HOW XGBOOST INVESTIGATES MARKET CHANGES

Group Project #2

AFM 423 Winter 2025

Melissa Meng, Cathy Ye, Kaylee Yuan

1.0 ABSTRACT

Stock return is influenced by many predictors and is measured by both traditional mathematical and machine learning models. As one of the machine learning models used to predict stock returns, boosted trees are built to minimize the training errors for models as each new tree is added in an optimal way. This research thus examines different parameter settings of XGBoost for the accuracy of predicting stock returns based on multi predictor variables. The model has been trained with 1,207 US stocks using 10 predictors from various categories such as momentum, liquidity, volatility, value, size, and dividend for the time before and after November 30th, 2018. Data preprocessing and backtesting was performed for the purpose of accuracy and efficiency. The results show that the XGBoost model achieved moderate predictive power with signs of overfitting and shifting factor relevance. These findings suggest evolving market dynamics over time and highlight the importance of adapting models to regime changes.

2.0 RESEARCH QUESTIONS

- What impact does each portion of your R codes have on the performance of your program?
- How do the characteristics of the dataset affect your algorithm?
- What impact do different parameter settings have on XGBoost's predictive accuracy in financial datasets?
- How does your framework/system implement an interesting application of the ML approach to FI?

This project investigates how machine learning can be effectively applied to factor investing by examining several key research questions.

First, we analyze the impact of each portion of R code on model performance and execution efficiency to ensure clarity, reproducibility, and streamlined experimentation. We additionally examine the impacts of financial dataset attributes, including noise, missing values, and non-stationarity, on learning and prediction performance of XGBoost. Also, we analyze the model's accuracy and generalization ability with regard to different settings of the model's hyperparameters to optimize the accuracy of the model and limit overfitting. Additionally, we analyze how the entire framework serves as an effective example of machine learning applications in factor investing, particularly in how multivariate regression analysis captures changes in the importance of factors between different market regimes before and after November 30th, 2018. Collectively, these questions focus our analysis and add to the understanding of the application of the machine learning method in factor investing.

3.0 THE APPLICATION OF THE ML APPROACH TO FACTOR INVESTING

Our project explores the application of a machine learning model, XGBoost, to Factor Investing by detecting and analyzing market shifts in different market behaviours within a certain time period. Traditionally, Factor Investing involves constructing portfolios based on some quantifiable market characteristics, such as value and momentum. But in complex financial markets, relationships between factors are more non-linear and subject to time effect and collinearity that traditional linear models may fail to capture. This motivates the use of some machine learning models such as XGBoost, which can intricate patterns directly from given data without strong assumptions about the underlying data generating processes.

The main goal of our study is to observe and evaluate how factor importance and their predictive relationships with returns have changed in a selected time period, particularly across a significant structural break in the market. We selected November 30th, 2018 as the

cutoff date to divide the historical time period into two segments: the time before November 30th, 2018 and the time after November 30th, 2018. This choice is motivated by a combination of global financial and economic events that marked a transition in market behaviours. During 2018, the global economy experienced several disruptions, such as rising trade tensions between the U.S. and China, volatile commodity prices, and increasing geopolitical uncertainty. In particular, the U.S. Federal Reserve has been continuing to tighten monetary policy with a rate hike in December 2018, which led to sharp market correction during the fourth quarter. Investors reallocated capital exhibited a profound change in sentiment around risk along with shifts on the importance of various factors, prompting many institutions to realign their strategies. In this regard, the date of November 30th, 2018 strikes as a simultaneously logical and significant cutoff date from which changes in market behavior are studied in the context of shifts in factor performance and market trends. The goal is to see whether or not changes in the macro environment translates, in a measurable sense, into changes in how relevant factors are, predictability of returns and overall market activity detectable by XGBoost during those two periods.

To achieve this, we applied the XGBoost algorithm over stock data with several fundamental and technical factors. The dataset contains frequently used predictors such as the earnings-per-share (Eps) ratio, price momentum measures, volatility, and firm size. We trained XGBoost on pre-cutoff data and tested on post-cutoff data to assess performance changes over time along with factor importance and model changes.

In this case, we optimize hyperparameters via cross-validation while managing lookahead bias and assigning 3-month forward returns as the target. This configuration supports the premise that the model emulates an actual investment strategy and allows us to analyze the effectiveness of the factors across different market conditions. By comparing the trained models' outputs across the two periods, we aim to assess whether certain factors

became more or less predictive, and to what extent XGBoost captures regime shifts that could inform dynamic portfolio allocation.

4.0 VARIABLES AND MEASURES

In the original dataset data_ml.xlsx, there are 1,207 stocks listed in the US stock market. The time period is sampled in monthly frequency, ranging from November 11, 1998 to March 31, 2019. There are a total of 244 months with no missing observations. However, the sample is not perfectly rectangular due to the non-constant number of characteristics of firms through the entire time period.

