Summary

This report presents a predictive model for the sign of S&P500 stock return using the 3-Month Volatility Index. First, we prove that return refers to the raw return as the models using raw return have higher predictive accuracy than excess return. We then prove that further dividing positive return on the predictive outcome does not improve the model accuracy. Moreover, we propose annualised logarithm return on diesel fuel as an explanatory variable and prove that it helps the model predictivity based on AIS, McFadden R^2, and Percent Correctly Predicted. The final result is that a binary probit model with a logarithm of VIX, a logarithm of VIX ratio, and annualised logarithm return on diesel fuel as its explanatory variables has the best accuracy in predicting the signs of raw return at about 82.302%.

1. Data

1.1 Data Collection

Appropriate datasets are needed to generate a model that can predict the sign of stock returns using the Volatility Index. All datasets used in this report are retrieved from the Federal Reserve Economic Data (FRED), with daily values from 1st January 2016 to 31st December 2019. A brief description of each dataset is explained below:

- <u>The Standard and Poor's 500 (SP500)</u>: Stock price index that contains 500 leading companies in the U.S.
- <u>CBOE S&P 500 3-Month Volatility Index (VXVCLS)</u>: Measurement of market volatility based on the prices of options in the S&P 500.
- <u>3-Month Treasury Bill Secondary Market Rate, Discount Basis (DTB3)</u>: The interest rate at which United States treasury bills are traded. We use this as a proxy for a risk-free rate.
- <u>Ultra-Low-Sulfur No. 2 Diesel Fuel Prices: New York Harbor (DDFUELNYH)</u>: It is a type of diesel fuel used for railroad locomotives, marine diesel fuel, on-road and non-road diesel fuel. EPA has mandated this ULSD for all diesel fuels (Wikipedia, 2024).

1.2 Data Processing, Creating Relevant Variables, Data Cleaning

To find the best predictive model, various functional forms and the use of lags are defined for analysis (*See Appendix A for defined forms and their descriptions*). First, the lagged variables of VIX are considered, as the previous day's fear index may potentially influence today's market. Daily changes in the volatility index are also defined to test whether the change in investor confidence level can predict better than the level of the fear index itself. Furthermore, the data we have on S&P 500 and diesel fuel is the price, so we have to convert the price into annualised log-return. The annualised log-return formula:

$$Annualised\ Log - Return = 1200 \times (\log(SP500_t) - \log(SP500_{t-1})) \tag{1.1}$$

We also define excess return or risk premium as annualised log-return or raw return minus the risk-free rate. In this report, we will construct two binary models and one trichotomous model. Thus, three outcome variables (y) are defined:

$$Positive\_Raw\_Return = \begin{cases} 1, & Annualised\ Log\ Return > 0 \\ 0, & Annualised\ Log\ Return \leq 0 \end{cases} \tag{1.2}$$

$$Positive\_Excess\_Return = \begin{cases} 1, & risk\ premium > 0 \\ 0, & risk\ premium \leq 0 \end{cases} \tag{1.3}$$

$$Tri\_Return = \begin{cases} 0, & Annualised\ Log\ Return < 0 \\ 1, & risk\ premium \leq 0 \\ 2, & risk\ premium > 0 \end{cases} \tag{1.4}$$

After all variables that need lags are defined, we can clean the data by filtering rows with "NA" and save it in a new table (*See Appendix A for data processing and cleaning code*). The descriptive statistics of core metrics are shown below:

	Mean	**Median**	**Min**	**Max**
Diesel Fuel Log Return	1.042	1.261	-90.136	132.478
Raw Log Return	0.61	0.761	-50.211	40.511
Excess Log Return	-0.697	-0.444	-51.701	38.141

2. Modelling

2.1 Exploring Functional Forms & Multicollinearity Concerns

When we want to construct a model that predicts the log return of S&P500 using the VIX as its covariates, we have to explore VIX non-linear transformations, such as quadratic and log transformation, its lagged values, and daily VIX change to find a model with the highest accuracy (*See Appendix B for combinations of VIX tested*). However, before proposing a model, we have to check the correlation of the coefficient between possible explanatory variables to prevent perfect multicollinearity from occurring. Multicollinearity happens when two or more independent variables in the model are highly correlated. This may inflate the standard errors and create unreliable significance tests (Siegel, 2016).

When testing models, it is important to note that linear relationships between covariates are acceptable to a certain degree. A benchmark for preventing multicollinearity is to avoid putting together variables with an absolute value of coefficient of correlation greater than 0.7 (Duda, 2022). We can see which variables possess multicollinearity using the correlation matrix below (*See Appendix A for the code*):

Correlation Matrix	VIX	VIX_1	VIX_2	VIXsq	VIX1sq	VIX2sq	diff_VIX	diff_VIX_2	diff_VIXsq	diff_VIX_2sq	logVIX	logVIX1	logVIX2	log_ratio_1	log_ratio_2
VIX	1	0.955	0.92	0.993	0.944	0.909	0.156	0.205	0.285	0.31	0.993	0.952	0.919	0.157	0.206
VIX_1	0.955	1	0.955	0.942	0.993	0.945	-0.143	0.005	0.145	0.198	0.954	0.993	0.952	-0.138	0.009
VIX_2	0.92	0.955	1	0.903	0.942	0.993	-0.111	-0.195	0.109	0.143	0.923	0.955	0.993	-0.111	-0.188
VIXsq	0.993	0.942	0.903	1	0.943	0.903	0.177	0.229	0.315	0.338	0.972	0.927	0.891	0.173	0.224
VIX1sq	0.944	0.993	0.942	0.943	1	0.944	-0.155	0.011	0.147	0.204	0.931	0.972	0.927	-0.145	0.015
VIX2sq	0.909	0.945	0.993	0.903	0.944	1	-0.115	-0.206	0.11	0.148	0.901	0.932	0.973	-0.112	-0.194
diff_VIX	0.156	-0.143	-0.111	0.177	-0.155	-0.115	1	0.669	0.47	0.375	0.136	-0.13	-0.106	0.985	0.659
diff_VIX_2	0.205	0.005	-0.195	0.229	0.011	-0.206	0.669	1	0.443	0.421	0.181	0	-0.182	0.671	0.987
diff_VIXsq	0.285	0.145	0.109	0.315	0.147	0.11	0.47	0.443	1	0.904	0.256	0.141	0.106	0.426	0.41
diff_VIX_2sq	0.31	0.198	0.143	0.338	0.204	0.148	0.375	0.421	0.904	1	0.282	0.191	0.136	0.337	0.397
logVIX	0.993	0.954	0.923	0.972	0.931	0.901	0.136	0.181	0.256	0.282	1	0.963	0.933	0.141	0.188
logVIX1	0.952	0.993	0.955	0.927	0.972	0.932	-0.13	0	0.141	0.191	0.963	1	0.963	-0.129	0.004
logVIX2	0.919	0.952	0.993	0.891	0.927	0.973	-0.106	-0.182	0.106	0.136	0.933	0.963	1	-0.108	-0.18
log_ratio_1	0.157	-0.138	-0.111	0.173	-0.145	-0.112	0.985	0.671	0.426	0.337	0.141	-0.129	-0.108	1	0.679
log_ratio_2	0.206	0.009	-0.188	0.224	0.015	-0.194	0.659	0.987	0.41	0.397	0.188	0.004	-0.18	0.679	1

2.2 Model Performance Metrics

To decide which models have the best predictive ability, we used three performance measurements, which are AIC, McFadden R^2, and Percent Correctly Predicted.

- **The Akaike Information Criterion (AIC**) is a statistical method that helps select a model that best explains the variance in the dependent variable with the fewest number of covariates (Manikantan, 2021). Lower AIC indicates a better fit as it helps to determine whether the cost of adding an extra explanatory variable is justified (Manikantan, 2021).
- **McFadden R^2** measures percentage improvement in the predicted probability of observing the independent variable when we use the information from explanatory variables. R^2 closer to 1 suggests a better fit. However it is important to note that it will always increase when more covariates are added to the model.
- **Percent Correctly Predicted or Hit Rate** measures the accuracy of the model. A higher value indicates higher predictive accuracy.

2.3 Binary Model

To find the best binary model, we tested two types of model, logit and probit model. Each model was assessed for its predictive capability on two binary models: Positive_Raw_Return and Positive_Excess_Return. Out of 296 models tested, we have found that the raw-return probit model with log of VIX and log ratio of VIX to its first lag has the best predictive ability (*See Appendix B for details of other models tested*). The model is written as:

$$\widehat{\Pr}(Positive\_Raw\_Return = 1) = \phi\left(\widehat{\beta_0} + \widehat{\beta_1}\, Log(VIX_t) + \widehat{\beta_2}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) \tag{2.1}$$

With Maximum Likelihood Estimation, we get:

$$\widehat{\Pr}(Positive\_Raw\_Return = 1) = \phi\left(\underset{(0.956)}{3.962} - \underset{(0.347)}{1.402}\, Log(VIX_t) - \underset{(3.155)}{49.008}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) \tag{2.2}$$

Based on the covariance matrix, the model inhibits small collinearity ($r = 0.141$), which does not indicate a high chance of multicollinearity. The rationale behind choosing this model is that it has the highest performance measurements in AIC, McFadden, and second best in Percent Correctly Predicted.

AIC	McFadden R^2	Percent Correctly Predicted
697.077	45.360%	81.216%

The model we proposed has a trade-off: Lower percent correctly predicted about 0.2% difference with the highest accuracy, but lowest AIC and highest R^2 among all 296 models (*see Appendix C*). As a result, additional research was done on different time periods with the same variable, and the proposed model has a higher hit rate on different datasets (*see Appendix D for details*). Thus, we tend to care more about AIC as the model with lower AIC generalises better and offers more robust insights across different datasets.

In addition, we can see from the Receiver Operating Characteristic curve, which shows the performance of a model at all thresholds (Google Developers, n.d.). We can see that the model that predicts the raw return has a larger area under the curve (AUC) than the model that predicts the sign of excess return.

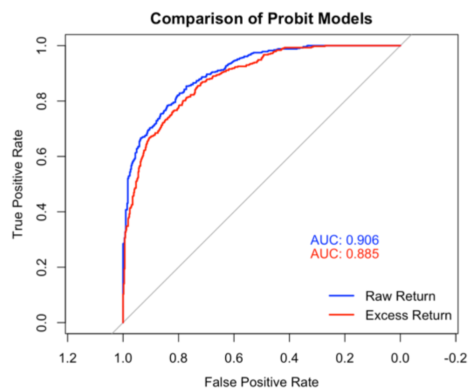

Figure 1: Comparison of Probit Models version 41, Output from R

As a result, we can deduce that, on average, model with raw return as its binary outcome has a higher predictive accuracy compared with model that has excess return as its outcome, holding all other covariates the same.

