NZ ESG PROJECT

Code Document

AUT – Dr. Prasad Hegde

Table of Contents

##### Variable creation – SAS Code1

1. [DECOMPOSED\_STK\_RET, CAPITAL\_INVESTED\_PER\_HOLDING, ESG flag](#variables_1)
2. [CALCULATING VALUE WEIGHTED ESG FUND RETURN](#variables_2)
3. [CALCULATING VALUE WEIGHTED NON ESG FUND RETURN](#variables_3)
4. [CALCULATING VALUE WEIGHTED ESG FUND RETURN DECOMPOSED](#variables_4)
5. [CALCULATING VALUE WEIGHTED NON ESG FUND RETURN](#variables_5)
6. [CALCULATING SUM\_ESG\_CAPITAL OF EACH FUND PER MONTH AND YEAR](#variables_6)
7. [CALCULATING TOTAL WEIGHT OF ESG HOLDINGS PER FUND](#variables_7)
8. [CALCULATING ESG TILT FOR THE FUND – inclusive](#variables_8)
9. [ESG TILT EXCLUSIVE](#variables_9)
10. [CALCULATION OF ESG FUND FLOWS FOR FUND PORTFOLIO](#variables_10)
11. [WINSORIZATION DONE - ESG\_FLOW\_PERCENT\_NZD](#variables_11)
12. [CALCULATION OF OVERALL FUND FLOWS FOR FUND PORTFOLIO](#variables_12)
13. [WINSORIZATION DONE - FLOW\_PERCENT\_NZD](#variables_13)
14. [CALCULATION OF VOLATILITY OF FUNDRETURNS](#variables_14)
15. [CALCULATING TRACKING ERROR](#variables_15)
16. [CREATING ALPHA GENERATION FLAG](#variables_16)
17. [CALCULATING TIME AND PICKING VARIABLES -TIMING\_PER\_STOCK\_MED, PICKING\_PER\_STOCK\_MED](#variables_17)
18. [Fund level Timing and Picking vars - SUM\_TIMING\_MED, SUM\_PICKING\_MED](#variables_18)
19. [CREATING WINSORIZED TIMING AND PICKING VARAIABLES](#variables_19)
20. [CREATING DATASET FROM ABOVE DATASET FOR (A) ESG STOCKS, (B) TOTAL STOCKS AND (C) TOTAL HOLDINGS IN A FUND PORTFOLIO](#variables_20)
21. [CALCULATING A) RATIO OF ESG STOCKS TO TOTAL STOCKS B) RATIO OF ESG STOCKS TO TOTAL HOLDINGS OR COMPONENTS HOLDINGS](#variables_21)
22. [COUNTING NON NZ STOCKS ON BASIS OF ISO CODE](#variables_22)
23. [NON\_NZ\_STOCK\_RATIO](#variables_23)
24. [COUNTING NO OF EQUITIES IN EACH FUND MONTH YEAR WHICH HAVE ESGSCORE >= 75](#variables_24)
25. [RATIO\_ESG75\_HLDSEQ\_NZD](#variables_25)
26. [ESGGE75\_CAP\_INV\_NZD, ESGGE75\_SUM\_WT\_NZD, ESGGE75\_VW\_RETURNS\_NZD](#variables_26)
27. [ESG\_SCORE\_AVG](#variables_27)
28. [ESG\_CONTRO\_AVG](#variables_28)
29. [CREATING HIGH LOW FLAG FOR EPU](#variables_29)
30. [TO CALCULATE NO OF FUNDS PER MONTH AND YEAR](#variables_30)
31. [INACTIVE FUND](#variables_31)
32. [ESG\_MINUS\_NON\_ESG\_T and ESG\_MINUS\_NON\_ESG\_P variables](#variables_32) – WE ARE NOT USING THIS CODE IN FINAL CALCULATIONS
33. [ESGGE75\_MINUS\_OTHR\_T and ESGGE75\_MINUS\_OTHR\_P variables](#variables_33)
34. [CALCULATING FORWARD ALPHAS](#variables_34)
35. [CREATING FLAG FOR HIGH AND LOW TIMIING AND HIGH AND LOW PICKING FUNDS](#variables_35)
36. [AVERAGE CH4 VARIABLES IN HIGH AND LOW PICKING ENVIRONMENT](#variables_36)
37. [ESG FUNDS](#variables_37)
38. [CREATING EXTREME VARIABLE - WHICH IS 1 WHEN THE NZX50 IS BELOW -5% OR ABOVE +5% MONTHLY RETURNS, OTHERWISE ZERO](#variables_38)
39. [ESG TRADING VARIABLE](#variables_39)
40. [CREATING RANKING FOR EACH FUND FOR TIMING VARIABLES – {QUINTILE\_5\_BOTH, QUINTILE\_1\_BOTH}](#variables_40)
41. [ESG FLOWS FOR GREATER THAN 75 – fund flows calculations](#variables_47)
42. [VALUE WEIGHTED ESG FUND RETURN – {VW\_HIGH\_ESG\_RETURNS } – where ESG score is GE 75](#variables_48)
43. [ESG\_MINUS\_NON\_ESG\_P](#variables_43)
44. [PICKING\_STKS\_Q1\_SUM, PICKING\_STKS\_Q1\_WIN](#variables_44)
45. [PICKING\_STKS\_Q5\_SUM, PICKING\_STKS\_Q5\_WIN](#variables_45)
46. [PICKING\_Q5\_MINUS\_Q1](#variables_46)
47. [ESG\_MINUS\_NON\_ESG\_TIM](#variables_47)
48. [TIMING\_STKS\_Q1\_SUM, TIMING\_STKS\_Q1\_WIN](#variables_48)
49. [TIMING\_STKS\_Q5\_SUM, TIMING\_STKS\_Q5\_WIN](#variables_49)
50. [TIMING\_Q5\_MINUS\_Q1](#variables_50)
51. [Q5\_FUNDS - definition](#variables_41)
52. [Q1\_FUNDS - definition](#_52)__Q1_FUNDS)

##### Datasets used in regression calculations2

1. [Definition of kiwisaver and open end funds](#_53)_Definition_of)
2. [TOP\_QUINTILE\_PICKING – dataset creation](#_54)_TOP_QUINTILE_PICKING_–)
3. [TOP\_QUINTILE\_TIMING – dataset creation](#_55)_TOP_QUINTILE_TIMING_–)
4. [BOTTOM\_QUINTILE\_PICKING – dataset creation](#_56)_BOTTOM_QUINTILE_PICKING_–)
5. [BOTTOM\_QUINTILE\_TIMING – dataset creation](#_57)_BOTTOM_QUINTILE_TIMING_–)
6. [ESG DATASET CREATION](#_58)_ESG_DATASET)
7. [test\_fundlevel\_20\_nov\_win\_set – main fundlevel dataset](#_59)_\“test_fundlevel_20_nov_win_set\”)

##### Macros created and used in the code3

1. **MACRO** [MERGE\_CODE\_1](#variables_60)
2. **MACRO** [IMPORT\_MACROS – for CSV datset](#variables_61)
3. **MACRO** [IMPORT\_MACRO – for stata dataset](#variables_62)
4. **MACRO** [PROC\_SUMMARY\_OVERALL – for overall funds](#variables_63)
5. **MACRO** [PROC\_SUMMARY\_KIWISAVER – for kiwisaver funds](#variables_64)
6. **MACRO** [PROC\_SUMMARY\_KIWIPENSION - for kiwipension funds](#variables_65)
7. **MACRO** [PROC\_SUMMARY\_ESG1 - for ESG rated holdings](#variables_66)
8. **MACRO** [PROC\_SUMMARY\_WEIGHT](#variables_67)
9. **MACRO** [PROC\_SUMMARY\_OPENEND for open end funds](#variables_68)
10. **MACRO** [CREATE\_FORWARD\_VAR – to create forward variables in the dataset](#variables_69)
11. **MACRO** [PROC\_SUMMARY\_HIGH\_PICKING](#variables_70)
12. **MACRO** [PROC\_SUMMARY\_LOW\_PICKING](#variables_71)
13. **MACRO** [PROC\_SUMMARY\_HIGH\_TIMING](#variables_72)
14. **MACRO** [PROC\_SUMMARY\_LOW\_TIMING](#variables_73)
15. **MACRO** [MERGE\_CODE\_2 - MERGE MACRO IF A OR B](#_73)_MERGE_CODE_2_)
16. **MACRO** [PICKING\_VARS\_ALPHA - CREATING MACRO TO CREATE AVERAGE CH4 VARIABLES IN HIGH AND LOW PICKING ENVIRONMENT](#variables_74)
17. **MACRO** [TIMING\_VARS\_ALPHA](#variables_75)
18. **MACRO** [TTEST\_MACRO](#variables_76) – macro to run ttest on multiple variables and get output in a dataset
19. **MACRO** [TTEST\_FUND\_STYLES\_FLAG\_EPU](#variables_77) – macro to run ttest on multiple variables when fund styles and EPU flag is taken in where condition

##### Code Variables created in Feb 20264

1. [Picking from >75 Score – ‘PICKING\_GE\_75’](#variables_78)
2. [Picking from <25 Score – ‘PICKING\_LE\_25’](#variables_79)
3. [ESG returns (Equal-weighted) – ‘EW\_ESG\_RETURNS’](#variables_80)

Stata code5

1. [Regression Equation for CH4\_alpha and related beta variables generation](#variables_81)
2. [Generating ch4\_alpha, size\_beta, momentum\_beta, market\_beta, value\_beta](#variables_82)
3. [Regression Equation for CAPM alpha beta variable generation](#variables_83)
4. [Regression Equation for generating Idiosyncratic volatility](#variables_85)
5. [Regression Equation for generating Value at risk variable](#variables_86)
6. [Regression to generate beta which is used in timing and picking calculations](#variables_87)
7. Very important to consider for any regression - [Generalized stata code to be used before running any regression](#variables_88)

### DECOMPOSED\_STK\_RET, CAPITAL\_INVESTED\_PER\_HOLDING, ESG flag

Intituion :- the code calculates a) decomposed stock return of each holding by multiplying fundreturns with holdings weight, b) capital invested per holding by the fund in the given portfolio and c) ESG flag which is 1 if ESG score is present for the holding else 0.

*Line no 827 in the codefile*

**DATA** HOLDINGS\_ESG\_FUND;

SET SD\_ALL\_DATA\_12;

STOCK\_WEIGHT\_DECIMAL = Percentholdings/**100**;

DECOMPOSED\_STK\_RET = STOCK\_WEIGHT\_DECIMAL \* Fundreturns\_NZD1;

/\*FUND RETURNS IS IN PERCENTAGE \*/

CAPITAL\_INVESTED\_PER\_HOLDING =

STOCK\_WEIGHT\_DECIMAL \* Netassets\_NZD;

IF ESGSCORE NE **.** THEN ESG = **1**;

ELSE ESG = **0**;

YY\_MON = INTNX('MONTH', MDY(MONTH, **1**, YEAR), **0**);

FORMAT YY\_MON YYMON7.; /\* FORMAT THE NEW DATE VARIABLE \*/

**RUN**;

### CALCULATING VALUE WEIGHTED ESG FUND RETURN

Intitution :- The code computes value-weighted ESG returns for the fund portfolio for a specified year and month by using the monthly returns of ESG holdings, with the capital invested in each holding serving as the weight.

*Line no 1002 in the codefile*

**PROC** **SUMMARY** DATA = NZD\_DATASET\_FINAL NWAY MISSING;

WHERE ESG = **1** AND MONTHLYRETURNS NE **.** ;

CLASS FUNDID YEAR MONTH;

WEIGHT CAPITAL\_INVESTED\_PER\_HOLDING;

VAR MONTHLYRETURNS;

OUTPUT OUT = ESG\_RETURNS\_VW MEAN (MONTHLYRETURNS) = VW\_ESG\_RETURNS;

**RUN**;

### CALCULATING VALUE WEIGHTED NON ESG FUND RETURN

Intitution :- The code calculates value-weighted returns for holdings without an ESG score in the fund portfolio for a given year and month, using the monthly returns and weighting them based on the capital invested in each holding.

*Line no 1011 in the codefile*

**PROC** **SUMMARY** DATA = NZD\_DATASET\_FINAL NWAY MISSING ;

WHERE ESG = **0** AND MONTHLYRETURNS NE **.** ;

CLASS FUNDID YEAR MONTH;

WEIGHT CAPITAL\_INVESTED\_PER\_HOLDING;

VAR MONTHLYRETURNS;

OUTPUT OUT = NON\_ESG\_RETURNS\_VW MEAN (MONTHLYRETURNS) = VW\_NON\_ESG\_RETURNS;

**RUN**;

### CALCULATING VALUE WEIGHTED ESG FUND RETURN DECOMPOSED

Intitution :- The code computes value-weighted ESG returns for the fund portfolio for a specified year and month by using the decomposed stock returns, with the capital invested in each holding serving as the weight.

*Line no 1036 in the codefile*

**PROC** **SUMMARY** DATA = NZD\_DATASET\_FINAL NWAY MISSING;

WHERE ESG = **1** ;

CLASS FUNDID YEAR MONTH;

WEIGHT CAPITAL\_INVESTED\_PER\_HOLDING;

VAR DECOMPOSED\_STK\_RET;

OUTPUT OUT = VW\_ESG\_RETURNS\_DECOMP MEAN (DECOMPOSED\_STK\_RET) = VW\_ESG\_RETURNS\_DECOMP;

**RUN**;

### CALCULATING VALUE WEIGHTED NON ESG FUND RETURN

Intitution :- The code calculates value-weighted returns for holdings without an ESG score in the fund portfolio for a given year and month, using their decomposed stock returns and weighting them based on the capital invested in each holding.

*Line no 1045 in the codefile*

**PROC** **SUMMARY** DATA = NZD\_DATASET\_FINAL NWAY MISSING ;

WHERE ESG = **0** ;

CLASS FUNDID YEAR MONTH;

WEIGHT CAPITAL\_INVESTED\_PER\_HOLDING;

VAR DECOMPOSED\_STK\_RET;

OUTPUT OUT = VW\_NON\_ESG\_RETURNS\_DECOMP MEAN (DECOMPOSED\_STK\_RET) = VW\_NON\_ESG\_RETURNS\_DECOMP;

**RUN**;

### CALCULATING SUM\_ESG\_CAPITAL OF EACH FUND PER MONTH AND YEAR

Intitution :- The code calculates the total capital invested by the fund portfolio in the holdings which have ESG ratings for specified year and month.

*Line no 1087 in the codefile*

%[***PROC\_SUMMARY\_ESG1***](#variables_66)(INPUT\_FILE=NZD\_DATASET\_FINAL, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=CAPITAL\_INVESTED\_PER\_HOLDING, OUTPUT\_FILE=ESG\_AUM\_MONTHLY, STATS\_OPS=SUM, CALCULATED\_VAR=SUM\_ESG\_CAPITAL\_NZD);

### CALCULATING TOTAL WEIGHT OF ESG HOLDINGS PER FUND

Intitution :- The code calculates the total percent weight allocated by a fund portfolio in the holdings which have ESG ratings for the specified year and month.

*Line no 1098 in the codefile*

%[***PROC\_SUMMARY\_ESG1***](#variables_66)(INPUT\_FILE=NZD\_DATASET\_FINAL, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=PERCENTHOLDINGS, OUTPUT\_FILE=ESG\_TOTAL\_HOLDINGS, STATS\_OPS=SUM, CALCULATED\_VAR=NZD\_ESG\_WT\_OF\_FUND);

### CALCULATING ESG TILT FOR THE FUND – inclusive

Intitution :- The code calculates the ESG tilt of a fund portfolio for the specified year and month by subtracting ESG weight of the fund portfolio with average industry weight in ESG holdings. A posiive value shows the fund is investing more as compared to industry trend while a negative value shows the fund is investing less as compared to industry trend.

