatrix}.$$

After constructing a bootstrap sample, recursive partitioning is performed by selecting the split that maximises the scaled Euclidean distance between the covariance matrices of the left and right child

nodes. The Euclidean distance is computed as:

$$d(\mathbf{\Sigma}^L, \mathbf{\Sigma}^R) = \sqrt{\sum_{i=1}^{q}\sum_{j=i}^{q}(\mathbf{\Sigma}_{ij}^L - \mathbf{\Sigma}_{ij}^R)^2}, \tag{5.2}$$

where $\mathbf{\Sigma}^L$ and $\mathbf{\Sigma}^R$ are the empirical covariance matrices of the left and right nodes, respectively.

In reality, the CRRF will evaluate all possible splits and compute 5.1, but for simplicity, let us assume the split that maximises 5.1 is $X_1 < 7$. It is important to understand how 5.1 is computed for any split. First, let us show how this is done for $X_1 < 7$. This starts by splitting the data into 2 child nodes: left and right. The left child node contains the observations and responses of:

$$\mathbf{X}_L = \begin{bmatrix} 5 & 2 & 78 & 80 \\ 3 & 1 & 65 & 70 \end{bmatrix},$$

and the right child node contains:

$$\mathbf{X}_R = \begin{bmatrix} 7 & 3 & 85 & 79 \\ 8 & 4 & 88 & 88 \\ 10 & 5 & 92 & 74 \end{bmatrix}.$$

For each node, the mean response is computed as:

$$\bar{\mathbf{Y}}_L = \frac{1}{2}\sum \mathbf{Y}_L = \begin{bmatrix} 71.5 & 75 \end{bmatrix}, \quad \bar{\mathbf{Y}}_R = \frac{1}{3}\sum \mathbf{Y}_R = \begin{bmatrix} 88.33 & 80.33 \end{bmatrix}.$$

The covariance matrices are then estimated using:

$$\mathbf{\Sigma}^L = \frac{1}{n_L - 1}\sum_{i \in L}(\mathbf{Y}_i - \bar{\mathbf{Y}}_L)(\mathbf{Y}_i - \bar{\mathbf{Y}}_L)^\top, \quad \mathbf{\Sigma}^R = \frac{1}{n_R - 1}\sum_{i \in R}(\mathbf{Y}_i - \bar{\mathbf{Y}}_R)(\mathbf{Y}_i - \bar{\mathbf{Y}}_R)^\top.$$

The computed covariance matrices for the left and right nodes are:

$$\mathbf{\Sigma}^L = \begin{bmatrix} 142.25 & 105.25 \\ 105.25 & 97.25 \end{bmatrix}, \quad \mathbf{\Sigma}^R = \begin{bmatrix} 41.67 & -10.67 \\ -10.67 & 21 \end{bmatrix}.$$

Applying the Euclidean distance formula to these matrices gives (to 2dp):

$$d(\mathbf{\Sigma}^L, \mathbf{\Sigma}^R) = \sqrt{(142.25 - 41.67)^2 + 2(105.25 - (-10.67))^2 + (97.25 - 21)^2} = 206.89.$$

Finally, the scaled Euclidean distance is:

$$\sqrt{n_L n_R} \times d(\mathbf{\Sigma}^L, \mathbf{\Sigma}^R) = 206.89\sqrt{2 \times 3} = 506.77$$

Since CRRF selects the split that maximises this distance, we can see the split at $X_1 = 7$ does lead to a large Euclidean distance between the response sample covariance matrices of each child node.

When predicting, each new observation is assigned to a terminal node based on its feature values. The prediction is computed using the BOP within the same node. For a new test point $\mathbf{x}^* = (6, 3)$, which falls into the left node, the relevant BOP observations here are:

$$BOP_{oob}(\mathbf{x}^*) = \{(78, 80), (65, 70)\}.$$

The predicted mean response is then given by:

$$\hat{\mathbf{Y}}^* = \frac{1}{|BOP_{oob}(x^*)|} \sum_{i \in BOP_{oob}(x^*)} \mathbf{Y}_i = \begin{bmatrix} 71.5 \\ 75 \end{bmatrix}.$$

The covariance structure of the predictions is estimated using:

$$\hat{\mathbf{\Sigma}}^* = \frac{1}{|BOP_{oob}(x^*)| - 1} \sum_{i \in BOP_{oob}(x^*)} (\mathbf{Y}_i - \hat{\mathbf{Y}}^*)(\mathbf{Y}_i - \hat{\mathbf{Y}}^*)^\top.$$

This results in

$$\hat{\mathbf{\Sigma}}^* = \begin{bmatrix} 142.25 & 105.25 \\ 105.25 & 97.25 \end{bmatrix}.$$

In this case, the prediction, $\hat{\mathbf{Y}}^*$, and its covariance matrix, $\hat{\mathbf{\Sigma}}^*$, correspond to $\bar{\mathbf{Y}}_L$ and $\mathbf{\Sigma}^L$ respectively because there is only 1 tree that has been made. For CRRF, many trees are formed from multiple bootstraps because each bootstrap leads to different splits being determined to maximise the Euclidean distance. As a result, the output $\hat{\mathbf{Y}}^*$ will be adjusted when more and more trees are added.

5.2.2 Code Explanation

CRRF came with a pre-set package in R called `CovReg`. This made its implementation simpler than manual coding in R. However, there were still some considerations.

To prevent over-fitting, explicit stopping criteria were implemented in the tree growth process. A node was set as terminal if either the depth limit of the tree was reached, `depth = 0`, or the number of samples in the node falls below the threshold, $n < 5$. In these cases, the node retains the mean and covariance of the responses, rather than splitting further. This procedure stops the tree from becoming too large and avoids overfitting.

Furthermore, to evaluate the performance of CRRF, a 5-fold CV was implemented using the ANRMSE. The data was randomly split into 5 folds of equal size, where each fold was used as a validation set once, and the remaining 4 folds formed the training set. This ensures that every observation is used in training and validation. At the end of the CV, the final ANRMSE of all the test folds is used to measure CRRF fit.

This code also uses parallelisation, which speeds up training by running multiple trees concurrently rather than sequentially. Parallel computing makes programs and processes run faster as more CPUs are used.[46] They are implemented using multi-threading, allowing trees to be trained simultaneously, thereby significantly reducing computation time.

5.2.3 Results

The CRRF model achieved an ANRMSE of 0.5238, the lowest value thus far. This model did not even consider interaction and polynomial terms, which were not explicitly tested as RFs already capture non-linear relationships and interactions through their splits.

The CRRF model was also analysed in how it determined predictor feature importance in modelling the 2 responses, ROE and sustainability score.

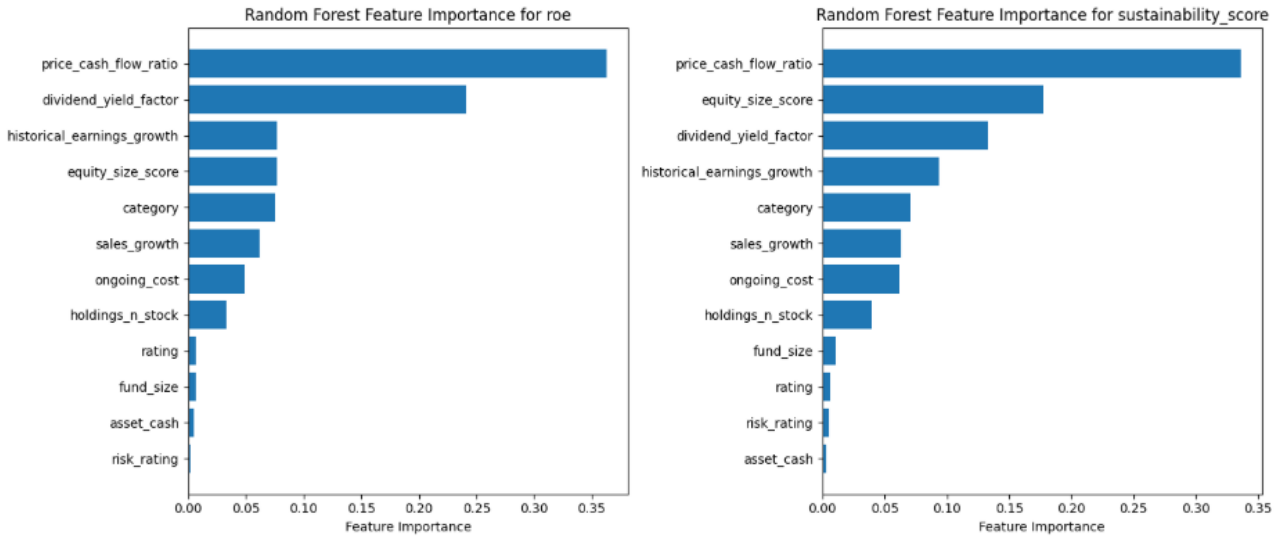

Figure 5.3: CRRF Feature Importance

For ROE, CRRF assigns the highest importance to `price_cash_flow_ratio`, `dividend_yield_factor`, and `equity_size_score`. Therefore, these features lead to the most significant changes in the covariance structure of responses. In contrast, `sales_growth`, `risk_rating`, and `asset_cash` have minimal influence, implying they do not meaningfully alter response covariances.

For sustainability score, `price_cash_flow_ratio` again is the strongest predictor, reinforcing its role in separating subgroups with differing response covariances. `equity_size_score` becomes more relevant here, which implies that it determines sustainability-linked covariance changes. `historical_earnings_growth` and `dividend_yield_factor` remain relevant but with smaller effect compared to `price_cash_flow_ratio`. `sales_growth` remains as important for sustainability score as for roe and `risk_rating` still has minimal importance.

Overall, the consistent importance of `price_cash_flow_ratio` highlights its strong role in dictating which equity funds should be invested in long-term based on the CRRF, whereas features with minimal covariance impact, like `risk_rating` and `asset_cash`, are not prioritised.

Although CRRFs have well-modelled this dataset, they have some drawbacks. First, they are computationally intensive because they need to calculate covariance matrices at each split. Second, although CV mitigates this somewhat, they are prone to over-fitting with irrelevant features. XGBoost can deal with all of these because it uses approximate tree learning and implements regularisation.

Chapter 6 XG Boost

This chapter investigated Extreme Gradient Boosting by first explaining its principles in the single-response case, including how it improves upon traditional gradient boosting. This method was then extended to the multivariate case using the Cholesky Decomposition to decorrelate the responses prior to fitting. The resulting model was then evaluated on the equity fund dataset to assess its performance.