However, it has been found that the model with VIX's first lag as its first covariate has the same goodness of fit measurements as model 2.2:

$$\widehat{\Pr}(Positive\_Raw\_Return = 1) = \phi\left(\widehat{\beta_0} + \widehat{\beta_1}\, Log(VIX_{t-1}) + \widehat{\beta_2}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) \tag{2.3}$$

Thus, we have actually found two best models with the same predictive ability (*see Appendix E for details*). We suspect that it was due to log(VIX) and log (VIX_{t-1}) inhibiting almost perfect multicollinearity ($r = 0.96$). As a result, they behave very similarly, holding all other covariates the same. The reason why we picked the model with log(VIX) was due to its practical implications. When we want to predict market movement, having a variable that reflects the most immediate market condition is often more beneficial. Therefore, we use model 2.1 as the best model.

2.4 Trichotomous Model

To find the trichotomous model, ordered logit and probit models are tested. Out of 147 models tested, we have found that the ordered probit model with log of VIX and log ratio of VIX to its first lag also has the best predictive ability (*See Appendix F for details of other models tested*). The model is written as:

$$\widehat{\Pr}(Tri\_Return = 0|x_t) = \Phi\left(\tau_1 - \widehat{\beta_1}\, Log(VIX_t) - \widehat{\beta_2}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) \tag{2.4}$$

$$\widehat{\Pr}(Tri\_Return = 1|x_t) = \Phi\left(\tau_2 - \widehat{\beta_1}\, Log(VIX_t) - \widehat{\beta_2}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) - \Phi\left(\tau_1 - \widehat{\beta_1}\, Log(VIX_t) - \widehat{\beta_2}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) \tag{2.5}$$

$$\widehat{\Pr}(Tri\_Return = 2|x_t) = 1 - \Phi\left(\tau_2 - \widehat{\beta_0} - \widehat{\beta_1}\, Log(VIX_t) - \widehat{\beta_2}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) \tag{2.6}$$

Using Maximum Likelihood Estimation,

$$\widehat{\Pr}(Tri\_Return = 0|x_t) = \Phi\left(\underset{(0.838)}{-3.073} + \underset{(0.304)}{1.082} Log(VIX_t) + \underset{(2.561)}{44.925}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) \tag{2.7}$$

$$\begin{aligned}\widehat{\Pr}(Tri\_Return = 1|x_t) = {} & \Phi\left(\underset{(0.836)}{-2.657} + \underset{(0.304)}{1.082} Log(VIX_t) + \underset{(2.561)}{44.925}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) - \\ & \Phi\left(\underset{(0.838)}{-3.073} + \underset{(0.304)}{1.082} Log(VIX_t) + \underset{(2.561)}{44.925}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right)\end{aligned} \tag{2.8}$$

$$\widehat{\Pr}(Tri\_Return = 2|x_t) = 1 - \Phi\left(\underset{(0.836)}{-2.657} + \underset{(0.304)}{1.082} Log(VIX_t) + \underset{(2.561)}{44.925}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right) \tag{2.9}$$

The trichotomous model has the same trade-off: Better AIC and McFadden or better Percent Correctly Predicted. Since we favour AIC and McFadden more, the model performance metric of model 2.4 is shown below:

AIC	McFadden R^2	Percent Correctly Predicted
1124.899	35.386%	74.919%

The same problem that occurs in the binary model also happens; the model with log(VIX_{t-1}) as its first covariates has the exact goodness of fit with the model proposed above (*see Appendix E for details)*. Therefore, the same reasoning applies to this trichotomous model as well.

2.5 Trichotomous Modelling Concern

When we want to calculate the percent correctly predicted, there is a concern about modeling error as the model never predicts the second outcome, y = 1. Thus, the actual y frequency distribution is constructed to ensure it is reasonable. The confusion matrix and bar chart are shown below:

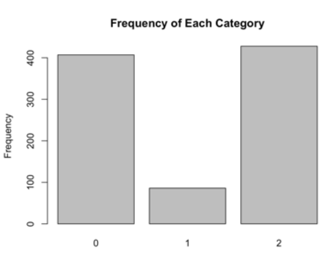

Figure 3: ROC curve of proposed model, Output from R

Matrix	**p=0**	**p=1**	**p=2**	**Row PCP**
y=0	338	0	69	83.05%
y=1	34	0	52	0
y=2	76	0	352	82.24%

Since the second outcome in the actual is significantly smaller than the other two outcomes, it would be reasonable for the model to predict incorrectly. However, this may imply that if the company decided to use this model to predict the return, it would rarely predict that the return would be positive but less than the risk-free rate, reducing accuracy. The main purpose of using the trichotomous model is to test whether the predictive ability is improved when we further divide the positive return. Therefore, we can deduce that we should continue to use the binary model instead, as it provides higher accuracy and simplicity.

2.6 Proposing Explanatory Variable

After exploring various models from binary and trichotomous, we can deduce that the raw return binary probit model has the best predictive ability. One might say that adding an explanatory variable may increase its ability. Thus, annualised log diesel fuel price return (*anlogp*) is used to test whether it helps the model. The intuition is that when fuel prices increase, demand for good movement increases. Higher demand can lead to higher business activities, revenue, and share prices. The proposed model became:

$$\widehat{\Pr}(Positive\_Raw\_Return = 1)_t = \phi\left(\underset{(0.964)}{3.884} - \underset{(0.350)}{1.378}\, Log(VIX_t) - \underset{(3.177)}{48.614}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right) + \underset{(0.003)}{0.008}\, anlogp\right) \quad (2.10)$$

We can see that it has better goodness of fit measurements and higher area under curve on ROC graph:

AIC	McFadden R^2	Percent Correctly Predicted
689.619	46.108%	82.302%

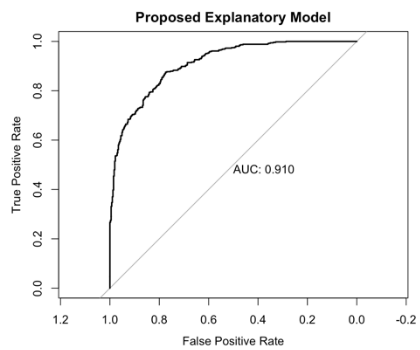

Figure 3: ROC curve of proposed model, Output from R

Moreover, the fuel price changes have a *p-value* of 0.002, which implies that the variable is significant under a 5% significance level, holding all else constant (*See Appendix G for the Wald Test*). Based on this result, we can infer that diesel fuel price helps the predictivity of the model.

We can deduce that the raw-return binary probit model on equation 2.10 has the best predictive ability among the 445 models explored. Directly interpreting each explanatory variable is not possible under the probit and logit model. However, we can interpret the sign. For the volatility index, this means that, on average, when the fear index increases, the probability of raw return being positive decreases, holding all else constant. The same goes for the VIX ratio; on average, when the ratio increases, the probability of a raw return positive will decrease, holding all else constant. However, for annual log diesel fuel return, on average, when the return increases, the probability of raw return positive increases, holding all else constant.

3. Conclusion

This report delves into various fear index models that could help predict signs of stock return. We have found that the binary probit model with raw return as its predicted outcome has the best predictive ability. It has also been proven that incorporating annualised log return on ultra-low-sulfur diesel fuel can further increase its predictive accuracy.

We can then answer the boss's uncertainty about which "returns" to refer to. According to the research done, the boss should refer to the return to raw return, which is the annualised log return without deducting risk-free rate. This is because the model has a higher accuracy in predicting whether the raw return is positive or negative rather than the excess. Furthermore, partitioning positive returns to return below the risk-free rate and above the risk-free rate will not lead to a more accurate prediction. This is based on the trichotomous model and has a lower predictive performance than the raw-return binary probit model. We believe further analysis of this model can be done for future research.

References

Duda, S. (2022, October 14). *Identifying and addressing multicollinearity in regression analysis*. Medium. https://scottmduda.medium.com/identifying-and-addressing-multicollinearity-in-regression-analysis-ca86a21a347e#:~:text=A%20quick%20way%20to%20identify

Federal Reserve Bank of St. Louis. (2024). *FRED Economic Data*. Stlouisfed.org; Federal Reserve Bank of St. Louis. https://fred.stlouisfed.org/

Google Developers. (n.d.). *Classification: ROC Curve and AUC | Machine Learning Crash Course*. Google Developers. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc#:~:text=An%20ROC%20curve%20(receiver%20operating

Manikantan, A. (2021, June 12). *Akaike Information Criterion: Model Selection*. Geek Culture. https://medium.com/geekculture/akaike-information-criterion-model-selection-c47df96ee9a8

Siegel, A. F. (2016). *Practical business statistics*. Elsevier. https://doi.org/10.1016/c2015-0-00463-4

Wikipedia. (2024, April 2). *Ultra-low-sulfur diesel*. Wikipedia. https://en.wikipedia.org/wiki/Ultra-low-sulfur_diesel

Appendix A
Data Cleaning & Correlation Matrix Code

The code for data cleaning is listed below. After data are cleaned, 921 observations remain.

```
#Clean Global Environment
rm(list=ls())

#Set Working Directory
setwd("/Users/bryanbastian/Desktop/Monash /ETF3600")

#Setup library
library(readxl)
library(dplyr)
library(tidyverse)
library(tangram)
library(lmtest)
library(sandwich)
library(car)
library(maxLik)
library(jtools)
library(pscl)
library(nnet)
library(fastDummies)
library(mlbench)
library(DescTools)
library(readxl)
library(olsrr)
library(ggpubr)
library(MASS)
library(pROC)
library(pscl)

#Load data needed
Stock_Price <- read.csv("SP500.csv", header = TRUE)
Volatility <- read.csv("VXVCLS.csv", header = TRUE)
Risk_Free <- read.csv("DTB3.csv", header = TRUE)
Extra <- read.csv("DDFUELNYH.csv", header = TRUE)

#Change the header name
colnames(Stock_Price) <- c("Date", "Price")
colnames(Volatility) <- c("Date", "VIX")
colnames(Risk_Free) <-c("Date", "Rf")
colnames(Extra) <- c("Date", "Ex")

#Merge the dataset
Stock <- merge(Stock_Price, Volatility, by.x ="Date", by.y = "Date")
Stock <- merge(Stock, Risk_Free, by.x = "Date", by.y = "Date")
Stock <- merge(Stock, Extra, by.x = "Date", by.y = "Date")

#Mark the missing value to NA
Stock[is.na(Stock) | Stock=="."] = NA
sum(is.na(Stock$Price))

#Convert DATE to Date class:
Stock[, 1] <- as.Date(Stock[, 1], format="%Y-%m-%d")

#Convert SP500, VIX, risk-free, and Diesel Fuel Price to numeric:
Stock[, 2] <- as.numeric(Stock[, 2])
Stock[, 3] <- as.numeric(Stock[, 3])
Stock[, 4] <- as.numeric(Stock[, 4])
Stock[, 5] <- as.numeric(Stock[, 5])
```

```
#Create time lag for VIX
Stock$VIX_t1 <- lag(Stock$VIX,1)
Stock$VIX_t2 <- lag(Stock$VIX,2)

#Create squared variables
Stock$VIXsq <- (Stock$VIX)^2
Stock$VIX1sq <- (Stock$VIX_t1)^2
Stock$VIX2sq <- (Stock$VIX_t2)^2

Stock$logVIX <- log(Stock$VIX)
Stock$logVIX1 <- log(Stock$VIX_t1)
Stock$logVIX2 <- log(Stock$VIX_t2)
Stock$logratio1 <- log(Stock$VIX/Stock$VIX_t1)
Stock$logratio2 <- log(Stock$VIX/Stock$VIX_t2)

#Defining annualised log returns, annualised log price rate, and excess return
Stock$Price_t1 <- lag(Stock$Price,1)
Stock$anlogr <- 1200*(log(Stock$Price)-log(Stock$Price_t1))
Stock$excess <- Stock$anlogr - Stock$Rf

Stock$Ex_t1 <- lag(Stock$Ex, 1)
Stock$anlogp <- 1200*(log(Stock$Ex)-log(Stock$Ex_t1))

#Defining binary models and trichotomous variable
Stock$rawreturn <- ifelse(Stock$anlogr>0,1,0)
Stock$excessreturn <- ifelse(Stock$excess>0,1,0)
Stock$tri <- ifelse(Stock$anlogr<0,0, ifelse(Stock$excess<=0, 1,
                                             ifelse(Stock$excess>0, 2, NA)))

#Filter the NA
tidy_Stock <- Stock
for (col in names(tidy_Stock)[-1]) {
  # Filter out rows where the column has NA values
  tidy_Stock <- tidy_Stock |> filter(!is.na(.data[[col]]))
}
remove(col)
summary(tidy_Stock)

#Find the difference in VIX

tidy_Stock$diff_VIX <- tidy_Stock$VIX - tidy_Stock$VIX_t1
tidy_Stock$diff_VIX_2 <- tidy_Stock$VIX - tidy_Stock$VIX_t2
tidy_Stock$diff_VIXsq <- tidy_Stock$diff_VIX^2
tidy_Stock$diff_VIX_2sq <- tidy_Stock$diff_VIX_2^2
```