*Line no 1103 in the codefile*

/\*STEP (9.3) : CALCULATING AVERAGE ESG HOLDINGS FOR THE MONTH AND YEAR

CALCULATING TOTAL INDUSTRY ESG WEIGHT FOR THE MONTH AND YEAR FOR OVERALL DATA \*/

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=ESG\_TOTAL\_HOLDINGS, CLASS\_VARS=YEAR MONTH, VARIABLES=NZD\_ESG\_WT\_OF\_FUND, OUTPUT\_FILE=INDUSTRY\_AVG, STATS\_OPS=MEAN, CALCULATED\_VAR=INDUSTRY\_AVG\_ESG\_WT\_NZD);

/\* STEP (9.4) : A)MERGING DATASET WITH MAIN DATASET

B)CALCULATING ESG TILT FOR THE FUND\*/

**PROC** **SORT** DATA = INDUSTRY\_AVG; BY YEAR MONTH; **RUN**;

**PROC** **SORT** DATA = ESG\_TOTAL\_HOLDINGS ; BY YEAR MONTH; **RUN**;

**DATA** ESG\_HOLDINGS\_TILT\_NZD;

MERGE ESG\_TOTAL\_HOLDINGS(IN=A) INDUSTRY\_AVG(IN=B);

BY YEAR MONTH;

IF A;

ESG\_TILT\_INCLUSIVE\_NZD =

NZD\_ESG\_WT\_OF\_FUND - INDUSTRY\_AVG\_ESG\_WT\_NZD;

DROP INDUSTRY\_AVG\_ESG\_WT\_NZD NZD\_ESG\_WT\_OF\_FUND;

**RUN**;

### ESG TILT EXCLUSIVE

Intitution :- The intitution is same as above code with only major difference that in industry weight calculation the fund ESG holdings weight is excluded. This gives a cleaner number which measures how much the fund tilts towards ESG compared to other industry participants without its own self influence in the industry trends.

*Line no 1132 in the codefile*

**PROC** **SQL**;

CREATE TABLE DATA\_12 AS

SELECT FUNDID, YEAR, MONTH, NZD\_ESG\_WT\_OF\_FUND, SUM(NZD\_ESG\_WT\_OF\_FUND)AS ESG\_WT\_SUM, COUNT (FUNDID) AS FUNDID\_COUNT

FROM DATA\_24

GROUP BY YEAR, MONTH;

**QUIT**;

**DATA** DATA\_33;

SET DATA\_12;

ESG\_WT\_EXCLUSIVE\_AVG = (ESG\_WT\_SUM - NZD\_ESG\_WT\_OF\_FUND) / (FUNDID\_COUNT - **1**);

ESG\_TILT\_EXCLUSIVE\_NZD =

NZD\_ESG\_WT\_OF\_FUND -ESG\_WT\_EXCLUSIVE\_AVG;

KEEP FUNDID YEAR MONTH FUNDID\_COUNT ESG\_WT\_EXCLUSIVE\_AVG ESG\_TILT\_EXCLUSIVE\_NZD;

**RUN**;

### CALCULATION OF ESG FUND FLOWS FOR FUND PORTFOLIO

Intitution :- The code calculates how much fund flow for specified year and month is deployed towards holdings which have ESG ratings for given fund portfolio.

*Line no 1152 in the codefile*

**Code A) dataset preparation**

/\* STEP (11.1) : CREATING DATASET FOR ESG FUND FLOWS CALCULATIONS \*/

**PROC** **SQL**;

CREATE TABLE ESGSET AS

SELECT DISTINCT FUNDID, YEAR, MONTH, VW\_ESG\_RETURNS\_DECOMP, SUM\_ESG\_CAPITAL\_NZD, NZD\_ESG\_WT\_OF\_FUND

FROM NZD\_DATASET\_FINAL

WHERE ESG =**1**;

**QUIT**;

**RUN**;

**Code B) variable calculations**

/\* STEP (11.2) : FUND ESG FLOW CALCULATIONS \*/

**DATA** ESGFUND\_FLOW\_NZD;

SET ESGSET;

BY FUNDID;

LAGGED\_ESGNETASSET = LAG(SUM\_ESG\_CAPITAL\_NZD); /\*CREATING NEW VARIABLE LAGGED\_NETASSET\*/

IF FIRST.FUNDID THEN LAGGED\_ESGNETASSET = **0**;

CHANGE\_ESGNETASSET = SUM\_ESG\_CAPITAL\_NZD - LAGGED\_ESGNETASSET;

IF NOT MISSING(SUM\_ESG\_CAPITAL\_NZD) AND SUM\_ESG\_CAPITAL\_NZD NE **0** AND NOT MISSING(CHANGE\_ESGNETASSET) THEN DO;

CHANGE\_ESGNETASSET\_PERCENT = (CHANGE\_ESGNETASSET / LAGGED\_ESGNETASSET) \* **100** ;

END;

ELSE CHANGE\_ESGNETASSET\_PERCENT = **.**;

IF NOT MISSING(VW\_ESG\_RETURNS\_DECOMP) AND VW\_ESG\_RETURNS\_DECOMP NE **0** THEN DO;

VW\_ESG\_RETURNS\_DECIMAL = VW\_ESG\_RETURNS\_DECOMP/**100**;

END;

ELSE VW\_ESG\_RETURNS\_DECIMAL = **.**;

ESG\_DOLLAR\_FLOW\_NZD = (SUM\_ESG\_CAPITAL\_NZD - (LAGGED\_ESGNETASSET \* (**1** + VW\_ESG\_RETURNS\_DECIMAL))) ;

IF NOT MISSING(LAGGED\_ESGNETASSET) AND LAGGED\_ESGNETASSET NE **0** AND NOT MISSING(ESG\_DOLLAR\_FLOW\_NZD) THEN DO;

ESG\_FLOW\_PERCENT\_NZD = (ESG\_DOLLAR\_FLOW\_NZD / LAGGED\_ESGNETASSET) \* **100**;

END;

ELSE ESG\_FLOW\_PERCENT\_NZD = **.**;

**RUN**;

### WINSORIZATION DONE - ESG\_FLOW\_PERCENT\_NZD

*Line no 1200 in the codefile*

**Code A) dataset preparation**

/\* STEP (11.5) :CREATING DATASET FOR CALCULATING WINSORIZED VALUES \*/

**PROC** **SQL**;

CREATE TABLE ESG\_FLOW\_PERCENT AS

SELECT DISTINCT FUNDID, YEAR, MONTH, ESG\_FLOW\_PERCENT\_NZD, BINARYVAR

FROM NZD\_DATASET\_FINAL;

**QUIT**;

**Code B) variable calculations**

/\* STEP (11.10) : WINSORIZATION FOR OVERALL FLOW TO CALCULATE AVG\_ESG\_FLOW\_PERCENT\_WIN ACROSS MONTH AND YEAR \*/

**PROC** **SORT** DATA = ESG\_FLOW\_PERCENT ; BY YEAR; **RUN**;

**PROC** **SUMMARY** DATA = ESG\_FLOW\_PERCENT ;

VAR ESG\_FLOW\_PERCENT\_NZD ;

OUTPUT OUT=\_STATS\_ P1=FLOW\_P1 P99=FLOW\_P99;

**RUN**;

**DATA** WANT\_ESG;

if \_N\_ = **1** then set \_STATS\_;

SET ESG\_FLOW\_PERCENT ;

IF ESG\_FLOW\_PERCENT\_NZD < FLOW\_P1 AND ESG\_FLOW\_PERCENT\_NZD NE **.** THEN ESG\_FLOW\_PERCENT\_WIN\_NZD = FLOW\_P1 ;

IF ESG\_FLOW\_PERCENT\_NZD > FLOW\_P99 AND ESG\_FLOW\_PERCENT\_NZD NE **.** THEN ESG\_FLOW\_PERCENT\_WIN\_NZD = FLOW\_P99 ;

IF ESG\_FLOW\_PERCENT\_WIN\_NZD = **.** THEN ESG\_FLOW\_PERCENT\_WIN\_NZD ESG\_FLOW\_PERCENT\_NZD ;

DROP \_TYPE\_ \_FREQ\_ ESG\_FLOW\_PERCENT\_NZD BINARYVAR FLOW\_P1 FLOW\_P99 ;

**RUN**;

### CALCULATION OF OVERALL FUND FLOWS FOR FUND PORTFOLIO

Intitution :- The code calculates the fund flow for specified year and month for each fund portfolio.

*Line no 1232 in the codefile*

**Code A) dataset preparation**

/\* STEP (12.1) : CREATING DATASET FOR OVERALL FUND FLOWS CALCULATIONS \*/

**PROC** **SQL**;

CREATE TABLE FUNDFLOW\_DATASET AS

SELECT DISTINCT FUNDID, YEAR, MONTH, Fundreturns\_NZD1, Netassets\_NZD

FROM NZD\_DATASET\_FINAL ;

**QUIT**;

**RUN**;

**Code B) variable calculations**

/\* STEP (12.2) : CALCULATING FUND FLOWS FOR THE DATASET \*/

**DATA** FLOW\_CALS\_NZD;

SET FUNDFLOW\_DATASET;

BY FUNDID;

LAGGED\_NETASSET = LAG(Netassets\_NZD);

IF FIRST.FUNDID THEN LAGGED\_NETASSET = **0**;

NETASSET\_CHANGE = Netassets\_NZD - LAGGED\_NETASSET;

IF NOT MISSING(Netassets\_NZD) AND Netassets\_NZD NE **0** AND NOT MISSING(NETASSET\_CHANGE) THEN DO;

CHANGE\_NETASSET\_PERCENT = ROUND((NETASSET\_CHANGE / Netassets\_NZD) \* **100**, **0.01**); END;

ELSE CHANGE\_NETASSET\_PERCENT = **.**;

IF NOT MISSING(Fundreturns\_NZD1) AND Fundreturns\_NZD1 NE **0** THEN DO;

FUNDRETURNS\_NEW\_VAL\_DECIMAL = Fundreturns\_NZD1/**100**;

END;

ELSE FUNDRETURNS\_NEW\_VAL\_DECIMAL = **.**;

FLOW\_DOLLAR\_VALUE\_NZD = (Netassets\_NZD -(LAGGED\_NETASSET \*(**1** + FUNDRETURNS\_NEW\_VAL\_DECIMAL)));

IF NOT MISSING(LAGGED\_NETASSET) AND LAGGED\_NETASSET NE **0** AND NOT MISSING(FLOW\_DOLLAR\_VALUE\_NZD) THEN DO;

FLOW\_PERCENT\_NZD = (FLOW\_DOLLAR\_VALUE\_NZD/LAGGED\_NETASSET)\***100** ;

END;

ELSE CHANGE\_NETASSET\_PERCENT = **.**;

**RUN**;

### WINSORIZATION DONE - FLOW\_PERCENT\_NZD

*Line no 1280 in the codefile*

/\* STEP (12.5) : CREATING DATASET OF DISTINCT VALUES TO CREATE FLOW PERCENT ACROSS YEAR AND MONTH \*/

**PROC** **SQL**;

CREATE TABLE FLOW\_PERCENT\_SET AS

SELECT DISTINCT FUNDID, YEAR, MONTH, FLOW\_PERCENT\_NZD, BINARYVAR

FROM NZD\_DATASET\_FINAL;

**QUIT**;

/\* STEP (12.10) : WINSORIZATION DONE FOR OVERALL FLOW TO CALCULATE AVG\_FLOW\_PERCENT ACROSS MONTH AND YEAR \*/

**PROC** **SUMMARY** DATA = FLOW\_PERCENT\_SET ;

VAR FLOW\_PERCENT\_NZD ;

OUTPUT OUT=\_STATS\_ P1=FLOW\_P1 P99=FLOW\_P99;

**RUN**;

/\* STEP (12.11) - CREATING WINSORIZED DATASET \*/

**DATA** WANT;

IF \_N\_ = **1** THEN SET \_STATS\_;

SET FLOW\_PERCENT\_SET;

IF FLOW\_PERCENT\_NZD < FLOW\_P1 AND FLOW\_PERCENT\_NZD NE **.** THEN FLOW\_PERCENT\_WIN\_NZD = FLOW\_P1;

IF FLOW\_PERCENT\_NZD > FLOW\_P99 AND FLOW\_PERCENT\_NZD NE **.** THEN FLOW\_PERCENT\_WIN\_NZD = FLOW\_P99;

IF FLOW\_PERCENT\_WIN\_NZD = **.** THEN FLOW\_PERCENT\_WIN\_NZD = FLOW\_PERCENT\_NZD;

DROP \_TYPE\_ \_FREQ\_ FLOW\_PERCENT\_NZD BINARYVAR FLOW\_P1 FLOW\_P99;

**RUN**;

### CALCULATION OF VOLATILITY OF FUNDRETURNS

Intitution :- The code calculates the trailing 12 month volatility (degree of variation of fund returns), updating every month.

*Line no 1416 in the codefile*

The dataset used in this code is different. It is continuos dataset of fundreturns.

**Code A) dataset preparation**

/\* STEP (13.5) : CREATING DATASET FOR VOLATILITY RETURNS \*/

**DATA** VOLATILITY\_1;

SET FILE\_SET.CAPM\_RETS\_FILE\_1;

KEEP FUNDID YEAR MONTH Fundreturns\_NZD1;

**RUN**;

**Code B) variable calculations**

/\* STEP (13.1) : CREATING VOLATILITY VALUES FOR THE DATA \*/

**PROC** **EXPAND** DATA = VOLATILITY\_1 OUT = RUNNING\_VOL;

BY FUNDID;

CONVERT Fundreturns\_NZD1 = ROLLING\_VOLATILITY\_NZD / TRANSFORMOUT=(MOVSTD **12**);

**RUN**;

/\* STEP (13.2) : CODE IS TO REMOVE FIRST 11 OBS OF ALL THE FUNDID SO THAT WE GET VOLATILITY FROM 12TH OBS.\*/

**DATA** VOLATILITY\_NZD;

SET RUNNING\_VOL;

BY FUNDID;

IF FIRST.FUNDID THEN COUNTER = **0**;

COUNTER + **1**;

IF COUNTER > **11** THEN OUTPUT;

KEEP FUNDID MONTH YEAR TIME Fundreturns\_NZD1 ROLLING\_VOLATILITY\_NZD ;

IF LAST.FUNDID THEN COUNTER = **0**;

**RUN**;

/\* STEP (13.3) : CREATING DATASET FROM ESG.VOLATILITY TO MERGE WITH TOTAL\_DATASET\_FINAL \*/

**DATA** TEMP\_VOLATILITY;

SET VOLATILITY\_NZD;

DROP Fundreturns\_NZD1 TIME;

**RUN**;

### CALCULATING TRACKING ERROR

Intitution :- The code calculates how much the fund’s returns deviate from its benchmark index for given year and month. We first calculate abnormal returns, which shows whether the fund have underperformed or outperformed the benchmark. Then it compute the moving 12 month std dev of abnormal returns.

*Line no 1476 in the codefile*

**Code A) dataset preparation**

/\* STEP 14.5 : USING FUNDRETURNS CONTINOUS DATASET TO CALCULATE VALUES \*/

**DATA** FUNDRETURNS\_FILE\_1;

SET FILE\_SET.CAPM\_RETS\_FILE\_1;

**RUN**;

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=FUNDRETURNS\_FILE\_1, INPUT\_FILE\_2=FILE\_SET.FUND\_DETAILS, SORT\_VARS=FUNDID, OUTPUT\_FILE=FUNDRETURNS\_FILE\_2);

/\* STEP 14.6 : MERGING MORNINGSTAR INDEX RETURNS \*/

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=FUNDRETURNS\_FILE\_2, INPUT\_FILE\_2=File\_set.Index\_returns, SORT\_VARS=Morningstar\_Index\_Id YEAR MONTH, OUTPUT\_FILE=FUNDRETURNS\_DATASET);

/\* STEP 14.7 : CLEANING ABOVE DATASET TO VOLATILITY CALCULATIONS \*/

**DATA** FUNDLEVEL\_SET;

SET FUNDRETURNS\_DATASET;

KEEP FUNDID YEAR MONTH Fundreturns\_NZD1 INDEX\_RETURNS1 Morningstar\_Index\_Id;

**RUN**;

**Code B) variable calculations**

/\* STEP 14.8 : CALCULATING TRACKING ERROR \*/

**DATA** TE\_SETUP1;

SET FUNDLEVEL\_SET;

AB\_RET = Fundreturns\_NZD1 - INDEX\_RETURNS1 ;

**RUN**;

**PROC** **SORT** DATA=TE\_SETUP1; BY FUNDID; **RUN**;

**PROC** **EXPAND** DATA=TE\_SETUP1 OUT = TE\_SETUP2 METHOD=NONE;

BY FUNDID ;

CONVERT AB\_RET = TRACK\_ERROR / TRANSFORMOUT=(NOMISS MOVSTD **12** TRIMLEFT **11** );/\* ROLLING 12 MONTH STDEV OF ABNORMAL\_RETURNS \*/

**RUN**;

### CREATING ACTIVE\_ALPHA GENERATION FLAG

Intitution :- The code calculates the level of active management of the fund. For the fund to be classified as active, it should have a higher tracking error (calculated on t-1) than the industry median else the active flag is 0.