6.1 Theory

Now, we move on to Extreme Gradient Boosting, which differs from Random Forests as it builds trees sequentially rather than independently. While Random Forests rely on bagging to reduce variance, XGBoost fits each new tree to the negative gradients of the loss function with respect to the current model's predictions. This iterative process forms an ensemble, refining predictions at every step.

6.1.1 Introduction

Gradient tree boosting builds a model by sequentially adding regression trees to minimise a predefined loss function. Each new tree is trained on the negative gradient of the current model, enabling the ensemble to iteratively refine its predictions.[47]

Formally, for a dataset with n observations and m features, we write:

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}, \quad \mathbf{x}_i \in \mathbb{R}^m,\ y_i \in \mathbb{R}, \quad i = 1, \ldots, n.$$

The prediction for each input, $\mathbf{x}_i$, is the sum of K regression trees:

$$\hat{y}_i = \phi(\mathbf{x}_i) = \sum_{k=1}^{K} f_k(\mathbf{x}_i), \quad f_k \in \mathcal{F}, \tag{6.1}$$

where each f_k is an individual regression tree in the ensemble and $\mathcal{F}$ is the space of all possible regression trees.[47]

One immediate point of concern is that the estimate will keep increasing as the number of regression trees does. However, each function f_k is not an independent prediction of y_i but rather a correction term trained to minimise the loss function with respect to the ensemble's current prediction.

For a general loss function $\ell(y_i, \hat{y}_i)$, this correction corresponds to the negative gradient, g_i, of the loss with respect to the current prediction:

$$f_k(\mathbf{x}_i) \approx -\frac{\partial \ell(y_i, \hat{y}_i)}{\partial \hat{y}_i}\Big|_{\hat{y}_i = \hat{y}_i^{(t-1)}} = -g_i.$$

In the case of the squared error loss function:

$$\ell(y_i, \hat{y}_i) = \frac{1}{2}(y_i - \hat{y}_i)^2, \tag{6.2}$$

the negative gradient simplifies to the residual $y_i - \hat{y}_i^{(t-1)}$.

The space of all possible regression trees can be formally written as:

$$\mathcal{F} = \left\{ f(\mathbf{x}) = w_{q(\mathbf{x})} \;\middle|\; q : \mathbb{R}^m \to \{1, \ldots, T\},\ w \in \mathbb{R}^T \right\}, \tag{6.3}$$

where T is the number of leaves in the tree, q is a function that maps an input $\mathbf{x}$ to its corresponding leaf index, and $w \in \mathbb{R}^T$ is a vector of leaf weights.[47] Each weight w_j corresponds to a real-valued score assigned to leaf j. Therefore, the leaf weights are the prediction scores returned by the tree. When an input $\mathbf{x}_i$ is passed through the tree, the structure q routes it to a specific leaf $q(\mathbf{x}_i)$, and the tree returns the corresponding score $w_{q(\mathbf{x}_i)}$ as its prediction.

Unlike classification trees, which output discrete class labels at each leaf, regression trees assign continuous numerical values, also referred to as prediction scores, to their leaves. These values are learned during training and represent the model's output for inputs that fall into the respective leaf.

Building on Equations 6.1 and 6.3, each regression tree has its own structure, $q^{(k)}$, and set of leaf scores, $w^{(k)}$.[47] Hence, the final model prediction is the sum of these scores across all trees:

$$\hat{y}_i = \sum_{k=1}^{K} w^{(k)}_{q^{(k)}(\mathbf{x}_i)}.$$

Extreme Gradient Boosting, otherwise known as XGBoost, improves gradient boosting by including regularisation explicitly in the loss function to address over-fitting and improve model predictions. Its objective function is:

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{t=1}^{T} \Omega(f_t), \tag{6.4}$$

where l represents the loss function, and $\Omega(f_t)$ is a regularisation term. The loss function in XGBoost regression can take various forms, such as the log loss. In this report, however, the unscaled squared error loss was used due to its simplicity in gradient derivation. See Equation 6.2. The omission of the $\frac{1}{2}$ has no impact on model performance, as the constant does not affect the direction of the gradient or the speed of convergence.

XGBoost uses regularisation to help control model complexity and mitigate over-fitting as follows:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2,$$

where T is the number of leaves in the tree, w_j are the leaf weights, γ is the pruning parameter which penalises additional leaves, and λ controls the regularisation of leaf weights.[47]

The pruning parameter can be selected automatically via CV or hyper-parameter tuning. However, it is ultimately set by the user. See Equation 6.5 for more on this parameter.

These weights are what XGBoost aims to determine, as they comprise the prediction values assigned to each leaf. XGBoost computes them using the Approximate Greedy Split-Finding Algorithm, which is detailed in Subsection 6.1.2.

XGBoost also incorporates shrinkage and column subsampling to reduce overfitting further. Shrinkage scales the leaf weights of newly added trees by a factor $\eta \in (0, 1]$, known as the learning rate. This reduces the influence of each individual tree, allowing subsequent trees more room to improve the model incrementally.[47] Column subsampling, in which only a random subset of features is considered when constructing each tree, not only acts as a form of regularisation but also improves computations of the parallel algorithm.[47]

XGBoost optimises the objective function using a second-order Taylor expansion. With constant

terms omitted, the loss function at iteration t is approximated as:

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2 \right] + \Omega(f_t),$$

where $g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ and $h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ are the first and second-order gradients of the loss function, respectively.[47]

To construct decision trees, XGBoost evaluates potential binary splits using the Greedy Algorithm for Split Finding. This algorithm involves maximising a gain function:

$$\text{Gain} = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma.^{47} \tag{6.5}$$

Each candidate split divides the training data into two subsets based on a feature threshold. The instances satisfying the condition (e.g., $x_j \leq s$) are assigned to the left node, L, while those that do not are assigned to the right node, R.

Here, G_L and H_L are the sums of first and second-order gradients for the left child, G_R and H_R are the corresponding sums for the right child, R, λ is a regularisation parameter, and γ is the pruning parameter.[47] A "child" refers to the rows selected when a decision tree splits at a node.

Equation 6.5 means that a node will only be split if the gain in the objective function exceeds the pruning parameter, γ. Higher values of γ encourage shallower trees by penalising complexity.

Each component within the Gain's bracket is called the (similarity) score. So, the Gain can be written as:

$$\text{Gain} = \frac{1}{2} \left[S_L + S_R - S_P \right] - \gamma,$$

where:

$$S_L = \frac{G_L^2}{H_L + \lambda}, \quad S_R = \frac{G_R^2}{H_R + \lambda}, \quad S_P = \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}.$$

Note: S_P is the score of the parent node of the new node being derived.

6.1.2 Iterative Steps of XGBoost

As outlined in Chen and Guestrin (2016), the exact Greedy Algorithm occurs by iterating over each feature dimension k, ranging from 1 to m, where m is the total number of predictors.

For each feature, potential split points are evaluated by sorting the training instances based on their corresponding feature values. During this process, the cumulative sums of gradients and Hessians for the left child node, denoted G_L and H_L, are initialised to zero and updated at each step as:

$$G_L \leftarrow G_L + g_j, \quad H_L \leftarrow H_L + h_j.$$

The corresponding values for the right child are computed as $G_R = G - G_L$ and $H_R = H - H_L$, where G and H are the total sums over the parent node.

The quality of each split is evaluated using the gain, which measures the reduction in loss. At each step, the algorithm compares the gain of the current split to the best one found so far:

$$\text{Gain}_{\text{new}} = S_L + S_R - S_P,$$

and retains the maximum.

Note: the pruning parameter γ is applied after split selection, not during the gain evaluation.

After evaluating all candidate splits, the algorithm selects the one with the highest gain. If no split yields a positive gain, the node remains a leaf. This approach ensures that each decision tree

grows in a way that minimises residual error at each step, leading to more accurate predictions.

The exact Greedy Algorithm is ideal for XGBoost, but due to its computational cost, an approximate greedy algorithm is used in practice. This method is similar to the prior one with some adjustments, which are explained below, again from Chen and Guestrin (2016).

The approximate greedy algorithm begins by iterating over each feature dimension $k = 1, \ldots, m$, where m is the total number of features.

For each feature k, a set of candidate splits $S_k = \{s_{k1}, s_{k2}, \ldots, s_{kl}\}$ is proposed by dividing the feature values into percentiles. This ensures the split proposals are distributed evenly across the feature range, capturing key points that are likely to lead to huge loss reduction. Split proposals can be generated globally or locally: in the global setting, split points are selected once per tree and reused at each node, while in the local setting, new split points are computed independently for each node.

After generating the candidate splits, the algorithm evaluates their quality. For each proposed split $s_{k,v}$ in feature k, the gradients and Hessians of instances falling between two consecutive split points are accumulated. The gradients and Hessians are then computed as:

$$G_{kv} \leftarrow= \sum_{j \in \{j | s_{k,v} \geq \mathbf{x}_{jk} > s_{k,v-1}\}} g_j$$
$$H_{kv} \leftarrow= \sum_{j \in \{j | s_{k,v} \geq \mathbf{x}_{jk} > s_{k,v-1}\}} h_j$$

where G_{kv} denotes the sum of gradients and H_{kv} the sum of Hessians over the interval between $s_{k,v-1}$ and $s_{k,v}$. This process approximates the exact split-finding procedure by evaluating only a limited subset of potential split points.

Once the gradients and Hessians have been computed for all proposed splits, the algorithm selects the split that maximises the gain. This follows the same logic as the exact greedy method but limits evaluation to the proposed split points, significantly reducing computational cost.

The same procedure described earlier is then applied but restricted to the proposed splits. While this approximation sacrifices some precision, it greatly improves efficiency, allowing XGBoost to scale to large datasets.