The Correlation Matrix Code:

```
##### Checking correlation between explanatory variables####
cortable <- cor(tidy_Stock[, c("VIX", "VIX_t1", "VIX_t2", "VIXsq", "VIX1sq", "VIX2sq",
                               "diff_VIX", "diff_VIX_2", "diff_VIXsq", "diff_VIX_2sq",
                               "logVIX", "logVIX1", "logVIX2", "logratio1", "logratio2")])
cortable <- round(cortable,3)
print(covtable)
```

Short Description on VIX Transformations Notation:

- VIX_t1: The volatility index 1 previous working day.
- VIX_t2: The volatility index 2 previous working days.
- VIXsq: The quadratic form of VIX.
- VIX1sq: The quadratic form of VIX_t1.
- VIX2sq: The quadratic form of VIX_t2.
- logVIX: The logarithm form of VIX.
- logVIX1: The logarithm form of VIX_t1, we use this for correlation matrix purpose, as in the modelling we use log(VIX_t1).
- logVIX2: The logarithm form of VIX_t2, we use this for correlation matrix purpose, as in the modelling we use log(VIX_t2).
- logratio1: Today's volatility index relative to yesterday VIX in logarithm form, we use this for correlation matrix purpose, as in the modelling we use log(VIX/VIX_t1).
- logratio2: Today's volatility index relative to two previous working days VIX in logarithm form, we use this for correlation matrix purpose, as in the modelling we use log(VIX/VIX_t2)).
- diff_VIX: 1 day change in VIX.
- diff_VIX_2: 2 days change in VIX.
- diff_VIXsq: The quadratic form of diff_VIX.
- diff_VIX_2sq: The quadratic form of diff_VIX_2.

Appendix B
All Combination Tested

For All Models, the testing follows the same pattern all the time, it follows 74 combinations:

Model_v1: $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX$

Model_v2 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX^2$

Model_v3 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX + \widehat{\beta_2} VIX^2$

Model_v4 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log(VIX)$

Model_v5 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_1$

Model_v6 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_1^2$

Model_v7 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_1 + \widehat{\beta_2}\, VIX_1^2$

Model_v8 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log(VIX_1)$

Model_v9 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_2$

Model_v10 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_2^2$

Model_v11 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_2 + \widehat{\beta_2}\, VIX_2^2$

Model_v12 : $y = \widehat{\beta_0} + \widehat{\beta_1} log(VIX_2)$

Model_v13 : $y = \widehat{\beta_0} + \widehat{\beta_1}(VIX - VIX_1)$

Model_v14 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, (VIX - VIX_1)^2$

Model_v15 : $y = \widehat{\beta_0} + \widehat{\beta_1}(VIX - VIX_1) + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v16 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log\left(\frac{VIX}{VIX_1}\right)$

Model_v17 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, (VIX - VIX_2)$

Model_v18 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, (VIX - VIX_2)^2$

Model_v19 : $y = \widehat{\beta_0} + \widehat{\beta_1}(VIX - VIX_2) + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v20 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log\left(\frac{VIX}{VIX_2}\right)$

Model_v21 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX + \widehat{\beta_2}(VIX - VIX_1)$

Model_v22 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX^2 + \widehat{\beta_2}(VIX - VIX_1)$

Model_v23 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log(VIX) + \widehat{\beta_2}(VIX - VIX_1)$

Model_v24 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_1 + \widehat{\beta_2}(VIX - VIX_1)$

Model_v25 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX1^2 + \widehat{\beta_2}(VIX - VIX_1)$

Model_v26 : $y = \widehat{\beta_0} + \widehat{\beta_1}\log(VIX_1) + \widehat{\beta_2}(VIX - VIX_1)$

Model_v27 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_2 + \widehat{\beta_2}(VIX - VIX_1)$

Model_v28 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_2^2 + \widehat{\beta_2}(VIX - VIX_1)$

Model_v29 : $y = \widehat{\beta_0} + \widehat{\beta_1}\log(VIX_2) + \widehat{\beta_2}(VIX - VIX_1)$

Model_v30 :$y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v31 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX^2 + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v32 : $y = \widehat{\beta_0} + \widehat{\beta_1}\log(VIX) + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v33 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_1 + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v34 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_1^2 + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v35 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log(VIX_1) + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v36 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX_2 + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v37 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX_2^2 + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v38 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log(VIX_2) + \widehat{\beta_2}(VIX - VIX_1)^2$

Model_v39 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v40: $y = \widehat{\beta_0} + \widehat{\beta_1} VIX^2 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v41 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log(VIX) + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v42 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX\_1 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v43 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, VIX_1^2 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v44 : $y = \widehat{\beta_0} + \widehat{\beta_1}\, log(VIX_1\ ) + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v45 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ VIX_2 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v46 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ VIX_2^2 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v47 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ log(VIX_2) + \widehat{\beta_2} log\left(\frac{VIX}{VIX_1}\right)$

Model_v48 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX + \widehat{\beta_2}(VIX - VIX_2)$

Model_v49 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX^2 + \widehat{\beta_2}(VIX - VIX_2)$

Model_v50 : $y = \widehat{\beta_0} + \widehat{\beta_1} log(VIX) + \widehat{\beta_2}(VIX - VIX_2)$

Model_v51 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX_1 + \widehat{\beta_2}(VIX - VIX_2)$

Model_v52 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX_1^2 + \widehat{\beta_2}(VIX - VIX\_2\ )$

Model_v53 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ log(VIX_1) + \widehat{\beta_2}(VIX - VIX_2)$

Model_v54 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ VIX_2 + \widehat{\beta_2}(VIX - VIX_2)$

Model_v55 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ VIX_2^2 + \widehat{\beta_2}(VIX - VIX_2)$

Model_v56 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ log(VIX_2) + \widehat{\beta_2}(VIX - VIX_2)$

Model_v57 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v58 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX^2 + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v59 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ log(VIX) + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v60 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ VIX_1 + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v61 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ VIX_1^2 + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v62 : $y = \widehat{\beta_0} + \widehat{\beta_1} log(VIX_1) + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v63 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ VIX\_2 + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v64 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX_2^2 + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v65 : $y = \widehat{\beta_0} + \widehat{\beta_1} log(VIX_2) + \widehat{\beta_2}(VIX - VIX_2)^2$

Model_v66 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Model_v67 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIXsq + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Model_v68 : $y = \widehat{\beta_0} + \widehat{\beta_1} log(VIX) + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Model_v69 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX_1 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Model_v70 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX_1^2 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Model_v71 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ log(VIX_1) + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Model_v72 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ VIX_2 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Model_v73 : $y = \widehat{\beta_0} + \widehat{\beta_1} VIX_2^2 + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Model_v74 : $y = \widehat{\beta_0} + \widehat{\beta_1}\ log(VIX_2) + \widehat{\beta_2} log\left(\frac{VIX}{VIX_2}\right)$

