Mutual Fund and Exchange-Traded Fund Analysis

Submitted in partial fulfilment of the requirements for the course of

STAT432 BASICS OF STATISTICAL LEARNING

By:

PROFESSOR HYOEUN LEE

Submitted by:

DHRUV NARESH BORDA

(UIN-675438725, NetID-borda2@illinois.edu)

PRANNOY KATHIRESAN

(UIN-650590434, NetID-prannoy2@illinois.edu)

UNIVERISTY OF ILLINOIS URBANA-CHAMPAIGN

DECEMBER 2023

Table of Content

List of Figures

List of Tables

Abstract

Regression and classification operations on quantitative and qualitative data from Mutual Funds and Exchange Traded Funds are the focus of this study. Different exchange traded funds and mutual funds' fund returns are forecasted. Through classification, the size and investment type of different funds in mutual funds and exchange traded funds are projected. For several methods of classification and regression, the MSE and classification error are determined. On both the mutual funds and exchange traded funds datasets, several procedures, including data cleaning, data visualization, data analysis, correlation analysis, regression, and classification, are carried out.

1. Introduction

1.1 Motivation for Analysis

Investments, which include giving up current assets in exchange for potential future returns, are a key tenet of the world economy. Investments come in a wide variety of forms, and they have been around since the sixteenth century. Following hundreds of years, an investment strategy—the notion of combining resources from multiple sources with the aim of jointly investing—emerged in 18th-century France.

Mutual Funds, which are actively managed by a fund manager and employ the assets within the fund to generate returns for the fund owners, are how these investment funds originally appeared in the United States. There is a cost associated with joining these funds because they require administration in order to provide returns. Exchange Traded Funds, or ETFs, are the fund investment instrument that later evolved to compete with Mutual Funds. ETFs, in contrast to mutual funds, are passively managed because they follow a particular market index. Investors that use ETFs expect to avoid wasting the resources needed to actively manage a fund by capturing the returns of the market as a whole.

Investors have disputed whether the active management approach of mutual funds can outperform the market and the passive investment approach of ETFs, but the answer is still up for debate. For each type of fund, we determined in our experiment which characteristics of each fund are most strongly associated to fund returns. Then, using the most connected characteristics for each fund, we sought to forecast whether or not future returns would be above average using the funds that had above average returns from our dataset. In the end, we discovered that, based on the fund descriptors that investors would have access to at the time of investing, ETFs have exceptionally high predictive accuracy while Mutual Funds have low predictive accuracy when attempting to determine if a Mutual Fund will have above average returns.

In addition to this predictive analysis, we looked at each fund's investing strategy and how it affected return on investment. Growth ETFs outperform growth mutual funds by a wide margin. Growth funds generate extraordinarily large returns in relatively short periods of time. Once again, we discovered that value ETFs outperformed value mutual funds in terms of total returns, with value funds exhibiting strong total returns across very long investment horizons.

1.2 Research Questions

The following set of questions influenced our flow of project and selection of variables.

1. What set of factors is most crucial to an MF and ETF's success?
2. Based on the observed characteristics, which type of fund has a greater overall success rate?
3. Is it possible to precisely predict a fund's success??

1.3 Description of Dataset

This dataset contains all US mutual funds together with their historical values and financial data gathered from Yahoo Finance. The financial figures as of November 2021 are referenced in this edition.

2,310 ETFs and 23,783 mutual funds are included in the dataset with following information:

1. General fund aspects (i.e., total net assets, fund family, inception date, etc.)
2. Portfolio indicators (i.e., cash, stocks, bonds, sectors, etc.)
3. Historical yearly and Quarterly returns (i.e., year-to-date, 1-year, 3-years, etc.)
4. Financial ratios (i.e., price/earnings, Treynor and Sharpe ratios, alpha, and beta)

The website https://finance.yahoo.com, which is open to the public, has had data scraped from it. Datasets enable numerous comparisons of portfolio choices made by investment managers in mutual funds and portfolio constraints placed on the indexes in exchange traded funds (ETFs). The 2017 ETF hype served as an inspiration, persuading several investors to invest in Exchange Traded Funds rather than Mutual Funds.

1.3.1 Data Limitations

One of the biggest challenges we faced during the analysis was the preponderance of "NAN" entries or empty values in columns.

Another data constraint we encountered while working on this project was the fact that the Mutual funds dataset was larger than the ETF dataset. This limited the amount of mutual fund information we could review before comparing them.

1.4 Data preprocessing

Based on the significance of the empty and unused columns relative to the response variables, we have eliminated them entirely. (i.e., “currency,” "years up,", "years down" "quarters up" etc.)

Since half of the columns have either missing data or data that is unrelated to our research, we have also deleted some of the extraneous category data. The data frame was finally divided into more manageable components for efficient model creation (E.g.: assets, fund return, ratios, sectors). We estimated the mean values for the entire column and utilized those to replace the missing values for certain of the columns, such as fund_treynor_ratio, that had missing values that could not be ignored and were essential to our analysis.

1.5 Data Analysis

1.5.1 Growth of MF and ETF over the years

The growth of ETF has been dramatically greater than that of MF over the past two decades, despite the fact that MF's quantity is 20 times more, as shown in Figure 1 Comparison of ETFs and MFs by Inception Year. As a result, ETF presents a significant alternative trade-off for MF and is thus important for investment manager research.

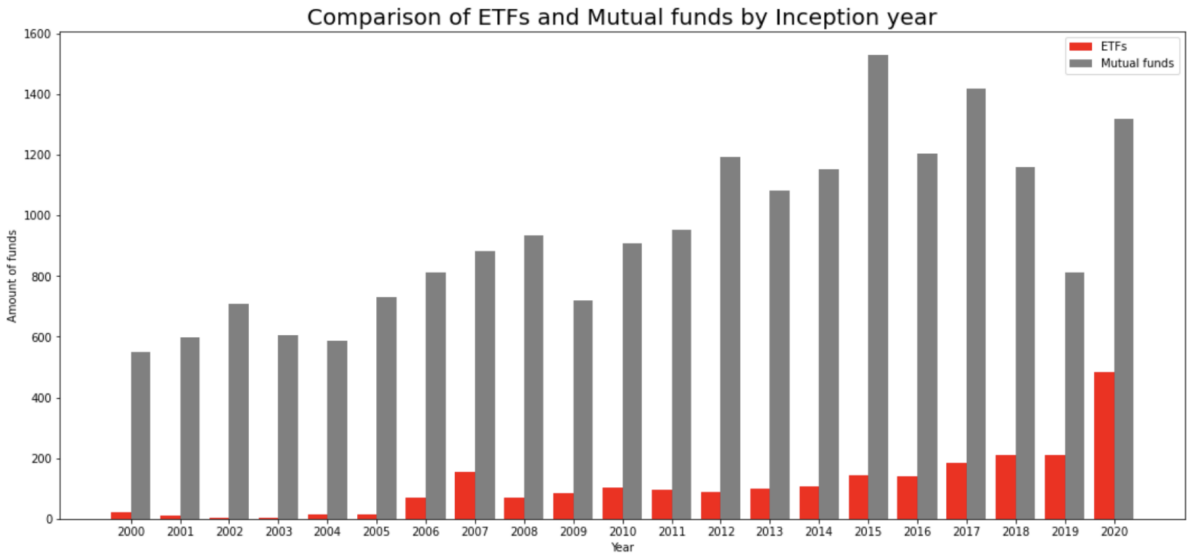

Figure 1 Comparison of ETFs and MFs by Inception Year

1.5.2 Fund Size

The fund size indicates how much capital the fund has committed. There are three types of fund sizes for various mutual funds and ETFs: large, medium, and small as depicted in Figure 2 Pie chart for ETF and MF fund size.

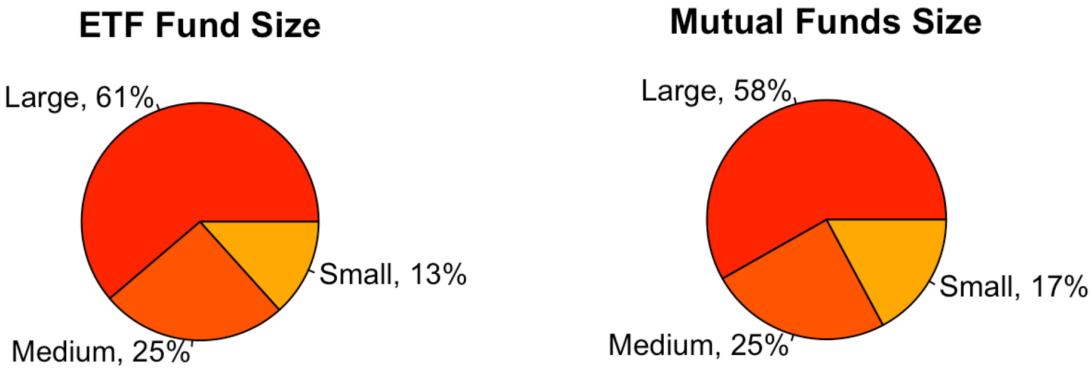

Figure 2 Pie chart for ETF and MF fund size

1.5.3 Investment Type

As showcased in Figure 3 Pie Chart for ETF and MF Investment Type the three investing classes for mutual funds and ETFs are blend, growth, and value. The stock types that a fund chooses to invest in are indicated by the investment type.

1. A growth fund is a diversified stock portfolio with a focus on capital appreciation and minimal to no dividend payments.
2. A value fund looks to invest in stocks that are thought to be inexpensive based on fundamental criteria.
3. A blend fund is a kind of equity mutual fund that holds both growth and value companies in varying proportions. Three funds give investors the chance to diversify their holdings among these well-liked investment types in a single portfolio.