In terms of every data point, there are 93 predictor variables in the sample. These attributes include most performance metrics such as liquidity, volatility, valuation, profitability, leverage, momentum, cash flow, and dividend yields. They align with established financial models and build foundations for research and investment purposes. The detailed descriptions of every 93 characteristics can be found in the Appendix section. Due to the size and efficiency of the research, only the following 10 features are chosen: "Mom_11M_Usd", "Debtequity", "Vol1Y_Usd", "Mkt_Cap_12M_Usd", "Div_Yld", "Pb", "Return_On_Capital", "Ebit_Ta", "Eps", and "Advt_12M_Usd". "Mom_11M_Usd" might capture momentum anomalies and reflect potential behavioural biases while “Debtequity” measures financial solvency and leverage risks in crises like 2008. "Vol1Y_Usd" demonstrates volatility and instability during the diverse market conditions whereas "Mkt_Cap_12M_Usd" represents the size effects of stocks. In addition, "Div_Yld" indicates shareholder returns and potential ability to attract more investors. "Pb" is also an important value factor related to High Minus Low (HML) that is discussed in the Fama-French three-factor model. "Return_On_Capital", "Ebit_Ta", and “Eps” are all essential factors to measure profitability, operational efficiency, and valuation for equity analysis. Finally,

"Advt_12M_Usd" refers to average daily trading volume, which greatly shows the liquidity of the US market. By selecting these 10 predictors, the model has a multi-dimensional framework for predicting stock returns, risks, and market trends over the entire period.

The response variable refers to the labels in the Appendix section. The four labels in the original dataset are “R1M_Usd”, “R3M_Usd”, “R6M_Usd”, and “R12M_Usd”. The study uses “R3M_Usd” as the response variable because there are too many zeros in other labels which might influence the accuracy of the final results. “R3M_Usd” means the stock returns forward 3 months. The returns are total returns that incorporate potential dividend payments during the sample periods.

5.0 EXPERIMENTAL METHODOLOGY

In this project, we apply the XGBoost algorithm to a supervised binary classification problem in Factor Investing, where the target is to predict whether a stock belongs to the top 30% performers over the next three months based on a selected set of financial features. We document below our full experimental setup to ensure that our process is reproducible and transparent.

5.1 Tools and Environment

All experiments were conducted using the R programming language, with a primary focus on machine learning and data analysis. The XGBoost package was used to train and test a gradient boosting classifier, which was selected due to its speed and efficacy with structured data. Data wrangling and manipulation were performed using the tidyverse suite, while lubridate was used to handle date formatting and filtering. Model performance was assessed using the pROC package, which enabled the calculation and visualization of Receiver Operating Characteristic (ROC) curves and Area Under the Curve (AUC) metrics.

Development and experimentation were carried out using both Visual Studio Code and RStudio. Code execution took place locally on both macOS and Windows systems to ensure platform compatibility. During training, evaluation metrics such as AUC were monitored in real time using xgb.cv() with 5-fold stratified cross-validation. Early stopping was enabled to determine the optimal number of boosting rounds and prevent overfitting.

Although formal timing tools were not employed, training time and performance progression were visually tracked via evaluation_log outputs and diagnostic plots. Random seed setting (set.seed(42)) was used to ensure the experiment could be replicated under the same conditions.

5.2 Dataset

As addressed in the Variables and Measures section, the dataset used was a financial time-series dataset stored in data_ml.RData. To avoid data leakage and simulate real-world forecasting conditions, the data was filtered to include only entries from July 2017 to June 2019. To ensure consistency in model inputs, only stocks with the most complete time series records were retained for training and testing. The data was split chronologically:

- Training set: Observations before November 30, 2018
- Testing set: Observations on or after November 30, 2018

The target variable was a binary classification label derived from 3-month forward returns (R3M_Usd), where values in the top quartile were labeled as 1 and those in the bottom quartile as 0. This framing allowed the model to distinguish between top and bottom-performing stocks.

5.3 Model Tuning

Hyperparameter tuning was performed using xgb.cv() with stratified 5-fold cross-validation. The tuning process was optimized for AUC, with early stopping rounds set

to 25 to terminate training when no further improvement was observed. Key hyperparameters adjusted included:

- max_depth = 6
- eta = 0.01 (learning rate)
- subsample = 0.8
- colsample_bytree = 0.8
- min_child_weight = 3
- gamma = 0.1
- Regularization: lambda = 0.5, alpha = 0.5

In addition, class imbalance was handled using the scale_pos_weight parameter, calculated as the ratio of negative to positive samples in the training data. This adjustment ensured the model did not favor the majority class during training. The best number of boosting rounds was selected based on validation AUC performance and was later used to train the final model on the full training set.