Appendix C
Binary Model Code and its Performance Metric Results

```r
##### RawReturnLogit######

#Create empty dataframe to store goodness of fit
RawLogit <- data.frame(
  AIC = numeric(),
  McFadden = numeric(),
  HitRate = numeric())

RawLModels <- list(
  Rlogit_v1 = glm(rawreturn~VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v2 = glm(rawreturn~VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v3 = glm(rawreturn~VIX+VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v4 = glm(rawreturn~log(VIX), data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v5 = glm(rawreturn~VIX_t1, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v6 = glm(rawreturn~VIX1sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v7 = glm(rawreturn~VIX_t1+VIX1sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v8 = glm(rawreturn~log(VIX_t1), data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v9 = glm(rawreturn~VIX_t2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v10 = glm(rawreturn~VIX2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v11 = glm(rawreturn~VIX_t2+VIX2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v12 = glm(rawreturn~log(VIX_t2), data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v13 = glm(rawreturn~diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v14 = glm(rawreturn~diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v15 = glm(rawreturn~diff_VIX+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v16 = glm(rawreturn~log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v17 = glm(rawreturn~diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v18 = glm(rawreturn~diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v19 = glm(rawreturn~diff_VIX_2+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v20 = glm(rawreturn~log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v21 = glm(rawreturn~VIX+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v22 = glm(rawreturn~VIXsq+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v23 = glm(rawreturn~log(VIX)+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v24 = glm(rawreturn~VIX_t1+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v25 = glm(rawreturn~VIX1sq+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v26 = glm(rawreturn~log(VIX_t1)+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v27 = glm(rawreturn~VIX_t2+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v28 = glm(rawreturn~VIX2sq+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v29 = glm(rawreturn~log(VIX_t2)+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v30 = glm(rawreturn~VIX+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v31 = glm(rawreturn~VIXsq+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v32 = glm(rawreturn~log(VIX)+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v33 = glm(rawreturn~VIX_t1+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v34 = glm(rawreturn~VIX1sq+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v35 = glm(rawreturn~log(VIX_t1)+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v36 = glm(rawreturn~VIX_t2+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v37 = glm(rawreturn~VIX2sq+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v38 = glm(rawreturn~log(VIX_t2)+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v39 = glm(rawreturn~VIX+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v40 = glm(rawreturn~VIXsq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v41 = glm(rawreturn~log(VIX)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v42 = glm(rawreturn~VIX_t1+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v43 = glm(rawreturn~VIX1sq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v44 = glm(rawreturn~log(VIX_t1)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v45 = glm(rawreturn~VIX_t2+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v46 = glm(rawreturn~VIX2sq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v47 = glm(rawreturn~log(VIX_t2)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v48 = glm(rawreturn~VIX+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v49 = glm(rawreturn~VIXsq+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v50 = glm(rawreturn~log(VIX)+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v51 = glm(rawreturn~VIX_t1+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v52 = glm(rawreturn~VIX1sq+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v53 = glm(rawreturn~log(VIX_t1)+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v54 = glm(rawreturn~VIX_t2+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v55 = glm(rawreturn~VIX2sq+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v56 = glm(rawreturn~log(VIX_t2)+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v57 = glm(rawreturn~VIX+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v58 = glm(rawreturn~VIXsq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v59 = glm(rawreturn~log(VIX)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v60 = glm(rawreturn~VIX_t1+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v61 = glm(rawreturn~VIX1sq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v62 = glm(rawreturn~log(VIX_t1)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v63 = glm(rawreturn~VIX_t2+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v64 = glm(rawreturn~VIX2sq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v65 = glm(rawreturn~log(VIX_t2)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),

  Rlogit_v66 = glm(rawreturn~VIX+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v67 = glm(rawreturn~VIXsq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v68 = glm(rawreturn~log(VIX)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v69 = glm(rawreturn~VIX_t1+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v70 = glm(rawreturn~VIX1sq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v71 = glm(rawreturn~log(VIX_t1)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v72 = glm(rawreturn~VIX_t2+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v73 = glm(rawreturn~VIX2sq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Rlogit_v74 = glm(rawreturn~log(VIX_t2)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")))


for (i in 1:length(RawLModels)){
  model = RawLModels[[i]]
  model_AIS = summary(model)
  model_McFadden = pR2(model)
  model_hitmiss = hitmiss(model)


  RawLogit <- rbind(RawLogit, data.frame(
    AIC = round(model_AIS$aic,3),
    McFadden = round(model_McFadden[4]*100,3),
    HitRate = round(model_hitmiss[1],3)
  ), make.row.names = FALSE)
}

####Check the table###
View(RawLogit)
```

```r
##### RawReturnProbit#####
#Create Empty Data Frame
RawProbit <- data.frame(
  AIC = numeric(),
  McFadden = numeric(),
  HitRate = numeric())

RawPModels <- list(
  RProbit_v1 = glm(rawreturn~VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v2 = glm(rawreturn~VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v3 = glm(rawreturn~VIX+VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v4 = glm(rawreturn~log(VIX), data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v5 = glm(rawreturn~VIX_t1, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v6 = glm(rawreturn~VIX1sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v7 = glm(rawreturn~VIX_t1+VIX1sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v8 = glm(rawreturn~log(VIX_t1), data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v9 = glm(rawreturn~VIX_t2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v10 = glm(rawreturn~VIX2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v11 = glm(rawreturn~VIX_t2+VIX2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v12 = glm(rawreturn~log(VIX_t2), data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v13 = glm(rawreturn~diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v14 = glm(rawreturn~diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v15 = glm(rawreturn~diff_VIX+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v16 = glm(rawreturn~log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v17 = glm(rawreturn~diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v18 = glm(rawreturn~diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v19 = glm(rawreturn~diff_VIX_2+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v20 = glm(rawreturn~log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v21 = glm(rawreturn~VIX+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v22 = glm(rawreturn~VIXsq+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v23 = glm(rawreturn~log(VIX)+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v24 = glm(rawreturn~VIX_t1+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v25 = glm(rawreturn~VIX1sq+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v26 = glm(rawreturn~log(VIX_t1)+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v27 = glm(rawreturn~VIX_t2+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v28 = glm(rawreturn~VIX2sq+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v29 = glm(rawreturn~log(VIX_t2)+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v30 = glm(rawreturn~VIX+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v31 = glm(rawreturn~VIXsq+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v32 = glm(rawreturn~log(VIX)+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v33 = glm(rawreturn~VIX_t1+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v34 = glm(rawreturn~VIX1sq+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v35 = glm(rawreturn~log(VIX_t1)+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v36 = glm(rawreturn~VIX_t2+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v37 = glm(rawreturn~VIX2sq+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v38 = glm(rawreturn~log(VIX_t2)+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v39 = glm(rawreturn~VIX+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v40 = glm(rawreturn~VIXsq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v41 = glm(rawreturn~log(VIX)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v42 = glm(rawreturn~VIX_t1+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v43 = glm(rawreturn~VIX1sq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v44 = glm(rawreturn~log(VIX_t1)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v45 = glm(rawreturn~VIX_t2+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v46 = glm(rawreturn~VIX2sq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v47 = glm(rawreturn~log(VIX_t2)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v48 = glm(rawreturn~VIX+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v49 = glm(rawreturn~VIXsq+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v50 = glm(rawreturn~log(VIX)+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v51 = glm(rawreturn~VIX_t1+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v52 = glm(rawreturn~VIX1sq+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v53 = glm(rawreturn~log(VIX_t1)+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v54 = glm(rawreturn~VIX_t2+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v55 = glm(rawreturn~VIX2sq+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v56 = glm(rawreturn~log(VIX_t2)+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v57 = glm(rawreturn~VIX+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v58 = glm(rawreturn~VIXsq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v59 = glm(rawreturn~log(VIX)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v60 = glm(rawreturn~VIX_t1+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v61 = glm(rawreturn~VIX1sq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v62 = glm(rawreturn~log(VIX_t1)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v63 = glm(rawreturn~VIX_t2+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v64 = glm(rawreturn~VIX2sq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v65 = glm(rawreturn~log(VIX_t2)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),

  RProbit_v66 = glm(rawreturn~VIX+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v67 = glm(rawreturn~VIXsq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v68 = glm(rawreturn~log(VIX)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v69 = glm(rawreturn~VIX_t1+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v70 = glm(rawreturn~VIX1sq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v71 = glm(rawreturn~log(VIX_t1)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v72 = glm(rawreturn~VIX_t2+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v73 = glm(rawreturn~VIX2sq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  RProbit_v74 = glm(rawreturn~log(VIX_t2)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")))


for (i in 1:length(RawPModels)){
  model = RawPModels[[i]]
  model_AIS = summary(model)
  model_McFadden = pR2(model)
  model_hitmiss = hitmiss(model)


  RawProbit <- rbind(RawProbit, data.frame(
    AIC = round(model_AIS$aic,3),
    McFadden = round(model_McFadden[4]*100,3),
    HitRate = round(model_hitmiss[1],3)
  ), make.row.names = FALSE)
}

####Check the table and find the summary of the best goodness of fit###
View(RawProbit)
best_fit_raw_probit = glm(rawreturn~log(VIX)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit"))

summary(best_fit_raw_probit)
```

Raw Return Logit			
Version	AIC	McFadden (%)	HitRate (%)
1	1246.157	1.789	57.655
2	1246.247	1.782	56.460
3	1248.119	1.792	57.438
4	1246.410	1.769	57.980
5	1268.778	0.000	55.700
6	1268.758	0.002	55.700
7	1270.058	0.057	55.700
8	1268.777	0.000	55.700
9	1268.766	0.001	55.700
10	1268.771	0.001	55.700
11	1270.723	0.005	55.700
12	1268.752	0.002	55.700
13	726.683	42.861	81.107
14	1261.415	0.582	56.460
15	727.413	42.961	81.325
16	714.695	43.809	80.890
17	1090.314	14.110	72.204
18	1264.357	0.350	56.678
19	1090.373	14.264	72.096
20	1074.044	15.397	71.336
21	709.691	44.363	81.433
22	709.613	44.369	81.325
23	710.086	44.331	81.433
24	709.691	44.363	81.433
25	709.101	44.409	81.325
26	710.562	44.294	81.433
27	711.676	44.206	80.999
28	710.789	44.276	80.782
29	712.596	44.133	81.542
30	1247.141	1.869	56.352
31	1247.426	1.847	56.895
32	1247.127	1.870	56.569
33	1262.640	0.644	56.895
34	1262.378	0.664	57.329
35	1262.886	0.624	56.678
36	1263.080	0.609	56.569
37	1263.034	0.612	56.678
38	1263.161	0.602	56.569
39	700.013	45.128	81.216
40	700.412	45.096	81.216
41	699.940	45.134	81.216
42	699.898	45.137	81.216
43	700.191	45.114	80.999
44	699.940	45.134	81.216
45	701.542	45.007	80.673

Raw Return Probit			
Version	AIC	McFadden (%)	HitRate (%)
1	1246.106	1.793	57.655
2	1246.184	1.787	56.46
3	1248.062	1.796	57.438
4	1246.376	1.771	57.98
5	1268.778	0	55.7
6	1268.758	0.002	55.7
7	1270.055	0.057	55.7
8	1268.777	0	55.7
9	1268.766	0.001	55.7
10	1268.771	0.001	55.7
11	1270.723	0.005	55.7
12	1268.751	0.002	55.7
13	728.168	42.744	81.107
14	1261.406	0.583	56.352
15	727.352	42.966	81.325
16	712.266	44.001	80.999
17	1102.82	13.122	71.444
18	1264.4	0.346	56.46
19	1102.34	13.318	72.204
20	1084.063	14.605	72.096
21	710.01	44.337	81.433
22	710.25	44.318	81.216
23	710.161	44.325	81.433
24	710.01	44.337	81.433
25	709.592	44.37	81.107
26	710.778	44.277	81.433
27	711.853	44.192	80.999
28	710.974	44.261	80.782
29	712.842	44.113	80.999
30	1247.068	1.875	56.352
31	1247.348	1.853	57.112
32	1247.064	1.875	56.46
33	1262.624	0.645	56.895
34	1262.357	0.666	57.329
35	1262.874	0.625	56.678
36	1263.087	0.608	56.678
37	1263.044	0.612	56.678
38	1263.165	0.602	56.569
39	697.268	45.345	81.325
40	697.81	45.302	81.107
41	697.077	45.360	81.216
42	697.1	45.358	81.325
43	697.476	45.328	80.999
44	697.077	45.360	81.216
45	698.885	45.217	80.782

46	701.533	45.008	80.456
47	701.666	44.997	80.890
48	1084.227	14.750	70.793
49	1085.168	14.675	71.444
50	1083.453	14.811	70.793
51	1092.270	14.114	71.987
52	1092.111	14.127	72.313
53	1092.314	14.110	72.204
54	1084.227	14.750	70.793
55	1083.710	14.791	71.118
56	1084.741	14.709	70.901
57	1247.990	1.802	57.329
58	1248.161	1.788	56.678
59	1248.115	1.792	57.655
60	1265.727	0.400	56.569
61	1265.482	0.419	56.786
62	1265.951	0.382	56.678
63	1266.158	0.365	56.569
64	1266.112	0.369	56.678
65	1266.223	0.360	56.569
66	1068.490	15.994	72.421
67	1069.213	15.937	72.204
68	1067.908	16.040	72.313
69	1076.014	15.399	71.770
70	1075.892	15.409	71.770
71	1076.043	15.397	71.336
72	1067.658	16.060	72.530
73	1067.447	16.077	72.421
74	1067.908	16.040	72.313