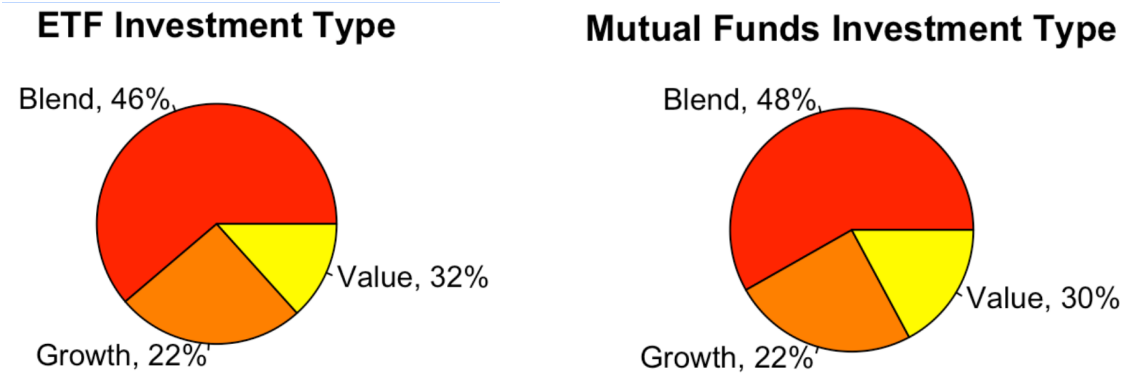

Figure 3 Pie Chart for ETF and MF Investment Type

1.5.4 Asset Distribution

The asset distribution shows the distinct types of investments that a fund makes. Stocks and bonds are the only investment options available with ETFs. While mutual funds invest in a variety of different assets, including cash, stocks, and bonds as shown in Figure 4 Pie Chart for ETF and MF Assets.

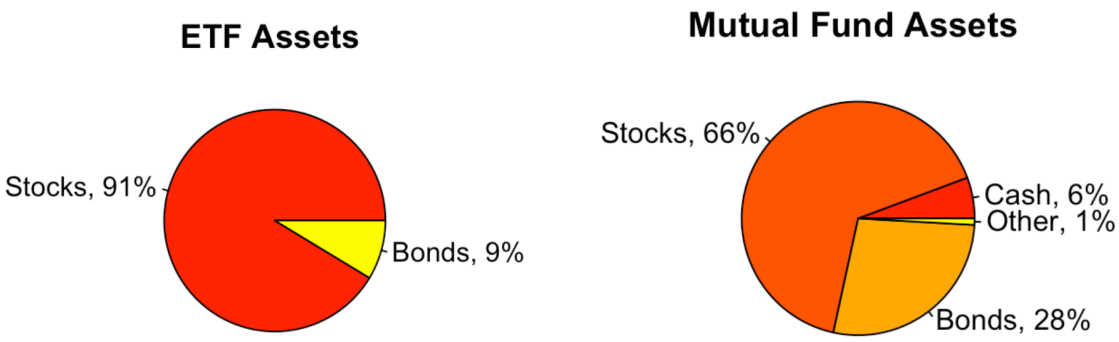

Figure 4 Pie Chart for ETF and MF Assets

1.5.5 Sector Distribution

Mutual funds and ETFs both invest in a variety of industries, including those related to energy, technology, healthcare, and more.

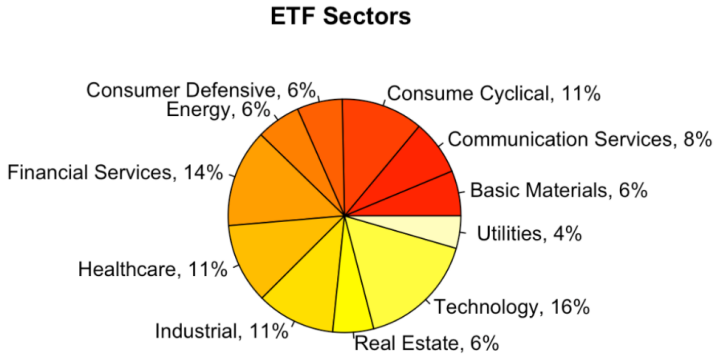

Figure 5 Pie Chart for ETF Investment Sectors

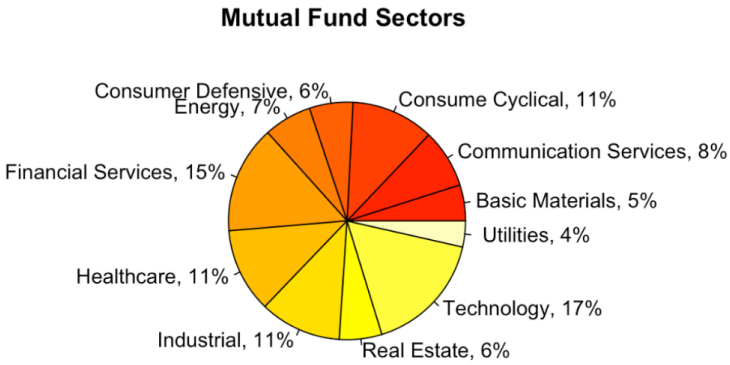

Figure 6 Pie Chart for MF Investment Sectors

1.5.6 Mutual Fund Risk Rating

Risk rating, which rates a fund's level of risk from 1 to 5 (Figure 7 Pie chart for Mutual Fund Morningstar Risk Rating), is a statistic specific to mutual funds. As shown in Figure 8 Categorical boxplot for MF Morningstar Risk Rating (1-5), as the riskiness of the fund rises, the whiskers lengthen. The riskier investments are also very volatile.

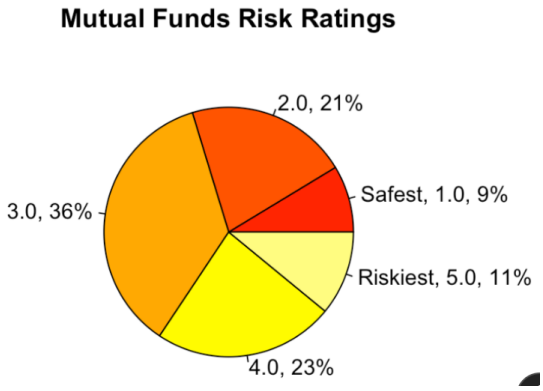

Figure 7 Pie chart for Mutual Fund Morningstar Risk Rating

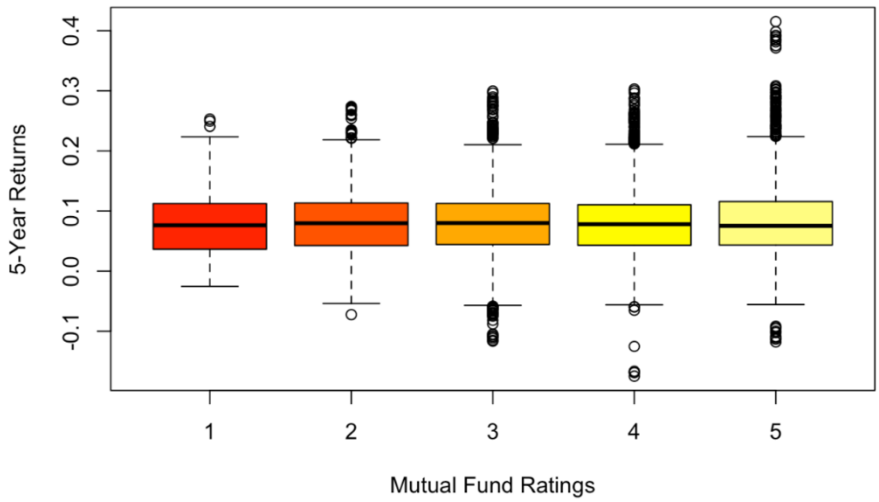

Figure 8 Categorical boxplot for MF Morningstar Risk Rating (1-5)

2. Statistical Methods

2.1 Correlation Analysis & Significance Testing

Following a test for significance between the response variable and various predictor factors, we decided to use 54 predictor variables for the analysis of exchange-traded funds (ETFs) and 149 predictor variables for the study of mutual funds.

2.1.1 ETF Correlation Plot

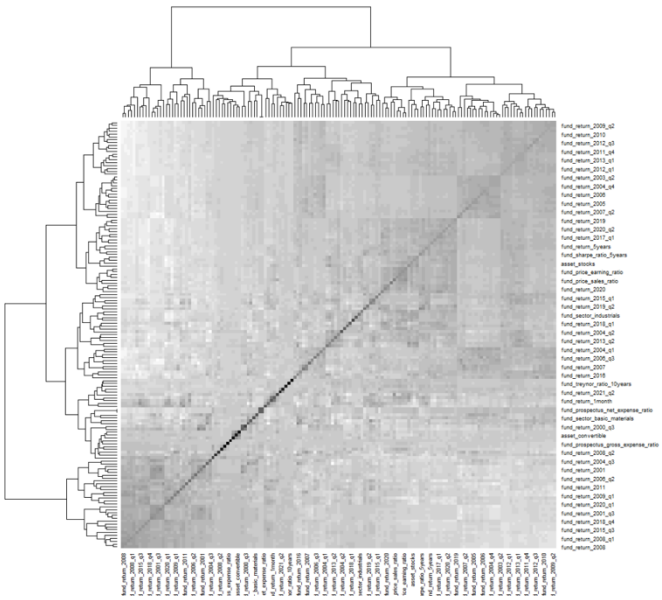

Figure 9 ETF Correlation Plot

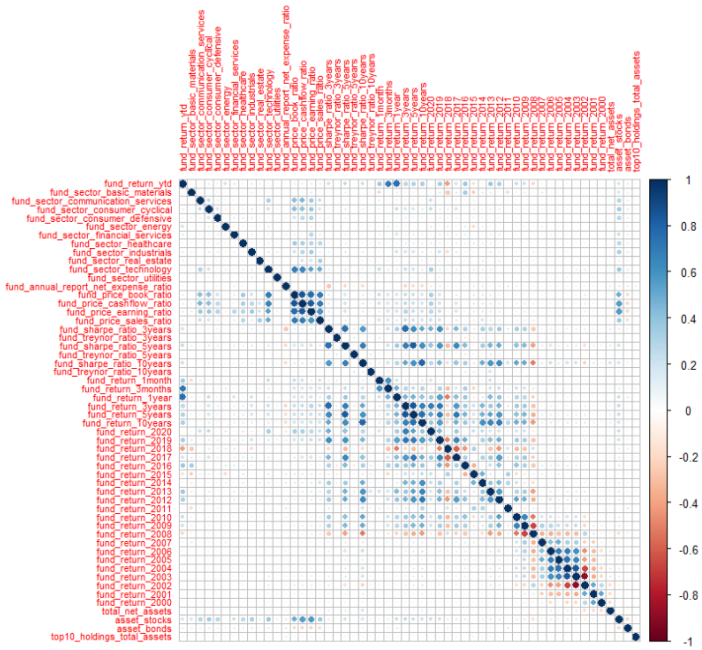

Figure 10 ETF Correlation Plot

The ETF Correlation plot consists of 54 correlated variables such as "asset_bonds", "fund_sector_energy", "fund_sector_industrials", "fund_return_1year", "fund_sector_real_estate", and so on. As observed from the Figure 9 ETF Correlation Plot, we have both the positive and negative correlations among the various variables of the ETF correlation plot.