5.4 Model Evaluation

Model performance was evaluated using the out-of-sample test set. Predictions were generated and classified using a 0.5 probability threshold. A confusion matrix was produced to assess classification performance, and ROC curves were plotted using the pROC package to visualize the trade-off between true positive and false positive rates.

The final model achieved an accuracy of approximately 62.9% on the test set, showing an improvement over baseline models. AUC was also computed to measure overall discriminative ability, and performance curves over boosting rounds were plotted using ggplot2 to visualize training and test AUC across iterations.

Feature importance was extracted using xgb.importance() and visualized to interpret which financial indicators had the greatest impact on the model's predictions.

5.5 Implementation Workflow in R

To ensure reproducibility and clarity of our methodology, we summarize below the main implementation workflow as executed in R. The entire process was modularized into distinct stages, each responsible for a key part of the machine learning pipeline.

5.5.1 Data Preprocessing

We began by installing and loading the necessary libraries: tidyverse, lubridate, xgboost, and pROC. The dataset data_ml.RData was then loaded and filtered to retain data between July 2017 and June 2019.

```
library(tidyverse)
library(lubridate)
library(xgboost)
library(pROC)

data_ml <- data_ml %>%
 filter(date > "2017-06-30", date < "2019-06-30") %>%
 arrange(stock_id, date)

# Filter only most complete stocks
stock_days <- data_ml %>%
 group_by(stock_id) %>%
 summarize(nb = n())
stock_ids_short <- stock_days$stock_id[stock_days$nb == max(stock_days$nb)]
data_ml <- filter(data_ml, stock_id %in% stock_ids_short)

# Select Features
features <- c(
 "Mom_11M_Usd", "Debtequity", "Vol1Y_Usd", "Mkt_Cap_12M_Usd",
 "Div_Yld", "Pb", "Return_On_Capital", "Ebit_Ta", "Eps", "Advt_12M_Usd"
)

# Scale Features
data_ml <- data_ml %>%
 group_by(date) %>%
 mutate(across(all_of(features), ~ scale(.)[, 1])) %>%
 ungroup()
```

5.5.2 Label Construction

The classification label was obtained from R3M_Usd. Stocks in the top quartile were labeled as 1 and those in the bottom quartile as 0.

```
q75 <- quantile(data_ml$R3M_Usd, 0.75, na.rm = TRUE)
q25 <- quantile(data_ml$R3M_Usd, 0.25, na.rm = TRUE)

data_ml <- data_ml %>%
 mutate(R3M_Usd_C = case_when(
   R3M_Usd >= q75 ~ 1,
   R3M_Usd <= q25 ~ 0,
   TRUE ~ NA_real_
 )) %>%
 drop_na(R3M_Usd_C)
```

5.5.3 Train-Test Split

The dataset has a cut-off date of 30 November 2018 and is split chronologically. Both training and testing datasets were converted into xgb.DMatrix objects, as required by XGBoost.

```
separation_date <- as.Date("2018-11-30")
train_data <- filter(data_ml, date < separation_date)
test_data <- filter(data_ml, date >= separation_date)

x_train <- as.matrix(train_data[, features])
x_test <- as.matrix(test_data[, features])
y_train <- train_data$R3M_Usd_C
y_test <- test_data$R3M_Usd_C

dtrain <- xgb.DMatrix(data = x_train, label = y_train)
dtest <- xgb.DMatrix(data = x_test, label = y_test)
```

5.5.4 XGBoost Model Configuration and Hyperparameter Tuning

Model tuning is done using cross-validation with early stopping to avoid overfitting.

```
params <- list(
```

```
 objective = "binary:logistic",
 eval_metric = "auc",
 max_depth = 6,
 eta = 0.01,
 subsample = 0.8,
 colsample_bytree = 0.8,
 scale_pos_weight = scale_pos_weight,
 lambda = 0.5,
 alpha = 0.5,
 min_child_weight = 3,
 gamma = 0.1
)

# ---- Model Tuning (Cross-Validation) ----
set.seed(42)
cv <- xgb.cv(
 params = params,
 data = dtrain,
 nrounds = 500,
 nfold = 5,
 stratified = TRUE,
 early_stopping_rounds = 25,
 verbose = 1,
 maximize = TRUE
)

best_nrounds <- cv$best_iteration
```

5.5.5 Final Model Training

We train the final model using the optimal boosting rounds obtained from CV.

```
# ---- Train Model ----
final_model <- xgb.train(
 params = params,
 data = dtrain,
 nrounds = best_nrounds,
 watchlist = list(train = dtrain, test = dtest),
 early_stopping_rounds = 25,
 verbose = 1,
 print_every_n = 10
)
```