*Line no 1505 in the codefile*

/\* STEP 14.9 : CALCULATING MEDIAN OF TRACKING ERROR \*/

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=TE\_SETUP2, CLASS\_VARS=YEAR MONTH, VARIABLES=TRACK\_ERROR, OUTPUT\_FILE=TE\_SETUP\_MEDIAN, STATS\_OPS=MEDIAN, CALCULATED\_VAR=MEDIAN\_TE);

/\* STEP 14.10 : CREATING LAG TRACKING ERROR \*/

**PROC** **SORT** DATA = TE\_SETUP\_MEDIAN ; BY YEAR MONTH; **RUN**;

**PROC** **EXPAND** DATA=TE\_SETUP2 OUT = TE\_SETUP3 METHOD=NONE;

BY FUNDID;

CONVERT TRACK\_ERROR = LAG\_TRACK\_ERROR / TRANSFORM=(LAG **1**);

**RUN**;

/\* STEP 14.11 : CREATING ALPHA GENERATION FLAG \*/

**PROC** **SORT** DATA = TE\_SETUP3 OUT = TE\_SETUP4; BY YEAR MONTH; **RUN**;

**DATA** FUND\_TRACKINGERROR;

MERGE TE\_SETUP4 (IN = A) TE\_SETUP\_MEDIAN (IN = B);

BY YEAR MONTH;

IF A;

IF LAG\_TRACK\_ERROR > MEDIAN\_TE THEN ACTIVE\_ALPHA = **1**;

/\*GREATER THAN MEDIAN THAN ACTIVE\*/

ELSE ACTIVE\_ALPHA = **0**;

IF LAG\_TRACK\_ERROR =**.** THEN ACTIVE\_ALPHA = **.**;

ELSE ACTIVE\_ALPHA =ACTIVE\_ALPHA ;

DROP \_FREQ\_ TIME Fundreturns\_NZD1 \_TYPE\_ INDEX\_RETURNS1 Morningstar\_Index\_Id;

**RUN**;

### CALCULATING TIME AND PICKING VARIABLES -TIMING\_PER\_STOCK\_MED, PICKING\_PER\_STOCK\_MED

Calculation :- The code evaluates a fund manager’s timing and stock-picking ability by analyzing individual holdings. First, it calculates the median holdings weight of each stock in the market for a given year and month. Then, it determines the fund’s active positioning by subtracting the median holdings from the fund’s specific holdings weight for that stock. Finally, this difference is multiplied by the beta (a measure of market sensitivity) and the NZX50 index returns, providing insight into how well the manager adjusts exposure based on expected market movements.

*Line no 1631 in the codefile*

**Code A) Calculating Median weight variable of SECID holdings in all fund portfolio**

**PROC** **SUMMARY** DATA = FILE\_033 NWAY MISSING;

CLASS SECID YEAR MONTH;

WEIGHT Netassets\_NZD;

VAR Percentholdings;

OUTPUT OUT = FILE\_123(DROP = \_TYPE\_ \_FREQ\_) MEDIAN(Percentholdings)=MED\_HOLDINGS\_SECID;

**RUN**;

*Line no 1643 in the codefile*

**Code B) Calculating Timing per stock and Picking per stock variables**

/\* STEP 11: CALCULATING TIME AND PICKING VARIABLES \*/

**DATA** STEP\_1;

SET TIME\_VARY\_DATA;

TIMING\_PER\_STOCK\_MED =

(Percentholdings - MED\_HOLDINGS\_SECID) \* (BETA \* NZX50returns\_NZD1\_FRWRD);

PICKING\_PER\_STOCK\_MED =

(Percentholdings - MED\_HOLDINGS\_SECID) \* (monthlyreturns\_FRWRD - (BETA\*NZX50returns\_NZD1\_FRWRD));

**RUN**;

### Fund level Timing and Picking vars - SUM\_TIMING\_MED, SUM\_PICKING\_MED

Calculation :- The above calculated timing and picking per stock is summed up for given year and month for each fund portfolio. The resulting variables are timing and picking of the fund portfolio.

*Line no 1649 in the codefile*

/\* STEP 12: CLEANING ABOVE DATASET \*/

**DATA** PICKING\_FILE\_MED;

SET STEP\_1;

DROP \_TYPE\_ \_FREQ\_;

**RUN**;

/\* STEP 13: TAKING SUM OF THE TIMING VARIABLES FOR THE MONTH AND YEAR PER FUND \*/

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=PICKING\_FILE\_MED, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=TIMING\_PER\_STOCK\_MED, OUTPUT\_FILE=DATA\_SKILL\_1, STATS\_OPS=SUM, CALCULATED\_VAR=SUM\_TIMING\_MED);

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=PICKING\_FILE\_MED, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=PICKING\_PER\_STOCK\_MED, OUTPUT\_FILE=DATA\_SKILL\_2, STATS\_OPS=SUM, CALCULATED\_VAR=SUM\_PICKING\_MED);

### CREATING WINSORIZED TIMING AND PICKING VARAIABLES

*Line no 1663 in the codefile*

/\* STEP 15: CALCULATING WINSORIZATION LEVELS FOR THE EW\_DATASET DATASET \*/

**PROC** **SORT** DATA = EW\_DATASET; BY YEAR MONTH; **RUN**;

**PROC** **MEANS** DATA = EW\_DATASET P1 P99 N NOPRINT;

BY YEAR;

VAR SUM\_TIMING\_MED SUM\_PICKING\_MED;

OUTPUT OUT = DATA\_1

P1= SUM\_TIMING\_MED\_P1 SUM\_PICKING\_MED\_P1

P99=SUM\_TIMING\_MED\_P99 SUM\_PICKING\_MED\_P99;

**RUN**;

/\* STEP 16: CREATING WINSORIZED TIMING AND PICKING VARAIABLES \*/

/\* MERGING PERCENTILE DATASET WITH EW\_DATASET \*/

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=EW\_DATASET, INPUT\_FILE\_2=DATA\_1, SORT\_VARS=YEAR, OUTPUT\_FILE=EW\_DATASET\_WIN);

**DATA** EW\_DATASET\_30;

SET EW\_DATASET\_WIN;

IF SUM\_TIMING\_MED < SUM\_TIMING\_MED\_P1 AND SUM\_TIMING\_MED NE **.** THEN SUM\_TIMING\_MED\_WIN = SUM\_TIMING\_MED\_P1;

IF SUM\_TIMING\_MED > SUM\_TIMING\_MED\_P99 AND SUM\_TIMING\_MED NE **.** THEN SUM\_TIMING\_MED\_WIN = SUM\_TIMING\_MED\_P99;

IF SUM\_TIMING\_MED\_WIN = **.** THEN SUM\_TIMING\_MED\_WIN = SUM\_TIMING\_MED;

IF SUM\_PICKING\_MED < SUM\_PICKING\_MED\_P1 AND SUM\_PICKING\_MED NE **.** THEN SUM\_PICKING\_MED\_WIN = SUM\_PICKING\_MED\_P1;

IF SUM\_PICKING\_MED > SUM\_PICKING\_MED\_P99 AND SUM\_PICKING\_MED NE **.** THEN SUM\_PICKING\_MED\_WIN = SUM\_PICKING\_MED\_P99;

IF SUM\_PICKING\_MED\_WIN = **.** THEN SUM\_PICKING\_MED\_WIN = SUM\_PICKING\_MED;

**RUN**;

### CREATING VARIABLES (A) ESG STOCKS, (B) TOTAL STOCKS AND (C) TOTAL HOLDINGS IN A FUND PORTFOLIO

Intitution :- The code calculates the a) total no of stocks in a fund portfolio which have ESG score or ratings, b) the total no of stocks in a fund portfolio and c) the total number of holdings in a fund portfolio, all for specified year and month.

*Line no 1974 in the codefile*

**Code a) dataset creation.**

**PROC** **SQL**;

CREATE TABLE HOLDINGS\_EQUITY AS

SELECT DISTINCT \*

FROM NZD\_HD\_ALL\_FINAL

WHERE TRIM(DETAILHOLDINGTYPE) IN (

'EQUITY',

'EQUITY - FUTURE',

'EQUITY - PRIVATE/',

'EQUITY - REIT',

'EQUITY - UNDEFINE',

'EQUITY - UNITS',

'EQUITY - WARRANTS', )

ORDER BY FUNDID, YEAR, MONTH;

**QUIT**;

**RUN**;

**Code b) variable creation.**

**PROC** **SQL**;

CREATE TABLE ESG\_EQUITY AS

SELECT FUNDID, YEAR, MONTH, SUM(CASE WHEN ESG=**1** THEN **1** ELSE **0** END) AS ESGHOLDINGS\_EQUITY\_NZD, COUNT(\*) AS TOTALHOLDINGS\_EQUITY\_NZD

FROM HOLDINGS\_EQUITY

GROUP BY FUNDID, YEAR, MONTH;

CREATE TABLE TOTAL\_HOLDINGS AS

SELECT FUNDID, YEAR, MONTH, COUNT(NAME)AS COMPONENT\_HOLDINGS\_NZD

FROM NZD\_HD\_ALL\_FINAL

GROUP BY FUNDID, YEAR, MONTH;

**QUIT**;

### CALCULATING A) RATIO OF ESG STOCKS TO TOTAL STOCKS B) RATIO OF ESG STOCKS TO TOTAL HOLDINGS OR COMPONENTS HOLDINGS

Intitution :- The code calculates the a) the ratio of ESG rated stocks to total number of stocks in a fund portfolio and b) the ratio of ESG stocks to total holdings in a fund portfolio, for given year and month.

*Line no 1990 in the codefile*

**DATA** FILE\_SET.ESGPROPORTION\_NZD\_STKRET\_WIN;

MERGE ESG\_EQUITY ( IN = A) TOTAL\_HOLDINGS (IN = B);

BY FUNDID YEAR MONTH;

IF A OR B;

RATIO\_ESG\_TO\_TOT\_EQUITY\_NZD = (ESGHOLDINGS\_EQUITY\_NZD / TOTALHOLDINGS\_EQUITY\_NZD) \* **100** ;

RATIO\_ESG\_TO\_TOT\_HOLDINGS\_NZD = (ESGHOLDINGS\_EQUITY\_NZD / COMPONENT\_HOLDINGS\_NZD) \* **100** ;

**RUN**;

### COUNTING NON NZ STOCKS ON BASIS OF ISO CODE

Intitution :- The code counts the number of foreign listed stocks in a fund portfolio for given year and month. The stock is identifed on basis of ISO code as listed in refintiv.

**Code A) dataset preparation**

*Line no 2010 in the codefile*

**DATA** NZD\_HD\_ALL\_FINAL;

SET File\_set.Nzd\_hd\_all\_final\_stkret\_win;

**RUN**;

%[***IMPORT\_MACROS***](#variables_61)(INPUTFILE='R:\SAS - Research\SWARN\_RESEARCH\SAS\_DATASETS\STOCKS\_LEVEL\_DATA\_REFINITIVE\NZ\_data\Exchange\_info.CSV', OUTPUT\_FILE=COUNTRY\_EXCHANGE);

**PROC** **SORT** DATA = COUNTRY\_EXCHANGE NODUPKEY; BY RICCODE; **RUN**;

%[***IMPORT\_MACROS***](#variables_61)(INPUTFILE='R:\SAS - Research\SWARN\_RESEARCH\SAS\_DATASETS\STOCKS\_LEVEL\_DATA\_REFINITIVE\NZ\_data\Country\_of\_HQ.CSV', OUTPUT\_FILE=Country\_of\_HQ);

**PROC** **SORT** DATA = Country\_of\_HQ NODUPKEY; BY RICCODE; **RUN**;

%[***IMPORT\_MACROS***](#variables_61)(INPUTFILE='R:\SAS - Research\SWARN\_RESEARCH\SAS\_DATASETS\STOCKS\_LEVEL\_DATA\_REFINITIVE\NZ\_data\ISO\_CODE\_NZD.CSV', OUTPUT\_FILE=ISO\_CODE\_NZD);

**PROC** **SORT** DATA = ISO\_CODE\_NZD NODUPKEY; BY RICCODE; **RUN**;

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=COUNTRY\_EXCHANGE, INPUT\_FILE\_2=Country\_of\_HQ, SORT\_VARS=RICCODE, OUTPUT\_FILE=TEMP\_FILE\_1);

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=TEMP\_FILE\_1, INPUT\_FILE\_2=ISO\_CODE\_NZD, SORT\_VARS=RICCODE, OUTPUT\_FILE=TEMP\_FILE\_2);

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=FILE\_SET.RIC\_CODE, INPUT\_FILE\_2=TEMP\_FILE\_2, SORT\_VARS=RICCODE, OUTPUT\_FILE=TEMP\_FILE\_3);

**PROC** **SORT** DATA = TEMP\_FILE\_3 OUT = FILE\_SET.STOCKS\_EXCHANGE\_FILE NODUPKEY; BY SECID; **RUN**;

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=NZD\_HD\_ALL\_FINAL, INPUT\_FILE\_2=FILE\_SET.STOCKS\_EXCHANGE\_FILE, SORT\_VARS=SECID, OUTPUT\_FILE=NZD\_HD\_ALL\_FINAL);

/\* CREATING DATASET FOR EQUITY STOCKS ONLY \*/

**PROC** **SQL**;

CREATE TABLE DATA\_123 AS

SELECT \*

FROM NZD\_HD\_ALL\_FINAL

WHERE TRIM(DETAILHOLDINGTYPE) IN (

'EQUITY',

'EQUITY - FUTURE',

'EQUITY - PRIVATE/',

'EQUITY - REIT',

'EQUITY - UNDEFINE',

'EQUITY - UNITS',

'EQUITY - WARRANTS', )

ORDER BY FUNDID, YEAR, MONTH;

**RUN**;

**Code B) variable calculations**

*Line no 2052 in the codefile*

/\* COUNTING NON NZ STOCKS ON BASIS OF ISO CODE \*/

**PROC** **SQL**;

CREATE TABLE COUNT\_12 AS

SELECT FUNDID, YEAR, MONTH, COUNT(\*) AS NON\_NZ\_STOCKS

FROM DATA\_123

WHERE Country\_of\_Exchange NE 'New Zealand' AND Country\_of\_Exchange NE 'NEW ZEALAND EXCHANGE LTD'

GROUP BY FUNDID, YEAR, MONTH;

**QUIT**;

### NON\_NZ\_STOCK\_RATIO

Intitution :- The code computes ratio of foreign listed stocks with total holdings of the portfolio.

*Line no 2065 in the codefile*

**DATA** FILE\_SET.ESGPROPORTION\_NZD\_STKRET\_WIN;

MERGE ESGPROPORTION\_NZD ( IN = A ) COUNT\_12 ( IN = B ) ;

BY FUNDID YEAR MONTH ;

IF A;

NON\_NZ\_STOCK\_RATIO =

(NON\_NZ\_STOCKS / COMPONENT\_HOLDINGS\_NZD) \* **100** ;

**RUN**;

### COUNTING NO OF EQUITIES IN EACH FUND MONTH YEAR WHICH HAVE ESGSCORE >= 75

Code Logic :- The code computes the total number of stocks which have ESG ratings >= 75 and are not categorised as sin stocks.

**Code A) dataset preparation**

*Line no 2217 in the codefile*

/\* STEP 15.1 : FILTERING EQUITY DATA FROM DATASET \*/

**PROC** **SQL**;

CREATE TABLE HOLDINGS\_EQUITY AS

SELECT DISTINCT \*

FROM NZD\_HD\_ALL\_FINAL

WHERE TRIM(DETAILHOLDINGTYPE) IN (

'EQUITY',

'EQUITY - FUTURE',

'EQUITY - PRIVATE/',

'EQUITY - REIT',

'EQUITY - UNDEFINE',

'EQUITY - UNITS',

'EQUITY - WARRANTS', )

ORDER BY FUNDID, YEAR, MONTH;

**QUIT**;

**RUN**;

**Code B) variable calculations**

*Line no 2234 in the codefile*

/\* STEP 15.4 : FILTERING DATA WHERE ESGSCORE >= 75 \*/

**DATA** TEST\_12;

SET HOLDINGS\_EQUITY ;

BY FUNDID YEAR MONTH;

IF ESGSCORE >= **75** AND sin\_stocks = **0** THEN

NEWSCORE = **1**;

ELSE

NEWSCORE = **0**;

**RUN**;

/\* STEP 15.6 : COUNTING NO OF EQUITIES IN EACH FUND MONTH YEAR WHICH HAVE ESGSCORE >= 75 \*/

**PROC** **SQL**;

CREATE TABLE TEST\_987 AS

SELECT FUNDID, YEAR, MONTH, COUNT(\*) AS ESGGE75\_NZD\_EQUITY

FROM TEST\_12

WHERE NEWSCORE = **1**

GROUP BY FUNDID, YEAR, MONTH;

**QUIT**;

### RATIO\_ESG75\_HLDSEQ\_NZD

Intitution :- The code computes the ratio of total number of stocks which have ESG ratings greater than 75 to the total ESG rated stocks.