2.1.2 Mutual Fund Correlation Plot

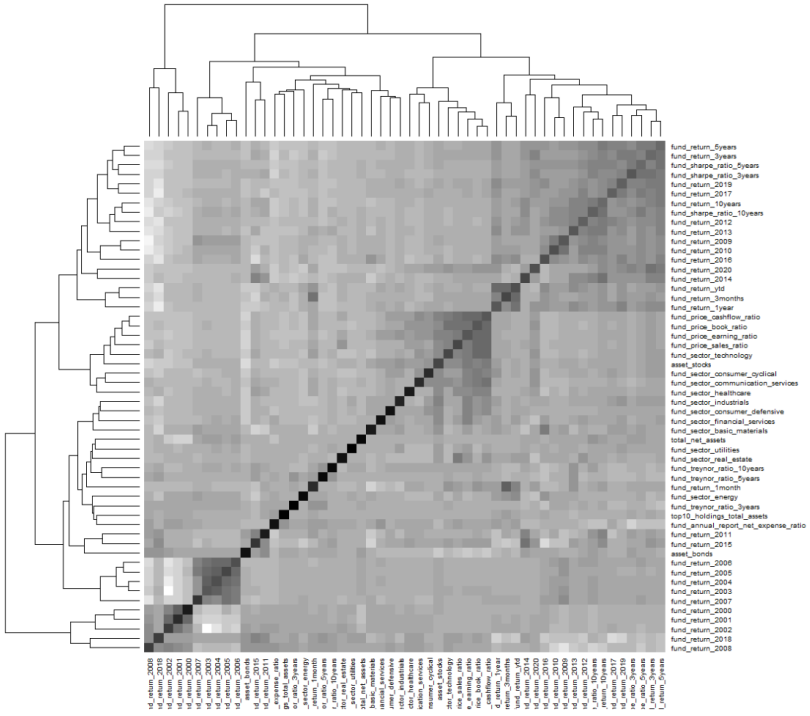

Figure 11 MF Correlation Plot

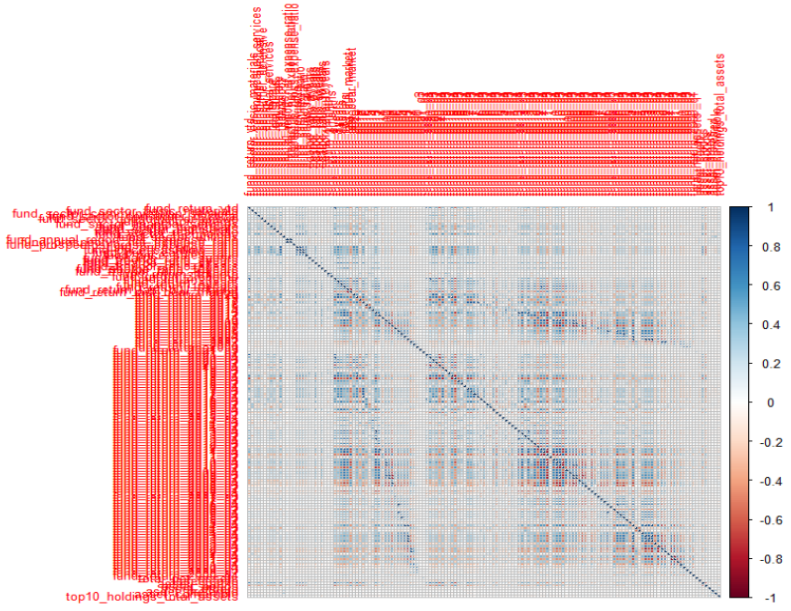

Figure 12 MF Correlation Plot

The Mutual fund correlation plot consists of 149 variables which are significant for determining the final output. Some of the significant variables are "fund_sharpe_ratio_10years", "fund_treynor_ratio_3years", "fund_return_2019", "total_net_assets", "asset_cash", "asset_bonds", "fund_sector_energy", and so on.

2.2 Regression

We used a variety of statistical techniques to forecast our quantitative variable, "fund_return_ytd" including the following models:

1. Linear regression
2. Best subset regression
3. KNN regression
4. Decision tree regression
    a. Cross validation pruning
    b. Random forest
5. Ridge regression
6. Lasso regression

Response variable: “fund_return_ytd”

Predictors variables:

1. **Qualitative variables**: “investment type” and “Size type”
2. **Quantitative variables:** "fund_sector_consumer_cyclical", "fund_price_cashflow_ratio", "fund_treynor_ratio_5years", "fund_sector_technology", "fund_price_sales_ratio" and so on.

The highly significant factors present in the datasets were used in regression analysis on the “fund_return_ytd”. Section 3 lists the MSE results for several regressions.

With its low MSE value (ETF: 0.00380 MF: 0.00020) and high R^2 value (ETF: 82.17%, MF: 96.6%), the Random Forest Regressor under decision tree regression performed well. Following importance chart for ETF and MF, as depicted in Figure 13 Important Predictors for ETF from Random Forest and Figure 14 Important Predictors for MF from Random Forest, represents the top predictor variables.

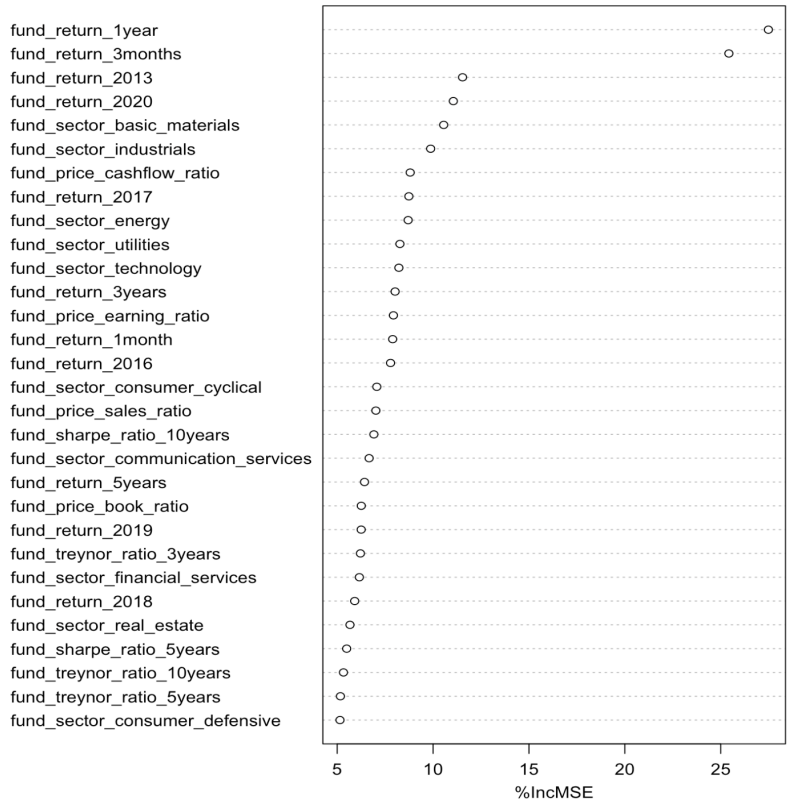

Figure 13 Important Predictors for ETF from Random Forest

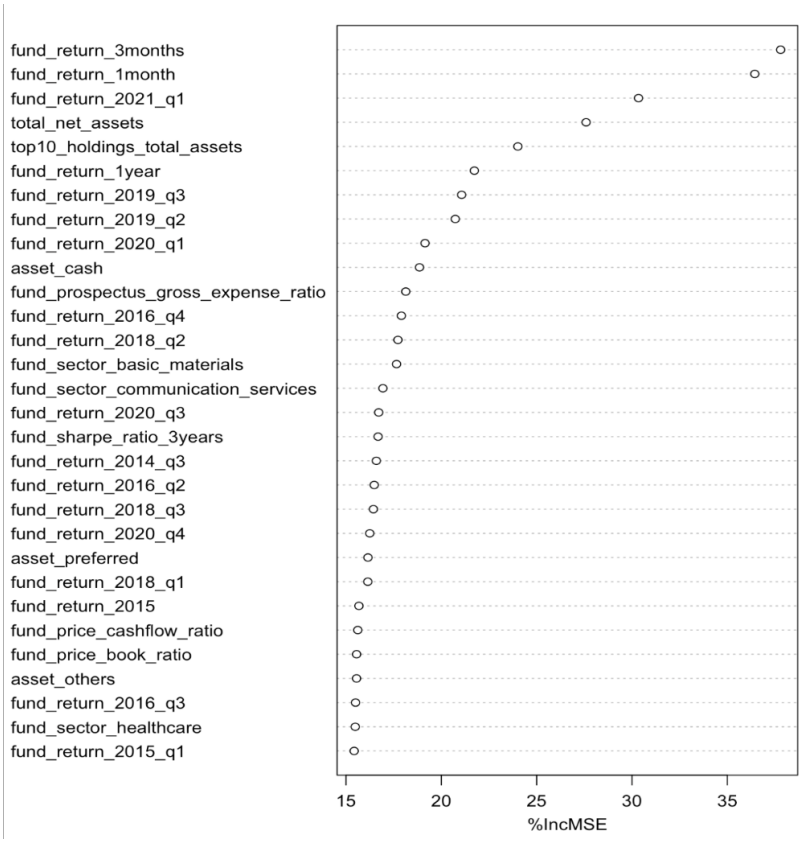

Figure 14 Important Predictors for MF from Random Forest

Predicted Value and Actual Value for ETF and MF can be observed from Figure 15 ETF Predicted Value vs Actual Value for Chosen Model and Figure 16 MF Predicted Value vs Actual Value for Chosen Model.