5.5.6 XGBoost Model Evaluation and ROC Curve

We evaluate performance using prediction accuracy and AUC with visual diagnostics.

```
# ---- Model Evaluation ----
# Generate predictions, confusion matrix, and AUC
preds <- predict(final_model, dtest)
pred_class <- ifelse(preds > 0.5, 1, 0)

confusionMatrix(factor(pred_class, levels = c(0, 1)), factor(y_test, levels = c(0, 1)))

roc_obj <- roc(y_test, preds)
roc_df <- data.frame(
 TPR = roc_obj$sensitivities,
 FPR = 1 - roc_obj$specificities
)

ggplot(roc_df, aes(x = FPR, y = TPR)) +
 geom_line(color = "steelblue", size = 1) +
 geom_abline(linetype = "dashed", color = "gray") +
 labs(
   title = paste0("ROC Curve (AUC = ", round(auc(roc_obj), 3), ")"),
   x = "False Positive Rate", y = "True Positive Rate"
 ) +
 theme_minimal()
```

5.5.7 Boosting Progress Visualization and Feature Importance

The learning curves help interpret model behavior over training rounds and we inspect which features influenced the model most.

```
eval_log <- data.frame(final_model$evaluation_log)

ggplot(eval_log, aes(iter)) +
 geom_line(aes(y = train_auc, color = "Train AUC")) +
 geom_line(aes(y = test_auc, color = "Test AUC")) +
 labs(title = "XGBoost AUC over Iterations", y = "AUC", x = "Boosting Rounds") +
 scale_color_manual(name = "Legend", values = c("Train AUC" = "blue", "Test AUC" =
"red")) +
 theme_minimal()
```

```
# ---- Feature Importance ----
# Visualize which features were most important to the model
importance_matrix <- xgb.importance(model = final_model)
xgb.plot.importance(importance_matrix, top_n = 10, rel_to_first = TRUE, xlab =
"XGBoost Feature Importance")
```

6.0 RESULTS AND DISCUSSION

6.1 Model Performance

The below Figure 1 shows a ROC Curve with an AUC of 0.63. The x-axis is False Positive Rate, the y-axis is True Positive Rate, and the dashed diagonal line represents a random classifier of AUC at 0.5.

The ROC Curve represents the classification performance of the trained XGBoost model in predicting binary outcomes (e.g. positive vs. negative returns) across the pre-selected market regime after November 30th, 2018. The ROC Curve has an AUC at around 0.63, meaning its modest ability to distinguish between outperforming and underperforming stocks. This result is valuable when examining model behaviours around certain cutoff dates.

By comparing the curve with the random classifier of AUC at 0.5, we assess the performance of the trained model and indicate market shifts and factor relevance with the test set. If the AUC significantly changes after November 30th, 2018, then it is suggesting that XGBoost is sensitive to dynamic changes in the market. For instance, it may capture shifts in many variables, such as momentum, liquidity, and volatility, influencing future returns.