*Line no 2265 in the codefile*

**DATA** FUNDLEVEL\_SET\_2;

MERGE FUNDLEVEL\_SET ( IN = A ) TEST\_987 ( IN = B );

BY FUNDID YEAR MONTH;

RATIO\_ESG75\_HLDSEQ\_NZD =

(ESGGE75\_NZD\_EQUITY / ESGHOLDINGS\_EQUITY\_NZD)\***100**;

DROP ESGHOLDINGS\_EQUITY\_NZD;

**RUN**;

### ESGGE75\_CAP\_INV\_NZD, ESGGE75\_SUM\_WT\_NZD, ESGGE75\_VW\_RETURNS\_NZD

Intitution :- The code computes a) the total capital invested by a fund portfolio in the stocks which have ESG ratings Ge 75, b) the total percent holdings weight of the stocks in the fund portfolio which have ESG ratings GE 75 and c) the total decomposed stock returns of stocks for the fund portfolio which have ESG ratings GE 75.

**Code A) dataset preparation**

*Line no 2277 in the codefile*

/\* STEP 15.10 : FILTERING DATA FROM NZD\_HD\_ALL\_FINAL WHERE ESGSCORE IS GE 75 \*/

**DATA** TEST\_21;

SET TEST\_12;

WHERE NEWSCORE = **1** ;

KEEP FUNDID YEAR MONTH SECID NAME DETAILHOLDINGTYPE NUMBEROFSHARES ;

**RUN**;

/\* STEP 15.11 : SELECTING DATASET FROM NZD\_DATASET\_FINAL ESGSCORE IS GE 75 AND NOT SIN\_STOCKS \*/

**DATA** TEST\_54;

SET NZD\_DATASET\_FINAL;

WHERE ESGSCORE >= **75** AND sin\_stocks = **0** ;

KEEP FUNDID YEAR MONTH SECID PERCENTHOLDINGS BINARYVAR CAPITAL\_INVESTED\_PER\_HOLDING DECOMPOSED\_STK\_RET ESGSCORE MONTHLYRETURNS;

**RUN**;

**PROC** **SORT** DATA = TEST\_54 ; BY FUNDID YEAR MONTH SECID; **RUN**;

**PROC** **SORT** DATA = TEST\_21 ; BY FUNDID YEAR MONTH SECID; **RUN**;

**DATA** ESGGE75\_DATA ;

MERGE TEST\_54 ( IN = A ) TEST\_21 ( IN = B ) ;

BY FUNDID YEAR MONTH SECID;

IF A AND B ;

**RUN**;

**Code B) variable calculations**

*Line no 2301 in the codefile*

/\* STEP 15.12 : CREATING SUMMARY STATS FROM ABOVE DATASET \*/

**PROC** **SUMMARY** DATA = ESGGE75\_DATA;

BY FUNDID YEAR MONTH ;

VAR CAPITAL\_INVESTED\_PER\_HOLDING ;

OUTPUT OUT = GHJKL(DROP=\_TYPE\_ \_FREQ\_)

SUM (CAPITAL\_INVESTED\_PER\_HOLDING) = ESGGE75\_CAP\_INV\_NZD

SUM (PERCENTHOLDINGS) = ESGGE75\_SUM\_WT\_NZD

SUM (DECOMPOSED\_STK\_RET) = ESGGE75\_VW\_RETURNS\_NZD ;

**RUN**;

### ESG\_SCORE\_AVG

Intitution :- The code computes average ESG score of a fund holdings for specified year and month for a fund portfolio.

*Line no 2394 in the codefile*

/\* CREATING AVERAGE ESG SCORE VARIABLE \*/

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=Nzd\_dataset\_final, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=ESGSCORE, OUTPUT\_FILE=ESG\_SCORE\_AVG\_FUND, STATS\_OPS=MEAN, CALCULATED\_VAR=ESG\_SCORE\_AVG);

### ESG\_CONTRO\_AVG

Intitution :- The code computes average ESG controversy score of a fund holdings for specified year and month for a fund portfolio.

*Line no 2396 in the codefile*

/\* CREATING AVERAGE ESG CONTROVERSY SCORE VARIABLE \*/

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=Nzd\_dataset\_final, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=ESG\_CONTROVERSY, OUTPUT\_FILE=ESG\_CONTRO\_AVG\_FUND, STATS\_OPS=MEAN, CALCULATED\_VAR=ESG\_CONTRO\_AVG);

### CREATING HIGH LOW FLAG FOR EPU

Code Logic – In the code we first calculate moving median of EPU scores with moving window of 12 periods or months. Second, we compare moving median value with EPU score. Third, we create a flag if current month EPU score is less than moving median EPU score then EPU\_HIGH is 0 else 1. This variable is a flag variable in dataset which could be used as filter.

**Code A) dataset preparation**

*Line no 2217 in the codefile*

/\* CREATING HIGH LOW FLAG FOR EPU \*/

**PROC** **SORT** DATA=New\_Zealand\_EPU\_Data\_1; BY YEAR MONTH; **RUN**;

/\* CREATING TIME SERIES VARIABLE \*/

**DATA** EPU\_DATA\_1;

SET New\_Zealand\_EPU\_Data\_1;

MODATE = MDY(MONTH,**1**,YEAR);

DROP VAR5 VAR6;

/\*FORMAT MODATE YYMMDD10.;\*/

**RUN**;

**Code B) variable calculations**

*Line no 2755 in the codefile*

**PROC** **EXPAND** DATA=EPU\_Data\_1 OUT=SETUP3 METHOD=NONE;

ID MODATE;

CONVERT EPU\_Scores= EPU\_MEDIAN / TRANSFORM=(NOMISS MOVMED **12**); **RUN**;

/\* CREATING EPU HIGH AND LOW FLAG \*/

**DATA** SETUP\_33;

SET SETUP3;

IF EPU\_Scores >= EPU\_MEDIAN THEN EPU\_HIGH = **1**;

ELSE EPU\_HIGH = **0**;

**RUN**;

### TO CALCULATE NO OF FUNDS PER MONTH AND YEAR

Code logic – The code calculates total number of funds in the dataset for given year and month. The code used is count function of SQL. In the second code there is a filter of Binaryvar which could be adjusted to (1,2) for kiwisaver funds and 3 for openend funds.

*Line no 2976 in the codefile*

/\* TO CALCULATE NO OF FUNDS PER MONTH AND YEAR \*/

/\* OVERALL \*/

**PROC** **SQL**;

CREATE TABLE DATA\_123 AS

SELECT DISTINCT YEAR, MONTH, COUNT(\*) AS FUNDID\_COUNT

FROM Fundlevel\_dataset\_truncated

GROUP BY YEAR, MONTH;

**QUIT**;

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=DATA\_123, CLASS\_VARS=YEAR, VARIABLES=FUNDID\_COUNT, OUTPUT\_FILE=DATA\_090, STATS\_OPS=MEAN, CALCULATED\_VAR=FUND\_AVERAGE);

/\* KIWISAVER \*/

**PROC** **SQL**;

CREATE TABLE DATA\_456 AS

SELECT DISTINCT YEAR, MONTH, COUNT(\*) AS FUNDID\_COUNT

FROM Fundlevel\_dataset\_truncated

WHERE BINARYVAR=**1**

GROUP BY YEAR, MONTH;

**QUIT**;

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=DATA\_456, CLASS\_VARS=YEAR, VARIABLES=FUNDID\_COUNT, OUTPUT\_FILE=DATA\_093, STATS\_OPS=MEAN, CALCULATED\_VAR=FUND\_AVERAGE);

### INACTIVE FUND

Code logic – The code calculates number of funds which have become inactive. To calculate this variable we have to first download the liquidation date of the fund and then run the below code to get average number of funds which are liquidated in specified year and month in the time series.

In the next code we have a where filter which can be used to calculate inactive as per binaryvar filter, i.e. (1,2) for kiwisaver funds and 3 for open end funds.

*Line no 2998 in the codefile*

/\* INACTIVE FUND \*/

**DATA** LIQ\_1;

SET File\_set.Liquidated\_info;

WHERE NOT MISSING(Liq\_month) AND NOT MISSING(Liq\_year)

AND (Liq\_year > **2007** AND Liq\_year < **2022**) OR

(Liq\_year = **2007** AND Liq\_month >= **7**) OR

(Liq\_year = **2022** AND Liq\_month <= **12**); ;

**RUN**;

**PROC** **SQL**;

CREATE TABLE DATA\_99 AS

SELECT DISTINCT Liq\_month, Liq\_year, COUNT(\*) AS FUNDID\_COUNT

FROM LIQ\_1

GROUP BY Liq\_month, Liq\_year;

**QUIT**;

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=DATA\_99, CLASS\_VARS=Liq\_year, VARIABLES=FUNDID\_COUNT, OUTPUT\_FILE=DATA\_093, STATS\_OPS=MEAN, CALCULATED\_VAR=INACTIVE\_FUND\_AVERAGE);

/\* KIWISAVER INACTIVE AFUNDS \*/

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=LIQ\_1, INPUT\_FILE\_2=File\_set.Binaryfile, SORT\_VARS=FUNDID, OUTPUT\_FILE=DATA\_KS\_INACTIVE);

**DATA** LIQ\_1;

SET DATA\_KS\_INACTIVE;

WHERE BINARYVAR = **1** AND NOT MISSING(Liq\_month) AND NOT MISSING(Liq\_year)

AND (Liq\_year > **2007** AND Liq\_year < **2022**) OR

(Liq\_year = **2007** AND Liq\_month >= **7**) OR

(Liq\_year = **2022** AND Liq\_month <= **12**); ;

**RUN**;

**PROC** **SQL**;

CREATE TABLE DATA\_99 AS

SELECT DISTINCT Liq\_month, Liq\_year, COUNT(\*) AS FUNDID\_COUNT

FROM LIQ\_1

GROUP BY Liq\_month, Liq\_year;

**QUIT**;

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=DATA\_99, CLASS\_VARS=Liq\_year, VARIABLES=FUNDID\_COUNT, OUTPUT\_FILE=DATA\_093, STATS\_OPS=MEAN, CALCULATED\_VAR=INACTIVE\_FUND\_AVERAGE);

### ESG\_MINUS\_NON\_ESG\_T and ESG\_MINUS\_NON\_ESG\_P variables

Code logic – The code calculates the difference between timing returns and picking returns generated by ESG rated holdings and unrated holdings.

**Code A) Timing variables**

*Line no 3183 in the codefile*

%[***PROC\_SUMMARY\_ESG1***](#variables_66)(INPUT\_FILE=Picking\_file\_stkret\_win, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=TIMING\_PER\_STOCK\_MED, OUTPUT\_FILE=FILE\_1, STATS\_OPS=SUM, CALCULATED\_VAR=SUM\_ESG\_TIMING\_TBL\_A);

**PROC** **SUMMARY** DATA=Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESG=**0**;

CLASS FUNDID YEAR MONTH;

OUTPUT OUT = FILE\_2 SUM(TIMING\_PER\_STOCK\_MED)=SUM\_NON\_ESG\_TIMING\_TBL\_A;

**RUN**;

**PROC** **SORT** DATA = FILE\_1; BY FUNDID YEAR MONTH; **RUN**;

**PROC** **SORT** DATA = FILE\_2; BY FUNDID YEAR MONTH; **RUN**;

**DATA** FILE\_3;

MERGE FILE\_1(IN=A) FILE\_2(IN=B);

BY FUNDID YEAR MONTH;

IF A OR B;

DROP \_TYPE\_ \_FREQ\_;

**RUN**;

/\* MERGING ABOVE FILES AND REMOVING EXTRA COLUMNS \*/

**PROC** **SORT** DATA = FILE\_3; BY FUNDID YEAR MONTH; **RUN**;

**PROC** **SORT** DATA = FUNDLEVEL\_DATASET\_TRUNCATED; BY FUNDID YEAR MONTH; **RUN**;

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

MERGE FUNDLEVEL\_DATASET\_TRUNCATED(IN=A) FILE\_3(IN=B);

BY FUNDID YEAR MONTH;

IF A;

DROP \_TYPE\_ \_FREQ\_;

ESG\_MINUS\_NON\_ESG\_T =

SUM\_ESG\_TIMING\_TBL\_A - SUM\_NON\_ESG\_TIMING\_TBL\_A;

**RUN**;

**Code B) Picking variables**

*Line no 3263 in the codefile*

%[***PROC\_SUMMARY\_ESG1***](#variables_66)(INPUT\_FILE=Picking\_file\_stkret\_win, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=PICKING\_PER\_STOCK\_MED, OUTPUT\_FILE=FILE\_5, STATS\_OPS=SUM, CALCULATED\_VAR=SUM\_ESG\_PICKING\_TBL\_A);

**PROC** **SUMMARY** DATA=Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESG=**0**;

CLASS FUNDID YEAR MONTH;

OUTPUT OUT = FILE\_6 SUM(PICKING\_PER\_STOCK\_MED)=SUM\_NON\_ESG\_PICKING\_TBL\_A;

**RUN**;

**PROC** **SORT** DATA = FILE\_5; BY FUNDID YEAR MONTH; **RUN**;

**PROC** **SORT** DATA = FILE\_6; BY FUNDID YEAR MONTH; **RUN**;

**DATA** FILE\_7;

MERGE FILE\_5(IN=A) FILE\_6(IN=B);

BY FUNDID YEAR MONTH;

IF A OR B;

DROP \_TYPE\_ \_FREQ\_;

**RUN**;

/\* MERGING ABOVE FILES AND REMOVING EXTRA COLUMNS \*/

**PROC** **SORT** DATA = FILE\_7; BY FUNDID YEAR MONTH; **RUN**;

**PROC** **SORT** DATA = FUNDLEVEL\_DATASET\_TRUNCATED; BY FUNDID YEAR MONTH; **RUN**;

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

MERGE FUNDLEVEL\_DATASET\_TRUNCATED(IN=A) FILE\_7(IN=B);

BY FUNDID YEAR MONTH;

IF A;

ESG\_MINUS\_NON\_ESG\_P =

SUM\_ESG\_PICKING\_TBL\_A - SUM\_NON\_ESG\_PICKING\_TBL\_A;

**RUN**;

\*\*\* ESG\_MINUS\_NON\_ESG\_T and ESG\_MINUS\_NON\_ESG\_P – both variables are winsorized at 1% levels on both tails before further calculations. \*\*\*

### ESGGE75\_MINUS\_OTHR\_T and ESGGE75\_MINUS\_OTHR\_P variables

Code logic – The code calculates the difference between timing and picking returns generated by holdings which have “ESG score greater than or equal to 75” and “ESG score less than 75”.