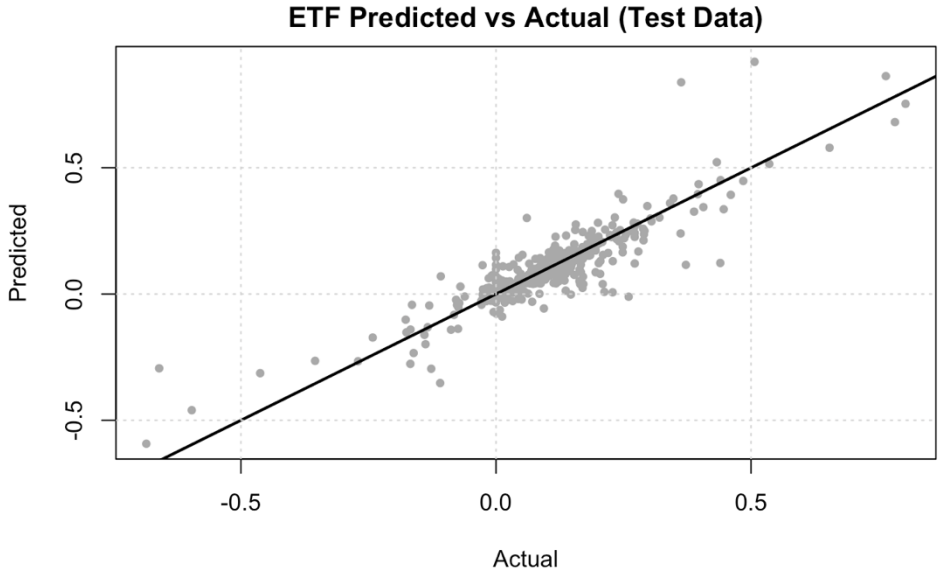

Figure 15 ETF Predicted Value vs Actual Value for Chosen Model

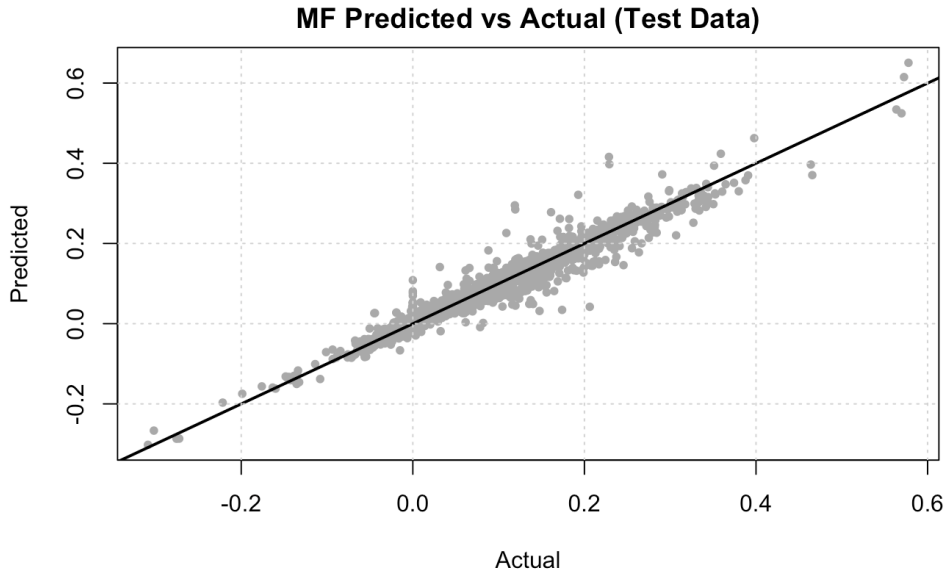

Figure 16 MF Predicted Value vs Actual Value for Chosen Model

2.3 Classification

We performed logistic regression and Linear discriminant analysis on the "fund_return_ytd", investment type and size type for both Mutual Funds and ETF dataset.

Response variables:

1. **Quantitative variables:** “fund_return_ytd”
2. **Qualitative variables**: “investment_type”, “size_type”

Predictors variables: "Fund_return_2018", “fund_price_earning_ratio", "top10_holdings_total_assets", "fund_sector_communication_services", "fund_sector_consumer_defensive", and so on.

Confusion Matrix for “fund_return_ytd” as response variable and Logistic Regression as chosen model is depicted in Table 1 Confusion Matrix for fund_return_ytd as Predictor Variable.

Table 1 Confusion Matrix for fund_return_ytd as Predictor Variable

ETF			MF		
Prediction	Reference		Prediction	Reference	
	High	low		High	Low
High	204	35	High	2297	67
low	17	206	Low	72	2320

Confusion Matrix for “investment_type” as response variable and LDA as chosen model is depicted in Table 2 Confusion Matrix for investment_type as Predictor Variable.

Table 2 Confusion Matrix for investment_type as Predictor Variable

ETF				MF			
Prediction	Reference			Prediction	Reference		
	Large	Medium	Small		Large	Medium	Small
Large	148	16	41	Large	1913	249	318
Medium	10	47	1	Medium	33	677	3
Small	11	3	54	Small	190	33	968

Confusion Matrix for “size_type” as response variable and LDA as chosen model is depicted in Table 3 Confusion Matrix for size_type as Predictor Variable

Table 3 Confusion Matrix for size_type as Predictor Variable

ETF				MF			
Prediction	Reference			Prediction	Reference		
	Large	Medium	Small		Large	Medium	Small
Large	193	37	19	Large	2404	195	43
Medium	16	30	14	Medium	196	725	172
Small	1	11	11	Small	32	100	496

Both for the MF and ETF response variables, the classification test error is anticipated, and it is tabulated in Section 3.

3. Results and Discussion

We began our analysis by conducting exploratory research. To find any notable similarities or differences, we compared ETFs to mutual funds when completing this analysis.

Visual comparisons between the various groups of variables were done first. Therefore, we separated the entire dataset into distinct data frames and carried on with our strategy. We identified the significant variables that will affect our response variables through comparisons and analysis.

For both the ETF and mutual fund datasets, we ran regression and classification tests. We then projected which test could explain the results better and identified which test is superior for making future predictions on both the ETF and Mutual fund datasets. The outcomes are displayed below.

Methods		**MSE**	
		ETF	**MF**
Linear Regression		0.00450	0.00025
Best Subsets Regression		0.00470	0.00350
KNN Regression		0.02330	0.00550
Decision Regression Tree	CV.pruning	0.00880	0.00200
	Random Forest	**0.00380**	**0.00020**
	Boosting	0.00500	0.00015
Ridge Regression		0.00420	0.00030
Lasso Regression		0.00460	0.00025

Figure 17 Metrics for Regression

The suitable model for classification is Logistic Regressor and LDA, depending on the predictor variable.

Response Variables	Methods	Classification Error (%)	
		ETF	MF
fund_return_ytd	Logistic Regression	11.260	2.020
investment_type	Linear Discriminant Analysis	22.300	22.400
size_type	Linear Discriminant Analysis	26.400	21.000

Figure 18 Metrics for Classification

As shown in For the first 3 years a fund stake is held, MFs have higher returns than ETFs. This is so that the mutual fund may make decisions more quickly and nimbly, resulting in returns that are more beneficial in the near term thanks to the active management style of mutual funds. After keeping the fund investment for 5 years, ETF returns begin to outpace mutual fund returns, From 5 to 10 years, ETF returns are bigger than those of mutual funds.

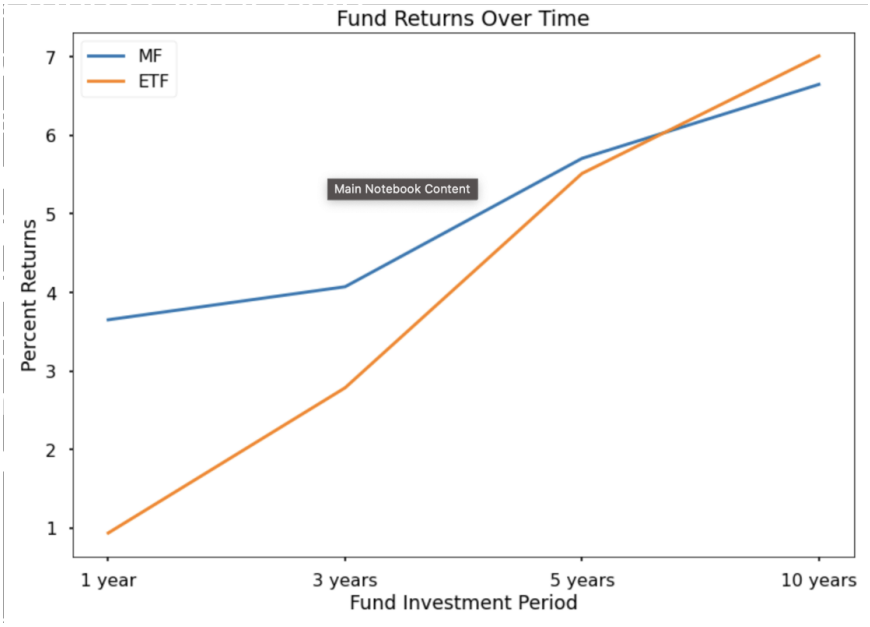

Figure 19 Fund Returns Over Time

We concluded that ETFs are more easily predictable than Mutual Funds after doing our analysis of the fund returns and predictor variables. The findings indicate that while our predictor factors for ETFs are all significant in forecasting a fund's success, they are not significant enough for mutual funds. These findings suggest that there are methods for calculating the predictability of ETFs, which may be a more reliable investment than mutual funds.