46	698.878	45.217	80.456
47	699.030	45.206	80.782
48	1096.843	13.752	69.381
49	1097.830	13.674	70.033
50	1095.984	13.820	70.141
51	1104.649	13.135	71.987
52	1104.347	13.159	71.987
53	1104.793	13.124	71.770
54	1096.843	13.752	69.381
55	1096.023	13.817	69.381
56	1097.585	13.694	69.815
57	1247.917	1.808	57.220
58	1248.081	1.795	56.569
59	1248.054	1.797	57.655
60	1265.788	0.395	56.569
61	1265.540	0.414	56.678
62	1266.011	0.377	56.352
63	1266.225	0.360	56.569
64	1266.184	0.363	56.569
65	1266.284	0.355	56.569
66	1078.992	15.164	71.661
67	1079.730	15.105	71.878
68	1078.356	15.214	71.878
69	1085.854	14.621	71.553
70	1085.552	14.645	71.553
71	1086.016	14.608	71.878
72	1077.998	15.242	71.444
73	1077.642	15.271	71.553
74	1078.356	15.214	71.878

For Excess Return Binary Model

```r
##### ExcessReturnLogit#####
#For Excess Logit
ExLogit <- data.frame(
  AIC = numeric(),
  McFadden = numeric(),
  HitRate = numeric())

ExLModels <- list(
  Elogit_v1 = glm(excessreturn~VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v2 = glm(excessreturn~VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v3 = glm(excessreturn~VIX+VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v4 = glm(excessreturn~log(VIX), data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v5 = glm(excessreturn~VIX_t1, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v6 = glm(excessreturn~VIX1sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v7 = glm(excessreturn~VIX_t1+VIX1sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v8 = glm(excessreturn~log(VIX_t1), data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v9 = glm(excessreturn~VIX_t2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v10 = glm(excessreturn~VIX2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v11 = glm(excessreturn~VIX_t2+VIX2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v12 = glm(excessreturn~log(VIX_t2), data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v13 = glm(excessreturn~diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v14 = glm(excessreturn~diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v15 = glm(excessreturn~diff_VIX+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v16 = glm(excessreturn~log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v17 = glm(excessreturn~diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v18 = glm(excessreturn~diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v19 = glm(excessreturn~diff_VIX_2+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v20 = glm(excessreturn~log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v21 = glm(excessreturn~VIX+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v22 = glm(excessreturn~VIXsq+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v23 = glm(excessreturn~log(VIX)+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v24 = glm(excessreturn~VIX_t1+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v25 = glm(excessreturn~VIX1sq+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v26 = glm(excessreturn~log(VIX_t1)+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v27 = glm(excessreturn~VIX_t2+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v28 = glm(excessreturn~VIX2sq+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v29 = glm(excessreturn~log(VIX_t2)+diff_VIX, data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v30 = glm(excessreturn~VIX+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v31 = glm(excessreturn~VIXsq+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v32 = glm(excessreturn~log(VIX)+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v33 = glm(excessreturn~VIX_t1+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v34 = glm(excessreturn~VIX1sq+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v35 = glm(excessreturn~log(VIX_t1)+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v36 = glm(excessreturn~VIX_t2+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v37 = glm(excessreturn~VIX2sq+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v38 = glm(excessreturn~log(VIX_t2)+diff_VIXsq, data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v39 = glm(excessreturn~VIX+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v40 = glm(excessreturn~VIXsq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v41 = glm(excessreturn~log(VIX)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v42 = glm(excessreturn~VIX_t1+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v43 = glm(excessreturn~VIX1sq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v44 = glm(excessreturn~log(VIX_t1)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v45 = glm(excessreturn~VIX_t2+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v46 = glm(excessreturn~VIX2sq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v47 = glm(excessreturn~log(VIX_t2)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v48 = glm(excessreturn~VIX+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v49 = glm(excessreturn~VIXsq+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v50 = glm(excessreturn~log(VIX)+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v51 = glm(excessreturn~VIX_t1+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v52 = glm(excessreturn~VIX1sq+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v53 = glm(excessreturn~log(VIX_t1)+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v54 = glm(excessreturn~VIX_t2+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v55 = glm(excessreturn~VIX2sq+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v56 = glm(excessreturn~log(VIX_t2)+diff_VIX_2, data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v57 = glm(excessreturn~VIX+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v58 = glm(excessreturn~VIXsq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v59 = glm(excessreturn~log(VIX)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v60 = glm(excessreturn~VIX_t1+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v61 = glm(excessreturn~VIX1sq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v62 = glm(excessreturn~log(VIX_t1)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v63 = glm(excessreturn~VIX_t2+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v64 = glm(excessreturn~VIX2sq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v65 = glm(excessreturn~log(VIX_t2)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="logit")),

  Elogit_v66 = glm(excessreturn~VIX+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v67 = glm(excessreturn~VIXsq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v68 = glm(excessreturn~log(VIX)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v69 = glm(excessreturn~VIX_t1+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v70 = glm(excessreturn~VIX1sq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v71 = glm(excessreturn~log(VIX_t1)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v72 = glm(excessreturn~VIX_t2+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v73 = glm(excessreturn~VIX2sq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")),
  Elogit_v74 = glm(excessreturn~log(VIX_t2)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="logit")))

for (i in 1:length(ExLModels)){
  model = ExLModels[[i]]
  model_AIS = summary(model)
  model_McFadden = pR2(model)
  model_hitmiss = hitmiss(model)


  ExLogit <- rbind(ExLogit, data.frame(
    AIC = round(model_AIS$aic,3),
    McFadden = round(model_McFadden[4]*100,3),
    HitRate = round(model_hitmiss[1],3)
  ), make.row.names = FALSE)
}

View(ExLogit)
```

```r
##### ExcessReturnProbit######
#For Excess Probit
ExProbit <- data.frame(
  AIC = numeric(),
  McFadden = numeric(),
  HitRate = numeric())

ExPModels <- list(
  EProbit_v1 = glm(excessreturn~VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v2 = glm(excessreturn~VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v3 = glm(excessreturn~VIX+VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v4 = glm(excessreturn~log(VIX), data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v5 = glm(excessreturn~VIX_t1, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v6 = glm(excessreturn~VIX1sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v7 = glm(excessreturn~VIX_t1+VIX1sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v8 = glm(excessreturn~log(VIX_t1), data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v9 = glm(excessreturn~VIX_t2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v10 = glm(excessreturn~VIX2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v11 = glm(excessreturn~VIX_t2+VIX2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v12 = glm(excessreturn~log(VIX_t2), data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v13 = glm(excessreturn~diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v14 = glm(excessreturn~diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v15 = glm(excessreturn~diff_VIX+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v16 = glm(excessreturn~log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v17 = glm(excessreturn~diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v18 = glm(excessreturn~diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v19 = glm(excessreturn~diff_VIX_2+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v20 = glm(excessreturn~log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v21 = glm(excessreturn~VIX+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v22 = glm(excessreturn~VIXsq+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v23 = glm(excessreturn~log(VIX)+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v24 = glm(excessreturn~VIX_t1+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v25 = glm(excessreturn~VIX1sq+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v26 = glm(excessreturn~log(VIX_t1)+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v27 = glm(excessreturn~VIX_t2+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v28 = glm(excessreturn~VIX2sq+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v29 = glm(excessreturn~log(VIX_t2)+diff_VIX, data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v30 = glm(excessreturn~VIX+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v31 = glm(excessreturn~VIXsq+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v32 = glm(excessreturn~log(VIX)+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v33 = glm(excessreturn~VIX_t1+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v34 = glm(excessreturn~VIX1sq+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v35 = glm(excessreturn~log(VIX_t1)+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v36 = glm(excessreturn~VIX_t2+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v37 = glm(excessreturn~VIX2sq+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v38 = glm(excessreturn~log(VIX_t2)+diff_VIXsq, data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v39 = glm(excessreturn~VIX+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v40 = glm(excessreturn~VIXsq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v41 = glm(excessreturn~log(VIX)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v42 = glm(excessreturn~VIX_t1+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v43 = glm(excessreturn~VIX1sq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v44 = glm(excessreturn~log(VIX_t1)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v45 = glm(excessreturn~VIX_t2+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v46 = glm(excessreturn~VIX2sq+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v47 = glm(excessreturn~log(VIX_t2)+log(VIX/VIX_t1), data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v48 = glm(excessreturn~VIX+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v49 = glm(excessreturn~VIXsq+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v50 = glm(excessreturn~log(VIX)+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v51 = glm(excessreturn~VIX_t1+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v52 = glm(excessreturn~VIX1sq+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v53 = glm(excessreturn~log(VIX_t1)+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v54 = glm(excessreturn~VIX_t2+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v55 = glm(excessreturn~VIX2sq+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v56 = glm(excessreturn~log(VIX_t2)+diff_VIX_2, data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v57 = glm(excessreturn~VIX+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v58 = glm(excessreturn~VIXsq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v59 = glm(excessreturn~log(VIX)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v60 = glm(excessreturn~VIX_t1+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v61 = glm(excessreturn~VIX1sq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v62 = glm(excessreturn~log(VIX_t1)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v63 = glm(excessreturn~VIX_t2+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v64 = glm(excessreturn~VIX2sq+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v65 = glm(excessreturn~log(VIX_t2)+diff_VIX_2sq, data=tidy_Stock, family=binomial(link="probit")),

  EProbit_v66 = glm(excessreturn~VIX+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v67 = glm(excessreturn~VIXsq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v68 = glm(excessreturn~log(VIX)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v69 = glm(excessreturn~VIX_t1+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v70 = glm(excessreturn~VIX1sq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v71 = glm(excessreturn~log(VIX_t1)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v72 = glm(excessreturn~VIX_t2+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v73 = glm(excessreturn~VIX2sq+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")),
  EProbit_v74 = glm(excessreturn~log(VIX_t2)+log(VIX/VIX_t2), data=tidy_Stock, family=binomial(link="probit")))


for (i in 1:length(ExPModels)){
  model = ExPModels[[i]]
  model_AIS = summary(model)
  model_McFadden = pR2(model)
  model_hitmiss = hitmiss(model)


  ExProbit <- rbind(ExProbit, data.frame(
    AIC = round(model_AIS$aic,3),
    McFadden = round(model_McFadden[4]*100,3),
    HitRate = round(model_hitmiss[1],3)
  ), make.row.names = FALSE)
}

#View Table
View(ExProbit)
```