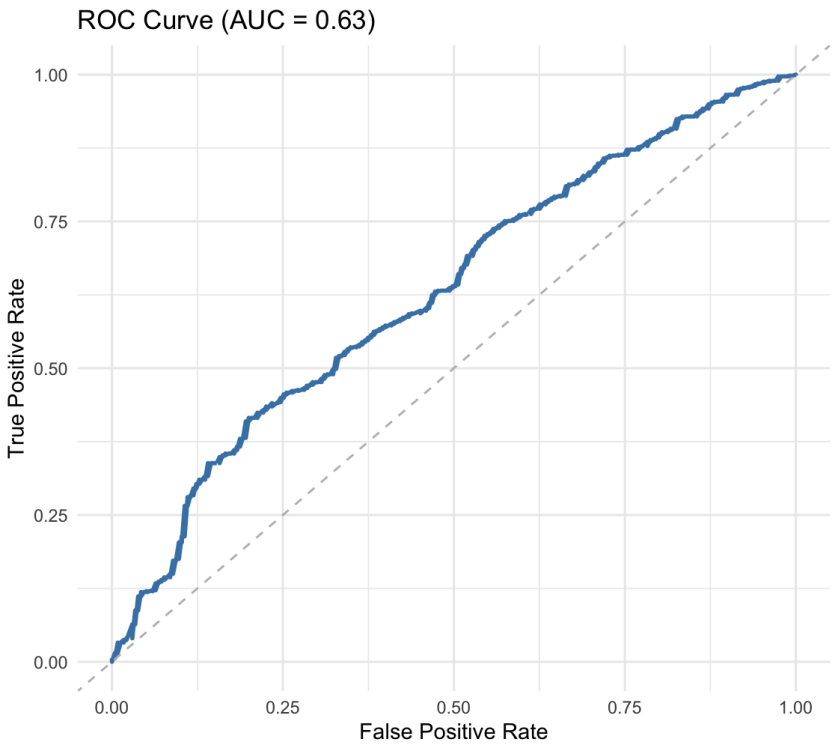

Figure 1. ROC Curve

The below Figure 2. AUC over Boosting rounds indicates the Area Under the Curve (AUC) performance of the XGBoost model on both the training and test datasets over 80 boosting rounds. The red line Training AUC increases dramatically from around 0.63 to 0.75 in the first 10 boosting rounds and then steadily reaches approximately 0.82 by round 80. This suggests that the model continues to fit the training data more accurately as more trees are added. In comparison, the blue line testing AUC shows very similar trends with huge increases from 0.42 to around 0.58 and then plateau around 0.63 in the end. After 20 boosting rounds, any additional rounds do not mean significant improvements for the model performance. After the usage of early stopping, the diminishing returns from further boosting offer limited generalization benefit. Overall, the plot helps to visualize the trade-off between model complexity and performance, illustrating the importance of early stopping to prevent overfitting.

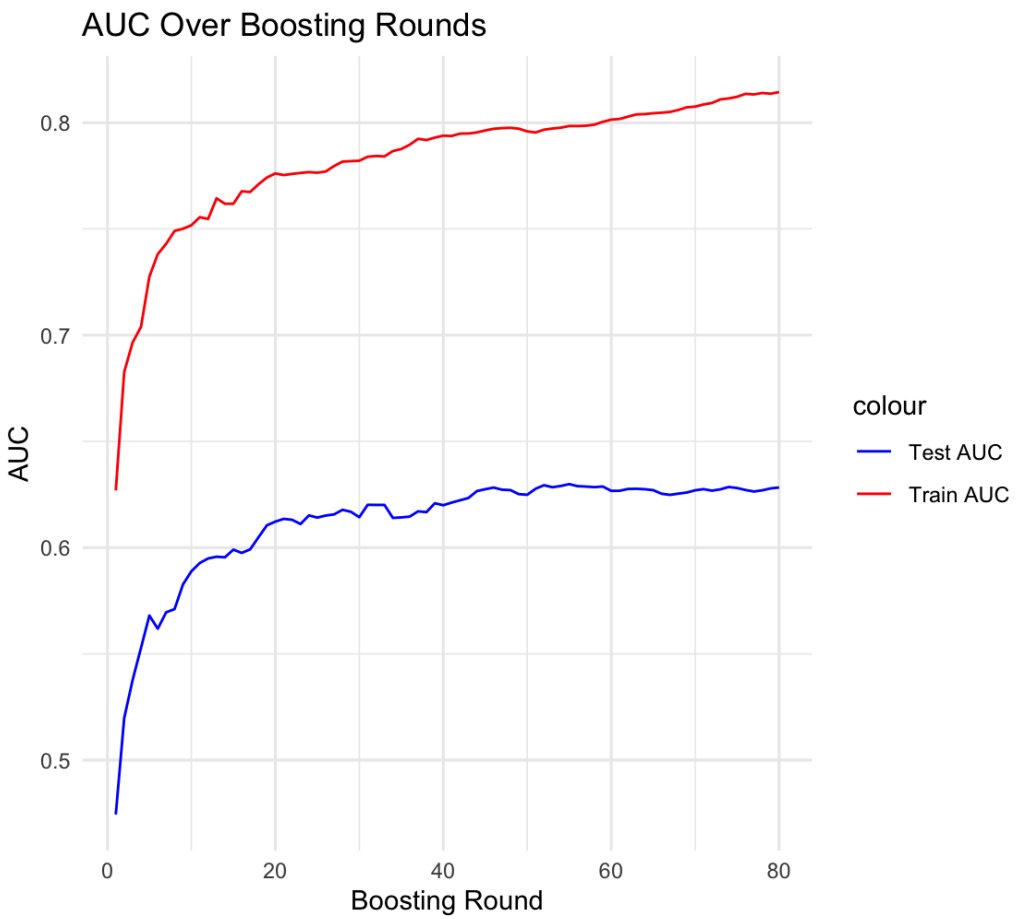

Figure 2. AUC over Boosting Rounds

6.2 Feature Importance

The research uses xgb.importance() to calculate the feature importance and visualize the influence of the chosen 10 financial characteristics on the XGboost model's predictions. Each time a variable is given for a tree split, it reduces the (training) losses and sums these gains during the testing periods. The feature importance is evaluated between 0 and 1 with bigger values suggesting greater gains.

As shown in Figure 3. XGBoost Feature Importance, "Ebit_Ta", "Debtequity", and "Vol1Y_Usd" are the top three influential predictors in the model (higher than 0.8). It demonstrates that profitability, leverage, and volatility are key drivers of short-term stock

return predictability. In contrast, the features “Pb”, “Return_On_Capital”, and “Mkt_Cap_12M_Usd” were ranked as the least influential predictors in the model, suggesting that valuation and firm size had comparatively weaker signals in this context.