Appendix

The code is accessible as followed:

--- title: "Stat 432 FA22 Project" author: 'Name: Dhruv Borda, Prannoy Kathiresan' netID: 'borda2@illinois.edu, prannoy2@illinois.edu' output: html_document: df_print: paged pdf_document: default ---
{r setup, include=FALSE} knitr::opts_chunk$set(echo = TRUE) rm(list=ls())

3.1.1 Loading Packages

{r} library(dplyr) library(tidyr) library(Metrics) library(caret) library(leaps) library(tree) library(boot) library(randomForest) library(gbm) library(glmnet) library(MASS) library(stringr)

3.1.2 Loading Raw Data-sets

{r} mf = read.csv("~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo Finance/data/MutualFunds.csv" , header = TRUE, na = c("N/A", ""), sep = ",") etf = read.csv("~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo Finance/data/ETFs.csv" , header = TRUE, na = c("N/A", ""), sep = ",")

4. Data Cleaning

{r} # Drop all the null columns etf = etf[,colSums(is.na(etf))<nrow(etf)] mf = mf[,colSums(is.na(mf))<nrow(mf)] # Drop not required variables drop.variables = c("quarters_up", "quarters_down", "top10_holdings", "years_up", "years_down", "currency") mf = mf[,!(names(mf) %in% drop.variables)] etf = etf[,!(names(etf) %in% drop.variables)]

{r} # Replacing Treynor Ratio NaN
mf$fund_treynor_ratio_10years[is.na(mf$fund_treynor_ratio_10years)] = mean(mf$fund_treynor_ratio_10years, na.rm = TRUE)
mf$fund_treynor_ratio_5years[is.na(mf$fund_treynor_ratio_5years)] = mean(mf$fund_treynor_ratio_5years, na.rm = TRUE)
mf$fund_treynor_ratio_3years[is.na(mf$fund_treynor_ratio_3years)] = mean(mf$fund_treynor_ratio_3years, na.rm = TRUE)
etf$fund_treynor_ratio_10years[is.na(etf$fund_treynor_ratio_10years)] = mean(etf$fund_treynor_ratio_10years, na.rm = TRUE)

etf$fund_treynor_ratio_5years[is.na(etf$fund_treynor_ratio_5years)] = mean(etf$fund_treynor_ratio_5years, na.rm = TRUE) etf$fund_treynor_ratio_3years[is.na(etf$fund_treynor_ratio_3years)] = mean(etf$fund_treynor_ratio_3years, na.rm = TRUE)

4.1.1 Cleaning by Categorical Data

{r} mf.categoryData = mf %>% dplyr::select(contains("category_")) etf.categoryData = etf %>% dplyr::select(contains("category_")) mf = mf %>% dplyr::select(-contains("category_")) etf = etf %>% dplyr::select(-contains("category_")) mf.categoryData = mf.categoryData %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.categoryData = etf.categoryData %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol) mf = mf %>% mutate(category_return_ytd = mf.categoryData$category_return_ytd) mf = mf %>% mutate(category_return_3years = mf.categoryData$category_return_3years) mf = mf %>% mutate(category_return_5years = mf.categoryData$category_return_5years) mf = mf %>% mutate(category_return_10years = mf.categoryData$category_return_10years) etf = etf %>% mutate(category_return_ytd = etf.categoryData$category_return_ytd) etf = etf %>% mutate(category_return_3years = etf.categoryData$category_return_3years) etf = etf %>% mutate(category_return_5years = etf.categoryData$category_return_5years) etf = etf %>% mutate(category_return_10years = etf.categoryData$category_return_10years)

4.1.2 Cleaning by Fund Return

{r} # Cleaning by Fund Return mf.fundReturn = mf %>% dplyr::select(contains("fund_return_")) etf.fundReturn = etf %>% dplyr::select(contains("fund_return_")) mf = mf %>% dplyr::select(-contains("fund_return_")) etf = etf %>% dplyr::select(-contains("fund_return_")) mf.fundReturn = mf.fundReturn %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.fundReturn = etf.fundReturn %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol) mf = mf %>% mutate(fund_return_ytd = mf.fundReturn$fund_return_ytd) etf = etf %>% mutate(fund_return_ytd = etf.fundReturn$fund_return_ytd) mf.fundReturn = mf.fundReturn %>% dplyr::select(-contains("fund_return_ytd")) etf.fundReturn = etf.fundReturn %>% dplyr::select(-contains("fund_return_ytd")) # Writing to file write.csv(mf.fundReturn,"~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo

Finance/data/ProcessedData/mf_returns.csv", row.names = FALSE) write.csv(etf.fundReturn,"~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo Finance/data/ProcessedData/etf_returns.csv", row.names = FALSE)

4.1.3 Cleaning by Ratios

{r} # Cleaning by Ratios mf.ratios = mf %>% dplyr::select(contains("_ratio")) etf.ratios = etf %>% dplyr::select(contains("_ratio")) mf = mf %>% dplyr::select(-contains("_ratio")) etf = etf %>% dplyr::select(-contains("_ratio")) mf.ratios = mf.ratios %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.ratios = etf.ratios %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol) # Writing to file write.csv(mf.ratios,"~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo Finance/data/ProcessedData/mf_ratios.csv", row.names = FALSE) write.csv(etf.ratios,"~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo Finance/data/ProcessedData/etf_ratios.csv", row.names = FALSE)

4.1.4 Cleaning by Sectors

{r} # Cleaning by Sectors mf.sectors = mf %>% dplyr::select(contains("sector")) etf.sectors = etf %>% dplyr::select(contains("sector")) mf = mf %>% dplyr::select(-contains("sector")) etf = etf %>% dplyr::select(-contains("sector")) mf.sectors = mf.sectors %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.sectors = etf.sectors %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol) # Writing to file write.csv(mf.sectors,"~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo Finance/data/ProcessedData/mf_sectors.csv", row.names = FALSE) write.csv(etf.sectors,"~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo Finance/data/ProcessedData/etf_sectors.csv", row.names = FALSE)

4.1.5 Cleaning by Assets

{r} # Cleaning by Assets mf.assets = mf %>% dplyr::select(contains("asset")) etf.assets = etf %>% dplyr::select(contains("asset")) mf = mf %>% dplyr::select(-contains("asset")) etf = etf %>% dplyr::select(-contains("asset")) mf.assets = mf.assets %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.assets = etf.assets %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol) # Writing to file write.csv(mf.assets,"~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds

dataset from Yahoo Finance/data/ProcessedData/mf_assets.csv", row.names = FALSE) write.csv(etf.assets,"~/UIUC/STAT432 Basics of Statistical Learning/Project/US Funds dataset from Yahoo Finance/data/ProcessedData/etf_assets.csv", row.names = FALSE)

4.1.6 Cleaning by SD, Beta and Credit

{r} # Cleaning by standard deviation mf.sd = mf %>% dplyr::select(contains("standard_deviation")) etf.sd = etf %>% dplyr::select(contains("standard_deviation")) mf = mf %>% dplyr::select(-contains("standard_deviation")) etf = etf %>% dplyr::select(-contains("standard_deviation")) mf.sd = mf.sd %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.sd = etf.sd %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol) # Cleaning by Beta mf.beta = mf %>% dplyr::select(contains("beta")) etf.beta = etf %>% dplyr::select(contains("beta")) mf = mf %>% dplyr::select(-contains("beta")) etf = etf %>% dplyr::select(-contains("beta")) mf.beta = mf.beta %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.beta = etf.beta %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol) # Cleaning by Credit mf.credit = mf %>% dplyr::select(contains("credit")) etf.credit = etf %>% dplyr::select(contains("credit")) mf = mf %>% dplyr::select(-contains("credit")) etf = etf %>% dplyr::select(-contains("credit")) mf.credit = mf.credit %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.credit = etf.credit %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol) # Cleaning by R squared mf.rsquare = mf %>% dplyr::select(contains("squared")) etf.rsquare = etf %>% dplyr::select(contains("squared")) mf = mf %>% dplyr::select(-contains("squared")) etf = etf %>% dplyr::select(-contains("squared")) mf.rsquare = mf.rsquare %>% mutate(fund_symbol = mf$fund_symbol) %>% relocate(fund_symbol) etf.rsquare = etf.rsquare %>% mutate(fund_symbol = etf$fund_symbol) %>% relocate(fund_symbol)

5. Data Visualization

5.1.1 Growth of ETF and MF

{r} # Seperating Inception Year etf.year = format(as.POSIXct(etf$inception_date, format = "%m/%d/%Y"), format="%Y") etf.year = as.data.frame(etf.year) etf.year = etf.year %>% mutate(etf.year = str_replace_all(etf.year, "00", "20")) etf.year = as.data.frame(as.numeric(etf.year$etf.year)) names(etf.year)[names(etf.year) ==

"as.numeric(etf.year$etf.year)"] = 'Year.ETF' mf.year = as.Date(mf$inception_date) mf.year = as.data.frame(as.numeric(format(mf.year, "%Y"))) names(mf.year)[names(mf.year) == "as.numeric(format(mf.year, \"%Y\"))"] = 'Year.MF' # Creating main inception year dataframe

5.1.2 Fund Size

{r} labels.etf.fund = c("Large", "Medium", "Small") pie(table(etf$size_type), labels = paste(labels.etf.fund,", ", round(prop.table(table(etf$size_type))*100), "%", sep = ""), col = heat.colors(5), main = "ETF Fund Size") labels.mf.fund = c("Large", "Medium", "Small") pie(table(mf$size_type), labels = paste(labels.mf.fund,", ", round(prop.table(table(mf$size_type))*100), "%", sep = ""), col = heat.colors(5), main = "Mutual Funds Size")

5.1.3 Investment Type

{r} labels.etf.investment = c("Blend", "Growth", "Value") pie(table(etf$size_type), labels = paste(labels.etf.investment,", ", round(prop.table(table(etf$investment_type))*100), "%", sep = ""), col = heat.colors(4), main = "ETF Investment Type") labels.mf.investment = c("Blend", "Growth", "Value") pie(table(mf$size_type), labels = paste(labels.mf.investment, ", ", round(prop.table(table(mf$investment_type))*100), "%", sep = ""), col = heat.colors(4), main = "Mutual Funds Investment Type")

5.1.4 Asset Distributions

{r} etf.assets[is.na(etf.assets)] = 0 category.etf.asset = c(sum(etf.assets$asset_stocks), sum(etf.assets$asset_bonds)) labels.etf.asset = c("Stocks", "Bonds") pie(category.etf.asset, labels = paste(labels.etf.asset,", ", round(prop.table(category.etf.asset)*100), "%", sep = ""), col = heat.colors(2), main = "ETF Assets ") mf.assets[is.na(mf.assets)] = 0 category.mf.asset = c(sum(mf.assets$asset_cash), sum(mf.assets$asset_stocks), sum(mf.assets$asset_bonds), sum(mf.assets$asset_others)) labels.mf.asset = c("Cash", "Stocks", "Bonds", "Other") pie(category.mf.asset, labels = paste(labels.mf.asset,", ",round(prop.table(category.mf.asset)*100), "%", sep = ""), col = heat.colors(5), main = "Mutual Fund Assets ")