Excess Return Logit			
Version	AIC	McFadden (%)	HitRate (%)
1	1268.520	0.603	54.072
2	1268.218	0.626	54.072
3	1270.132	0.633	54.180
4	1268.885	0.574	54.072
5	1272.838	0.263	54.832
6	1272.608	0.281	55.049
7	1274.442	0.294	54.397
8	1273.162	0.238	54.940
9	1273.619	0.202	54.289
10	1273.644	0.200	54.072
11	1275.618	0.202	54.289
12	1273.723	0.194	54.289
13	788.552	38.330	79.262
14	1274.426	0.138	53.529
15	787.950	38.535	79.587
16	780.354	38.975	79.262
17	1116.255	12.571	67.752
18	1275.219	0.076	53.529
19	1115.199	12.812	67.210
20	1102.867	13.624	68.187
21	777.601	39.348	78.936
22	777.658	39.344	79.153
23	777.795	39.333	78.827
24	777.601	39.348	78.936
25	777.420	39.363	79.153
26	778.026	39.315	78.827
27	779.485	39.200	78.610
28	779.124	39.229	79.045
29	779.956	39.163	78.610
30	1270.288	0.621	53.963
31	1270.067	0.638	53.963
32	1270.554	0.600	54.072
33	1271.590	0.518	53.637
34	1271.229	0.547	53.855
35	1272.083	0.480	53.529
36	1272.854	0.419	54.397
37	1272.862	0.418	53.855
38	1273.022	0.406	54.072
39	774.918	39.559	79.262
40	775.188	39.538	79.370
41	774.801	39.568	78.827
42	774.979	39.555	79.262
43	775.310	39.529	79.370
44	774.801	39.568	78.827
45	776.172	39.461	78.936

Excess Return Probit			
Version	AIC	McFadden (%)	HitRate (%)
1	1268.498	0.604	54.072
2	1268.189	0.629	54.072
3	1270.099	0.636	54.180
4	1268.868	0.575	53.963
5	1272.839	0.263	54.832
6	1272.608	0.281	55.049
7	1274.438	0.295	54.397
8	1273.165	0.237	54.940
9	1273.621	0.202	54.289
10	1273.645	0.200	54.072
11	1275.620	0.202	54.289
12	1273.725	0.193	54.289
13	788.865	38.306	79.262
14	1274.386	0.141	53.529
15	787.205	38.593	79.587
16	779.271	39.060	79.262
17	1127.451	11.691	67.644
18	1275.187	0.0790	53.529
19	1124.290	12.097	66.884
20	1112.429	12.872	68.187
21	778.088	39.310	78.719
22	778.288	39.294	79.045
23	778.149	39.305	78.719
24	778.088	39.310	78.719
25	777.989	39.318	79.153
26	778.441	39.282	78.719
27	780.005	39.159	78.610
28	779.718	39.182	78.936
29	780.421	39.127	78.610
30	1270.239	0.625	53.963
31	1270.015	0.642	53.963
32	1270.506	0.604	54.072
33	1271.536	0.523	53.637
34	1271.171	0.551	53.855
35	1272.034	0.484	53.529
36	1272.830	0.421	54.397
37	1272.840	0.42	53.963
38	1272.996	0.408	54.072
39	774.010	39.631	79.262
40	774.341	39.605	79.370
41	773.828	39.645	78.827
42	774.059	39.627	79.262
43	774.441	39.597	79.370
44	773.828	39.645	78.827
45	775.328	39.527	78.936

46	776.305	39.450	79.045
47	776.090	39.467	78.827
48	1116.185	12.734	69.055
49	1116.528	12.707	69.055
50	1115.880	12.758	69.598
51	1116.076	12.743	67.101
52	1115.397	12.796	67.644
53	1116.649	12.698	67.101
54	1116.185	12.734	69.055
55	1115.891	12.757	69.055
56	1116.442	12.714	69.381
57	1270.512	0.603	54.072
58	1270.218	0.626	54.072
59	1270.848	0.577	54.072
60	1272.521	0.445	54.506
61	1272.149	0.475	54.832
62	1273.021	0.406	54.289
63	1273.868	0.339	53.746
64	1273.856	0.340	53.855
65	1274.038	0.326	53.963
66	1103.498	13.731	69.381
67	1103.695	13.716	69.381
68	1103.316	13.746	69.707
69	1102.234	13.831	67.535
70	1101.573	13.883	67.427
71	1102.830	13.784	67.318
72	1103.209	13.754	69.490
73	1103.086	13.764	69.490
74	1103.316	13.746	69.707

46	775.502	39.513	78.827
47	775.204	39.537	78.936
48	1128.000	11.805	68.947
49	1128.296	11.782	68.838
50	1127.708	11.828	69.381
51	1125.846	11.975	66.884
52	1124.870	12.051	66.992
53	1126.686	11.909	66.667
54	1128.000	11.805	68.947
55	1127.621	11.835	68.838
56	1128.295	11.782	69.055
57	1270.485	0.605	54.072
58	1270.189	0.629	54.072
59	1270.824	0.579	53.963
60	1272.502	0.447	54.506
61	1272.129	0.476	54.723
62	1273.000	0.408	54.397
63	1273.850	0.341	53.637
64	1273.840	0.342	53.855
65	1274.017	0.328	53.746
66	1113.520	12.944	69.164
67	1113.687	12.930	69.055
68	1113.349	12.957	69.490
69	1110.270	13.199	67.644
70	1109.304	13.275	67.644
71	1111.142	13.130	67.427
72	1113.220	12.967	69.490
73	1113.053	12.980	69.490
74	1113.349	12.957	69.490

Appendix D
Predictive Performance Measurement from Different Time Period

Additional Dataset was collected between 1/1/2020 to 31/12/2023, to test which model performs better in a different dataset. The code for the calculation is exactly the same, just the dataset imported have different time period.

For our best model (Better AIC and McFadden R^2, but lower Hit Rate):

$$\widehat{\Pr}(Positive\_Raw\_Return = 1)_t = \phi\left(\widehat{\beta_0} + \widehat{\beta_1}\, Log(VIX_t) + \widehat{\beta_2}\, Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right)$$

The goodness of fit measurements:

AIC	McFadden R^2	Percent Correctly Predicted
790.804	38.312%	80.911%

Compared with the model with highest Hit Rate (model 21) on our dataset:

$$\widehat{\Pr}(Positive\_Raw\_Return = 1)_t = \phi\left(\widehat{\beta_0} + \widehat{\beta_1}\, VIX_t + \widehat{\beta_2}\,(VIX - VIX_{t-1}\,)\right)$$

AIC	McFadden R^2	Percent Correctly Predicted
876.145	31.604%	79.93%

As a result, we used the first model with highest AIC and McFadden as it generalises more across different datasets.

Appendix E
Comparison of Two Highest Accuracy Probit Binary and Probit Trichotomous

For Binary Probit Model:

Model:

$$\widehat{\Pr}(Positive\_Raw\_Return = 1)_t = \phi\left(3.962 - 1.402\ Log(VIX_t) - 49.008\ Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right)$$

AIC: 697.077
McFadden R^2: 45.360%
Hit-Miss Table:

	$y = 0$	$y = 1$
$\hat{y} = 0$	315	80
$\hat{y} = 1$	93	433

Percent Correctly Predicted = 81.216%
Percent Correctly Predicted = 77.206%, for y = 0
Percent Correctly Predicted = 84.405%, for y = 1

Model:

$$\widehat{\Pr}(Positive\_Raw\_Return = 1)_t = \phi\left(3.962 - 1.402\ Log(VIX_{t-1}) - 50.410\ Log\left(\frac{VIX_t}{VIX_{t-1}}\right)\right)$$

AIC: 697.077
McFadden R^2: 45.360%
Hit-Miss Table:

	$y = 0$	$y = 1$
$\hat{y} = 0$	315	80
$\hat{y} = 1$	93	433

Percent Correctly Predicted = 81.216%
Percent Correctly Predicted = 77.206%, for y = 0
Percent Correctly Predicted = 84.405%, for y = 1

For Trichotomous Model:
For clarity purpose, the full model will not be listed:

$$y \sim log(VIX_t) + log\left(\frac{VIX_t}{VIX_{t-1}}\right)$$

AIC: 1124.899
McFadden R^2: 35.386%
Hit-Miss Table:

	$p = 1$	$p = 2$	$p = 3$
$y = 1$	338	0	69
$y = 2$	34	0	52
$y = 3$	76	0	352

Percent Correctly Predicted, Fitted Model: 74.92%
Percent Correctly Predicted, Null Model : 46.47%

For clarity purpose, the full model will not be listed:

$$y \sim log(VIX_{t-1}) + log\left(\frac{VIX_t}{VIX_{t-1}}\right)$$

AIC: 1124.899
McFadden R^2: 35.386%
Hit-Miss Table:

	$p = 1$	$p = 2$	$p = 3$
$y = 1$	338	0	69
$y = 2$	34	0	52
$y = 3$	76	0	352

Percent Correctly Predicted, Fitted Model: 74.92%
Percent Correctly Predicted, Null Model : 46.47%