**Code A) Timing variables**

*Line no 3344 in the codefile*

/\* STEP 1: CREATING FLAG FOR ESGSCORE GE 75 \*/

**DATA** PICKING\_1;

SET PICKING\_FILE\_MED\_ESG\_STKRET;

IF ESGSCORE >= **75** THEN FLAG\_ESGGE=**1**;

ELSE FLAG\_ESGGE=**0**;

**RUN**;

/\* STEP 2: CALCULATING SUM TIMING FOR ESG GREATER THAN 75 SCORE STOCKS \*/

**PROC** **SUMMARY** DATA=PICKING\_1 NWAY MISSING;

WHERE FLAG\_ESGGE=**1**;

CLASS FUNDID YEAR MONTH;

OUTPUT OUT = FILE\_9 SUM(TIMING\_PER\_STOCK\_MED\_ESG)=SUM\_ESGGE75\_TIMING\_TBL\_B;

**RUN**;

/\* STEP 3: CALCULATING SUM TIMING FOR ESG GREATER THAN 75 SCORE STOCKS \*/

**PROC** **SUMMARY** DATA=PICKING\_1 NWAY MISSING;

WHERE FLAG\_ESGGE=**0**;

CLASS FUNDID YEAR MONTH;

OUTPUT OUT = FILE\_10 SUM(TIMING\_PER\_STOCK\_MED\_ESG)=SUM\_ESGOTHR\_TIMING\_TBL\_B;

**RUN**;

/\* *Line no 3379 in the codefile* **\*/**

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

MERGE FUNDLEVEL\_DATASET\_TRUNCATED(IN=A) FILE\_11(IN=B);

BY FUNDID YEAR MONTH;

IF A;

ESGGE75\_MINUS\_OTHR\_T =

SUM\_ESGGE75\_TIMING\_TBL\_B - SUM\_ESGOTHR\_TIMING\_TBL\_B;

**RUN**;

**Code B) Picking variables**

/\* *Line no 3440 in the codefile* **\*/**

/\* STEP 2: CALCULATING SUM TIMING FOR ESG GREATER THAN 75 SCORE STOCKS \*/

**PROC** **SUMMARY** DATA=PICKING\_1 NWAY MISSING;

WHERE FLAG\_ESGGE=**1**;

CLASS FUNDID YEAR MONTH;

OUTPUT OUT = FILE\_12 SUM(PICKING\_PER\_STOCK\_MED\_ESG)=SUM\_ESGGE75\_PICKING\_TBL\_B;

**RUN**;

/\* STEP 3: CALCULATING SUM TIMING FOR ESG GREATER THAN 75 SCORE STOCKS \*/

**PROC** **SUMMARY** DATA=PICKING\_1 NWAY MISSING;

WHERE FLAG\_ESGGE=**0**;

CLASS FUNDID YEAR MONTH;

OUTPUT OUT = FILE\_13 SUM(PICKING\_PER\_STOCK\_MED\_ESG)=SUM\_ESGOTHR\_PICKING\_TBL\_B;

**RUN**;

/\* *Line no 3469 in the codefile* **\*/**

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

MERGE FUNDLEVEL\_DATASET\_TRUNCATED(IN=A) FILE\_14(IN=B);

BY FUNDID YEAR MONTH;

IF A;

ESGGE75\_MINUS\_OTHR\_P =

SUM\_ESGGE75\_PICKING\_TBL\_B - SUM\_ESGOTHR\_PICKING\_TBL\_B;

**RUN**;

\*\*\* ESGGE75\_MINUS\_OTHR\_T and ESGGE75\_MINUS\_OTHR\_P – both variables are winsorized at 1% levels on both tails before further calculations. \*\*\*

### CALCULATING FORWARD ALPHAS

Code logic – The code generates the forward alphas for (t+1), (t+2), (t+3) and (t+12) months. The generated alphas are used in regression for calculaing predictability.

**Code A) dataset preparation**

/\* *Line no 3714 in the codefile* **\*/**

/\* STEP 1: IMPORTING ALPHAS FILE \*/

%[***IMPORT\_MACRO***](#variables_62)(INPUT\_FILE='R:\SAS - Research\SWARN\_RESEARCH\SAS\_DATASETS\FUND\_MONTH\_DATA\FUND\_NZD\Supporting\_files\Alphas\_file\_STKRET\_WIN.dta', DBMS\_TYPE=DTA, OUTPUT\_DATASET=Alphas\_file);

**PROC** **SORT** DATA = Alphas\_file; BY FUNDID YEAR MONTH; **RUN**;

**PROC** **CONTENTS** DATA = Alphas\_file; **RUN**;

/\* STEP 2: CLEANING DATA \*/

**DATA** ALPHAS\_FILE\_1;

SET ALPHAS\_FILE;

KEEP FUNDID YEAR MONTH ch4\_alpha;

**RUN**;

**Code B) variable calculations**

/\* STEP 3: CREATING FORWARD ALPHAS \*/

**PROC** **SORT** DATA = ALPHAS\_FILE\_1; BY FUNDID; **RUN**;

**PROC** **EXPAND** DATA=ALPHAS\_FILE\_1 OUT=EXPAND\_1 METHOD=NONE;

BY FUNDID;

CONVERT ch4\_alpha = CH4\_ALPHA\_1 / TRANSFORMOUT=(LEAD **1**);

CONVERT ch4\_alpha = CH4\_ALPHA\_3 / TRANSFORMOUT=(LEAD **3**);

CONVERT ch4\_alpha = CH4\_ALPHA\_6 / TRANSFORMOUT=(LEAD **6**);

CONVERT ch4\_alpha = CH4\_ALPHA\_12 / TRANSFORMOUT=(LEAD **12**);

**RUN**;

### CREATING FLAG FOR HIGH AND LOW TIMIING AND HIGH AND LOW PICKING FUNDS

Code logic – To calculate whether fund has outperformed market in generating timing and picking returns, we did the following code. First we calculated median picking and timing for the dataset for each year and month. Second we compared timing and picking generated by the fund with median picking and timing calculated for the year and month. Third, if timing or picking generated by fund is greater than median picking or timing, we flag it as high\_timing or high\_picking = 1 variable else 0.

*Line no 3749 in the codefile*

/\* STEP 1: CALCULATING MEDIAN TIMING \*/

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED, CLASS\_VARS=YEAR MONTH, VARIABLES=SUM\_TIMING\_MED\_WIN, OUTPUT\_FILE=FILE\_19, STATS\_OPS=MEDIAN, CALCULATED\_VAR=MEDIAN\_TIMING\_WIN);

/\* STEP 2: CALCULATING MEDIAN PICKING \*/

%[***PROC\_SUMMARY\_OVERALL***](#variables_63)(INPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED, CLASS\_VARS=YEAR MONTH, VARIABLES=SUM\_PICKING\_MED\_WIN, OUTPUT\_FILE=FILE\_20, STATS\_OPS=MEDIAN, CALCULATED\_VAR=MEDIAN\_PICKING\_WIN);

*Line no 3770 in the codefile*

/\* STEP 5: CREATING FLAG FOR HIGH AND LOW TIMIING AND HIGH AND LOW PICKING FUNDS \*/

**DATA** SET\_12;

SET FUNDLEVEL\_DATASET\_TRUNCATED;

IF SUM\_TIMING\_MED\_WIN = **.** THEN DELETE;

IF SUM\_TIMING\_MED\_WIN > MEDIAN\_TIMING\_WIN THEN HIGH\_TIMING = **1**;

ELSE HIGH\_TIMING = **0**;

KEEP FUNDID YEAR MONTH HIGH\_TIMING ;

**RUN**;

**DATA** SET\_14;

SET FUNDLEVEL\_DATASET\_TRUNCATED;

IF SUM\_PICKING\_MED\_WIN = **.** THEN DELETE;

IF SUM\_PICKING\_MED\_WIN > MEDIAN\_PICKING\_WIN THEN HIGH\_PICKING = **1**;

ELSE HIGH\_PICKING =**0**;

KEEP FUNDID YEAR MONTH HIGH\_PICKING;

**RUN**;

### AVERAGE CH4 VARIABLES IN HIGH AND LOW PICKING ENVIRONMENT

**The code is macro based as lot of variables with repeating nature is calculated.**

**The following macro is creating 3 variables(highlighted) in the dataset in one call.**

Code Logic – The code calculates average CH4 alpha variable generated by high timing or picking funds and low timing and picking funds for each year and month. In the next step, the code calculates the difference between these high timing and low timing returns generated for each year and month. The process is repeated for (t+1), (t+2), (t+3) and (t+12) ch4 alpha variables.

**Code A) Picking variables**

*Line no 3794 in the codefile*

/\* STEP 1: CREATING MACRO TO CREATE AVERAGE CH4 VARIABLES IN HIGH AND LOW PICKING ENVIRONMENT \*/

**%MACRO** [PICKING\_VARS\_ALPHA](#variables_74)(INPUT\_VAR, INPUT\_CHTIME);

%[***PROC\_SUMMARY\_HIGH\_PICKING***](#variables_70)(INPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED, CLASS\_VARS=YEAR MONTH, VARIABLES=&INPUT\_VAR., OUTPUT\_FILE=FILE\_1, STATS\_OPS=MEAN, CALCULATED\_VAR=YEAR\_&INPUT\_CHTIME.\_HIGH\_PICKING);

%[***PROC\_SUMMARY\_LOW\_PICKING***](#variables_71)(INPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED, CLASS\_VARS=YEAR MONTH, VARIABLES=&INPUT\_VAR., OUTPUT\_FILE=FILE\_2, STATS\_OPS=MEAN, CALCULATED\_VAR=YEAR\_&INPUT\_CHTIME.\_LOW\_PICKING);

%[***MERGE\_CODE\_2***](#variables_74)(INPUT\_FILE\_1=FILE\_1, INPUT\_FILE\_2=FILE\_2, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=FILE\_3);

DATA FILE\_4;

SET FILE\_3;

DROP \_TYPE\_ \_FREQ\_;

&INPUT\_CHTIME.\_HML\_PICKING = YEAR\_&INPUT\_CHTIME.\_HIGH\_PICKING - YEAR\_&INPUT\_CHTIME.\_LOW\_PICKING;

RUN;

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=FUNDLEVEL\_DATASET\_TRUNCATED, INPUT\_FILE\_2=FILE\_4, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED);

PROC DATASETS LIB = WORK; /\*DELETING INTERMEDIARY DATASET FOR AVOIDING FILE CONFUSION\*/

DELETE FILE\_1 FILE\_2 FILE\_3 FILE\_4;

QUIT;

**%MEND** [PICKING\_VARS\_ALPHA](#variables_74);

/\* STEP 2: CREATING PICKING VARIABLES - AVERAGE YEARLY \*/

%[***PICKING\_VARS\_ALPHA***](#variables_74)(INPUT\_VAR=CH4\_ALPHA\_1, INPUT\_CHTIME=CH4T1);

%[***PICKING\_VARS\_ALPHA***](#variables_74)(INPUT\_VAR=CH4\_ALPHA\_3, INPUT\_CHTIME=CH4T3);

%[***PICKING\_VARS\_ALPHA***](#variables_74)(INPUT\_VAR=CH4\_ALPHA\_6, INPUT\_CHTIME=CH4T6);

%[***PICKING\_VARS\_ALPHA***](#variables_74)(INPUT\_VAR=CH4\_ALPHA\_12, INPUT\_CHTIME=CH4T12);

**Code A) Timing variables**

*Line no 3824 in the codefile*

/\* STEP 3: CREATING MACRO TO CREATE AVERAGE CH4 VARIABLES IN HIGH AND LOW TIMING ENVIRONMENT \*/

**%MACRO** [TIMING\_VARS\_ALPHA](#variables_75)(INPUT\_VAR, INPUT\_CHTIME);

%[***PROC\_SUMMARY\_HIGH\_TIMING***](#variables_72)(INPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED, CLASS\_VARS=YEAR MONTH, VARIABLES=&INPUT\_VAR., OUTPUT\_FILE=FILE\_1, STATS\_OPS=MEAN, CALCULATED\_VAR=YEAR\_&INPUT\_CHTIME.\_HIGH\_TIMING);

%[***PROC\_SUMMARY\_LOW\_TIMING***](#variables_73)(INPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED, CLASS\_VARS=YEAR MONTH, VARIABLES=&INPUT\_VAR., OUTPUT\_FILE=FILE\_2, STATS\_OPS=MEAN, CALCULATED\_VAR=YEAR\_&INPUT\_CHTIME.\_LOW\_TIMING);

%[***MERGE\_CODE\_2***](#variables_74)(INPUT\_FILE\_1=FILE\_1, INPUT\_FILE\_2=FILE\_2, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=FILE\_3);

DATA FILE\_4;

SET FILE\_3;

DROP \_TYPE\_ \_FREQ\_;

&INPUT\_CHTIME.\_HML\_TIMING = YEAR\_&INPUT\_CHTIME.\_HIGH\_TIMING -YEAR\_&INPUT\_CHTIME.\_LOW\_TIMING;

RUN;

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=FUNDLEVEL\_DATASET\_TRUNCATED, INPUT\_FILE\_2=FILE\_4, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED);

PROC DATASETS LIB = WORK; /\*DELETING INTERMEDIARY DATASET FOR AVOIDING FILE CONFUSION\*/

DELETE FILE\_1 FILE\_2 FILE\_3 FILE\_4;

QUIT;

**%MEND** [TIMING\_VARS\_ALPHA](#variables_75);

/\* STEP 4: CREATING PICKING VARIABLES - AVERAGE YEARLY \*/

%[***TIMING\_VARS\_ALPHA***](#variables_75)(INPUT\_VAR=CH4\_ALPHA\_1, INPUT\_CHTIME=CH4T1);

%[***TIMING\_VARS\_ALPHA***](#variables_75)(INPUT\_VAR=CH4\_ALPHA\_3, INPUT\_CHTIME=CH4T3);

%[***TIMING\_VARS\_ALPHA***](#variables_75)(INPUT\_VAR=CH4\_ALPHA\_6, INPUT\_CHTIME=CH4T6);

%[***TIMING\_VARS\_ALPHA***](#variables_75)(INPUT\_VAR=CH4\_ALPHA\_12, INPUT\_CHTIME=CH4T12);

### ESG FUNDS

This code defines ESG funds.

*Line no 4349 in the codefile*

/\* \*\*\* CREATING ESG FLAG FOR STATA \*\*\* \*/

**PROC** **SORT** DATA = File\_set.Fundlevel\_dataset\_stkret\_win OUT = FUNDLEVEL\_DATASET\_TRUNCATED; BY FUNDID YEAR MONTH; **RUN**;

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

SET FUNDLEVEL\_DATASET\_TRUNCATED;

IF NZD\_ESG\_WT\_OF\_FUND >= **50** THEN ESG\_FUND = **1**;

ELSE ESG\_FUND = **0**;

**RUN**;

### CREATING EXTREME VARIABLE - WHICH IS 1 WHEN THE NZX50 IS BELOW -5% OR ABOVE +5% MONTHLY RETURNS, OTHERWISE ZERO

*Line no 4648 in the codefile*

/\* CREATING EXTREME VARIABLE - WHICH IS 1 WHEN THE NZX50 IS BELOW -5% OR ABOVE +5% MONTHLY RETURNS, OTHERWISE ZERO \*/

**DATA** NZX50\_RETURNS;

SET Fund\_set.Nzx50returns\_nzd;

DROP FUNDID;

**RUN**;

%[***MERGE\_CODE\_1***](#variables_60)(INPUT\_FILE\_1=File\_set.Fundlevel\_dataset\_stkret\_win, INPUT\_FILE\_2=NZX50\_RETURNS, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED);

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

SET FUNDLEVEL\_DATASET\_TRUNCATED;

IF NZX50returns\_NZD1 >= **5** OR NZX50returns\_NZD1 <= -**5** THEN EXTREME = **1**;

ELSE EXTREME = **0**;

**RUN**;

### ESG TRADING VARIABLES

Code A) dataset creation

*Line no 4690 in the codefile*

**DATA** NZD\_DATASET\_FINAL;

SET File\_set.Nzd\_dataset\_final\_stkret\_win;

**RUN**;

**DATA** MONTHLYRETURNS\_DATA;

SET File\_set.Refinitiv\_all\_data;

**RUN**;

**Code B) creating lag values for stockreturns in the portfolio and netassets of the fund**

/\* CREATING LAG VALUES FOR MONTHLYRETURNS \*/

**PROC** **SORT** DATA = MONTHLYRETURNS\_DATA; BY SECID YEAR MONTH; **RUN**;

**DATA** STOCK\_RETURNS;

SET MONTHLYRETURNS\_DATA;

LAG\_STKRETURNS = LAG(monthlyreturns);

IF FIRST.SECID THEN DO;

LAG\_STKRETURNS = **.**;

END;

KEEP SECID YEAR MONTH LAG\_STKRETURNS;

**RUN**;

/\* CREATING LAG VALUES FOR NETASSETS \*/

**DATA** NETASSETS\_DATA;

SET File\_set.Fundlevel\_merge\_stkret\_win;

**RUN**;

/\* CREATING LAG VALUES FOR NETASSETS \*/

**DATA** NETASSETS\_LAG\_DATA;

SET NETASSETS\_DATA;

LAG\_NETASSETS = LAG(Netassets\_NZD);

IF FIRST.FUNDID THEN DO;

LAG\_NETASSETS = **.**;

END;

KEEP FUNDID YEAR MONTH LAG\_NETASSETS;

**RUN**;

**Code C) Creating each holdings flag in the portfolio in created dataset.**

**DATA** SET\_2;

SET SET\_1;

BY FUNDID SECID YEAR MONTH;

LAG\_Percentholdings = LAG(Percentholdings);

LAG\_WT\_HLDGS\_DEC = LAG(STOCK\_WEIGHT\_DECIMAL);

/\* INITIALIZE BOTH THESE FLAGS \*/

Fund\_Old\_Position\_in\_stk = **1**;

Fund\_New\_Position\_in\_stk = **0**;

IF FIRST.FUNDID OR FIRST.SECID THEN DO;

LAG\_Percentholdings = **0**;

LAG\_WT\_HLDGS\_DEC = **0**;

Fund\_New\_Position\_in\_stk = **1**;

Fund\_Old\_Position\_in\_stk = **0**;

END;