5.1.5 Sector Distribution

{r} etf.sectors[is.na(etf.sectors)] = 0 category.etf.sector =
c(sum(etf.sectors$fund_sector_basic_materials),
sum(etf.sectors$fund_sector_communication_services),
sum(etf.sectors$fund_sector_consumer_cyclical),
sum(etf.sectors$fund_sector_consumer_defensive), sum(etf.sectors$fund_sector_energy),
sum(etf.sectors$fund_sector_financial_services), sum(etf.sectors$fund_sector_healthcare),
sum(etf.sectors$fund_sector_industrials), sum(etf.sectors$fund_sector_real_estate),
sum(etf.sectors$fund_sector_technology), sum(etf.sectors$fund_sector_utilities))
labels.etf.sector = c("Basic Materials", "Communication Services", "Consume Cyclical",
"Consumer Defensive", "Energy", "Financial Services","Healthcare", "Industrial", "Real
Estate", "Technology"," Utilities") pie(category.etf.sector, labels = paste(labels.etf.sector,", ",
round(prop.table(category.etf.sector)*100), "%", sep = ""), col = heat.colors(11), main =
"ETF Sectors ") mf.sectors[is.na(mf.sectors)] = 0 category.mf.sectors =
c(sum(mf.sectors$fund_sector_basic_materials),
sum(mf.sectors$fund_sector_communication_services),
sum(mf.sectors$fund_sector_consumer_cyclical),
sum(mf.sectors$fund_sector_consumer_defensive), sum(mf.sectors$fund_sector_energy),
sum(mf.sectors$fund_sector_financial_services), sum(mf.sectors$fund_sector_healthcare),
sum(mf.sectors$fund_sector_industrials), sum(mf.sectors$fund_sector_real_estate),
sum(mf.sectors$fund_sector_technology), sum(mf.sectors$fund_sector_utilities))
labels.mf.sectors = c("Basic Materials", "Communication Services", "Consume Cyclical",
"Consumer Defensive", "Energy", "Financial Services","Healthcare", "Industrial", "Real
Estate", "Technology"," Utilities") pie(category.mf.sectors, labels = paste(labels.mf.sectors,",
",round(prop.table(category.mf.sectors)*100), "%", sep = ""), col = heat.colors(11), main =
"Mutual Fund Sectors ")

5.1.6 Mutual Fund Risk Rating

{r} labels.mf.risk = c("Safest, 1.0", "2.0", "3.0", "4.0", "Riskiest, 5.0")
pie(table(mf$morningstar_risk_rating), labels = paste(labels.mf.risk, ", ", round(
prop.table(table(mf$morningstar_risk_rating))*100), "%", sep = ""), col = heat.colors(5),
main = "Mutual Funds Risk Ratings") boxplot(mf.fundReturn$fund_return_5years ~

mf$morningstar_risk_rating, col = heat.colors(5), xlab = "Mutual Fund Ratings", ylab = "5-Year Returns")

{r} labels.etf.category = as.data.frame(table(etf$fund_category)) labels.etf.category = labels.etf.category[order(labels.etf.category$Freq, decreasing = FALSE),] barplot(tail(labels.etf.category$Freq, 20), horiz = TRUE, col = rev(heat.colors(20)), names = tail(labels.etf.category$Var1, 20), las = 1, main = "ETF Distribution by Category", xlab = "Number of ETF") labels.mf.category = as.data.frame(table(mf$fund_category)) labels.mf.category = labels.mf.category[order(labels.mf.category$Freq, decreasing = FALSE),] barplot(tail(labels.mf.category$Freq, 20), horiz = TRUE, col = rev(heat.colors(20)), names = tail(labels.mf.category$Var1, 20), las = 1, main = "Mutual Fund Distribution by Category", xlab = "Number of MF")

6. Data Analysis

6.1 Categorical Analysis

6.1.1 Sector Correlation Analysis

{r} etf.sector.correlation = cbind(etf$fund_return_ytd, etf.sectors) names(etf.sector.correlation)[names(etf.sector.correlation) == 'etf$fund_return_ytd'] = 'fund_return_ytd' etf.sector.correlation[is.na(etf.sector.correlation)] = 0 labels.etf.sector.correlation = cor(etf.sector.correlation$fund_return_ytd, etf.sector.correlation[, -c(1,2)]) barplot(labels.etf.sector.correlation, names = names(labels.etf.sector.correlation), las = 2, main = "Correlation of Sector with ETF Returns",) mf.sector.correlation = cbind(mf$fund_return_ytd, mf.sectors) names(mf.sector.correlation)[names(mf.sector.correlation) == 'mf$fund_return_ytd'] = 'fund_return_ytd' mf.sector.correlation[is.na(mf.sector.correlation)] = 0 labels.mf.sector.correlation = cor(mf.sector.correlation$fund_return_ytd, mf.sector.correlation[, -c(1,2)]) barplot(labels.mf.sector.correlation, col = "grey", names = names(labels.mf.sector.correlation), las = 2, main = "Correlation of Sector with MF Returns",) df.barplot.correlation = as.data.frame(cbind(t(data.frame(labels.etf.sector.correlation)), t(data.frame(labels.mf.sector.correlation)))) names(df.barplot.correlation)[names(df.barplot.correlation) == 'V1'] = 'ETF' names(df.barplot.correlation)[names(df.barplot.correlation) == 'V2'] = 'MF'

barplot(t(as.matrix(df.barplot.correlation[, 1:2])), beside = TRUE, names.arg = df.barplot.correlation$lengthclass, las = 2, legend.text = TRUE, ylab = "Correlation with Fund returns",)

6.1.2 Sector Regression Analysis

{r} lm.etf.sector = lm(etf.sector.correlation$fund_return_ytd ~ ., data = subset(etf.sector.correlation, select = -c(fund_symbol))) summary(lm.etf.sector) lm.mf.sector = lm(mf.sector.correlation$fund_return_ytd ~ ., data = subset(mf.sector.correlation, select = -c(fund_symbol))) summary(lm.mf.sector)

6.1.3 Ratio Correlation Analysis

{r} etf.ratios.correlation = cbind(etf$fund_return_ytd, etf.ratios) names(etf.ratios.correlation)[names(etf.ratios.correlation) == 'etf$fund_return_ytd'] = 'fund_return_ytd' etf.ratios.correlation[is.na(etf.ratios.correlation)] = 0 mf.ratios.correlation = cbind(mf$fund_return_ytd, mf.ratios) names(mf.ratios.correlation)[names(mf.ratios.correlation) == 'mf$fund_return_ytd'] = 'fund_return_ytd' mf.ratios.correlation[is.na(mf.ratios.correlation)] = 0

6.1.4 Ratio Regression Analysis

{r} lm.etf.ratios = lm(etf.ratios.correlation$fund_return_ytd ~ ., data = subset(etf.ratios.correlation, select = -c(fund_symbol))) summary(lm.etf.ratios) lm.mf.ratio = lm(mf.ratios.correlation$fund_return_ytd ~ ., data = subset(mf.ratios.correlation, select = -c(fund_symbol))) summary(lm.mf.ratio)

6.1.5 Return Correlation Analysis

{r} etf.return.correlation = cbind(etf$fund_return_ytd, etf.fundReturn) names(etf.return.correlation)[names(etf.return.correlation) == 'etf$fund_return_ytd'] = 'fund_return_ytd' etf.return.correlation[is.na(etf.return.correlation)] = 0 mf.return.correlation = cbind(mf$fund_return_ytd, mf.fundReturn) names(mf.return.correlation)[names(mf.return.correlation) == 'mf$fund_return_ytd'] = 'fund_return_ytd' mf.return.correlation[is.na(mf.return.correlation)] = 0

6.1.6 Return Regression Analysis

{r} lm.etf.return = lm(etf.return.correlation$fund_return_ytd ~ ., data = subset(etf.return.correlation, select = -c(fund_symbol))) summary(lm.etf.return) lm.mf.return = lm(mf.return.correlation$fund_return_ytd ~ ., data = subset(mf.return.correlation, select = -c(fund_symbol))) summary(lm.mf.return)

6.1.7 Assets Correlation Analysis

{r} etf.asset.correlation = cbind(etf$fund_return_ytd, etf.assets) names(etf.asset.correlation)[names(etf.asset.correlation) == 'etf$fund_return_ytd'] = 'fund_return_ytd' etf.asset.correlation[is.na(etf.asset.correlation)] = 0 mf.asset.correlation = cbind(mf$fund_return_ytd, mf.assets) names(mf.asset.correlation)[names(mf.asset.correlation) == 'mf$fund_return_ytd'] = 'fund_return_ytd' mf.asset.correlation[is.na(mf.asset.correlation)] = 0

6.1.8 Asset Regression Analysis

{r} lm.etf.asset = lm(etf.asset.correlation$fund_return_ytd ~ ., data = subset(etf.asset.correlation, select = -c(fund_symbol))) summary(lm.etf.asset) lm.mf.asset = lm(mf.asset.correlation$fund_return_ytd ~ ., data = subset(mf.asset.correlation, select = -c(fund_symbol))) summary(lm.mf.asset)

6.2 Correlation Analysis

{r} etf.correlation = cbind(etf$fund_return_ytd, subset(etf.sectors, select = -c(fund_symbol)), subset(etf.ratios, select = -c(fund_symbol)), subset(etf.fundReturn, select = -c(fund_symbol)), subset(etf.assets, select = -c(fund_symbol))) names(etf.correlation)[names(etf.correlation) == 'etf$fund_return_ytd'] = 'fund_return_ytd' etf.correlation[is.na(etf.correlation)] = 0 set.seed(432) etf.correlation_idx = sample(nrow(etf.correlation), size = 0.8*nrow(etf.correlation)) etf.correlation_trn = etf.correlation[etf.correlation_idx,] # training data etf.correlation_tst = etf.correlation[-etf.correlation_idx,] # test data mf.correlation = cbind(mf$fund_return_ytd, subset(mf.sectors, select = -c(fund_symbol)), subset(mf.ratios, select = -c(fund_symbol)), subset(mf.fundReturn, select = -c(fund_symbol)), subset(mf.assets, select = -c(fund_symbol))) names(mf.correlation)[names(mf.correlation) == 'mf$fund_return_ytd'] = 'fund_return_ytd' mf.correlation[is.na(mf.correlation)] = 0 set.seed(432) mf.correlation_idx =

sample(nrow(mf.correlation), size = 0.8*nrow(mf.correlation)) mf.correlation_trn = mf.correlation[mf.correlation_idx,] # training data mf.correlation_tst = mf.correlation[-mf.correlation_idx,] # test data