Appendix F
Trichotomous Model Code and its Performance Metric Results

```r
##### TriLogit#####
####Ordered Logit

tri = as.factor(tri)
levels(tri)

TriLogit <- data.frame(
  AIC = numeric(),
  McFadden = numeric(),
  HitMiss = numeric())

TriLModels <- list(
  TriLogit_v1 = polr(tri~VIX, method ="logistic"),
  TriLogit_v2 = polr(tri~VIXsq, method ="logistic"),
  TriLogit_v3 = polr(tri~VIX+VIXsq, method ="logistic"),
  TriLogit_v4 = polr(tri~log(VIX), method ="logistic"),

  TriLogit_v5 = polr(tri~VIX_t1, method ="logistic"),
  TriLogit_v6 = polr(tri~VIX1sq, method ="logistic"),
  TriLogit_v7 = polr(tri~VIX_t1+VIX1sq, method ="logistic"),
  TriLogit_v8 = polr(tri~log(VIX_t1), method ="logistic"),

  TriLogit_v9 = polr(tri~VIX_t2, method ="logistic"),
  TriLogit_v10 = polr(tri~VIX2sq, method ="logistic"),
  TriLogit_v11 = polr(tri~VIX_t2+VIX2sq, method ="logistic"),
  TriLogit_v12 = polr(tri~log(VIX_t2), method ="logistic"),

  TriLogit_v13 = polr(tri~diff_VIX, method ="logistic"),
  TriLogit_v14 = polr(tri~diff_VIXsq, method ="logistic"),
  TriLogit_v15 = polr(tri~diff_VIX+diff_VIXsq, method ="logistic"),
  TriLogit_v16 = polr(tri~log(VIX/VIX_t1), method ="logistic"),

  TriLogit_v17 = polr(tri~diff_VIX_2, method ="logistic"),
  TriLogit_v18 = polr(tri~diff_VIX_2sq, method ="logistic"),
  TriLogit_v19 = polr(tri~diff_VIX_2+diff_VIX_2sq, method ="logistic"),
  TriLogit_v20 = polr(tri~log(VIX/VIX_t2), method ="logistic"),

  TriLogit_v21 = polr(tri~VIX+diff_VIX, method ="logistic"),
  TriLogit_v22 = polr(tri~VIXsq+diff_VIX, method ="logistic"),
  TriLogit_v23 = polr(tri~log(VIX)+diff_VIX, method ="logistic"),
  TriLogit_v24 = polr(tri~VIX_t1+diff_VIX, method ="logistic"),
  TriLogit_v25 = polr(tri~VIX1sq+diff_VIX, method ="logistic"),
  TriLogit_v26 = polr(tri~log(VIX_t1)+diff_VIX, method ="logistic"),
  TriLogit_v27 = polr(tri~VIX_t2+diff_VIX, method ="logistic"),
  TriLogit_v28 = polr(tri~VIX2sq+diff_VIX, method ="logistic"),
  TriLogit_v29 = polr(tri~log(VIX_t2)+diff_VIX, method ="logistic"),

  TriLogit_v30 = polr(tri~VIX+diff_VIXsq, method ="logistic"),
  TriLogit_v31 = polr(tri~VIXsq+diff_VIXsq, method ="logistic"),
  TriLogit_v32 = polr(tri~log(VIX)+diff_VIXsq, method ="logistic"),
  TriLogit_v33 = polr(tri~VIX_t1+diff_VIXsq, method ="logistic"),
  TriLogit_v34 = polr(tri~VIX1sq+diff_VIXsq, method ="logistic"),
  TriLogit_v35 = polr(tri~log(VIX_t1)+diff_VIXsq, method ="logistic"),
  TriLogit_v36 = polr(tri~VIX_t2+diff_VIXsq, method ="logistic"),
  TriLogit_v37 = polr(tri~VIX2sq+diff_VIXsq, method ="logistic"),
  TriLogit_v38 = polr(tri~log(VIX_t2)+diff_VIXsq, method ="logistic"),
  TriLogit_v39 = polr(tri~VIX+log(VIX/VIX_t1), method ="logistic"),
  TriLogit_v40 = polr(tri~VIXsq+log(VIX/VIX_t1), method ="logistic"),
  TriLogit_v41 = polr(tri~log(VIX)+log(VIX/VIX_t1), method ="logistic"),
  TriLogit_v42 = polr(tri~VIX_t1+log(VIX/VIX_t1), method ="logistic"),
  TriLogit_v43 = polr(tri~VIX1sq+log(VIX/VIX_t1), method ="logistic"),
  TriLogit_v44 = polr(tri~log(VIX_t1)+log(VIX/VIX_t1), method ="logistic"),
  TriLogit_v45 = polr(tri~VIX_t2+log(VIX/VIX_t1), method ="logistic"),
  TriLogit_v46 = polr(tri~VIX2sq+log(VIX/VIX_t1), method ="logistic"),
  TriLogit_v47 = polr(tri~log(VIX_t2)+log(VIX/VIX_t1), method ="logistic"),

  TriLogit_v48 = polr(tri~VIX+diff_VIX_2, method ="logistic"),
  TriLogit_v49 = polr(tri~VIXsq+diff_VIX_2, method ="logistic"),
  TriLogit_v50 = polr(tri~log(VIX)+diff_VIX_2, method ="logistic"),
  TriLogit_v51 = polr(tri~VIX_t1+diff_VIX_2, method ="logistic"),
  TriLogit_v52 = polr(tri~VIX1sq+diff_VIX_2, method ="logistic"),
  TriLogit_v53 = polr(tri~log(VIX_t1)+diff_VIX_2, method ="logistic"),
  TriLogit_v54 = polr(tri~VIX_t2+diff_VIX_2, method ="logistic"),
  TriLogit_v55 = polr(tri~VIX2sq+diff_VIX_2, method ="logistic"),
  TriLogit_v56 = polr(tri~log(VIX_t2)+diff_VIX_2, method ="logistic"),

  TriLogit_v57 = polr(tri~VIX+diff_VIX_2sq, method ="logistic"),
  TriLogit_v58 = polr(tri~VIXsq+diff_VIX_2sq, method ="logistic"),
  TriLogit_v59 = polr(tri~log(VIX)+diff_VIX_2sq, method ="logistic"),
  TriLogit_v60 = polr(tri~VIX_t1+diff_VIX_2sq, method ="logistic"),
  TriLogit_v61 = polr(tri~VIX1sq+diff_VIX_2sq, method ="logistic"),
  TriLogit_v62 = polr(tri~log(VIX_t1)+diff_VIX_2sq, method ="logistic"),
  TriLogit_v63 = polr(tri~VIX_t2+diff_VIX_2sq, method ="logistic"),
  TriLogit_v64 = polr(tri~VIX2sq+diff_VIX_2sq, method ="logistic"),
  TriLogit_v65 = polr(tri~log(VIX_t2)+diff_VIX_2sq, method ="logistic"),

  TriLogit_v66 = polr(tri~VIX+log(VIX/VIX_t2), method ="logistic"),
  TriLogit_v67 = polr(tri~VIXsq+log(VIX/VIX_t2), method ="logistic"),
  TriLogit_v68 = polr(tri~log(VIX)+log(VIX/VIX_t2), method ="logistic"),
  TriLogit_v69 = polr(tri~VIX_t1+log(VIX/VIX_t2), method ="logistic"),
  TriLogit_v70 = polr(tri~VIX1sq+log(VIX/VIX_t2), method ="logistic"),
  TriLogit_v71 = polr(tri~log(VIX_t1)+log(VIX/VIX_t2), method ="logistic"),
  TriLogit_v72 = polr(tri~VIX_t2+log(VIX/VIX_t2), method ="logistic"),
  TriLogit_v73 = polr(tri~VIX2sq+log(VIX/VIX_t2), method ="logistic"),
  TriLogit_v74 = polr(tri~log(VIX_t2)+log(VIX/VIX_t2), method ="logistic"))

for (i in 1:length(TriLModels)){
  model = TriLModels[[i]]
  model_AIC = AIC(model)
  model_McFadden = pR2(model)[4]
  predicted_classes <- predict(model, tidy_Stock, type = "class")
  actual_classes <- tidy_Stock$tri
  correct_predictions <- predicted_classes == actual_classes
  accuracy <- sum(correct_predictions) / length(correct_predictions) * 100

  TriLogit <- rbind(TriLogit, data.frame(
    AIC = round(model_AIC,3),
    McFadden = round(model_McFadden*100,3),
    HitMiss = round(accuracy,3)
  ), make.row.names = FALSE)
}
View(TriLogit)
```

```r
##### TriProbit######
#Ordered Probit
TriProbit <- data.frame(
  AIC = numeric(),
  McFadden = numeric(),
  HitMiss = numeric())

TriPModels <- list(
  TriProbit_v1 = polr(tri~VIX, method ="probit"),
  TriProbit_v2 = polr(tri~VIXsq, method ="probit"),
  TriProbit_v3 = polr(tri~VIX+VIXsq, method ="probit"),
  TriProbit_v4 = polr(tri~log(VIX), method ="probit"),

  TriProbit_v5 = polr(tri~VIX_t1, method ="probit"),
  TriProbit_v6 = polr(tri~VIX1sq, method ="probit"),
  TriProbit_v7 = polr(tri~VIX_t1+VIX1sq, method ="probit"),
  TriProbit_v8 = polr(tri~log(VIX_t1), method ="probit"),

  TriProbit_v9 = polr(tri~VIX_t2, method ="probit"),
  TriProbit_v10 = polr(tri~VIX2sq, method ="probit"),
  TriProbit_v11 = polr(tri~VIX_t2+VIX2sq, method ="probit"),
  TriProbit_v12 = polr(tri~log(VIX_t2), method ="probit"),

  TriProbit_v13 = polr(tri~diff_VIX, method ="probit"),
  TriProbit_v14 = polr(tri~diff_VIXsq, method ="probit"),
  TriProbit_v15 = polr(tri~diff_VIX+diff_VIXsq, method ="probit"),
  TriProbit_v16 = polr(tri~log(VIX/VIX_t1), method ="probit"),

  TriProbit_v17 = polr(tri~diff_VIX_2, method ="probit"),
  TriProbit_v18 = polr(tri~diff_VIX_2sq, method ="probit"),
  TriProbit_v19 = polr(tri~log(VIX/VIX_t2), method ="probit"),

  TriProbit_v20 = polr(tri~VIX+diff_VIX, method ="probit"),
  TriProbit_v21 = polr(tri~VIXsq+diff_VIX, method ="probit"),
  TriProbit_v22 = polr(tri~log(VIX)+diff_VIX, method ="probit"),
  TriProbit_v23 = polr(tri~VIX_t1+diff_VIX, method ="probit"),
  TriProbit_v24 = polr(tri~VIX1sq+diff_VIX, method ="probit"),
  TriProbit_v25 = polr(tri~log(VIX_t1)+diff_VIX, method ="probit"),
  TriProbit_v26 = polr(tri~VIX_t2+diff_VIX, method ="probit"),
  TriProbit_v27 = polr(tri~VIX2sq+diff_VIX, method ="probit"),
  TriProbit_v28 = polr(tri~log(VIX_t2)+diff_VIX, method ="probit"),

  TriProbit_v29 = polr(tri~VIX+diff_VIXsq, method ="probit"),
  TriProbit_v30 = polr(tri~VIXsq+diff_VIXsq, method ="probit"),
  TriProbit_v31 = polr(tri~log(VIX)+diff_VIXsq, method ="probit"),
  TriProbit_v32 = polr(tri~VIX_t1+diff_VIXsq, method ="probit"),
  TriProbit_v33 = polr(tri~VIX1sq+diff_VIXsq, method ="probit"),
  TriProbit_v34= polr(tri~log(VIX_t1)+diff_VIXsq, method ="probit"),
  TriProbit_v35 = polr(tri~VIX_t2+diff_VIXsq, method ="probit"),
  TriProbit_v36 = polr(tri~VIX2sq+diff_VIXsq, method ="probit"),
  TriProbit_v37 = polr(tri~log(VIX_t2)+diff_VIXsq, method ="probit"),

  TriProbit_v38 = polr(tri~VIX+log(VIX/VIX_t1), method ="probit"),
  TriProbit_v39 = polr(tri~VIXsq+log(VIX/VIX_t1), method ="probit"),
  TriProbit_v42 = polr(tri~VIX1sq+log(VIX/VIX_t1), method ="probit"),
  TriProbit_v43 = polr(tri~log(VIX_t1)+log(VIX/VIX_t1), method ="probit"),
  TriProbit_v44 = polr(tri~VIX_t2+log(VIX/VIX_t1), method ="probit"),
  TriProbit_v45 = polr(tri~VIX2sq+log(VIX/VIX_t1), method ="probit"),
  TriProbit_v46 = polr(tri~log(VIX_t2)+log(VIX/VIX_t1), method ="probit"),

  TriProbit_v47 = polr(tri~VIX+diff_VIX_2, method ="probit"),
  TriProbit_v48 = polr(tri~VIXsq+diff_VIX_2, method ="probit"),
  TriProbit_v49 = polr(tri~log(VIX)+diff_VIX_2, method ="probit"),
  TriProbit_v50 = polr(tri~VIX_t1+diff_VIX_2, method ="probit"),
  TriProbit_v51 = polr(tri~VIX1sq+diff_VIX_2, method ="probit"),
  TriProbit_v52 = polr(tri~log(VIX_t1)+diff_VIX_2, method ="probit"),
  TriProbit_v53 = polr(tri~VIX_t2+diff_VIX_2, method ="probit"),
  TriProbit_v54 = polr(tri~VIX2sq+diff_VIX_2, method ="probit"),
  TriProbit_v55 = polr(tri~log(VIX_t2)+diff_VIX_2, method ="probit"),

  TriProbit_v56 = polr(tri~VIX+diff_VIX_2sq, method ="probit"),
  TriProbit_v57 = polr(tri~VIXsq+diff_VIX_2sq, method ="probit"),
  TriProbit_v58 = polr(tri~log(VIX)+diff_VIX_2sq, method ="probit"),
  TriProbit_v59 = polr(tri~VIX_t1+diff_VIX_2sq, method ="probit"),
  TriProbit_v60 = polr(tri~VIX1sq+diff_VIX_2sq, method ="probit"),
  TriProbit_v61 = polr(tri~log(VIX_t1)+diff_VIX_2sq, method ="probit"),
  TriProbit_v62 = polr(tri~VIX_t2+diff_VIX_2sq, method ="probit"),
  TriProbit_v63 = polr(tri~VIX2sq+diff_VIX_2sq, method ="probit"),
  TriProbit_v64 = polr(tri~log(VIX_t2)+diff_VIX_2sq, method ="probit"),

  TriProbit_v65 = polr(tri~VIX+log(VIX/VIX_t2), method ="probit"),
  TriProbit_v66 = polr(tri~VIXsq+log(VIX/VIX_t2), method ="probit"),
  TriProbit_v67 = polr(tri~log(VIX)+log(VIX/VIX_t2), method ="probit"),
  TriProbit_v68 = polr(tri~VIX_t1+log(VIX/VIX_t2), method ="probit"),
  TriProbit_v69 = polr(tri~VIX1sq+log(VIX/VIX_t2), method ="probit"),
  TriProbit_v70 = polr(tri~log(VIX_t1)+log(VIX/VIX_t2), method ="probit"),
  TriProbit_v71 = polr(tri~VIX_t2+log(VIX/VIX_t2), method ="probit"),
  TriProbit_v72 = polr(tri~VIX2sq+log(VIX/VIX_t2), method ="probit"),
  TriProbit_v73 = polr(tri~log(VIX_t2)+log(VIX/VIX_t2), method ="probit"))

for (i in 1:length(TriPModels)){
  model = TriPModels[[i]]
  model_AIC = AIC(model)
  model_McFadden = pR2(model)[4]
  predicted_classes <- predict(model, tidy_Stock, type = "class")
  actual_classes <- tidy_Stock$tri
  correct_predictions <- predicted_classes == actual_classes
  accuracy <- sum(correct_predictions) / length(correct_predictions) * 100

  TriProbit <- rbind(TriProbit, data.frame(
    AIC = round(model_AIC,3),
    McFadden = round(model_McFadden*100,3),
    HitMiss = round(accuracy,3)
  ), make.row.names = FALSE)
}

View(TriProbit)
best_fit_tri_probit <- polr(tri~log(VIX)+log(VIX/VIX_t1), method ="probit")
summary(best_fit_tri_probit)
pR2(best_fit_tri_probit)
hitmiss(best_fit_tri_probit)
```