After identifying the optimal split, the corresponding leaf weight is computed as:

$$w_j = -\frac{G_j}{H_j + \lambda},$$

where G_j is the sum of gradients for all instances in the leaf, H_j is the sum of Hessians for all instances in the leaf, and λ is the regularisation parameter.[47]
Each instance's prediction is updated as follows:

$$\hat{y}^{(t)} = \hat{y}^{(t-1)} + \eta w_j,$$

where $\hat{y}^{(t-1)}$ represents the previous prediction, w_j is the weight of the leaf the instance falls into, and η is the learning rate, which controls the step size of updates.[47]

6.1.3 Cholesky Decomposition

Although a prior distribution is not required to apply Cholesky decomposition, assuming a multivariate normal distribution (MVN) helps in satisfying its requirements, which are a positive definite symmetric matrix. The probability density function of an MVN distribution is given by:

$$f(\mathbf{Y}|\boldsymbol{\mu}_\mathbf{Y}, \Sigma_\mathbf{Y}) = \frac{1}{\sqrt{(2\pi)^D |\Sigma_\mathbf{Y}|}} \exp\left(-\frac{1}{2}(\mathbf{Y} - \boldsymbol{\mu}_\mathbf{Y})^\top \Sigma_\mathbf{Y}^{-1} (\mathbf{Y} - \boldsymbol{\mu}_\mathbf{Y})\right),$$

where $\boldsymbol{\mu}_{\mathbf{Y}} \in \mathbb{R}^D$ represents a vector of conditional means, $\Sigma_{\mathbf{Y}}$ is the positive definite symmetric $D \times D$ response covariance matrix, D is the number of responses, and $|\cdot|$ denotes the determinant.[48]

In the bivariate case ($D = 2$), the local response covariance matrix for instance i, denoted $\boldsymbol{\Sigma}_{i\mathbf{y}}$, is:

$$\boldsymbol{\Sigma}_{i\mathbf{y}} = \begin{bmatrix} \sigma_{i,1}^2(\mathbf{x}) & \rho_i(\mathbf{x})\sigma_{i,1}(\mathbf{x})\sigma_{i,2}(\mathbf{x}) \\ \rho_i(\mathbf{x})\sigma_{i,1}(\mathbf{x})\sigma_{i,2}(\mathbf{x}) & \sigma_{i,2}^2(\mathbf{x}) \end{bmatrix},$$

where $\rho_i(\mathbf{x})$ is the conditional correlation between the two responses, the diagonal entries represent the variances, and the off-diagonal entries represent the covariances, for each $i = 1, \ldots, N$.[48]

To ensure positive definiteness of the response covariance matrix, $\boldsymbol{\Sigma}_{\mathbf{Y}}$, a common and efficient approach is to use the Cholesky decomposition, which factorises the matrix as follows:

$$\boldsymbol{\Sigma}_{\mathbf{Y}} = \mathbf{L}\mathbf{L}^\top,$$

where $\mathbf{L} \in \mathbb{R}^{D \times D}$ is a lower triangular matrix.[48] This decomposition guarantees positive definiteness as long as all diagonal elements ℓ_{ii} of $\mathbf{L}$ are strictly positive. The $D(D-1)/2$ off-diagonal elements ℓ_{ij} (for $j < i$) may take any value in $\mathbb{R}$.

The Cholesky decomposition uses the following algorithm to derive $\mathbf{L}$:

1. Initialise $\mathbf{L}$ as an $n \times n$ zero matrix.
2. For each row i (from 1 to n):
   - Compute the diagonal element L_{ii}:
   $$L_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2}$$
   - For each column j (below diagonal, $j > i$):
   $$L_{ji} = \frac{A_{ji} - \sum_{k=1}^{i-1} L_{jk}L_{ik}}{L_{ii}}$$
3. Return $\mathbf{L}$.

Cholesky decomposition can be used to remove intercorrelation between response variables by applying the transformation $\mathbf{L}^{-1}\mathbf{Y}^\top$, which produces decorrelated responses.[49] This decorrelation allows XGBoost to be applied independently to each transformed response. After prediction, the original correlated response space is recovered by reapplying $\mathbf{L}$, which reintroduces the covariance structure.

There does exist a version of XGBoost, Multi-Output XGBoost, which supports MRR with an in-built mechanism. However, its methodology still needs to be fleshed out.

6.2 Application

6.2.1 Small-Scale Example

We will work with the familiar dataset evaluating Mathematics and Science Scores:

X_1: Hours Studied	X_2: Time Spent on Papers	Y_1: Math Scores	Y_2: Science Scores
5	2	78	80
7	3	85	79
8	4	88	88
3	1	65	70
10	5	92	74

Table 6.1: Study Time vs. Exam Scores

We have:

$$\mathbf{X} = \begin{bmatrix} 5 & 7 & 8 & 3 & 10 \\ 2 & 3 & 4 & 1 & 5 \end{bmatrix}^\top, \quad \mathbf{Y} = \begin{bmatrix} 78 & 85 & 88 & 65 & 92 \\ 80 & 79 & 88 & 70 & 74 \end{bmatrix}^\top, \quad \bar{\mathbf{Y}} = \begin{bmatrix} 81.6 & 78.2 \end{bmatrix}^\top.$$

First, we need the response covariance matrix, $\mathbf{\Sigma_Y}$:

$$\mathbf{\Sigma_Y} = \frac{1}{n-1}(\mathbf{Y} - \bar{\mathbf{Y}})^\top(\mathbf{Y} - \bar{\mathbf{Y}}) = \begin{bmatrix} 112.30 & 37.85 \\ 37.85 & 46.20 \end{bmatrix},$$

so each response value has its column mean subtracted from it. The matrix is symmetric and positive definite because its (1,2) and (2,1) entries are the same, and it has positive eigenvalues.

Next comes the Cholesky Decomposition of the response covariance matrix. This subsection is for clarity, so for ease of decomposition, let us show Cholesky Decomposition on a "nicer" matrix:

$$\mathbf{A} = \begin{bmatrix} 4 & 2 \\ 2 & 3 \end{bmatrix}.$$

We aim to find a lower triangular matrix $\mathbf{L}$ such that:

$$\mathbf{A} = \mathbf{L}\mathbf{L}^\top, \quad \mathbf{L} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix}$$

The first step is to compute L_{11} using the formula for diagonal elements:

$$L_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2} \Rightarrow L_{11} = \sqrt{A_{11}} = \sqrt{4} = 2 \Rightarrow \mathbf{L} = \begin{bmatrix} 2 & 0 \\ L_{21} & L_{22} \end{bmatrix}$$

Next, compute L_{21}. For off-diagonal elements:

$$L_{ji} = \frac{A_{ji} - \sum_{k=1}^{i-1} L_{jk}L_{ik}}{L_{ii}} \Rightarrow L_{21} = \frac{A_{21}}{L_{11}} = \frac{2}{2} = 1 \Rightarrow \mathbf{L} = \begin{bmatrix} 2 & 0 \\ 1 & L_{22} \end{bmatrix}$$

Finally, compute L_{22}, going back to our diagonal element formula:

$$L_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2} \Rightarrow L_{22} = \sqrt{A_{22} - L_{21}^2} = \sqrt{3-1} = \sqrt{2} \Rightarrow \mathbf{L} = \begin{bmatrix} 2 & 0 \\ 1 & \sqrt{2} \end{bmatrix}$$

See A.2.7 for verification this decomposition is correct.

Applying this algorithm to $\mathbf{\Sigma_Y}$ gives, when passing it through R's inbuilt `chol` function:

$$\mathbf{\Sigma_Y} = \mathbf{L}\mathbf{L}^\top = \begin{bmatrix} 10.60 & 0 \\ 3.57 & 5.78 \end{bmatrix} \begin{bmatrix} 10.60 & 3.57 \\ 0 & 5.78 \end{bmatrix}$$

Now we can decorrelate the responses using $\mathbf{L}^{-1}$:

$$\tilde{\mathbf{Y}}^\top = \mathbf{L}^{-1}\mathbf{Y}^\top = \begin{bmatrix} 7.36 & 8.02 & 8.30 & 6.13 & 8.68 \\ 9.29 & 8.71 & 10.09 & 8.32 & 7.43 \end{bmatrix}$$

Here the off-diagonal elements of the covariance matrix of the transformed responses are zero, and it leaves the identity matrix, $\mathbf{I}_2$:

$$\text{Cov}(\tilde{\mathbf{Y}}) \approx \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Now that we have decorrelated the responses, we can apply an XGBoost model to each response without needing to consider the response covariance matrix.

Let us apply XGBoost, with a squared error loss function, to response $\tilde{Y}_1$, Mathematics Scores against the predictors X_1 and X_2. XGBoost starts by initialising predictions for all samples. The initial prediction is typically 0.5, but let us start here instead by taking the mean of $\tilde{Y}_1$:

$$\hat{Y}_1^{(0)} = \frac{7.36 + 8.02 + 8.30 + 6.13 + 8.68}{5} = \frac{38.49}{5} = 7.70. \tag{6.6}$$

Thus, the initial prediction for all samples is:

$$\hat{Y}_1^{(0)} = \begin{bmatrix} 7.70 & 7.70 & 7.70 & 7.70 & 7.70 \end{bmatrix}^\top.$$

Next, the residuals are calculated as the difference between the actual values $\tilde{Y}_1$ and the initial predictions (to 2dp):

$$\begin{aligned} r_1 = \tilde{Y}_1 - \hat{Y}_1^{(0)} &= \begin{bmatrix} 7.36 & 8.02 & 8.30 & 6.13 & 8.68 \end{bmatrix}^\top - \begin{bmatrix} 7.70 & 7.70 & 7.70 & 7.70 & 7.70 \end{bmatrix}^\top \\ &= \begin{bmatrix} -0.34 & 0.32 & 0.60 & -1.57 & 0.98 \end{bmatrix}^\top. \end{aligned}$$

The residuals are the errors in the initial predictions, which will be used to fit the first XGBoost tree. The residuals are taken because these are half the gradients when using the squared error loss function. The Gradients, g_i, and Hessians, h_i, are needed because they are critical in evaluating the Score and computing the Gain. The gradients are calculated as:

$$g_i = \frac{\partial L}{\partial \hat{Y}_i} = \frac{\partial}{\partial \hat{Y}_i} \sum_{i=1}^{n} (\hat{Y}_i - Y_i)^2 = 2(Y_i - \hat{Y}_i) = 2r_i,$$

where the summation vanishes because we are looking at each i^{th} element when differentiating. From the residuals, the gradients are (to 2dp):

$$g = \begin{bmatrix} -0.68 & 0.64 & 1.21 & -3.13 & 1.96 \end{bmatrix}^\top.$$

For squared error loss, the second-order gradient (Hessian) is:

$$h_i = \frac{\partial^2 L}{\partial \hat{Y}_i^2} = 2.$$

Since we are using the squared error loss function, the Hessian for all points is:

$$h = \begin{bmatrix} 2 & 2 & 2 & 2 & 2 \end{bmatrix}^\top$$

Now, we train a small decision tree to predict the gradients. This starts with the parent node, P, which contains all the data. Each tree starts with a single leaf, and all residuals go to that leaf.[50] All that needs to be done here is to compute its score using the exact residuals:

$$S_P = \frac{(\sum g_P)^2}{\sum h_P + \lambda} = 0.$$

where g_p, h_p are the Gradients and Hessians of the parent node, respectively. This makes sense because, due to this loss function, the gradients are proportional to the residuals, which always sum to zero. When the gradients in a node are very different, they cancel each other out and the Score is relatively small and you expect large gradients when initialising.[50] In contrast, when the gradients are similar, or there is just one of them, they do not cancel out, and the Score is relatively large.[50]

XGBoost will then evaluate splits after this by comparing the Gain for each split in X_1 and X_2 in this dataset and finding the split which maximises this. Maximising the Gain means there is a better splitting of the gradients into clusters of similar values.[50] Iterating through all potential split points can be computationally expensive, so this is where a weighted quantile sketch approximates the best-split points using weighted quantiles. See A.2.7 for more. This speeds up training by reducing the number of candidate split points.