6.3 Regression

6.3.1 Linear Regression

{r} set.seed(432) lm.etf = lm(etf.correlation_trn$fund_return_ytd ~ ., data = etf.correlation_trn) summary(lm.etf) lm.etf.predict = predict(lm.etf, newdata = etf.correlation_tst) mse(lm.etf.predict, etf.correlation_tst$fund_return_ytd) plot(x = etf.correlation_tst$fund_return_ytd, y = lm.etf.predict, pch = 20, col = "darkgrey", main = "ETF Predicted vs Actual (Test Data)", xlab = "Actual", ylab = "Predicted") abline(a = 0, b = 1, lwd = 2) grid()

{r} set.seed(432) lm.mf = lm(mf.correlation_trn$fund_return_ytd ~ ., data = mf.correlation_trn) summary(lm.mf) lm.mf.predict = predict(lm.mf, newdata = mf.correlation_tst) mse(lm.mf.predict, mf.correlation_tst$fund_return_ytd) plot(x = mf.correlation_tst$fund_return_ytd, y = lm.mf.predict, pch = 20, col = "darkgrey", main = "MF Predicted vs Actual (Test Data)", xlab = "Actual", ylab = "Predicted") abline(a = 0, b = 1, lwd = 2) grid()

6.3.2 Best Subsets Regression

{r} set.seed(432) regit.etf = regsubsets(etf.correlation_trn$fund_return_ytd ~ ., data = etf.correlation_trn, nvmax = 10 , really.big=T) regit.etf.summary = summary(regit.etf) # Best subset according to BIC criterion which.min(regit.etf.summary$bic) # Variables included in the best subset coef(regit.etf,10) # BIC vs model size plot plot(regit.etf.summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l") points(10,regit.etf.summary$bic[10],col="blue",cex=1.5,pch=20) regit.etf.tst.mat = model.matrix(etf.correlation_tst$fund_return_ytd ~ ., data = etf.correlation_tst ,really.big=T) regit.etf.coefi = coef(regit.etf, 10) regit.etf.pred = regit.etf.tst.mat[,names(regit.etf.coefi)] %*% regit.etf.coefi # Calculate MSE mse(etf.correlation_tst$fund_return_ytd, regit.etf.pred)

{r eval=FALSE, include=FALSE} regit.mf = regsubsets(mf.correlation_trn$fund_return_ytd ~ ., data = mf.correlation_trn , nvmax = 10, really.big=T) regit.mf.summary = summary(regit.mf) # Best subset according to BIC criterion

which.min(regit.mf.summary$bic) # Variables included in the best subset coef(regit.mf, 8) # BIC vs model size plot plot(regit.mf.summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l") points(8,regit.mf.summary$bic[8],col="blue",cex=1.5,pch=20) regit.mf.tst.mat = model.matrix(mf.correlation_tst$fund_return_ytd ~ ., data = mf.correlation_tst ,really.big=T) regit.mf.coefi = coef(regit.mf, 8) regit.mf.pred = tst.mat[,names(regit.mf.coefi)] %*% regit.mf.coefi # Calculate MSE mse(mf.correlation_tst$fund_return_ytd, regit.mf.pred)

6.3.3 KNN Regression

{r eval=FALSE, include=FALSE} knn.etf = knnreg(etf.correlation_trn$fund_return_ytd ~ ., data = etf.correlation_trn, k =10) knn.etf.pred = predict(knn.etf, newdata = etf.correlation_tst) mse(etf.correlation_tst$fund_return_ytd, knn.etf.pred)

{r} k = seq(1, 100) etf.correlation.2 =etf.correlation # Model development set.seed(432) knn.etf = lapply(k, function(x){ knnreg(etf.correlation_trn$fund_return_ytd ~ ., data = etf.correlation_trn, k = x)}) knn.etf.pred=lapply(knn.etf, predict, newdata = etf.correlation_tst) # RMSE knn.etf.tst_mse=sapply(knn.etf.pred, mse, actual = etf.correlation_tst$fund_return_ytd) plot(k, knn.etf.tst_mse, type='l', xlab="k", ylab="Validation RMSE", ylim=c(min(knn.etf.tst_mse), max(knn.etf.tst_mse)), col='blue')

{r} knn.mf = knnreg(mf.correlation_trn$fund_return_ytd ~ ., data = mf.correlation_trn, k = 10) knn.mf.pred = predict(knn.mf, newdata = mf.correlation_tst) mse(mf.correlation_tst$fund_return_ytd, knn.mf.pred)

{r eval=FALSE, include=FALSE} k = seq(1, 100) # Model development set.seed(432) knn.mf = lapply(k, function(x){ knnreg(mf.correlation_trn$fund_return_ytd ~ ., data = mf.correlation_trn, k = x)}) knn.mf.pred=lapply(knn.mf, predict, newdata = mf.correlation_tst) # RMSE knn.mf.tst_mse=sapply(knn.mf.pred, mse, actual = mf.correlation_tst$fund_return_ytd) plot(k, knn.mf.tst_mse, type='l', xlab="k", ylab="Validation RMSE", ylim=c(min(knn.mf.tst_mse), max(knn.mf.tst_mse)), col='blue')

6.3.4 Decision Regression Tree

{r} # Developing model set.seed(432) tree.etf = tree(etf.correlation_trn$fund_return_ytd ~ ., data = etf.correlation_trn, control = tree.control(nobs = nrow(etf.correlation_trn), mindev = 0)) # Results #summary(tree.etf) plot(tree.etf) text(tree.etf, pretty = 0) # To improve the

accuracy we do cross validaion cv.tree.etf = cv.tree(tree.etf) cv.tree.etf$size[which.min(cv.tree.etf$dev)] cv.tree.etf$k[which.min(cv.tree.etf$dev)] plot(cv.tree.etf$size, cv.tree.etf$dev, type = "b", xlab='Tree Size', ylab='Error Rate', main = 'Cross Validation: Error Vs Size') plot(cv.tree.etf$k, cv.tree.etf$dev, type = "b", xlab='k', ylab='Error Rate', main = 'Cross Validation: Error Vs k')

{r} # Developing model set.seed(432) tree.mf = tree(mf.correlation_trn$fund_return_ytd ~ ., data = mf.correlation_trn, control = tree.control(nobs = nrow(mf.correlation_trn), mindev = 0)) # Results #summary(tree.etf) plot(tree.mf) text(tree.mf, pretty = 0) # To improve the accuracy we do cross validaion cv.tree.mf = cv.tree(tree.mf) cv.tree.mf$size[which.min(cv.tree.mf$dev)] cv.tree.mf$k[which.min(cv.tree.mf$dev)] plot(cv.tree.mf$size, cv.tree.mf$dev, type = "b", xlab='Tree Size', ylab='Error Rate', main = 'Cross Validation: Error Vs Size') plot(cv.tree.mf$k, cv.tree.mf$dev, type = "b", xlab='k', ylab='Error Rate', main = 'Cross Validation: Error Vs k')

Cross-Validation Pruning

{r} set.seed(432) prune.tree.etf <- prune.tree(tree.etf, k = 7) plot(prune.tree.etf) text(prune.tree.etf, pretty = 0) prune.tree.etf.pred <- predict(prune.tree.etf, newdata = etf.correlation_tst) mse(prune.tree.etf.pred, etf.correlation_tst$fund_return_ytd)

{r} set.seed(432) prune.tree.mf <- prune.tree(tree.mf, k = 7) plot(prune.tree.mf) text(prune.tree.mf, pretty = 0) prune.tree.mf.pred <- predict(prune.tree.mf, newdata = mf.correlation_tst) mse(prune.tree.mf.pred, mf.correlation_tst$fund_return_ytd)

Random Forest

{r} # Development of model set.seed(432) rf.etf = randomForest(etf.correlation_trn$fund_return_ytd ~ ., data = etf.correlation_trn, mtry = 10, ntree = 500, importance = TRUE) rf.etf rf.etf.pred = predict(rf.etf, newdata = etf.correlation_tst) mse(rf.etf.pred, etf.correlation_tst$fund_return_ytd) rf.imp.etf = as.data.frame(importance(rf.etf)) rf.imp.etf[order(rf.imp.etf$`%IncMSE`, decreasing = TRUE),] varImpPlot(rf.etf)

{r} # Development of model set.seed(432) rf.mf = randomForest(mf.correlation_trn$fund_return_ytd ~ ., data = mf.correlation_trn, mtry = 10, ntree = 500, importance = TRUE) rf.mf rf.mf.pred = predict(rf.mf, newdata = mf.correlation_tst) mse(rf.mf.pred, mf.correlation_tst$fund_return_ytd) rf.imp.mf =

as.data.frame(importance(rf.mf)) rf.imp.mf[order(rf.imp.mf$`%IncMSE`, decreasing = TRUE),] varImpPlot(rf.mf)

Boosting

{r} gbmGrid = expand.grid(interaction.depth = c(1, 2, 3), n.trees = c(500, 1000, 1500), shrinkage = c(0.001, 0.01, 0.1), n.minobsinnode = 10) # Model Development set.seed(432) gbm.etf = train(fund_return_ytd ~ ., data = etf.correlation_trn, method = "gbm", trControl = trainControl(method = "cv", number = 10), verbose = FALSE, tuneGrid = gbmGrid,) # Results plot(gbm.etf) gbm.etf$bestTune mse(predict(gbm.etf, etf.correlation_tst), etf.correlation_tst$fund_return_ytd)

{r} gbmGrid = expand.grid(interaction.depth = 3, n.trees = c(100, 500), shrinkage = c(0.01, 0.1), n.minobsinnode = 10) # Model Development set.seed(432) gbm.mf = train(fund_return_ytd ~ ., data = mf.correlation_trn, method = "gbm", trControl = trainControl(method = "cv", number = 10), verbose = FALSE, tuneGrid = gbmGrid,) # Results plot(gbm.mf) gbm.mf$bestTune mse(predict(gbm.mf, mf.correlation_tst), mf.correlation_tst$fund_return_ytd)