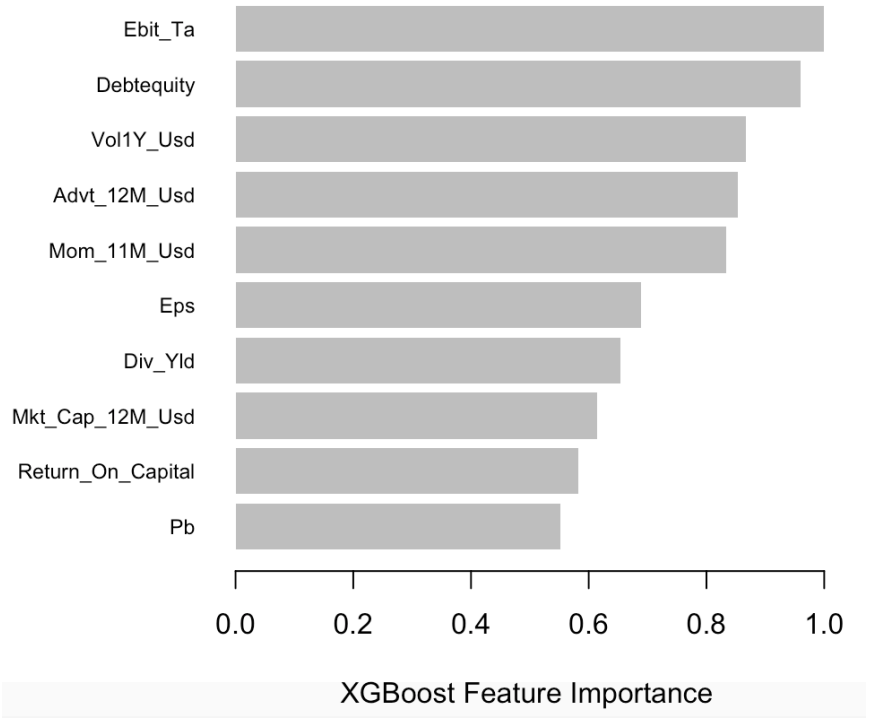

Figure 3. XGBoost Feature Importance

7.0 CONCLUSIONS

Our research using XGBoost shares several meaningful thoughts in terms of factor investing dynamics, particularly around the date of November 30th, 2018.

First, the ROC curve demonstrates that the model captures some predictive signal while the given financial data is noisy and nonlinear. This modest performance of AUC of 0.63 is still valuable in tracking how market behavior and factor importance shifts across the time. The second graph plots the AUC over 80 boosting rounds and compares the differences between the training and test model performance. It may suggest overfitting and reduced generalizability of the trained model on test data. The gap sufficiently reflects model instability in dynamic regimes, which means there is a mismatch between historical (training

set) and future (testing set) data due to macroeconomic changes, policy shifts, or investor sentiment transitions around the cutoff date. The flattening of the Test AUC curve shows that although our model keeps improving in the training set, it fails to adapt to structural shifts in the underlying market behaviour. The third plot, XGBoost importance, allows us to identify dominant signals, where profitability (Ebit_Ta), momentum and liquidity factors dominate the signals in the selected periods. In contrast, value-weighted factors, such as price-to-book (Pb) and dividend yield (Div_Yld), indicate weaker feature impacts. This may reflect potential big macroeconomic events around November 30th, 2018, including rising interest rates, trade tensions, or evolving market sentiment.

Overall, the given results strengthen the motivation of the XGboost method on factor investing. The gap between training and testing performance as shown in the second graph, combined with changing factor importance, suggests that XGBoost can be considered as a valuable tool for identifying market shifts in dynamic market regimes with various factors. Using this tool to analyze financial data in different time periods allows us to assess and quantify changes in market themes, investor sentiment and factor behavior before and after November 30th, 2018, sufficiently providing predictive and executable insights for adaptive factor investing.

8.0 RELATED WORK

Group Project 1 and 2 are highly related, therefore the reader is referred to our GPR # 1 report for related work.

APPENDIX

Exhibit 1 DATA APPENDIX

Predictor Name	Brief Description
Advt_12M_Usd	average daily volume in amount in USD over 12 months
Advt_3M_Usd	average daily volume in amount in USD over 3 months
Advt_6M_Usd	average daily volume in amount in USD over 6 months
Asset_Turnover	total sales on average assets
Bb_Yld	buyback yield
Bv	book value
Capex_Ps_Cf	capital expenditure on price to sale cash flow
Capex_Sales	capital expenditure on sales
Cash_Div_Cf	cash dividends cash flow
Cash_Per_Share	cash per share
Cf_Sales	cash flow per share
Debtequity	debt to equity
Div_Yld	dividend yield