Trichotomous Model

For Trichotomous Probit Model, model 19 is unable to converge, so the probit model has 73 tests.

Trichotomous Logit Model			
Version	AIC	McFadden (%)	HitMiss (%)
1	1718.758	0.915	50.923
2	1718.414	0.934	50.489
3	1720.397	0.935	50.489
4	1719.314	0.882	49.729
5	1733.576	0.057	45.711
6	1733.387	0.068	45.820
7	1734.901	0.096	46.471
8	1733.780	0.045	45.928
9	1733.953	0.035	45.928
10	1733.932	0.037	45.494
11	1735.927	0.037	45.603
12	1734.025	0.031	45.928
13	1149.253	33.861	74.810
14	1730.043	0.262	47.340
15	1147.663	34.069	75.027
16	1138.723	34.470	74.919
17	1541.092	11.193	64.604
18	1731.928	0.153	47.448
19	1541.032	11.312	64.169
20	1524.647	12.144	64.495
21	1134.297	34.842	75.136
22	1134.415	34.835	74.919
23	1134.508	34.830	75.136
24	1134.297	34.842	75.136
25	1134.030	34.858	74.919
26	1134.867	34.809	75.136
27	1136.845	34.695	75.027
28	1136.344	34.724	74.810
29	1137.428	34.661	75.136
30	1720.094	0.953	51.031
31	1719.927	0.963	50.271
32	1720.428	0.934	49.946
33	1729.416	0.414	47.557
34	1729.003	0.438	47.991
35	1729.871	0.387	47.448
36	1730.349	0.360	47.666
37	1730.281	0.364	48.100
38	1730.525	0.349	47.557
39	1128.729	35.164	74.701
40	1129.112	35.142	74.484
41	1128.604	35.171	75.027
42	1128.768	35.162	74.701
43	1129.191	35.138	74.484
44	1128.604	35.171	75.027
45	1130.577	35.057	74.484

Trichotomous Probit Model			
Version	AIC	McFadden (%)	HitMiss (%)
1	1718.782	0.913	50.923
2	1718.489	0.930	50.380
3	1720.483	0.930	50.271
4	1719.284	0.884	49.186
5	1733.605	0.056	45.494
6	1733.423	0.066	45.711
7	1734.961	0.093	46.471
8	1733.802	0.044	45.928
9	1733.974	0.034	45.385
10	1733.956	0.035	45.603
11	1735.954	0.035	45.603
12	1734.042	0.030	45.928
13	1149.57	33.843	74.810
14	1730.080	0.260	47.231
15	1146.215	34.153	75.027
16	1135.751	34.642	74.919
17	1556.648	10.293	64.604
18	1731.944	0.152	47.557
20	1537.854	11.380	64.604
21	1133.583	34.883	75.136
22	1133.969	34.861	74.701
23	1133.576	34.884	74.919
24	1133.583	34.883	75.136
25	1133.456	34.891	74.701
26	1134.064	34.856	75.027
27	1135.980	34.745	75.136
28	1135.487	34.773	74.593
29	1136.607	34.708	75.027
30	1720.102	0.952	50.706
31	1719.981	0.959	49.946
32	1720.390	0.936	50.163
33	1729.468	0.411	47.666
34	1729.065	0.434	48.100
35	1729.917	0.385	47.448
36	1730.439	0.354	47.448
37	1730.384	0.358	48.317
38	1730.601	0.345	47.340
39	1125.141	35.372	74.701
40	1125.655	35.342	74.484
41	1124.899	35.386	74.919
42	1125.144	35.372	74.701
43	1125.660	35.342	74.593
44	1124.899	35.386	74.919
45	1127.031	35.262	74.593
46	1127.203	35.253	74.810

46	1130.712	35.050	75.027
47	1130.516	35.061	74.593
48	1537.892	11.494	65.147
49	1538.595	11.453	64.712
50	1537.304	11.528	65.038
51	1542.338	11.236	64.712
52	1541.825	11.266	64.712
53	1542.700	11.215	64.712
54	1537.892	11.494	65.147
55	1537.479	11.517	64.929
56	1538.290	11.471	65.038
57	1720.668	0.920	50.706
58	1720.383	0.936	50.380
59	1721.126	0.893	49.186
60	1731.507	0.293	48.100
61	1731.095	0.317	47.883
62	1731.955	0.267	47.448
63	1732.496	0.235	47.666
64	1732.414	0.240	47.557
65	1732.664	0.226	47.666
66	1522.296	12.396	64.929
67	1522.776	12.368	64.929
68	1521.897	12.419	64.821
69	1525.781	12.194	64.929
70	1525.293	12.222	65.038
71	1526.151	12.173	64.712
72	1521.729	12.429	64.929
73	1521.574	12.438	64.821
74	1521.897	12.419	64.821

47	1126.951	35.267	74.593
48	1554.059	10.558	64.387
49	1554.740	10.519	64.495
50	1553.443	10.594	64.169
51	1557.200	10.377	64.387
52	1556.443	10.420	64.712
53	1557.766	10.344	64.929
54	1554.059	10.558	64.387
55	1553.411	10.596	64.495
56	1554.613	10.526	64.495
57	1720.683	0.919	50.706
58	1720.449	0.932	50.271
59	1721.089	0.895	49.294
60	1731.613	0.287	48.100
61	1731.216	0.310	47.991
62	1732.046	0.262	47.340
63	1732.605	0.229	47.991
64	1732.537	0.233	48.208
65	1732.756	0.220	47.774
66	1536.137	11.595	64.712
67	1536.599	11.568	64.712
68	1535.718	11.619	64.712
69	1538.161	11.478	64.712
70	1537.406	11.522	64.929
71	1538.757	11.444	64.929
72	1535.468	11.634	64.604
73	1535.190	11.650	64.712
74	1535.718	11.619	64.712

Appendix G
Significance Test on the Proposed Explanatory Variable

Using the asymptotic normality property of Maximum Likelihood Estimation, Wald Test is conducted to test whether Ultra-Low-Sulfur Diesel Price Change is a significant predictor of return at 5% significance level.

Step 1: Define Hypotheses

$H_0: \beta_{\text{anlogp}} = 0$

$H_A: \beta_{\text{anlogp}} \neq 0$

Step 2: Determine the test statistics

The z-statistic:

$$z - statistic = \frac{\hat{\beta}_{anlogp}}{\mathrm{se}(\hat{\beta}_{anlogp})} \overset{a}{\sim} N(0, 1)\ under\ H_0$$

Step 3: Significance level of test

$\alpha = 0.05$

Step 4: Decision Rule

Reject H_0 if $|z| > z_{\left(\frac{0.05}{2}\right)} = 1.96$ or if p-value < 0.05

Step 5: The value of test statistics

$$z - statistic = \frac{0.0085}{0.00276} = 3.08$$

$$p - value = 0.002$$

Step 6: Conclusion

Since $3.08 > 1.96$ and $0.002 < 0.05$, we reject H_0.

We can reject H_0 at 5% significance level. The sample does provide enough evidence against H_0. That is, on average, the annualised log Ultra-Low-Sulfur Diesel Return is a significant predictor on the probability of positive raw return, holding all else constant.

The Code is attached below:

```
> ##### Considering Explanatory Variable#####
> RProbit_proposed = glm(rawreturn~log(VIX)+log(VIX/VIX_t1)+anlogp, data=tidy_Stock, family=binomial(link="probit"))
Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred
> summary(RProbit_proposed)

Call:
glm(formula = rawreturn ~ log(VIX) + log(VIX/VIX_t1) + anlogp,
    family = binomial(link = "probit"), data = tidy_Stock)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6370  -0.5675   0.0551   0.6059   2.6442

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)       3.884829   0.963543   4.032 5.53e-05 ***
log(VIX)         -1.378300   0.349742  -3.941 8.12e-05 ***
log(VIX/VIX_t1) -48.614026   3.177380 -15.300  < 2e-16 ***
anlogp            0.008499   0.002755   3.086  0.00203 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1264.78  on 920  degrees of freedom
Residual deviance:  681.62  on 917  degrees of freedom
AIC: 689.62

Number of Fisher Scoring iterations: 7
```

In case needed, for the full version of the R-script and Output in csv, please refer to: https://github.com/BryanSambada/ETX3600_Project