/\* INITIALIZE SELL OFF FLAG \*/

IF LAST.FUNDID OR LAST.SECID THEN DO;

IF YY\_MON < LIQ\_YR\_MON THEN DO:

Stk\_postion\_sold\_off = **1**;

END;

ELSE DO;

Stk\_postion\_sold\_off = **0**;

END;

**RUN**;

**Code D) Creating variables at stock level.**

**DATA** SET\_3;

SET SET\_2;

BY FUNDID SECID YEAR MONTH;

PERCENT\_CHNG\_WT = Percentholdings - LAG\_Percentholdings; /\* EQUATION IS (PERCENT \* PERCENT ) \*/

TRADING\_RET = PERCENT\_CHNG\_WT \* LAG\_STKRETURNS;

/\* ALL VARIABLES ARE IN PERCENTAGES, ANSWER IS IN DECIMAL \*/

CHNG\_WT\_HLDGS\_DEC = STOCK\_WEIGHT\_DECIMAL - LAG\_WT\_HLDGS\_DEC; /\* EQUATION IS (DECIMAL \* DECIAML) \*/

TRADING\_VAL = CHNG\_WT\_HLDGS\_DEC \* LAG\_NETASSETS;

/\* ALL VARIABLES ARE IN DECIMAL POINTS \*/

**RUN**;

**Code E) Creating variables at Fund level – a)overall holdings b)ESG holdings c)holdings where ESGscore >= 75 d) holdings where ESGscore <= 25.**

/\* SUMMATION OF FUND TRADING RETURN AND FUND TRADING VALUE \*/

**PROC** **SORT** DATA = SET\_3; BY FUNDID YEAR MONTH; **RUN**;

**PROC** **SUMMARY** DATA = SET\_3 NWAY MISSING;

CLASS FUNDID YEAR MONTH;

VAR TRADING\_RET TRADING\_VAL;

OUTPUT OUT = DATA\_1(DROP = \_TYPE\_ \_FREQ\_)

SUM(TRADING\_RET) = FUND\_TRADING\_RET

SUM(TRADING\_VAL) = FUND\_TRADING\_VAL;

**RUN**;

/\* SUMMATION OF FUND ESG TRADING WHERE ESGSCORE IS GE 75 \*/

**PROC** **SUMMARY** DATA = SET\_3 NWAY MISSING;

WHERE NOT MISSING(ESGSCORE);

CLASS FUNDID YEAR MONTH;

VAR TRADING\_RET TRADING\_VAL;

OUTPUT OUT = DATA\_2(DROP = \_TYPE\_ \_FREQ\_)

SUM(TRADING\_RET) = ESG\_TRDNG\_RET

SUM(TRADING\_VAL) = ESG\_TRDNG\_VALUE;

**RUN**;

/\* SUMMATION OF FUND ESG TRADING WHERE ESGSCORE IS GE 75 \*/

/\* SUMMATION OF FUND ESG TRADING WHERE ESGSCORE IS GE 25 \*/

**PROC** **SUMMARY** DATA = SET\_3 NWAY MISSING;

WHERE ESGSCORE >= **75**;

CLASS FUNDID YEAR MONTH;

VAR TRADING\_RET TRADING\_VAL;

OUTPUT OUT = DATA\_3(DROP = \_TYPE\_ \_FREQ\_)

SUM(TRADING\_RET) = HIGH\_ESG\_TRDNG\_RET

SUM(TRADING\_VAL) = HIGH\_ESG\_TRDNG\_VALUE;

**RUN**;

**PROC** **SUMMARY** DATA = SET\_3 NWAY MISSING;

WHERE ESGSCORE <= **25**;

CLASS FUNDID YEAR MONTH;

VAR TRADING\_RET TRADING\_VAL;

OUTPUT OUT = DATA\_4(DROP = \_TYPE\_ \_FREQ\_)

SUM(TRADING\_RET) = LOW\_ESG\_TRDNG\_RET

SUM(TRADING\_VAL) = LOW\_ESG\_TRDNG\_VALUE;

**RUN**;

**Code F) Creating ratio variables at Fund level – a)overall holdings b)ESG holdings c)holdings where ESGscore >= 75 d) holdings where ESGscore <= 25. Also creating flag variables.**

/\* CREATING ESG TRADING VARIABLE \*/

**DATA** TRADING\_DATA;

SET MERGE\_AVG;

RATIO\_HIGH\_ESG\_TRADING\_RET = (HIGH\_ESG\_TRDNG\_RET / FUND\_TRADING\_RET) \***100**;

RATIO\_HIGH\_ESG\_TRADING\_VALUE = (HIGH\_ESG\_TRDNG\_VALUE / FUND\_TRADING\_VAL) \* **100**;

RATIO\_LOW\_ESG\_TRADING\_RET = (LOW\_ESG\_TRDNG\_RET / FUND\_TRADING\_RET) \* **100**;

RATIO\_LOW\_ESG\_TRADING\_VALUE = (LOW\_ESG\_TRDNG\_VALUE / FUND\_TRADING\_VAL) \* **100**;

HML\_ESG\_TRADING\_RET = COALESCE(HIGH\_ESG\_TRDNG\_RET,**0**) - COALESCE(LOW\_ESG\_TRDNG\_RET,**0**);

HML\_ESG\_TRADING\_VALUE = COALESCE(HIGH\_ESG\_TRDNG\_VALUE,**0**) - COALESCE(LOW\_ESG\_TRDNG\_VALUE,**0**);

/\* INITIALIZE FLAG VARIABLES \*/

HIGH\_ESG\_GREATER\_TRADING\_VALUE = **0**;

LOW\_ESG\_GREATER\_TRADING\_VALUE = **0**;

IF HML\_ESG\_trading\_value > **0** THEN DO;

HIGH\_ESG\_GREATER\_TRADING\_VALUE = **1**;

END;

IF HML\_ESG\_trading\_value < **0** THEN DO;

LOW\_ESG\_GREATER\_TRADING\_VALUE = **1**;

END;

**RUN**;

### ESG FLOWS FOR GREATER THAN 75 – fund flows calculations

Code logic – The code is divided into multiple parts. The code is used to calculate fund flows received by holdings which have ESG score greater than or equal to 75 in the fund portfolio for specified year and month.

**Code A) Creating fund flows where the ESG score is greater than 75**.

The calculated final values are winsorized at p1 and p99 levels.

*Line no 5100 in the codefile.*

/\* ESG FLOWS FOR GREATER THAN 75 \*/

/\* STEP (6.4) : CALCULATING VALUE WEIGHTED ESG FUND RETURN DECOMPOSED PER FUND PER MONTH PER YEAR \*/

**PROC** **SUMMARY** DATA = NAD\_DATASET\_FINAL NWAY MISSING;

WHERE ESGSCORE >= **75** ;

CLASS FUNDID YEAR MONTH;

WEIGHT CAPITAL\_INVESTED\_PER\_HOLDING;

VAR DECOMPOSED\_STK\_RET;

OUTPUT OUT = VW\_ESG\_RETURNS\_DECOMP MEAN (DECOMPOSED\_STK\_RET) = VW\_ESG\_RETS\_DECOMP\_GE75;

**RUN**;

/\*STEP (8.1) : CALCULATING SUM\_ESG\_CAPITAL OF EACH FUND PER MONTH AND YEAR FOR OVERALL DATA\*/

**PROC** **SUMMARY** DATA = NAD\_DATASET\_FINAL NWAY MISSING;

WHERE ESGSCORE >= **75** ;

CLASS FUNDID YEAR MONTH;

VAR CAPITAL\_INVESTED\_PER\_HOLDING;

OUTPUT OUT = SUM\_ESG\_CAP\_GE75 SUM (CAPITAL\_INVESTED\_PER\_HOLDING) = SUM\_ESG\_CAP\_GE75;

**RUN**;

/\* STEP (11.2) : FUND ESG FLOW CALCULATIONS \*/

**DATA** ESGGE75\_FUND\_FLOW;

SET ESG\_GE75\_FLOW\_CALSET;

BY FUNDID;

LAGGED\_ESGNETASSET = LAG(SUM\_ESG\_CAP\_GE75);

IF FIRST.FUNDID THEN LAGGED\_ESGNETASSET = **0**;

CHANGE\_ESGNETASSET = SUM\_ESG\_CAP\_GE75 - LAGGED\_ESGNETASSET;

IF NOT MISSING(SUM\_ESG\_CAP\_GE75) AND SUM\_ESG\_CAP\_GE75 NE **0** AND NOT MISSING(CHANGE\_ESGNETASSET) THEN DO;

CHANGE\_ESGNETASSET\_PERCENT = (CHANGE\_ESGNETASSET / LAGGED\_ESGNETASSET) \* **100** ;

END;

ELSE CHANGE\_ESGNETASSET\_PERCENT = **.**;

IF NOT MISSING(VW\_ESG\_RETS\_DECOMP\_GE75) AND VW\_ESG\_RETS\_DECOMP\_GE75 NE **0** THEN DO;

VW\_ESG\_RETURNS\_DECIMAL = VW\_ESG\_RETS\_DECOMP\_GE75/**100**;

END;

ELSE VW\_ESG\_RETURNS\_DECIMAL = **.**;

ESGGE75\_DOLLAR\_FLOW = (SUM\_ESG\_CAP\_GE75 - (LAGGED\_ESGNETASSET \* (**1** + VW\_ESG\_RETURNS\_DECIMAL))) ;

IF NOT MISSING(LAGGED\_ESGNETASSET) AND LAGGED\_ESGNETASSET NE **0** AND NOT MISSING(ESGGE75\_DOLLAR\_FLOW) THEN DO;

ESGGE75\_FLOW\_PRCNT = (ESGGE75\_DOLLAR\_FLOW / LAGGED\_ESGNETASSET) \* **100**;

END;

ELSE ESGGE75\_FLOW\_PRCNT = **.**;

**RUN**;

**Code B) Creating ratios of the above calculated values with overall fund flows.**

The calculated final values are winsorized at p1 and p99 levels.

**DATA** TEST\_2;

MERGE TEST\_1(IN=A) SET\_1(IN=B);

BY FUNDID YEAR MONTH;

IF A;

HIGH\_ESG\_FLOW\_RATIO\_DOLLAR = ESGGE75\_DOLLAR\_FLOW / FLOW\_DOLLAR\_VALUE\_NZD;

HIGH\_ESG\_FLOW\_RATIO\_PRCNT = ESGGE75\_FLOW\_PRCNT\_WIN / FLOW\_PERCENT\_WIN\_NZD;

**RUN**;

### VALUE WEIGHTED ESG FUND RETURN – {VW\_HIGH\_ESG\_RETURNS } – where ESG score is GE 75

Code logic – The code calculates value weighted returns for holdings which have ESG score ratings greater than or equal to 75. The weight variable is the capital invested per holding.

*Line no 5100 in the codefile.*

/\* STEP (6.1) : CALCULATING VALUE WEIGHTED ESG FUND RETURN PER FUND PER MONTH PER YEAR \*/

**PROC** **SUMMARY** DATA = NAD\_DATASET\_FINAL NWAY MISSING;

WHERE ESG = **1** AND MONTHLYRETURNS NE **.** AND ESGSCORE >= **75**;

CLASS FUNDID YEAR MONTH;

WEIGHT CAPITAL\_INVESTED\_PER\_HOLDING;

VAR MONTHLYRETURNS;

OUTPUT OUT = ESG\_RETURNS\_VW

MEAN (MONTHLYRETURNS) = VW\_HIGH\_ESG\_RETURNS;

**RUN**;

### CREATING RANKING FOR EACH FUND FOR TIMING VARIABLES – {QUINTILE\_5\_BOTH, QUINTILE\_1\_BOTH}

Code logic – The code divided the data into 5 equal parts by ranking. The quintile 1 is lowest 20% of values (i.e. these funds have generated lowest 20% timing or picking values in the dataset) and quintile 5 is highest 20%(i.e. these funds have generated highest 20% either timing or picking values in the dataset).

In next code, we created a flag variable with \_BOTH for quintile 1 and 5. It classifies funds which are in top 20% in both timing and picking and bottom 20% in both timing and picking.

*Line no 4909 in the codefile*

**DATA** Fundlevel\_p1\_dataset;

SET File\_set.Fundlevel\_p1\_dataset;

**RUN**;

/\* CREATING RANKING FOR EACH FUND FOR TIMING VARAIBLES \*/

**PROC** **SORT** DATA = Fundlevel\_p1\_dataset; BY YEAR MONTH; **RUN**;

**PROC** **RANK** DATA = Fundlevel\_p1\_dataset OUT = DATA\_1 GROUPS = **5**;

BY YEAR MONTH;

VAR SUM\_TIMING\_MED\_WIN;

RANKS TIMING\_RANK;

**RUN**;

/\* CREATING RANKING FOR EACH FUND FOR PICKING VARAIBLES \*/

**PROC** **SORT** DATA = DATA\_1; BY YEAR MONTH; **RUN**;

**PROC** **RANK** DATA = DATA\_1 OUT = DATA\_2 GROUPS = **5**;

BY YEAR MONTH;

VAR SUM\_PICKING\_MED\_WIN;

RANKS PICKING\_RANK;

**RUN**;

/\* RANKING VARIABLE NAMING 1 TO 5 IN ASCENDING ORDER \*/

**DATA** Fundlevel\_p1\_dataset;

SET DATA\_2;

TIMING\_RANK = TIMING\_RANK+**1**;

PICKING\_RANK = PICKING\_RANK+**1**;

**RUN**;

/\* COMPARING RANKING OF TIMING AND PICKING AND CREATING FLAG \*/

**DATA** Fundlevel\_p1\_dataset;

SET Fundlevel\_p1\_dataset;

QUINTILE\_5\_BOTH = **0**;

QUINTILE\_1\_BOTH = **0**;

IF (TIMING\_RANK=**5**) AND (PICKING\_RANK = **5**) THEN DO;

QUINTILE\_5\_BOTH = **1**; /\* top quintile \*/

END;

IF (TIMING\_RANK = **1**) AND (PICKING\_RANK = **1**) THEN DO;

QUINTILE\_1\_BOTH = **1**; /\* bottom quintile \*/

END;

**RUN**;

### ESG\_MINUS\_NON\_ESG\_P

Code logic – This code calculates the a) total picking generated by funds which have ESG score ratings and b) total picking generated by funds which don’t have ESG ratings. In the next step the code calculates c) the difference between the two variables and d) the ratio of picking generated by ESG rated stocks with overall picking generated by the fund. The code is same as variable no 32.

*Line no 5528 in the codefile*

**Code a)** creating decomposed picking variable which is sum of all ESG rated holdings for picking variable.

/\* Step 1: CREATING DECOMPOSED ESG PICKING VARIABLE \*/

%[***PROC\_SUMMARY\_ESG1***](#variables_66)(INPUT\_FILE=Picking\_file\_stkret\_win, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=PICKING\_PER\_STOCK\_MED, OUTPUT\_FILE=FILE\_5, STATS\_OPS=SUM, CALCULATED\_VAR=PICKING\_ESG\_DECOMP);

**Code b)** creating decomposed picking variable which is sum of all non – ESG rated holdings for picking variable. Also, winsorize the both values at p1 and p99.

/\* STEP 2: CREATING DECOMPOSED NON ESG PICKING VARIABLE \*/

**PROC** **SUMMARY** DATA=Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESG=**0**;

CLASS FUNDID YEAR MONTH;

OUTPUT OUT = FILE\_6 SUM(PICKING\_PER\_STOCK\_MED)=PICKING\_NON\_ESG\_DECOMP;

**RUN**;

**Code c)** calculate the variable ESG minus NON ESG picking.

/\* STEP 7: CREATING RATIO VARIABLE - ESG\_PICKING/OVERALL\_PICKING \*/

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

SET FUNDLEVEL\_DATASET\_TRUNCATED;

ESG\_MINUS\_NON\_ESG\_P = COALESCE(PICKING\_ESG\_DECOMP\_WIN, **0**) - COALESCE(PICKING\_NON\_ESG\_DECOMP\_WIN, **0**);

RATIO\_ESG\_OVERALL\_P = PICKING\_ESG\_DECOMP\_WIN/SUM\_PICKING\_MED\_WIN;

**RUN**;

### PICKING\_STKS\_Q1\_SUM, PICKING\_STKS\_Q1\_WIN

Code logic - The code calculates sum of picking of all holdings which have ESG score <= 25 and is classified as Q1. The calculated variable is winsorized at p1 and p99 levels.