Dps	dividend per share
Ebit_Bv	EBIT on book value
Ebit_Noa	EBIT on non operating asset
Ebit_Oa	EBIT on operating asset
Ebit_Ta	EBIT on total asset
Ebitda_Margin	EBITDA margin
Eps	earnings per share
Eps_Basic	earnings per share basic
Eps_Basic_Gr	earnings per share growth
Eps_Contin_Oper	earnings per share continuing operations
Eps_Dil	earnings per share diluted
Ev	enterprise value
Ev_Ebitda	enterprise value on EBITDA
Fa_Ci	fixed assets on common equity
Fcf	free cash flow
Fcf_Bv	free cash flow on book value
Fcf_Ce	free cash flow on capital employed

Fcf_Margin	free cash flow margin
Fcf_Noa	free cash flow on net operating assets
Fcf_Oa	free cash flow on operating assets
Fcf_Ta	free cash flow on total assets
Fcf_Tbv	free cash flow on tangible book value
Fcf_Toa	free cash flow on total operating assets
Fcf_Yld	free cash flow yield
Free_Ps_Cf	free cash flow on price sales
Int_Rev	intangibles on revenues
Interest_Expense	interest expense coverage
Mkt_Cap_12M_Usd	average market capitalization over 12 months in USD
Mkt_Cap_3M_Usd	average market capitalization over 3 months in USD
Mkt_Cap_6M_Usd	average market capitalization over 6 months in USD
Mom_11M_Usd	price momentum 12-1 months in USD
Mom_5M_Usd	price momentum 6-1 months in USD
Mom_Sharp_11M_Usd	price momentum 12-1 months in USD divided by volatility

Mom_Sharp_5M_Usd	price momentum 6-1 months in USD divided by volatility
Nd_Ebitda	net debt on EBITDA
Net_Debt	net debt
Net_Debt_Cf	net debt on cash flow
Net_Margin	net margin
Netdebtyield	net debt yield
Ni	net income
Ni_Avail_Margin	net income available margin
Ni_Oa	net income on operating asset
Ni_Toa	net income on total operating asset
Noa	net operating asset
Oa	operating asset
Ocf	operating cash flow
Ocf_Bv	operating cash flow on book value
Ocf_Ce	operating cash flow on capital employed
Ocf_Margin	operating cash flow margin
Ocf_Noa	operating cash flow on net operating assets

Ocf_Oa	operating cash flow on operating assets
Ocf_Ta	operating cash flow on total assets
Ocf_Tbv	operating cash flow on tangible book value
Ocf_Toa	operating cash flow on total operating assets
Op_Margin	operating margin
Op_Prt_Margin	net margin 1Y growth
Oper_Ps_Net_Cf	cash flow from operations per share net
Pb	price to book
Pe	price earnings
Ptx_Mgn	margin pretax
Recurring_Earning_Total_Assets	recurring earnings on total assets
Return_On_Capital	return on capital
Rev	revenue
Roa	return on assets
Roc	return on capital
Roce	return on capital employed
Roe	return on equity

Sales_Ps	price to sales
Share_Turn_12M	average share turnover 12 months
Share_Turn_3M	average share turnover 3 months
Share_Turn_6M	average share turnover 6 months
Ta	total assets
Tev_Less_Mktcap	total enterprise value less market capitalization
Tot_Debt_Rev	total debt on revenue
Total_Capital	total capital
Total_Debt	total debt
Total_Debt_Capital	total debt on capital
Total_Liabilities_Total_Assets	total liabilities on total assets
Vol1Y_Usd	volatility of returns over one year
Vol3Y_Usd	volatility of returns over 3 years
R1M_Usd	return forward 1 month (LABEL)
R3M_Usd	return forward 3 months (LABEL)
R6M_Usd	return forward 6 months (LABEL)
R12M_Usd	return forward 12 months (LABEL)