Now, let us show how the gain is computed for an example split: $X_1 < 7$. $X_1 < 7$ splits the data into 2 sets of predictors and corresponding transformed responses:

X_1: Hours Studied	X_2: Time Spent on Papers	$\tilde{Y}_1$: Mathematics Scores	r_0: Residuals
5	2	7.36	-0.34
3	1	6.13	-1.57

Table 6.2: Study Time vs. Mathematics Scores for $X_1 < 7$

X_1: Hours Studied	X_2: Time Spent on Papers	$\tilde{Y}_1$: Mathematics Scores	r_0: Residuals
7	3	8.02	0.32
8	4	8.3	0.60
10	5	8.68	0.98

Table 6.3: Study Time vs. Mathematics Scores for $X_1 \geq 7$

Science Scores, $\hat{Y}_2$, have been removed as we are just modelling $\tilde{Y}_1$ here, and the initial residuals are placed for clarity when calculating the gradients and Hessians.

Let us call 6.2 the "left", L, node and 6.3 the "right", R, node. From each node, calculate the cumulative gradients and Hessians as:

$$G_L = \sum_{i=1}^{2} 2(r_i)_L = -3.81, \quad H_L = \sum_{i=1}^{2} 2 = 4, G_R = \sum_{i=1}^{2} 2(r_i)_R = 3.81, \quad H_R = \sum_{i=1}^{3} 2 = 6.$$

This is used to calculate the similarity scores for the left and right nodes with $\lambda = 0.1$:

$$S_L = \frac{G_L^2}{H_L + \lambda} = 3.62, \quad S_R = \frac{G_R^2}{H_R + \lambda} = 2.42.$$

Now compute the Gain for $X_1 < 7$ in comparison with the parent node:

$$\text{Gain} = S_L + S_R - S_P = 6.04.$$

For simplicity, assume the tree split of $X_1 < 7$ gives the optimal gain so we can adjust the predictions. The leaf weight differs by the child node each row falls in and is calculated as follows (to 2dp):

$$w_L = -\frac{G_L}{H_L + \lambda} = 0.95, \quad w_R = -\frac{G_R}{H_R + \lambda} = -0.63$$

The predictions are then adjusted using the leaf weight with the learning rate η, which we set as 0.3. Here, the initial prediction was the mean, from 6.6. So, the final transformed predictions are:

$$\hat{Y}_{1,L}^{(1)} = \hat{Y}_1^{(0)} + \eta \times w_L = 7.98, \quad \hat{Y}_{1,R}^{(1)} = \hat{Y}_1^{(0)} + \eta \times w_R = 7.51.$$

These need to be converted back to the original response space with the original correlation structure, using: $\mathbf{L}\hat{Y}_{1,L}^{(1)}$ or $\mathbf{L}\hat{Y}_{1,R}^{(1)}$. This gives the following original predictions (to 2dp):

$$\text{Left Node} : \hat{Y}_1^{(1)} = 84.55, \quad \text{Right Node} : \hat{Y}_1^{(1)} = 79.61.$$

6.2.2 Code Explanation

This section describes how Cholesky-Gaussian XGBoost was applied to the equity fund dataset. The primary implementation detail, not covered thus far, involved incorporating k-fold CV.

The dataset was partitioned into five folds using `KFold()`, with shuffling enabled to ensure randomness in CV. For each fold, the empirical response covariance matrix of the target variables was computed using `numpy.cov()` to capture interdependencies between responses. Cholesky decomposition was then applied via `numpy.linalg.cholesky()`, producing a lower triangular matrix $\mathbf{L}$. Its inverse, $\mathbf{L}^{-1}$, was computed using `numpy.linalg.inv()` to decorrelate the response variables prior to model training. Each decorrelated response variable was then modelled independently using an XGBoost regression model implemented with `XGBRegressor()`. The models were trained on the transformed responses using `model.fit(X_train, Y_train)`, and predictions were generated on the validation set using `model.predict(X_val)`. As the predictions were produced in the transformed space, they were mapped back to the original scale by multiplying with the Cholesky factor via `numpy.dot()`. This step ensured that the predicted responses retained their original correlation structure. The adjusted predictions were subsequently used to compute the ANRMSE across all folds.

6.2.3 Results

XGBoost achieved an ANRMSE of 0.426, indicating an excellent fit. This was the best performance among all the models tested and highlights XGBoost's effectiveness in modelling complex datasets.

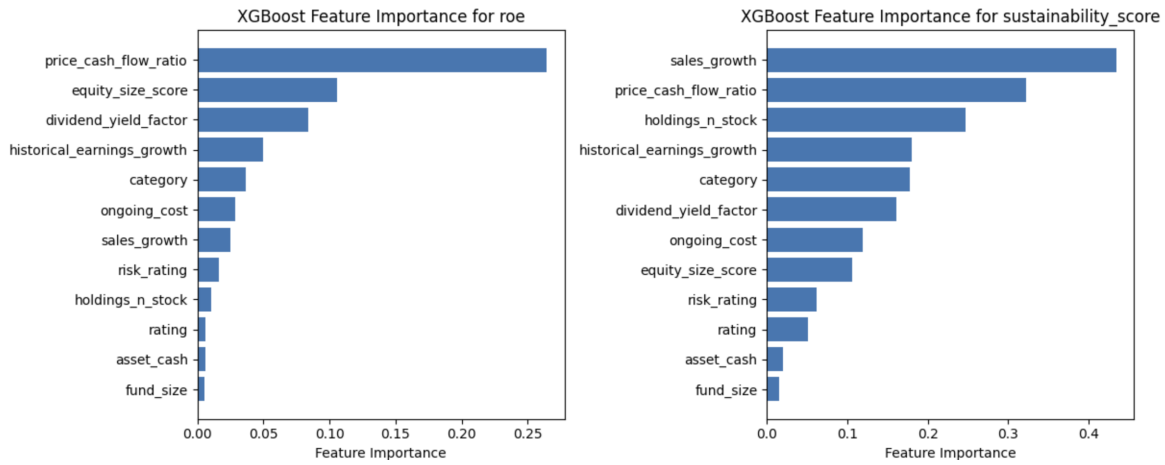

Figure 6.1: XGBoost Feature Importance

The most important feature in explaining the ROE was the `price_cash_flow_ratio`. This means that, following decorrelation, this ratio had the greatest impact in accounting for variation in the response. However, due to the Cholesky transformation, some of its influence may have been partially transferred to the sustainability score when mapping back to the original response space.

Other notable contributors included the `equity_size_score` and the `dividend_yield_factor`, reflecting their substantial role in predicting the decorrelated target. On the other hand, variables such as asset cash and fund size had relatively low feature importance, indicating that these features provided little additional information once dependencies between responses had been taken into account.

The most impactful feature for the sustainability score was `sales_growth`. Unlike ROE, where a dominant predictor emerged, sustainability appeared to be influenced by a few predictors, as indicated by the more even distribution of feature importance. The `price_cash_flow_ratio` also ranked highly, similar to its importance in predicting ROE. This supports that prior to decorrelation, ROE and the sustainability score shared some correlation, which was separated using Cholesky decomposition.

Other significant features for the sustainability score included `holdings_n_stock` and `historical_earnings_growth`, reinforcing that sustainability was influenced by multiple predictors rather than a single dominant one.

XGBoost proved highly effective on the equity fund dataset, but it has limitations in certain cases. One notable drawback is that its interpretation can be challenging due to its inherent complexity.[51]

Chapter 7 Conclusion

7.1 Summary of Findings and Model Comparisons

7.1.1 Summary of Findings

Here is a table summarising the key figures generated from this report from the model methods:

Table 7.1: Model Fit Comparison on Equity Fund Dataset

Model	ANRMSE	Model Fit
Chapter 3: Linear Regression		
Model 1: MRLR with all predictors	0.7606	Unsatisfactory
Model 2: Forward Selection	0.7606	Unsatisfactory
Model 3: Backward Selection	0.7924	Unsatisfactory
Model 4: Bidirectional Selection	0.7606	Unsatisfactory
Model 5: Bidirectional Selection with Interaction Terms	0.5613	Good
Model 6: Bidirectional Selection with Non-Linear Terms	0.6893	Satisfactory
Chapter 4: Shrinkage Methods		
Model 7: MRCE	0.7612	Unsatisfactory
Model 8: RRRR	0.7510	Unsatisfactory
Model 9: MRCE including Polynomial Terms	0.6781	Satisfactory
Model 10: RRRR including Polynomial Terms	0.7276	Unsatisfactory
Model 11: MRCE including Interaction Terms	0.6777	Satisfactory
Model 12: RRRR including Interaction Terms	0.7231	Unsatisfactory
Chapter 5: Random Forests		
Model 13: Covariance Regression with Random Forests	0.5238	Good
Chapter 6: XGBoost		
Model 14: Cholesky-Decomposition XGBoost	0.4267	Excellent

The model fit was defined based on criteria set out in Chapter 3 (see subsection 3.2.1).

7.1.2 Model Comparisons

The MRLR model using all predictors performed poorly, and all the stepwise selection methods (forwards, backwards and bi-directional) failed to improve performance significantly. Backward selection performed the worst, suggesting that eliminating variables here led to worse model performance.