*Line no 5594 in the codefile*

/\* ESGSCORE =< 25 \*/

**PROC** **SUMMARY** DATA = Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESGSCORE <= **25**;

CLASS FUNDID YEAR MONTH;

VAR PICKING\_PER\_STOCK\_MED;

OUTPUT OUT = DATA\_18(DROP = \_TYPE\_ \_FREQ\_) SUM(PICKING\_PER\_STOCK\_MED) = PICKING\_STKS\_Q1\_SUM;

**RUN**;

### PICKING\_STKS\_Q5\_SUM, PICKING\_STKS\_Q5\_WIN

Code logic- The code calculates sum of picking of all holdings which have ESG score >= 75 and is classified as Q5. The calculated variable is winsorized at p1 and p99 levels.

*Line no 5560 in the codefile*

/\* ESGSCORE >= 75 \*/

**PROC** **SUMMARY** DATA = Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESGSCORE >= **75**;

CLASS FUNDID YEAR MONTH;

VAR PICKING\_PER\_STOCK\_MED;

OUTPUT OUT = DATA\_19(DROP = \_TYPE\_ \_FREQ\_) SUM(PICKING\_PER\_STOCK\_MED) = PICKING\_STKS\_Q5\_SUM;

**RUN**;

### PICKING\_Q5\_MINUS\_Q1

Code logic - The code calculates the difference between picking of Q5 stocks(ESG score ge 75) and Q1(ESG score le 25) stocks.

*Line no 5641 in the codefile*

/\* CALCULATING RATIOS \*/

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

SET FUNDLEVEL\_DATASET\_TRUNCATED;

PICKING\_Q5\_MINUS\_Q1 = COALESCE(PICKING\_STKS\_Q5\_WIN, **0**) - COALESCE(PICKING\_STKS\_Q1\_WIN, **0**);

RATIO\_PICKING\_Q1\_OVERALL = PICKING\_STKS\_Q1\_WIN/SUM\_PICKING\_MED\_WIN;

RATIO\_PICKING\_Q5\_OVERALL = PICKING\_STKS\_Q5\_WIN/SUM\_PICKING\_MED\_WIN;

**RUN**;

1. ESG\_MINUS\_NON\_ESG\_TIM

Code logic – This code calculates the a) total timing generated by funds which have ESG score ratings and b) total timing generated by funds which don’t have ESG ratings. In the next step the code calculates c) the difference between the two variables and d) the ratio of timing generated by ESG rated stocks with overall timing generated by the fund. The code is same as variable no 32.

*Line no 5673 in the codefile*

**Code a)** creating decomposed timing variable which is sum of all ESG rated holdings for timing variable.

/\* Step 1: CREATING DECOMPOSED ESG TIMING VARIABLE \*/

%[***PROC\_SUMMARY\_ESG1***](#variables_66)(INPUT\_FILE=Picking\_file\_stkret\_win, CLASS\_VARS=FUNDID YEAR MONTH, VARIABLES=TIMING\_PER\_STOCK\_MED, OUTPUT\_FILE=FILE\_1, STATS\_OPS=SUM, CALCULATED\_VAR=TIMING\_ESG\_DECOMP);

**Code b)** creating decomposed timing variable which is sum of all non – ESG rated holdings for timing variable. Also, winsorize the both values at p1 and p99.

**PROC** **SUMMARY** DATA=Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESG=**0**;

CLASS FUNDID YEAR MONTH;

OUTPUT OUT = FILE\_2 SUM(TIMING\_PER\_STOCK\_MED)=TIMING\_NON\_ESG\_DECOMP;

**RUN**;

**Code c)** calculate the variable ESG minus NON ESG timing.

/\* STEP 8: CREATING RATIO VARIABLE - ESG\_PICKING/OVERALL\_PICKING \*/

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

SET FUNDLEVEL\_DATASET\_TRUNCATED;

ESG\_MINUS\_NON\_ESG\_TIM = COALESCE(TIMING\_ESG\_DECOMP\_WIN, **0**) - COALESCE(TIMING\_NON\_ESG\_DECOMP\_WIN, **0**);

RATIO\_ESG\_OVERALL\_TIM = TIMING\_ESG\_DECOMP\_WIN/SUM\_TIMING\_MED\_WIN;

**RUN**;

### 48) TIMING\_STKS\_Q1\_SUM, TIMING\_STKS\_Q1\_WIN

The code calculates sum of timing of all holdings which have ESG score <= 25 and is classified as Q1. The calculated variable is winsorized at p1 and p99 levels.

*Line no 5734 in the codefile*

/\* ESGSCORE =< 25 \*/

**PROC** **SUMMARY** DATA = Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESGSCORE <= **25**;

CLASS FUNDID YEAR MONTH;

VAR TIMING\_PER\_STOCK\_MED;

OUTPUT OUT = DATA\_20(DROP = \_TYPE\_ \_FREQ\_) SUM(TIMING\_PER\_STOCK\_MED) = TIMING\_STKS\_Q1\_SUM;

**RUN**;

### 49) TIMING\_STKS\_Q5\_SUM, TIMING\_STKS\_Q5\_WIN

The code calculates sum of picking of all holdings which have ESG score >= 75 and is classified as Q5. The calculated variable is winsorized at p1 and p99 levels.

*Line no 5742 in the codefile*

/\* ESGSCORE >= 75 \*/

**PROC** **SUMMARY** DATA = Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESGSCORE >= **75**;

CLASS FUNDID YEAR MONTH;

VAR TIMING\_PER\_STOCK\_MED;

OUTPUT OUT = DATA\_21(DROP = \_TYPE\_ \_FREQ\_) SUM(TIMING\_PER\_STOCK\_MED) = TIMING\_STKS\_Q5\_SUM;

**RUN**;

### 50) TIMING\_Q5\_MINUS\_Q1

The code calculates the difference between timing of Q5 stocks(ESG score GE 75) and Q1 stocks(ESG score LE 25).

*Line no 5783 in the codefile*

/\* CALCULATING RATIOS \*/

**DATA** FUNDLEVEL\_DATASET\_TRUNCATED;

SET FUNDLEVEL\_DATASET\_TRUNCATED;

TIMING\_Q5\_MINUS\_Q1 = COALESCE(TIMING\_STKS\_Q5\_WIN, **0**) - COALESCE(TIMING\_STKS\_Q1\_WIN, **0**);

RATIO\_TIMING\_Q1\_OVERALL = TIMING\_STKS\_Q1\_WIN/SUM\_TIMING\_MED\_WIN;

RATIO\_TIMING\_Q5\_OVERALL = TIMING\_STKS\_Q5\_WIN/SUM\_TIMING\_MED\_WIN;

**RUN**;

### 51) Q5\_FUNDS – definition

The dataset created is a temporary dataset in work library for table calculations.

*Line no 5222 in the codefile*

**DATA** Q5\_FUNDS;

SET File\_set.TEST\_FUNDLEVEL\_20\_NOV\_WIN\_SET;

WHERE QUINTILE\_5\_BOTH = **1**;

IF MISSING(RATIO\_HIGH\_ESG\_TRADING\_RET) AND NOT MISSING(RATIO\_HIGH\_ESG\_TRADING\_VALUE) THEN DELETE;

IF MISSING(RATIO\_HIGH\_ESG\_TRADING\_VALUE) AND NOT MISSING(RATIO\_HIGH\_ESG\_TRADING\_RET) THEN DELETE;

**RUN**;

### 52) Q1\_FUNDS - definition

The dataset created is a temporary dataset in work library for table calculations.

*Line no 5236 in the codefile*

**DATA** Q1\_FUNDS;

SET File\_set.TEST\_FUNDLEVEL\_20\_NOV\_WIN\_SET;

WHERE QUINTILE\_1\_BOTH = **1**;

IF MISSING(RATIO\_HIGH\_ESG\_TRADING\_RET) AND NOT MISSING(RATIO\_HIGH\_ESG\_TRADING\_VALUE) THEN DELETE;

IF MISSING(RATIO\_HIGH\_ESG\_TRADING\_VALUE) AND NOT MISSING(RATIO\_HIGH\_ESG\_TRADING\_RET) THEN DELETE;

**RUN**;

### 53) Definition of overall, kiwisaver and open end funds

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* OVERALL FUNDS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

/\* MODEL 1 \*/

reghdfe SUM\_PICKING\_MED\_WIN\_scaled NZD\_ESG\_WT\_OF\_FUND\_SCALED log\_age log\_tna TOTAL\_COST\_RATIO if p1!=., absorb(FundID FIRM\_NAME\_FLAG)

est store t1

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* KIWISAVER FUNDS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/

/\* MODEL 1 \*/

reghdfe SUM\_PICKING\_MED\_WIN\_scaled NZD\_ESG\_WT\_OF\_FUND\_SCALED log\_age log\_tna TOTAL\_COST\_RATIO if (p1 != .) & (Binaryvar == 1 | Binaryvar == 2), absorb(FundID FIRM\_NAME\_FLAG)

est store t1

### 54) TOP\_QUINTILE\_PICKING – dataset creation

The dataset created is used in regression in stata for top quintile picking.

Line no 5325 in the codefile.

/\* Subset data for top quintile (RANK 5) FUNDS \*/

/\* PICKING \*/

**DATA** File\_set.TOP\_QUINTILE\_PICKING;

SET File\_set.Test\_fundlevel\_20\_nov\_win\_set;

WHERE PICKING\_RANK = **5**;

**RUN**;

**PROC** **SORT** DATA = File\_set.TOP\_QUINTILE\_PICKING; BY FUNDID YEAR MONTH; **RUN**;

### 55) TOP\_QUINTILE\_TIMING – dataset creation

The dataset created is used in regression in stata for top quintile timing.

Line no 5325 in the codefile.

/\* TIMING \*/

**DATA** File\_set.TOP\_QUINTILE\_TIMING;

SET File\_set.Test\_fundlevel\_20\_nov\_win\_set;

WHERE TIMING\_RANK = **5**;

**RUN**;

**PROC** **SORT** DATA = File\_set.TOP\_QUINTILE\_TIMING; BY FUNDID YEAR MONTH; **RUN**;

### 56) BOTTOM\_QUINTILE\_PICKING – dataset creation

The dataset created is used in regression in stata for bottom quintile picking.

Line no 5340 in the codefile.

/\* SUBSET DATA FOR BOTTOM QUINTILE (RANK 1) FUNDS \*/

/\* PICKING \*/

**DATA** File\_set.BOTTOM\_QUINTILE\_PICKING;

SET File\_set.Test\_fundlevel\_20\_nov\_win\_set;

WHERE PICKING\_RANK = **1**;

**RUN**;

**PROC** **SORT** DATA = File\_set.BOTTOM\_QUINTILE\_PICKING; BY FUNDID YEAR MONTH; **RUN**;

### 57) BOTTOM\_QUINTILE\_TIMING – dataset creation

The dataset created is used in regression in stata for bottom quintile timing.

Line no 5347 in the codefile.

/\* TIMING \*/

**DATA** File\_set.BOTTOM\_QUINTILE\_TIMING;

SET File\_set.Test\_fundlevel\_20\_nov\_win\_set;

WHERE TIMING\_RANK = **1**;

**RUN**;

**PROC** **SORT** DATA = File\_set.BOTTOM\_QUINTILE\_TIMING; BY FUNDID YEAR MONTH; **RUN**;

### 58) ESG DATASET CREATION

The dataset created is used in regression in Stata for ESG-motivated funds calculations.

Line no 5347 in the codefile.

/\* ESG DATASET CREATION \*/

**PROC** **SUMMARY** DATA=FUNDLEVEL\_P1\_DATASET NWAY MISSING ;

CLASS YEAR MONTH;

VAR ESGGE75\_SUM\_WT\_NZD;

OUTPUT OUT = FILE\_1 /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

MEDIAN(ESGGE75\_SUM\_WT\_NZD) = ESGGE\_75\_MEDIAN;

**RUN**;

**PROC** **SORT** DATA = FUNDLEVEL\_P1\_DATASET OUT = LINK\_FILE; BY YEAR MONTH; **RUN**;

**PROC** **SORT** DATA = FILE\_1; BY YEAR MONTH; **RUN**;

**DATA** FUNDLEVEL\_P1\_DATASET;

MERGE LINK\_FILE (IN=A) FILE\_1 (IN=B);

BY YEAR MONTH;

IF A;

DROP \_TYPE\_ \_FREQ\_;

**RUN**;

**DATA** SET\_12;

SET FUNDLEVEL\_P1\_DATASET;

IF ESGGE\_75\_MEDIAN = **.** THEN DELETE;

IF ESGGE75\_SUM\_WT\_NZD > ESGGE\_75\_MEDIAN THEN ESG\_75\_FUND = **1**;

ELSE ESG\_75\_FUND = **0**;

**RUN**;

**DATA** File\_set.FUNDLEVEL\_ESG75\_DATASET;

SET SET\_12;

WHERE ESG\_75\_FUND = **1**;

**RUN**;

### 59) “test\_fundlevel\_20\_nov\_win\_set” – main fundlevel dataset

This is the fund-level dataset created and is used for final calculations and regressions. The dataset has 12764 obs and all the variables created. The dataset is also named as “Fundlevel\_Final\_Dec2024” and is made as backup dataset.

### 60) MACRO MERGE\_CODE\_1

/\* MERGE GENERAL MACRO \*/

**%MACRO** MERGE\_CODE\_1(INPUT\_FILE\_1, INPUT\_FILE\_2, SORT\_VARS, OUTPUT\_FILE);

PROC SORT DATA = &INPUT\_FILE\_1. OUT = LINK\_FILE; BY &SORT\_VARS.; RUN;

PROC SORT DATA = &INPUT\_FILE\_2.; BY &SORT\_VARS.; RUN;

DATA &OUTPUT\_FILE.;

MERGE LINK\_FILE (IN=A) &INPUT\_FILE\_2. (IN=B);

BY &SORT\_VARS.;

IF A;

RUN;

PROC DATASETS LIB = WORK; /\*DELETING INTERMEDIARY DATASET FOR SPACE MANAGEMENT\*/

DELETE LINK\_FILE; /\*COOMENT IT OUT IF WANT TO VIEW LINK\_FILE\*/

QUIT;

**%MEND** MERGE\_CODE\_1;

### 61) MACRO IMPORT\_MACROS – for CSV datset

/\* IMPORT MACRO 1 \*/

**%MACRO** IMPORT\_MACROS(INPUTFILE, OUTPUT\_FILE);

FILENAME REFFILE "&INPUTFILE.";

PROC IMPORT DATAFILE=REFFILE

DBMS=CSV

REPLACE

OUT=&OUTPUT\_FILE.;

GETNAMES=YES;

GUESSINGROWS=**33000**;

RUN;

PROC CONTENTS DATA = &OUTPUT\_FILE.; RUN;

**%MEND** IMPORT\_MACROS;

### **62)** **MACRO** IMPORT\_MACRO – for stata dataset

/\* IMPORT MACRO 2 - FOR STATA DATASET \*/

**%MACRO** IMPORT\_MACRO(INPUT\_FILE, DBMS\_TYPE, OUTPUT\_DATASET);

FILENAME REFFILE "&INPUT\_FILE.";

OPTIONS MSGLEVEL=I; /\* RUN V9 ENGINE OPTIONS IF REQURED \*/

PROC IMPORT DATAFILE=REFFILE

DBMS=&DBMS\_TYPE.