Exhibit 2 R Code

```r
if (!require(tidyverse)) {
  install.packages("tidyverse")
}
if (!require(lubridate)) {
  install.packages("lubridate")
}
if (!require(xgboost)) {
  install.packages("xgboost")
}
if (!require(pROC)) {
  install.packages("pROC")
}

library(tidyverse)
library(lubridate)
library(xgboost)
library(pROC)

# ---- Data Preprocessing ----
# replace the path to your data_ml.RData file
load("/Users/melmeng/Library/Mobile Documents/com~apple~CloudDocs/Mel/4B/AFM 423/H5/data_ml.RData")

data_ml <- data_ml %>%
  filter(date > "2017-06-30", date < "2019-06-30") %>%
  arrange(stock_id, date)

# Filter only most complete stocks
stock_days <- data_ml %>% group_by(stock_id) %>% summarize(nb = n())
stock_ids_short <- stock_days$stock_id[stock_days$nb == max(stock_days$nb)]
data_ml <- filter(data_ml, stock_id %in% stock_ids_short)

# Select Features
features <- c("Mom_11M_Usd", "Debtequity", "Vol1Y_Usd", "Mkt_Cap_12M_Usd",
              "Div_Yld", "Pb", "Return_On_Capital", "Ebit_Ta", "Eps", "Advt_12M_Usd")

# Scale Features
data_ml <- data_ml %>%
  group_by(date) %>%
  mutate(across(all_of(features), ~ scale(.)[, 1])) %>%
  ungroup()

# Define Binary Label
q75 <- quantile(data_ml$R3M_Usd, 0.75, na.rm = TRUE)
q25 <- quantile(data_ml$R3M_Usd, 0.25, na.rm = TRUE)

data_ml <- data_ml %>%
  mutate(R3M_Usd_C = case_when(
    R3M_Usd >= q75 ~ 1,
    R3M_Usd <= q25 ~ 0,
    TRUE ~ NA_real_
  )) %>%
  drop_na(R3M_Usd_C)

# ---- Train-Test Split ----
separation_date <- as.Date("2018-11-30")
train_data <- filter(data_ml, date < separation_date)
test_data  <- filter(data_ml, date >= separation_date)

x_train <- as.matrix(train_data[, features])
x_test  <- as.matrix(test_data[, features])
y_train <- train_data$R3M_Usd_C
y_test  <- test_data$R3M_Usd_C

dtrain <- xgb.DMatrix(data = x_train, label = y_train)
dtest  <- xgb.DMatrix(data = x_test, label = y_test)
```

```r

# ---- Class Imbalance Handling ----
scale_pos_weight <- sum(y_train == 0) / sum(y_train == 1)

# ---- Model Setup ----
params <- list(
  objective = "binary:logistic",
  eval_metric = "auc",
  max_depth = 6,
  eta = 0.01,
  subsample = 0.8,
  colsample_bytree = 0.8,
  scale_pos_weight = scale_pos_weight,
  lambda = 0.5,
  alpha = 0.5,
  min_child_weight = 3,
  gamma = 0.1
)

# ---- Model Tuning (Cross-Validation) ----
set.seed(42)
cv <- xgb.cv(
  params = params,
  data = dtrain,
  nrounds = 500,
  nfold = 5,
  stratified = TRUE,
  early_stopping_rounds = 25,
  verbose = 1,
  maximize = TRUE
)

best_nrounds <- cv$best_iteration

```

```r
# ---- Train Model ----
final_model <- xgb.train(
  params = params,
  data = dtrain,
  nrounds = best_nrounds,
  watchlist = list(train = dtrain, test = dtest),
  early_stopping_rounds = 25,
  verbose = 1,
  print_every_n = 10
)

# ---- Model Evaluation ----
# Generate predictions, confusion matrix, and AUC
preds <- predict(final_model, dtest)
pred_class <- ifelse(preds > 0.5, 1, 0)

confusionMatrix(factor(pred_class, levels = c(0, 1)), factor(y_test, levels = c(0, 1)))

roc_obj <- roc(y_test, preds)
roc_df <- data.frame(
  TPR = roc_obj$sensitivities,
  FPR = 1 - roc_obj$specificities
)

ggplot(roc_df, aes(x = FPR, y = TPR)) +
  geom_line(color = "steelblue", size = 1) +
  geom_abline(linetype = "dashed", color = "gray") +
  labs(title = paste0("ROC Curve (AUC = ", round(auc(roc_obj), 3), ")"),
       x = "False Positive Rate", y = "True Positive Rate") +
  theme_minimal()

# ---- Training Log Visualization ----
eval_log <- data.frame(final_model$evaluation_log)

```

```r
ggplot(eval_log, aes(iter)) +
  geom_line(aes(y = train_auc, color = "Train AUC")) +
  geom_line(aes(y = test_auc, color = "Test AUC")) +
  labs(title = "XGBoost AUC over Iterations", y = "AUC", x = "Boosting Rounds") +
  scale_color_manual(name = "Legend", values = c("Train AUC" = "blue", "Test AUC" = "red")) +
  theme_minimal()

# ---- Feature Importance ----
# Visualize which features were most important to the model
importance_matrix <- xgb.importance(model = final_model)
xgb.plot.importance(importance_matrix, top_n = 10, rel_to_first = TRUE, xlab = "XGBoost Feature Importance")
```

References

Praphutikul, T., & Limpiyakorn, Y. (2023, May 9). XGBoost for smart portfolio management based on Multi Factor Stock Selection. https://dl.acm.org/doi/fullHtml/10.1145/3605423.3605442

Wang, M. (2022, January 1). (PDF) a portfolio strategy based on XGBoost regression and Monte Carlo method. https://www.researchgate.net/publication/368498536_A_Portfolio_Strategy_Based_on_XGBoost_Regression_and_Monte_Carlo_Method

Zolotareva, E. (2021). Aiding long-term investment decisions with XGBoost Machine Learning Model. https://arxiv.org/pdf/2104.09341