However, the introduction of interaction terms, alongside bi-directional stepwise selection, led to a huge improvement, resulting in a "Good" model fit. Similarly, the addition of non-linear terms, also selected through bi-directional stepwise selection, improved upon the MRLR model, with just the standard predictors, and achieved a "Satisfactory" fit.

The shrinkage methods aimed to improve regression stability by using penalisation techniques. However, the MRCE and RRRR models, with the regular features, performed similarly to their counterparts in standard regression, again returning an "Unsatisfactory" fit.

Introducing polynomial terms in MRCE improved performance, but the same approach for RRRR was less effective. Adding interaction terms improved MRCE but did not improve RRRR as much.

The performance of shrinkage methods was also noteworthy because, despite being specifically designed to handle multiple responses, particularly under multicollinearity or high-dimensional settings, both MRCE and RRRR performed as poorly as, or even worse than, the MRLR models. This can be attributed to the structure of the dataset: after pre-processing (see Chapter 2), the predictors were not highly correlated, and the data was not high-dimensional ($p = 14, n = 1269$). In this case, the advantages of regularisation and rank reduction are less impactful.

Tree-based models significantly improved over linear and shrinkage-based methods. The Random Forest model was significantly better than previous models and was evaluated as a "Good" fit model. Progress was also found with XGBoost, which was the first "Excellent" model.

The strong performance of XGBoost and CRRF, particularly in comparison to the limited improvement achieved by including interaction terms in regression models, indicates that the equity fund responses, sustainability score and ROE, are driven by subtle, conditional interactions among predictors. The only marginal multivariate normality, evidenced by Mardia's test (see Subsection 2.3.5), further corroborates the existence of non-linear effects and differential impacts. Overall, the findings suggest that the most effective modelling approaches are those that accommodate flexible, local structure and interaction effects, as opposed to approaches based on rigid parametric or distributional assumptions.

Ultimately, the results suggest that the predictors influence fund outcomes in ways that are not purely additive or globally consistent. Instead, relationships appear to be context-dependent. This means that the effect of one variable likely changes based on the values of others. This aligns with the success of models like XGBoost and CRRF, which are designed to capture these conditional structures. Conversely, the poor performance of linear and shrinkage-based models reflects their inability to adapt to such nuanced, data-specific patterns.

7.2 Challenges and Limitations

This report faced different methodological and practical challenges. The first of which was finding an appropriate dataset to evaluate these models. This was a challenge because it involved finding a dataset that satisfied all the assumptions underpinning each MRR model. In particular, priority was given to datasets with independent observations, multivariate normality between responses, full rank in the predictor matrix, and a moderate degree of correlation between response variables. These criteria ensured that the models could be appropriately applied and compared.

In retrospect, several limitations of the equity fund dataset itself became clearer through the modelling process. Although the dataset had a reasonably large number of observations, the number of predictors, $p = 14$, limited the potential benefits of the shrinkage methods used (MRCE and RRRR).

Missing data presented another challenge. In the report, linear model imputation for numerical

variables and random forests for the categorical ones. While these approaches are more sophisticated than simple imputation, they assume that data are missing at random, that is, the missingness depends only on observed values. This assumption may not hold, particularly if some variables are missing not at random and influenced by unobserved or sensitive information.

In retrospect, a formal assessment of the missingness mechanism would have been carried out. For example, Hotelling's multivariate t-test or absolute standardised mean differences could have been used to test if the distributions of observed covariates differed between missing and non-missing groups.

Ultimately, more advanced imputation techniques, such as multiple imputation via chained equations, provide an improved method of dealing with missing data.

Another limitation in this report was the relatively low degree of feature engineering. For example, frequency encoding was used for the equity category feature, which treated each fund category as a separate level. However, many of these categories could have been effectively grouped — for instance, by region (e.g., Sweden, Switzerland), sector (e.g., Consumer Goods), or market capitalisation (e.g., Large-Cap). Grouping them based on this basis might have improved performance on all of the models.

While some models incorporated interaction terms and polynomial features, a more systematic feature engineering process, such as using principal component analysis, could have improved model performance or revealed complex patterns. With more time and hence familiarity with the financial context of the dataset, this could have been explored further.

Moving beyond the dataset, another challenge was determining what constitutes a valid MRR model. Initially, models were included that, while multivariate in their structure, did not strictly qualify as multiple-response. For example, multivariate ridge regression (see subsection 4.1.3) applies a penalty across the full coefficient matrix, regularising all responses jointly, but it does not explicitly incorporate the response covariance matrix. Therefore, it is not a proper MRR model.

Finally, one of the most significant limitations was the interpretability of more complex models, particularly those in later chapters. While highly accurate, models such as XGBoost are often regarded as "black boxes," as their internal decision structures are not easily interpretable.[52] However, this lack of interpretability does not diminish their utility: the ability to predict accurately can still inform decision-making even if the model's internal mechanisms are not clear.

7.3 Report Overview and Future Work

This report began by examining MRLR, extending SRLR. MRLR was then combined with stepwise selection and sequential MANOVA to identify relevant predictors. The analysis then progressed to shrinkage-based methods, namely MRCE and RRRR, both of which introduced regularisation to improve model stability. Following this, tree-based approaches were explored through CRRFs, which aimed to maximise Euclidean distance between nodes during splits. Finally, XGBoost was adapted to multiple responses via the Cholesky-Gaussian decomposition, allowing for the decorrelation of response variables and the use of independent models on each response.

While this report has extensively covered several MRR models on a single dataset, there remain multiple avenues for future work. One direction would be to apply these models across a broad set of datasets to understand how performance varies under different conditions, such as varying levels of multicollinearity, response correlation, and sample size. This could help to further clarify which types of data are best suited for each model and why certain models perform better in specific scenarios.

Alternatively, future work could involve expanding the set of models applied to this dataset, for example, neural networks can be extended to MRR, and hence evaluated on this dataset, using Gaussian Processes. Ultimately, this could improve both predictions and insight into the underlying structure of the equity fund data.

Bibliography

1. James Chen. Investing in equity funds: A beginner's guide. *Investopedia*, April 2024. Reviewed by Gordon Scott, Fact checked by Ariel Courage.

2. B.R. Kumar. Sustainable finance and investment: The intersection of profitability and environmental impact. *Library Progress (International)*, 44:16477–16485, 10 2024.

3. Jason Fernando. Return on equity (roe) calculation and what it means. *Investopedia. URL: https://www. investopedia. com/terms/r/returnonequity. asp (date of access: 14.12. 2023)*, 2023.

4. Will Kenton. Net income (ni): Definition, uses, and formula, 2024. Investopedia. Updated June 25, 2024. Reviewed by Khadija Khartit and fact-checked by Kirsten Rohrs Schmitt.

5. Mitchell Franklin, Patty Graybeal, and Dixon Cooper. *Principles of accounting. Volume 1: Financial accounting*. OpenStax, 2019.

6. James Chen. Morningstar sustainability rating: Definition and how it works, 2023. Investopedia. Updated November 30, 2023. Reviewed by Gordon Scott.

7. Larry Fink. Larry fink's 2021 letter to ceos. `https://www.blackrock.com/corporate/about-us/sustainability-resilience-research`, January 2021. Chairman and CEO of BlackRock.

8. Kanti V Mardia, John T Kent, and Charles C Taylor. *Multivariate analysis*. John Wiley & Sons, 2024.

9. Stockopedia. Risk rating, 2024. Accessed: 27 November 2024.

10. Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning with Applications in R*. Springer, 2 edition, 2021.

11. AMG Funds LLC. Equity style analysis: Growth vs. value, 2023. Accessed: 27 November 2024.

12. Adam Hayes. Growth stock: What it is, examples, vs. value stock, December 2023. Updated 11 December 2023. Reviewed by Chip Stapleton. Fact checked by Kirsten Rohrs Schmitt. Accessed: 27 November 2024.

13. Hitul Adatiya. Eda on european mf dataset, 2022. Version 2 of 4. Accessed: 27 November 2024. Licensed under Apache 2.0 open source.

14. Chris B. Murphy. Operating costs definition: Formula, types, and real-world examples, 2024. Updated 28 June 2024. Reviewed by David Kindness. Fact checked by Amanda Jackson. Accessed: 27 November 2024. Part of the series: The Evolution of Accounting and its Terminology.

15. Michael H. Kutner, Christopher J. Nachtsheim, John Neter, and William Li. *Applied Linear Statistical Models*. McGraw-Hill/Irwin, 5 edition, 2005.

16. David A. Belsley, Edwin Kuh, and Roy E. Welsch. *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. Wiley, 1980.

17. Shangzhi Hong and Henry S Lynn. Accuracy of random-forest-based imputation of missing data in the presence of non-normality, non-linearity, and interaction. *BMC medical research methodology*, 20:1–12, 2020.

18. Cátia M. Salgado, Carlos Azevedo, Hugo Proença, and Susana M. Vieira. Missing data. In Ross C. Mitchell, editor, *Secondary Analysis of Electronic Health Records*, pages 143–162. Springer, Cham, 2016. Open Access chapter distributed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License.

19. openfunds. Fund ratios and exposures, 2020. Accessed: 2025-04-03.

20. Richard Johnson and Dean Wichern. Multivariate linear regression models: Section 7.7. In *Applied Multivariate Statistical Analysis: Pearson New International Edition*, pages 360–429. Pearson Education, Limited, 6th edition, 2013. Accessed: 27 November 2024.

21. Smith Gary. Step from stepwise. *Journal of Big Data*, 5(1):1–12, 2018.

22. Gudmund R. Iversen. Multivariate analysis of variance and covariance (manova and mancova). In Michael S. Lewis-Beck, Alan Bryman, and Tim Futing Liao, editors, *The SAGE Encyclopedia of Social Science Research Methods*, volume 2, pages 702–703. SAGE Publications, Inc., Thousand Oaks, CA, 2004.

23. Newsom. Multivariate analysis of variance. *Psy 522/622 Multiple Regression and Multivariate Quantitative Methods*, 2024. Winter 2024 Lecture Notes.

24. STAT 505 Pennsylvania State University. Lesson 8: Multivariate analysis of variance (manova), 2024.

25. Daniel N Moriasi, Jeffrey G Arnold, Michael W Van Liew, Ronald L Bingner, R Daren Harmel, and Tamie L Veith. Model evaluation guidelines for systematic quantification of accuracy in watershed simulations. *Transactions of the ASABE*, 50(3):885–900, 2007.