6.3.5 Ridge Regression

{r} model.control = trainControl(method = "cv", number = 10) # Ridge regression Model Development ridge.grid = expand.grid(lambda = seq(from=0,to=0.1,by=0.005), alpha = 0) set.seed(432) ridge.etf = train(fund_return_ytd ~ ., data = etf.correlation_trn, method = "glmnet", tuneGrid = ridge.grid, trControl = model.control, prePoc=c("center","scale")) plot(ridge.etf) ridge.etf$bestTune ridge.etf.rmse = ridge.etf$results$RMSE[ridge.etf$results$alpha == ridge.etf$bestTune$alpha & ridge.etf$results$lambda == ridge.etf$bestTune$lambda] ridge.etf.rmse^2 mse(predict(ridge.etf, etf.correlation_tst), etf.correlation_tst$fund_return_ytd)

{r} model.control = trainControl(method = "cv", number = 10) # Ridge regression Model Development ridge.grid = expand.grid(lambda = seq(from=0,to=0.1,by=0.005), alpha = 0) set.seed(432) ridge.mf = train(fund_return_ytd ~ ., data = mf.correlation_trn, method = "glmnet", tuneGrid = ridge.grid, trControl = model.control, prePoc=c("center","scale")) plot(ridge.mf) ridge.mf$bestTune ridge.mf.rmse = ridge.mf$results$RMSE[ridge.mf$results$alpha == ridge.mf$bestTune$alpha & ridge.mf$results$lambda ==

ridge.mf$bestTune$lambda] ridge.mf.rmse^2 mse(predict(ridge.mf, mf.correlation_tst), mf.correlation_tst$fund_return_ytd)

6.3.6 Lasso Regression

{r} model.control = trainControl(method = "cv", number = 10) # Ridge regression Model Development lasso.grid = expand.grid(lambda = seq(from=0,to=0.1,by=0.005), alpha = 1) set.seed(432) lasso.etf = train(fund_return_ytd ~ ., data = etf.correlation_trn, method = "glmnet", tuneGrid = lasso.grid, trControl = model.control, prePoc=c("center","scale")) plot(lasso.etf) lasso.etf$bestTune lasso.etf.rmse = lasso.etf$results$RMSE[lasso.etf$results$alpha == lasso.etf$bestTune$alpha & lasso.etf$results$lambda == lasso.etf$bestTune$lambda] lasso.etf.rmse^2 mse(predict(lasso.etf, etf.correlation_tst), etf.correlation_tst$fund_return_ytd)

{r} model.control = trainControl(method = "cv", number = 10) # Ridge regression Model Development lasso.grid = expand.grid(lambda = seq(from=0,to=0.1,by=0.005), alpha = 1) set.seed(432) lasso.mf = train(fund_return_ytd ~ ., data = mf.correlation_trn, method = "glmnet", tuneGrid = lasso.grid, trControl = model.control, prePoc=c("center","scale")) plot(lasso.mf) lasso.mf$bestTune lasso.mf.rmse = lasso.mf$results$RMSE[lasso.mf$results$alpha == lasso.mf$bestTune$alpha & lasso.mf$results$lambda == lasso.mf$bestTune$lambda] lasso.mf.rmse^2 mse(predict(lasso.mf, mf.correlation_tst), mf.correlation_tst$fund_return_ytd)

6.4 Classification

{r} etf.correlation = cbind(etf$fund_return_ytd, etf$investment_type, etf$size_type, subset(etf.sectors, select = -c(fund_symbol)), subset(etf.ratios, select = -c(fund_symbol)), subset(etf.fundReturn, select = -c(fund_symbol)), subset(etf.assets, select = -c(fund_symbol))) names(etf.correlation)[names(etf.correlation) == 'etf$fund_return_ytd'] = 'fund_return_ytd' etf.correlation[is.na(etf.correlation)] = 0 set.seed(432) etf.correlation_idx = sample(nrow(etf.correlation), size = 0.8*nrow(etf.correlation)) etf.correlation_trn = etf.correlation[etf.correlation_idx,] # training data etf.correlation_tst = etf.correlation[-etf.correlation_idx,] # test data mf.correlation = cbind(mf$fund_return_ytd, mf$investment_type, mf$size_type, subset(mf.sectors, select = -c(fund_symbol)), subset(mf.ratios, select = -c(fund_symbol)), subset(mf.fundReturn, select = -c(fund_symbol)), subset(mf.assets, select = -c(fund_symbol)))

names(mf.correlation)[names(mf.correlation) == 'mf$fund_return_ytd'] = 'fund_return_ytd' mf.correlation[is.na(mf.correlation)] = 0 set.seed(432) mf.correlation_idx = sample(nrow(mf.correlation), size = 0.8*nrow(mf.correlation)) mf.correlation_trn = mf.correlation[mf.correlation_idx,] # training data mf.correlation_tst = mf.correlation[-mf.correlation_idx,] # test data

6.4.1 Logistic Regression

{r warning=FALSE} etf.logit = etf.correlation etf.logit[is.na(etf.logit)] = 0 etf.logit$fund_return_ytd = factor(ifelse(etf.logit$fund_return_ytd < mean(etf.logit$fund_return_ytd), "Low", "High")) # Generate random indices set.seed(432) etf.logit.tst_idx = sample(nrow(etf.logit),size=0.2*nrow(etf.logit)) # Split data into 'test data' and 'train data' etf.logit.tst = etf.logit[etf.logit.tst_idx,] etf.logit.trn = etf.logit[-etf.logit.tst_idx,] # Model development based on training data set.seed(432) etf.logit.model <- glm(fund_return_ytd ~ ., data = etf.logit.trn, family = "binomial") etf.logit.model.pred = predict(etf.logit.model, etf.logit.tst, type = "response") etf.logit.model.prob = as.factor(ifelse(etf.logit.model.pred > 0.5, "Low", "High")) # Confusion Matrix confusionMatrix(etf.logit.model.prob, etf.logit.tst$fund_return_ytd)

{r warning=FALSE} mf.logit = mf.correlation mf.logit[is.na(mf.logit)] = 0 mf.logit$fund_return_ytd = factor(ifelse(mf.logit$fund_return_ytd < mean(mf.logit$fund_return_ytd), "Low", "High")) # Generate random indices set.seed(432) mf.logit.tst_idx = sample(nrow(mf.logit),size=0.2*nrow(mf.logit)) # Split data into 'test data' and 'train data' mf.logit.tst = mf.logit[mf.logit.tst_idx,] mf.logit.trn = mf.logit[-mf.logit.tst_idx,] # Model development based on training data set.seed(432) mf.logit.model <- glm(fund_return_ytd ~ ., data = mf.logit.trn, family = "binomial") mf.logit.model.pred = predict(mf.logit.model, mf.logit.tst, type = "response") mf.logit.model.prob = as.factor(ifelse(mf.logit.model.pred > 0.5, "Low", "High")) # Confusion Matrix confusionMatrix(mf.logit.model.prob, mf.logit.tst$fund_return_ytd)

6.4.2 Linear Discriminant Analysis

{r warning=FALSE} etf.logit = etf.correlation etf.logit[is.na(etf.logit)] = 0 etf.logit$`etf$investment_type` = as.factor(etf.logit$`etf$investment_type`) # Generate random indices set.seed(432) etf.logit.tst_idx = sample(nrow(etf.logit),size=0.2*nrow(etf.logit)) # Split data into 'test data' and 'train data'

etf.logit.tst = etf.logit[etf.logit.tst_idx,] etf.logit.trn = etf.logit[-etf.logit.tst_idx,] # Model development based on training data set.seed(432) etf.logit.model <- lda(`etf$investment_type` ~ ., data = etf.logit.trn, family = "binomial") etf.logit.model.pred = predict(etf.logit.model, etf.logit.tst, type = "response") # Confusion Matrix confusionMatrix(etf.logit.model.pred$class, etf.logit.tst$`etf$investment_type`)

{r warning=FALSE} etf.logit = etf.correlation etf.logit[is.na(etf.logit)] = 0 etf.logit$`etf$size_type` = as.factor(etf.logit$`etf$size_type`) # Generate random indices set.seed(432) etf.logit.tst_idx = sample(nrow(etf.logit),size=0.2*nrow(etf.logit)) # Split data into 'test data' and 'train data' etf.logit.tst = etf.logit[etf.logit.tst_idx,] etf.logit.trn = etf.logit[-etf.logit.tst_idx,] # Model development based on training data set.seed(432) etf.logit.model <- lda(`etf$size_type` ~ ., data = etf.logit.trn, family = "binomial") etf.logit.model.pred = predict(etf.logit.model, etf.logit.tst, type = "response") # Confusion Matrix confusionMatrix(etf.logit.model.pred$class, etf.logit.tst$`etf$size_type`)

{r warning=FALSE} mf.logit = mf.correlation mf.logit[is.na(mf.logit)] = 0 mf.logit$`mf$investment_type` = as.factor(mf.logit$`mf$investment_type`) # Generate random indices set.seed(432) mf.logit.tst_idx = sample(nrow(mf.logit),size=0.2*nrow(mf.logit)) # Split data into 'test data' and 'train data' mf.logit.tst = mf.logit[mf.logit.tst_idx,] mf.logit.trn = mf.logit[-mf.logit.tst_idx,] # Model development based on training data set.seed(432) mf.logit.model <- lda(`mf$investment_type` ~ ., data = mf.logit.trn, family = "binomial") mf.logit.model.pred = predict(mf.logit.model, mf.logit.tst, type = "response") # Confusion Matrix confusionMatrix(mf.logit.model.pred$class, mf.logit.tst$`mf$investment_type`)

{r warning=FALSE} mf.logit = mf.correlation mf.logit[is.na(mf.logit)] = 0 mf.logit$`mf$size_type` = as.factor(mf.logit$`mf$size_type`) # Generate random indices set.seed(432) mf.logit.tst_idx = sample(nrow(mf.logit),size=0.2*nrow(mf.logit)) # Split data into 'test data' and 'train data' mf.logit.tst = mf.logit[mf.logit.tst_idx,] mf.logit.trn = mf.logit[-mf.logit.tst_idx,] # Model development based on training data set.seed(432) mf.logit.model <- lda(`mf$size_type` ~ ., data = mf.logit.trn, family = "binomial") mf.logit.model.pred = predict(mf.logit.model, mf.logit.tst, type = "response") # Confusion Matrix confusionMatrix(mf.logit.model.pred$class, mf.logit.tst$`mf$size_type`)