26. Kristin L. Sainani. Multivariate regression: The pitfalls of automated variable selection. *PM&R*, 5(9):791–794, 2013.

27. Ashin Mukherjee and Ji Zhu. Reduced rank ridge regression and its kernel extensions. *Statistical analysis and data mining: the ASA data science journal*, 4(6):612–622, 2011.

28. Ian T Jolliffe. *Principal component analysis for special types of data*. Springer, 2002.

29. Adam J Rothman, Elizaveta Levina, and Ji Zhu. Sparse multivariate regression with covariance estimation. *Journal of Computational and Graphical Statistics*, 19(4):947–962, 2010.

30. Jian Guo, Elizaveta Levina, George Michailidis, and Ji Zhu. Joint estimation of multiple graphical models. *Biometrika*, 98(1):1–15, 02 2011.

31. Frederik Questier, Raf Put, Danny Coomans, Beata Walczak, and Yvan Vander Heyden. The use of cart and multivariate regression trees for supervised and unsupervised feature selection. *Chemometrics and Intelligent Laboratory Systems*, 76(1):45–54, 2005.

32. Elsevier Inc. *Data Science Process*. Elsevier, Amsterdam, Netherlands, 2019. DOI: https://doi.org/10.1016/B978-0-12-814761-0.00002-2.

33. Leo Breiman. Statistics department university of california berkeley, ca 94720. 2001. *Random Forests*, 2001.

34. Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8):832–844, 1998.

35. Quebec Centre for Biodiversity Science. Qcbs r workshop series: Workshop 10: Advanced multivariate analyses in r, 2023. Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International Licenses all content.

36. G. De'ath. Multivariate regression trees: A new technique for modeling species-environment relationships. *Ecological Society of America*, 88:2783–2792, 2002.

37. Mark Segal and Yuanyuan Xiao. Multivariate random forests. *WIREs Data Mining and Knowledge Discovery*, 1(1):80–87, 2011.

38. Cansu Alakus, Denis Larocque, and Aurélie Labbe. Covariance regression with random forests. *BMC bioinformatics*, 24(1):258, 2023.

39. L Breiman, JH Friedman, RA Olshen, and CJ Stone. Classification and regression trees. boca raton, florida: Chapman hall/crc, 1984.

40. Hoora Moradian, Denis Larocque, and François Bellavance. L_1 l 1 splitting rules in survival forests. *Lifetime data analysis*, 23:671–691, 2017.

41. Sami Tabib and Denis Larocque. Non-parametric individual treatment effect estimation for survival data with random forests. *Bioinformatics*, 36(2):629–636, 2020.

42. Cansu Alakuş, Denis Larocque, Sébastien Jacquemont, Fanny Barlaam, Charles-Olivier Martin, Kristian Agbogba, Sarah Lippé, and Aurélie Labbe. Conditional canonical correlation estimation based on covariates with random forests. *Bioinformatics*, 37(17):2714–2721, 2021.

43. Susan Athey, Julie Tibshirani, and Stefan Wager. Generalized random forests. *Project Euclid*, 2019.

44. Benjamin Lu and Johanna Hardin. A unified framework for random forest prediction error estimation. *Journal of Machine Learning Research*, 22(8):1–41, 2021.

45. Cansu Alakus, Denis Larocque, and Aurelie Labbe. Rfpredinterval: An r package for prediction intervals with random forests and boosted forests. *arXiv preprint arXiv:2106.08217*, 2021.

46. N Azizah, LS Riza, and Y Wihardi. Implementation of random forest algorithm with parallel computing in r. In *Journal of Physics: Conference Series*, volume 1280, page 022028. IOP Publishing, 2019.

47. Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

48. Alexander März. Multi-target xgboostlss regression. *arXiv preprint arXiv:2210.06831*, 2022.

49. DSG Pollock and Emi Mise. The cholesky decomposition of a toeplitz matrix and a wiener-kolmogorov filter for seasonal adjustment. Technical report, 2020.

50. StatQuest with Josh Starmer. XGBoost Part 1 (of 4): Regression, 2019. [Online; accessed 6-March-2025].

51. XGBoosting. Xgboost advantages and disadvantages (pros vs cons), 2024. Accessed: 2025-04-11.

52. Jo Jackson. Multivariate techniques: Advantages and disadvantages. `https://classroom.synonym.com/multivariate-techniques-advantages-and-disadvantages-123456.html`, 2018. Updated September 05, 2018. Accessed December 06, 2024.

53. Alexander Von Eye and G Anne Bogat. Testing the assumption of multivariate normality. *Psychology Science*, 46:243–258, 2004.

Chapter A Appendices

A.1 Additional Data Visualisations

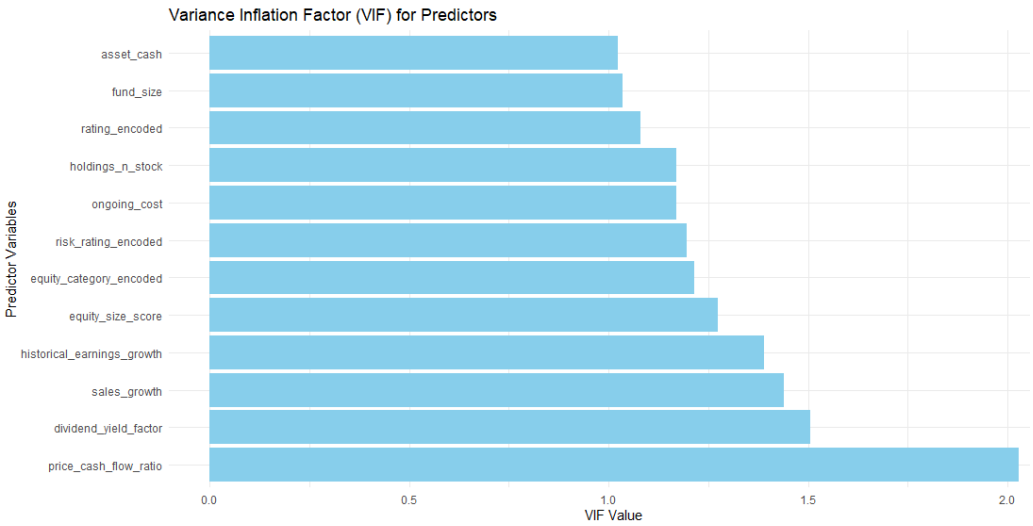

Figure A.1: VIF Values post Dataset Cleaning

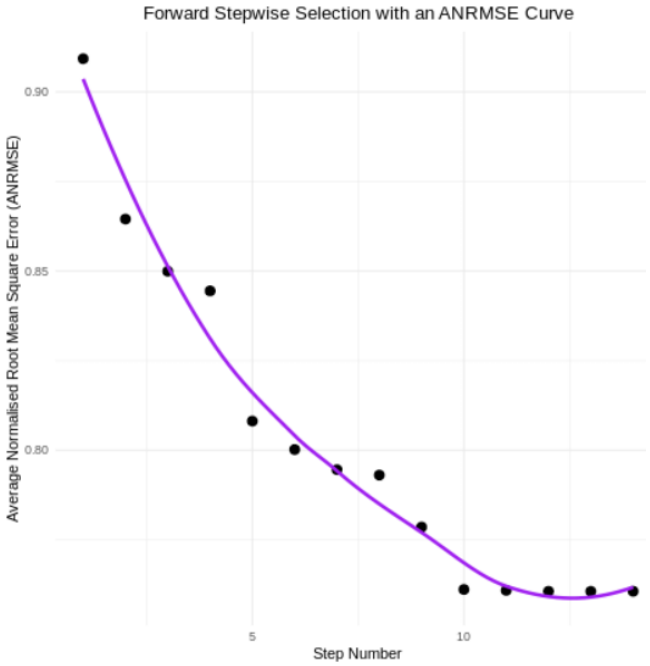

Figure A.2: Forward Stepwise Selection with an ANRMSE Curve

A.2 Extra Chapter Insights

A.2.1 Introduction

Correlation Between Math Scores and Science Scores

X_1: Hours Studied	X_2: Time Spent on Papers	Y_1: Math Scores	Y_2: Science Scores
5	2	78	80
7	3	85	79
8	4	88	88
3	1	65	70
10	5	92	74

Table A.1: Study Time vs. Exam Scores

Let each pair (x_i, y_i) represent Math (x) and Science (y) scores:

$$(x_1, y_1) = (78, 80), (x_2, y_2) = (85, 79), (x_3, y_3) = (88, 88), (x_4, y_4) = (65, 70), (x_5, y_5) = (92, 74).$$

Next, compute the means. Let $\bar{x}$ be the mean of the Math scores, and $\bar{y}$ be the mean of the Science scores:

$$\bar{x} = \frac{78 + 85 + 88 + 65 + 92}{5} = 81.6, \quad \bar{y} = \frac{80 + 79 + 88 + 70 + 74}{5} = 78.2.$$

For each pair (x_i, y_i), compute $\big(x_i - \bar{x}\big)$ and $\big(y_i - \bar{y}\big)$. For clarity, the table below shows these values:

x_i	y_i	$x_i - \bar{x}$	$y_i - \bar{y}$
78	80	$78 - 81.6 = -3.6$	$80 - 78.2 = 1.8$
85	79	$85 - 81.6 = 3.4$	$79 - 78.2 = 0.8$
88	88	$88 - 81.6 = 6.4$	$88 - 78.2 = 9.8$
65	70	$65 - 81.6 = -16.6$	$70 - 78.2 = -8.2$
92	74	$92 - 81.6 = 10.4$	$74 - 78.2 = -4.2$

Next, multiply each pair of deviations and sum them:

$$\sum_{i=1}^{5}(x_i - \bar{x})(y_i - \bar{y}) = (-3.6)(1.8) + (3.4)(0.8) + (6.4)(9.8) + (-16.6)(-8.2) + (10.4)(-4.2) = 151.4.$$

Next, compute the Sum of Squares for Each Variable:

$$\sum_{i=1}^{5}(x_i - \bar{x})^2 = (-3.6)^2 + (3.4)^2 + (6.4)^2 + (-16.6)^2 + (10.4)^2 = 449.2,$$

$$\sum_{i=1}^{5}(y_i - \bar{y})^2 = (1.8)^2 + (0.8)^2 + (9.8)^2 + (-8.2)^2 + (-4.2)^2 = 184.8.$$

Finally, apply the Pearson Correlation Formula. The Pearson correlation coefficient r is given by

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \ \sum(y_i - \bar{y})^2}} = \frac{151.4}{\sqrt{449.2 \times 184.8}} \approx 0.526.$$

Thus, the correlation between the Math and Science scores in this dataset is approximately 0.526, indicating a moderate positive relationship.

A.2.2 Exploratory Data Analysis

Mardia's Test for Multivariate Normality

Mardia's test evaluates whether a set of multivariate data follows a multivariate normal distribution by examining two key properties: multivariate skewness and multivariate kurtosis.[53]

The multivariate skewness statistic is defined as:

$$\beta_{1,d} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \left[(x_i - \bar{x})^\top S^{-1} (x_j - \bar{x}) \right]^3,$$

where x_i is the observation vector for the i-th observation, $\bar{x}$ is the sample mean vector, S is the sample covariance matrix, N is the number of observations and d is the number of response variables.[53]

Under the null hypothesis of multivariate normality, the skewness statistic $\beta_{1,d}$ is asymptotically distributed as chi-squared with $\frac{d(d+1)(d+2)}{6}$ degrees of freedom.[53] The p-value is calculated as:

$$p = P\left(\chi^2_{\text{df}} > \beta_{1,d}\right),$$

which corresponds to the upper-tail probability of the chi-squared distribution.

The multivariate kurtosis statistic is defined as:

$$\beta_{2,d} = \frac{1}{N} \sum_{i=1}^{N} \left[(x_i - \bar{x})^\top S^{-1} (x_i - \bar{x}) \right]^2,$$

For a multivariate normal distribution, the expected value of $\beta_{2,d}$ is $d(d+2)$, and the variance is approximately:

$$\text{Var}(\beta_{2,d}) = \frac{8d(d+2)}{N}.$$[53]

To assess deviation from normality, the kurtosis statistic is standardised:

$$Z = \frac{\beta_{2,d} - d(d+2)}{\sqrt{\text{Var}(\beta_{2,d})}}.$$

The p-value is then computed from the standard normal distribution as: $p = 2P(Z > |z|)$. This two-tailed test detects whether the kurtosis significantly deviates from the expected value under multivariate normality.

For the equity fund dataset, the sample covariance matrix is the response covariance matrix between ROE and sustainability score. $d = 2$ because there are 2 responses and $N = 1248$, which is the number of equity funds in the dataset.

A.2.3 Multiple Response Linear Regression

Covariance Expansion

$$\text{Cov}(\mathbf{XB}) = \text{E}[(\mathbf{XB} - \text{E}[\mathbf{XB}])(\mathbf{XB} - \text{E}[\mathbf{XB}])^\top].$$

Since expectation is linear: $\mathrm{E}[\mathbf{XB}] = \mathrm{E}[\mathbf{X}]\mathbf{B}$, the centered term becomes:

$$(\mathbf{XB} - \mathrm{E}[\mathbf{X}]\mathbf{B}) = (\mathbf{X} - \mathrm{E}[\mathbf{X}])\mathbf{B}.$$

Substitute this into the covariance definition:

$$\mathrm{Cov}(\mathbf{XB}) = \mathrm{E}\left[(\mathbf{X} - \mathrm{E}[\mathbf{X}])\mathbf{B} \cdot (\mathbf{X} - \mathrm{E}[\mathbf{X}])^\top \mathbf{B}^\top\right].$$

Factor out $\mathbf{B}$ because it is constant:

$$\mathrm{Cov}(\mathbf{XB}) = \mathbf{B}^\top \mathrm{E}\left[(\mathbf{X} - \mathrm{E}[\mathbf{X}])(\mathbf{X} - \mathrm{E}[\mathbf{X}])^\top\right]\mathbf{B}.$$

The middle term is just $\mathrm{Cov}(\mathbf{X})$, therefore: $\mathrm{Cov}(\mathbf{XB}) = \mathbf{B}^\top \mathrm{Cov}(\mathbf{X})\mathbf{B}$.

A.2.4 Ridge Regression

Closed-Form Coefficient Estimator Derivation

We begin with the ridge regression loss function:

$$\mathcal{L}(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2,$$

where $\mathbf{X} \in \mathbb{R}^{n\times p}$, $\mathbf{Y} \in \mathbb{R}^{n\times 1}$, and $\boldsymbol{\beta} \in \mathbb{R}^{p\times 1}$.

To find the closed-form solution, we differentiate $\mathcal{L}(\boldsymbol{\beta})$ with respect to $\boldsymbol{\beta}$ and set the gradient equal to zero giving:

$$\begin{aligned}\nabla_{\boldsymbol{\beta}}\mathcal{L}(\boldsymbol{\beta}) = -2\mathbf{X}^\top(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) + 2\lambda\boldsymbol{\beta} = 0,\\ \mathbf{X}^\top\mathbf{Y} = \mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} + \lambda\boldsymbol{\beta},\\ (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})\boldsymbol{\beta} = \mathbf{X}^\top\mathbf{Y}.\end{aligned}$$

Hence, the closed-form solution for ridge regression is:

$$\hat{\boldsymbol{\beta}}^{\text{Ridge}} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{Y}.$$

A.2.5 Reduced Rank Ridge Regression

Minimisation Problem Simplification

Starting with the penalised objective:

$$\|\mathbf{Y} - \mathbf{XB}\|_F^2 + \lambda\|\mathbf{B}\|_F^2.$$

Because, the above is the sum of two Frobenius norms, this can be written as a single Frobenius norm of a block matrix as follows:

$$= \left\| \begin{pmatrix} \mathbf{Y} - \mathbf{XB} \\ -\sqrt{\lambda}\mathbf{B} \end{pmatrix} \right\|_F^2.$$

Now note that:

$$\begin{pmatrix} \mathbf{Y} - \mathbf{XB} \\ -\sqrt{\lambda}\mathbf{B} \end{pmatrix} = \begin{pmatrix} \mathbf{Y} \\ \mathbf{0} \end{pmatrix} - \begin{pmatrix} \mathbf{X} \\ \sqrt{\lambda}\mathbf{I} \end{pmatrix} \mathbf{B}.$$

Therefore, the penalised problem can be written as an equivalent least squares problem on an augmented dataset:

$$\|\mathbf{Y} - \mathbf{XB}\|_F^2 + \lambda\|\mathbf{B}\|_F^2 = \left\| \begin{pmatrix} \mathbf{Y} \\ \mathbf{0} \end{pmatrix} - \begin{pmatrix} \mathbf{X} \\ \sqrt{\lambda}\mathbf{I} \end{pmatrix} \mathbf{B} \right\|_F^2 = \|\mathbf{Y}^* - \mathbf{X}^*\mathbf{B}\|_F^2.$$

Hence, the original penalised objective can be rewritten as the following unconstrained least squares objective on an augmented dataset:

$$\hat{\mathbf{B}}(\lambda, r) = \underset{\text{rank}(\mathbf{B}) \leq r}{\arg\min} \ \|\mathbf{Y}^* - \mathbf{X}^*\mathbf{B}\|_F^2.$$

Mean Centring and the Response-Covariance Matrix

The response matrix $\mathbf{Y}$ can be written as the sum of its mean-centred component and the mean matrix: $\mathbf{Y} = (\mathbf{Y} - \bar{\mathbf{Y}}) + \bar{\mathbf{Y}}$. In this decomposition, $\mathbf{Y} - \bar{\mathbf{Y}}$ is the mean-centred version of $\mathbf{Y}$, while $\bar{\mathbf{Y}}$ is a matrix where each row is the mean response vector $\bar{\mathbf{y}}$.

Substituting the decomposition $\mathbf{Y} = (\mathbf{Y} - \bar{\mathbf{Y}}) + \bar{\mathbf{Y}}$ into $\mathbf{Y}^\top\mathbf{Y}$, gives $[(\mathbf{Y} - \bar{\mathbf{Y}}) + \bar{\mathbf{Y}}]^\top[(\mathbf{Y} - \bar{\mathbf{Y}}) + \bar{\mathbf{Y}}]$. Expanding this product results in

$$\mathbf{Y}^\top\mathbf{Y} = (\mathbf{Y} - \bar{\mathbf{Y}})^\top(\mathbf{Y} - \bar{\mathbf{Y}}) + \bar{\mathbf{Y}}^\top(\mathbf{Y} - \bar{\mathbf{Y}}) + (\mathbf{Y} - \bar{\mathbf{Y}})^\top\bar{\mathbf{Y}} + \bar{\mathbf{Y}}^\top\bar{\mathbf{Y}}.$$

The two cross terms $\bar{\mathbf{Y}}^\top(\mathbf{Y} - \bar{\mathbf{Y}})$ and $(\mathbf{Y} - \bar{\mathbf{Y}})^\top\bar{\mathbf{Y}}$ are equal to zero. This vanishing comes from the fact that the mean response matrix $\bar{\mathbf{Y}}$ is constant across all rows, meaning that the sum of all deviations from the mean is equal to zero. Since:

$$\sum_{i=1}^{n}(\mathbf{y}_i - \bar{\mathbf{y}}) = 0,$$

it follows that when multiplied by $\bar{\mathbf{Y}}$, the result also vanishes:

$$\sum_{i=1}^{n}\bar{\mathbf{y}}^\top(\mathbf{y}_i - \bar{\mathbf{y}}) = \bar{\mathbf{y}}^\top\sum_{i=1}^{n}(\mathbf{y}_i - \bar{\mathbf{y}}) = 0.^{28}$$

Since this holds for all response variables, it means that $\bar{\mathbf{Y}}^\top(\mathbf{Y} - \bar{\mathbf{Y}}) = 0 \quad \text{and} \quad (\mathbf{Y} - \bar{\mathbf{Y}})^\top\bar{\mathbf{Y}} = 0$. Substituting these results back into the expansion simplifies the equation to:

$$\mathbf{Y}^\top\mathbf{Y} = (\mathbf{Y} - \bar{\mathbf{Y}})^\top(\mathbf{Y} - \bar{\mathbf{Y}}) + \bar{\mathbf{Y}}^\top\bar{\mathbf{Y}}.$$

The term $\bar{\mathbf{Y}}^\top\bar{\mathbf{Y}}$ can be further simplified. Since each row of $\bar{\mathbf{Y}}$ is equal to the mean response vector $\bar{\mathbf{y}}$, and there are n such rows, the sum simplifies as follows:

$$\bar{\mathbf{Y}}^\top\bar{\mathbf{Y}} = n\bar{\mathbf{y}}^\top\bar{\mathbf{y}}.$$

This scaling by n occurs because the same mean vector is repeated for all observations. Substituting this result into the equation gives:

$$\mathbf{Y}^\top\mathbf{Y} = (\mathbf{Y} - \bar{\mathbf{Y}})^\top(\mathbf{Y} - \bar{\mathbf{Y}}) + n\bar{\mathbf{Y}}^\top\bar{\mathbf{Y}}.^{28}$$

This equation shows that $\mathbf{Y}^\top\mathbf{Y}$ consists of the variance of the mean-centred data plus the contribution

of the mean structure. In its final form, this is:

$$\mathbf{Y}^\top \mathbf{Y} = (n-1)\mathbf{\Sigma_Y} + n\bar{\mathbf{Y}}^\top \bar{\mathbf{Y}},$$

where $\bar{\mathbf{Y}}^\top \bar{\mathbf{Y}}$ captures the mean effects of $\mathbf{Y}$.

Projection Simplification

Starting with:

$$\left(\sum_{i=1}^{\tau} d_i \mathbf{u}_i \mathbf{v}_i^\top\right)\left(\sum_{j=1}^{r} \mathbf{v}_j \mathbf{v}_j^\top\right) = \sum_{i=1}^{\tau} d_i \mathbf{u}_i \left(\mathbf{v}_i^\top \sum_{j=1}^{r} \mathbf{v}_j \mathbf{v}_j^\top\right).$$

The inner expression simplifies using orthonormality:

- If $i \leq r$:
$$\mathbf{v}_i^\top \sum_{j=1}^{r} \mathbf{v}_j \mathbf{v}_j^\top = \mathbf{v}_i^\top.$$
- If $i > r$:
$$\mathbf{v}_i^\top \sum_{j=1}^{r} \mathbf{v}_j \mathbf{v}_j^\top = \mathbf{0}^\top.$$

So the entire sum becomes:

$$\sum_{i=1}^{r} d_i \mathbf{u}_i \mathbf{v}_i^\top.$$

Singular Value Decomposition

Given the matrix $\hat{\mathbf{Y}}^*_{\mathbf{R}}$:

$$\hat{\mathbf{Y}}^*_{\mathbf{R}} = \begin{bmatrix} -5.67 & 2.96 & 4.19 & -14.29 & 12.81 & 0.74 & -0.62 \\ -1.82 & -0.91 & 2.28 & -2.73 & 3.19 & -0.23 & 0.55 \end{bmatrix}^\top.$$

We aim to compute its Singular Value Decomposition: $\hat{\mathbf{Y}}^*_{\mathbf{R}} = \mathbf{UDV}^\top$. First, compute the Gram matrix:

$$\hat{\mathbf{Y}}^{*\top}_R \hat{\mathbf{Y}}^*_R = \begin{bmatrix} 427.7629 & 96.57640 \\ 96.57640 & 27.33693 \end{bmatrix}.$$

This symmetric matrix is used to compute the eigenvalues and eigenvectors, which in turn provide the singular values. These singular values are the square roots of the eigenvalues of $\hat{\mathbf{Y}}^{*\top}_R \hat{\mathbf{Y}}^*_R$. The eigenvalues of $\hat{\mathbf{Y}}^{*\top}_R \hat{\mathbf{Y}}^*_R$ are: $\lambda_1 = 449.838519$ and $\lambda_2 = 5.261272$.

Hence:

$$\mathbf{D} = \begin{bmatrix} \sqrt{449.838519} & 0 \\ 0 & \sqrt{5.261272} \end{bmatrix} = \begin{bmatrix} 21.209397 & 0 \\ 0 & 2.293746 \end{bmatrix}.$$

The eigenvectors of $\hat{\mathbf{Y}}^{*\top}_R \hat{\mathbf{Y}}^*_R$ are the matrix $\mathbf{V}$:

$$\mathbf{V} = \begin{bmatrix} 0.9748562 & -0.2228349 \\ 0.2228349 & 0.9748562 \end{bmatrix}.$$

Now, we compute $\mathbf{U}$ using the formula: $\mathbf{U} = \hat{\mathbf{Y}}^*_{\mathbf{R}}\mathbf{V}\mathbf{S}^{-1}$. Since $\mathbf{V}$ is an orthonormal matrix, we have $\mathbf{V}^\top = \mathbf{V}^{-1}$, leading to (2dp):

$$\mathbf{U} = \begin{bmatrix} -0.28 & 0.13 & 0.22 & -0.69 & 0.62 & 0.03 & -0.02 \\ -0.22 & -0.68 & 0.56 & 0.23 & 0.11 & -0.17 & 0.29 \end{bmatrix}^\top.$$

A.2.6 Multivariate Regression with Covariance Estimation

Negative Log-Likelihood Derivation

Starting with the familiar MRLR model:

$$\mathbf{Y}_{(n\times m)} = \mathbf{X}_{(n\times p)}\mathbf{B}_{(p\times m)} + \mathbf{E}_{(n\times m)},$$

where p no longer includes the column of ones.

Each row $\mathbf{y}_i \in \mathbb{R}^{1\times m}$ of $\mathbf{Y}$ follows:

$$\mathbf{y}_i|\mathbf{x}_i \sim \mathcal{N}_m(\mathbf{x}_i\mathbf{B}, \mathbf{\Sigma_E}),$$

The probability density function for a single observation is:

$$p(\mathbf{y}_i \mid \mathbf{x}_i) = \frac{1}{(2\pi)^{m/2}|\mathbf{\Sigma_E}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{y}_i - \mathbf{x}_i\mathbf{B})^\top \mathbf{\Sigma_E}^{-1}(\mathbf{y}_i - \mathbf{x}_i\mathbf{B})\right).$$

Assuming independence across observations, the total log-likelihood becomes:

$$\log L(\mathbf{B}, \mathbf{\Sigma_E}) = \sum_{i=1}^{n} \log p(\mathbf{y}_i \mid \mathbf{x}_i)$$

$$= -\frac{nm}{2}\log(2\pi) - \frac{n}{2}\log|\mathbf{\Sigma_E}| - \frac{1}{2}\sum_{i=1}^{n}(\mathbf{y}_i - \mathbf{x}_i\mathbf{B})^\top \mathbf{\Sigma_E}^{-1}(\mathbf{y}_i - \mathbf{x}_i\mathbf{B}) \qquad (*)$$

Subbing the residual matrix, $\mathbf{E} = \mathbf{Y} - \mathbf{XB}$, into the quadratic form gives:

$$\sum_{i=1}^{n}(\mathbf{y}_i - \mathbf{x}_i\mathbf{B})^\top \mathbf{\Sigma_E}^{-1}(\mathbf{y}_i - \mathbf{x}_i\mathbf{B}) = \operatorname{tr}\left(\mathbf{E}^\top\mathbf{E}\mathbf{\Sigma_E}^{-1}\right)$$

$$= \operatorname{tr}\left((\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})\mathbf{\Omega}\right),$$

where $\mathbf{\Omega} = \mathbf{\Sigma_E}^{-1}$.

Substituting back into (*), the log-likelihood becomes:

$$\log L(\mathbf{B}, \mathbf{\Omega}) = -\frac{nm}{2}\log(2\pi) + \frac{n}{2}\log|\mathbf{\Omega}| - \frac{1}{2}\operatorname{tr}\left((\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})\mathbf{\Omega}\right).$$

Ignoring the constant term, $\frac{nm}{2}\log(2\pi)$, the negative log-likelihood (NLL) becomes:

$$\text{NLL}(\mathbf{B}, \mathbf{\Omega}) = \frac{1}{2}\operatorname{tr}\left((\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})\mathbf{\Omega}\right) - \frac{n}{2}\log|\mathbf{\Omega}|.$$

Finally, multiplying and dividing the trace term by n, and omitting the constant $\frac{1}{2}$, we obtain the simplified negative log-likelihood:

$$g(\mathbf{B}, \mathbf{\Omega}) = \operatorname{tr}\left(\frac{1}{n}(\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})\mathbf{\Omega}\right) - \log|\mathbf{\Omega}|.$$

This is the negative log-likelihood function (up to a constant) for the MRLR model.

A.2.7 Cholesky-Gaussian XGBoost

Weighted Quantile Sketches

In the approximate split-finding algorithm, as already stated, candidate split points are chosen based on percentiles of the feature values. The rank function $r_k(z)$ calculates the proportion of instances where feature k is less than z, weighted by second-order gradient statistics h.[47] The algorithm aims to find split points $s_{k1}, s_{k2}, \ldots, s_{kl}$ such that the difference between the ranks of consecutive points is small, controlled by ϵ, which is an approximation factor.[47] This process ensures that the split points are well-distributed across the range of the feature values.

The Hessian, h_i, is used as a weight for each instance or row in the dataset, meaning that data points with larger Hessians carry more influence in determining the split. This weighting reflects the importance of the instance in minimising loss. Higher-weighted instances show where the model struggles, guiding the algorithm to focus on these critical areas during split finding.

Traditional quantile sketch algorithms work for datasets where all instances have equal weights.[47] However, no existing algorithm efficiently handles weighted data. This creates challenges in approximating split points for datasets with imbalanced or sparse data distributions. Randomised sorting or heuristic approaches have been used to address this issue, but these methods lack theoretical guarantees.[47] As a result, they may fail to identify the optimal split points, leading to suboptimal model performance. To overcome this limitation, XGBoost introduces a novel weighted quantile sketch algorithm that can handle weighted data with provable theoretical guarantees.[47] This algorithm ensures accurate split findings by accounting for the distribution of weights across the dataset.

Cholesky-Decomposition Verification

We verify by computing:

$$\mathbf{L}\mathbf{L}^\top = \begin{bmatrix} 2 & 0 \\ 1 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 0 & \sqrt{2} \end{bmatrix}$$

$$= \begin{bmatrix} (2 \times 2) + (0 \times 0) & (2 \times 1) + (0 \times \sqrt{2}) \\ (1 \times 2) + (\sqrt{2} \times 0) & (1 \times 1) + (\sqrt{2} \times \sqrt{2}) \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 2 \\ 2 & 3 \end{bmatrix} = \mathbf{A}$$

As desired.