OUT=&OUTPUT\_DATASET.;

RUN;

**%MEND** IMPORT\_MACRO;

### 63) PROC\_SUMMARY\_OVERALL – for overall funds

/\* PROC SUMMARY MACRO 1 \*/

**%MACRO** PROC\_SUMMARY\_OVERALL(INPUT\_FILE, CLASS\_VARS, VARIABLES,OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA = &INPUT\_FILE. NWAY MISSING;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.)=&CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_OVERALL;

### 64) MACRO PROC\_SUMMARY\_KIWISAVER – for kiwisaver funds

/\* PROC SUMMARY MACRO 2 \*/

**%MACRO** PROC\_SUMMARY\_KIWISAVER(INPUT\_FILE, CLASS\_VARS, VARIABLES, OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA = &INPUT\_FILE. NWAY MISSING;

WHERE BINARYVAR = **1**;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.)=&CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_KIWISAVER;

65) PROC\_SUMMARY\_KIWIPENSION - for kiwipension funds

/\* PROC SUMMARY MACRO 6 \*/

**%MACRO** PROC\_SUMMARY\_KIWIPENSION(INPUT\_FILE, CLASS\_VARS, VARIABLES,OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA = &INPUT\_FILE. NWAY MISSING;

WHERE BINARYVAR IN(**1**,**2**);

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.)=&CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_KIWIPENSION;

66) PROC\_SUMMARY\_ESG1 - for ESG rated holdings

/\* PROC SUMMARY MACRO 3 \*/

**%MACRO** PROC\_SUMMARY\_ESG1(INPUT\_FILE, CLASS\_VARS, VARIABLES, OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA=&INPUT\_FILE. NWAY MISSING ;

WHERE ESG =**1**;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT = &OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.) = &CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_ESG1;

### 67) PROC\_SUMMARY\_WEIGHT

/\* PROC SUMMARY MACRO 4 \*/

**%MACRO** PROC\_SUMMARY\_WEIGHT(INPUT\_FILE, WEIGHT\_VAR, CLASS\_VARS, VARIABLES, OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA=&INPUT\_FILE. NWAY MISSING;

WEIGHT &WEIGHT\_VAR.;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.) = &CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_WEIGHT;

68) PROC\_SUMMARY\_OPENEND for open end funds

/\* PROC SUMMARY MACRO 5 \*/

**%MACRO** PROC\_SUMMARY\_OPENEND(INPUT\_FILE, CLASS\_VARS, VARIABLES, OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA = &INPUT\_FILE. NWAY MISSING;

WHERE BINARYVAR = **3**;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.)=&CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_OPENEND;

### 69) CREATE\_FORWARD\_VAR – to create forward variables in the dataset

/\* MACRO TO CREATE CARRY FORWARD VARIABLES \*/

**%MACRO** CREATE\_FORWARD\_VAR(INPUT\_FILE, VAR1, VAR2, VAR3, OUTPUT\_FILE, VAR\_TOBE\_CARRY, FILTER\_VAR);

PROC SORT DATA = &INPUT\_FILE. OUT = DATA\_123; BY &VAR1. DESCENDING &VAR2. DESCENDING &VAR3.; RUN;

DATA &OUTPUT\_FILE.;

SET DATA\_123;

BY &VAR1. DESCENDING &VAR2. DESCENDING &VAR3.;

TEMP\_LAG\_VAR = LAG(&VAR\_TOBE\_CARRY.);

IF FIRST.&FILTER\_VAR. THEN &VAR\_TOBE\_CARRY.\_FRWRD = **.**;

ELSE &VAR\_TOBE\_CARRY.\_FRWRD = TEMP\_LAG\_VAR;

DROP TEMP\_LAG\_VAR;

RUN;

**%MEND** CREATE\_FORWARD\_VAR;

### 70) PROC\_SUMMARY\_HIGH\_PICKING

/\* PROC SUMMARY MACRO 6 \*/

**%MACRO** PROC\_SUMMARY\_HIGH\_PICKING(INPUT\_FILE, CLASS\_VARS, VARIABLES, OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA = &INPUT\_FILE. NWAY MISSING;

WHERE HIGH\_PICKING = **1**;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.)=&CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_HIGH\_PICKING;

### 71) PROC\_SUMMARY\_LOW\_PICKING

/\* PROC SUMMARY MACRO 7 \*/

**%MACRO** PROC\_SUMMARY\_LOW\_PICKING(INPUT\_FILE, CLASS\_VARS, VARIABLES, OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA = &INPUT\_FILE. NWAY MISSING;

WHERE HIGH\_PICKING = **0**;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.)=&CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_LOW\_PICKING;

### 72) PROC\_SUMMARY\_HIGH\_TIMING

/\* PROC SUMMARY MACRO 8 \*/

**%MACRO** PROC\_SUMMARY\_HIGH\_TIMING(INPUT\_FILE, CLASS\_VARS, VARIABLES, OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA = &INPUT\_FILE. NWAY MISSING;

WHERE HIGH\_TIMING = **1**;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.)=&CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_HIGH\_TIMING;

### 73) PROC\_SUMMARY\_LOW\_TIMING

/\* PROC SUMMARY MACRO 9 \*/

**%MACRO** PROC\_SUMMARY\_LOW\_TIMING(INPUT\_FILE, CLASS\_VARS, VARIABLES, OUTPUT\_FILE, STATS\_OPS, CALCULATED\_VAR);

PROC SUMMARY DATA = &INPUT\_FILE. NWAY MISSING;

WHERE HIGH\_TIMING = **0**;

CLASS &CLASS\_VARS.;

VAR &VARIABLES.;

OUTPUT OUT=&OUTPUT\_FILE. /\*(DROP=\_TYPE\_ \_FREQ\_)\*/

&STATS\_OPS.(&VARIABLES.)=&CALCULATED\_VAR.;

RUN;

**%MEND** PROC\_SUMMARY\_LOW\_TIMING;

### 73) MERGE\_CODE\_2 - MERGE MACRO IF A OR B

/\* MERGE MACRO IF A OR B \*/

**%MACRO** MERGE\_CODE\_2(INPUT\_FILE\_1, INPUT\_FILE\_2, SORT\_VARS, OUTPUT\_FILE);

PROC SORT DATA = &INPUT\_FILE\_1. OUT = LINK\_FILE; BY &SORT\_VARS.; RUN;

PROC SORT DATA = &INPUT\_FILE\_2.; BY &SORT\_VARS.; RUN;

DATA &OUTPUT\_FILE.;

MERGE LINK\_FILE (IN=A) &INPUT\_FILE\_2. (IN=B);

BY &SORT\_VARS.;

IF A OR B;

RUN;

PROC DATASETS LIB = WORK; /\*DELETING INTERMEDIARY DATASET FOR SPACE MANAGEMENT\*/

DELETE LINK\_FILE;

QUIT;

**%MEND** MERGE\_CODE\_2;

### 74) PICKING\_VARS\_ALPHA - CREATING MACRO TO CREATE AVERAGE CH4 VARIABLES IN HIGH AND LOW PICKING ENVIRONMENT

/\* CREATING MACRO TO CREATE AVERAGE CH4 VARIABLES IN HIGH AND LOW PICKING ENVIRONMENT \*/

**%MACRO** PICKING\_VARS\_ALPHA(INPUT\_FILE, INPUT\_VAR, INPUT\_CHTIME);

%***PROC\_SUMMARY\_HIGH\_PICKING***(INPUT\_FILE=&INPUT\_FILE., CLASS\_VARS=YEAR MONTH, VARIABLES=&INPUT\_VAR., OUTPUT\_FILE=FILE\_1, STATS\_OPS=MEAN, CALCULATED\_VAR=YEAR\_&INPUT\_CHTIME.\_HIGH\_PICKING);

%***PROC\_SUMMARY\_LOW\_PICKING***(INPUT\_FILE=&INPUT\_FILE., CLASS\_VARS=YEAR MONTH, VARIABLES=&INPUT\_VAR., OUTPUT\_FILE=FILE\_2, STATS\_OPS=MEAN, CALCULATED\_VAR=YEAR\_&INPUT\_CHTIME.\_LOW\_PICKING);

%***MERGE\_CODE\_2***(INPUT\_FILE\_1=FILE\_1, INPUT\_FILE\_2=FILE\_2, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=FILE\_3);

DATA FILE\_4;

SET FILE\_3;

DROP \_TYPE\_ \_FREQ\_;

&INPUT\_CHTIME.\_HML\_PICKING = COALESCE(YEAR\_&INPUT\_CHTIME.\_HIGH\_PICKING, **0**) - COALESCE(YEAR\_&INPUT\_CHTIME.\_LOW\_PICKING, **0**);

RUN;

%***MERGE\_CODE\_1***(INPUT\_FILE\_1=SET\_FUNDS\_ESG\_DATA, INPUT\_FILE\_2=FILE\_4, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=SET\_FUNDS\_ESG\_DATA);

**%MEND** PICKING\_VARS\_ALPHA;

### 75) MACRO TIMING\_VARS\_ALPHA

/\* STEP 3: CREATING MACRO TO CREATE AVERAGE CH4 VARIABLES IN HIGH AND LOW TIMING ENVIRONMENT \*/

**%MACRO** TIMING\_VARS\_ALPHA(INPUT\_VAR, INPUT\_CHTIME);

%***PROC\_SUMMARY\_HIGH\_TIMING***(INPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED, CLASS\_VARS=YEAR MONTH, VARIABLES=&INPUT\_VAR., OUTPUT\_FILE=FILE\_1, STATS\_OPS=MEAN, CALCULATED\_VAR=YEAR\_&INPUT\_CHTIME.\_HIGH\_TIMING);

%***PROC\_SUMMARY\_LOW\_TIMING***(INPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED, CLASS\_VARS=YEAR MONTH, VARIABLES=&INPUT\_VAR., OUTPUT\_FILE=FILE\_2, STATS\_OPS=MEAN, CALCULATED\_VAR=YEAR\_&INPUT\_CHTIME.\_LOW\_TIMING);

%***MERGE\_CODE\_2***(INPUT\_FILE\_1=FILE\_1, INPUT\_FILE\_2=FILE\_2, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=FILE\_3);

DATA FILE\_4;

SET FILE\_3;

DROP \_TYPE\_ \_FREQ\_;

&INPUT\_CHTIME.\_HML\_TIMING = YEAR\_&INPUT\_CHTIME.\_HIGH\_TIMING - YEAR\_&INPUT\_CHTIME.\_LOW\_TIMING;

RUN;

%***MERGE\_CODE\_1***(INPUT\_FILE\_1=FUNDLEVEL\_DATASET\_TRUNCATED, INPUT\_FILE\_2=FILE\_4, SORT\_VARS=YEAR MONTH, OUTPUT\_FILE=FUNDLEVEL\_DATASET\_TRUNCATED);

PROC DATASETS LIB = WORK; /\*DELETING INTERMEDIARY DATASET FOR AVOIDING FILE CONFUSION\*/

DELETE FILE\_1 FILE\_2 FILE\_3 FILE\_4;

QUIT;

**%MEND** TIMING\_VARS\_ALPHA;

### 76) MACRO TTEST\_MACRO

/\* TTEST MACRO TO GET DATA IN DATASET \*/

**%MACRO** TTEST\_MACRO(INPUT\_VAR\_FOR\_CALS, INPUT\_FILE, OUTPUT\_FILE);

/\* COUNT THE NUMBER OF VARIABLES TO LOOP THROUGH \*/

%LET INPUT\_VAR\_FOR\_CALS\_NUM = %SYSFUNC(COUNTW(&INPUT\_VAR\_FOR\_CALS));

/\* INITIALIZE FLAG FOR FIRST ITERATION \*/

%LET FIRST\_ITER = 1;

/\* LOOP THROUGH EACH VARIABLE \*/

%DO I = **1** %TO &INPUT\_VAR\_FOR\_CALS\_NUM;

%LET INPUT\_VAR = %SCAN(&INPUT\_VAR\_FOR\_CALS, &I);

/\* RUNNING PROC TTEST \*/

ODS OUTPUT TTests = TTEST\_&I; /\* TTEST OUTPUT \*/

ODS OUTPUT Statistics = MEAN\_&I; /\* MEAN OUTPUT \*/

PROC TTEST DATA = &INPUT\_FILE.;

VAR &INPUT\_VAR;

RUN;

/\* MERGE TTEST AND MEAN OUTPUTS \*/

%***MERGE\_CODE\_2***(INPUT\_FILE\_1=MEAN\_&I, INPUT\_FILE\_2=TTEST\_&I, SORT\_VARS=Variable, OUTPUT\_FILE=Results\_&I);

/\* ADD IDENTIFIER FOR VARIABLE \*/

DATA Results\_&INPUT\_VAR;

LENGTH Variable $**32**;

SET Results\_&I;

VARIABLE = "&INPUT\_VAR"; /\* Store current variable name \*/

KEEP VARIABLE Mean N tValue Probt; /\* Retain necessary columns \*/

RUN;

/\* APPEND RESULTS: HANDLE FIRST ITERATION SEPARATELY \*/

%IF &FIRST\_ITER = **1** %THEN %DO;

DATA &OUTPUT\_FILE.;

SET Results\_&INPUT\_VAR;

RUN;

%LET FIRST\_ITER = 0; /\* Disable the first iteration flag \*/

%END;

%ELSE %DO;

PROC APPEND BASE=&OUTPUT\_FILE. DATA=Results\_&INPUT\_VAR FORCE;

RUN;

%END;

/\* DELETE TEMPORARY DATASETS \*/

PROC DATASETS LIBRARY=WORK NOLIST;

DELETE MEAN\_&I TTEST\_&I Results\_&I;

QUIT;

%END;

**%MEND** TTEST\_MACRO;

%LET INPUT\_VAR\_FOR\_CALS = PICKING\_ESG\_DECOMP\_WIN PICKING\_NON\_ESG\_DECOMP\_WIN ESG\_MINUS\_NON\_ESG\_P PICKING\_STKS\_Q1\_WIN PICKING\_STKS\_Q5\_WIN PICKING\_Q5\_MINUS\_Q1

TIMING\_ESG\_DECOMP\_WIN TIMING\_NON\_ESG\_DECOMP\_WIN ESG\_MINUS\_NON\_ESG\_TIM TIMING\_STKS\_Q1\_WIN TIMING\_STKS\_Q5\_WIN TIMING\_Q5\_MINUS\_Q1;

### 77) MACRO TTEST\_FUND\_STYLES\_FLAG\_EPU

This macro to be used when fund styles and EPU flag is to be used in where condition.

/\* TTEST AND MEANS \*/

**%MACRO** TTEST\_FUND\_STYLES\_FLAG\_EPU(INPUT\_EPU\_HIGH, FUND\_STYLE\_INPUT, INPUT\_VAR\_FOR\_CALS, OUTPUT\_FILE);

/\* COUNTING NO OF FUND STYLES TO RUN THE DO LOOP \*/

%LET FUND\_STYLE\_NUMS = %SYSFUNC(COUNTW(&FUND\_STYLE\_INPUT));

/\* INITIALIZE AN EMPTY FLAG TO TRACK THE FIRST ITERATION \*/

%LET FIRST\_ITER = 1;

/\* LOOPING THROUGH EACH FUND STYLE \*/

%DO I = 1 %TO &FUND\_STYLE\_NUMS;

%LET FUND\_STYLE = %SCAN(&FUND\_STYLE\_INPUT, &I);

/\* RUNNING PROC TTESTS AND STORE THE FILES \*/

ODS OUTPUT TTests = TTEST\_&FUND\_STYLE; /\* PUT RESULTS IN THIS FILE FIRST \*/

ODS OUTPUT Statistics = MEAN\_&FUND\_STYLE;

PROC TTEST DATA=FUNDLEVEL\_DATASET\_TRUNCATED;

WHERE EPU\_HIGH = &INPUT\_EPU\_HIGH. AND %UNQUOTE(&FUND\_STYLE) = 1;

VAR &INPUT\_VAR\_FOR\_CALS.;

RUN;

/\* MERGE BOTH THE FILES \*/

%***MERGE\_CODE\_2***(INPUT\_FILE\_1=MEAN\_&FUND\_STYLE, INPUT\_FILE\_2=TTEST\_&FUND\_STYLE, SORT\_VARS=Variable, OUTPUT\_FILE=Results\_&FUND\_STYLE);

/\* ADD A FUND\_STYLE IDENTIFIER IN ODS FILE AND CLEAN THE DATASET FOR REQUIRED VARIABLES ONLY \*/

DATA Results\_&FUND\_STYLE;

LENGTH FUND\_STYLE $**32**;

SET Results\_&FUND\_STYLE;

FUND\_STYLE = "&FUND\_STYLE";

/\* ADD THE CURRENT FUND STYLE AS A VARIABLE \*/

KEEP FUND\_STYLE Variable Mean N tValue Probt;

RUN;

/\* APPEND THE RESULTS IN ANOTHER DATABASE - THIS LOOP IS NEEDED TO MAKE DATA STRUCTURE PROPER IN THE FILE \*/

%IF &FIRST\_ITER = **1** %THEN %DO;

DATA &OUTPUT\_FILE.;

SET Results\_&FUND\_STYLE;

RUN;

%LET FIRST\_ITER = 0;

/\* MAKING FIRST\_ITER FLLAG AS FALSE \*/

%END;

%ELSE %DO;

/\* APPEND RESULTS IN ANOTHER DATASET \*/

PROC APPEND BASE=&OUTPUT\_FILE. DATA=Results\_&FUND\_STYLE FORCE;

RUN;

%END;

PROC DATASETS LIBRARY=WORK NOLIST;

DELETE MEAN\_&FUND\_STYLE TTEST\_&FUND\_STYLE Results\_&FUND\_STYLE;

QUIT;

%END;

**%MEND** TTEST\_FUND\_STYLES\_FLAG\_EPU;

### 78) Picking from >75 Score – ‘PICKING\_GE\_75’

Code logic – Calculating sum of picking returns yielded by stocks which have ESGscore >= 75 for specified fund portfolio per month and year.

**DATA** Picking\_file\_stkret\_win;

SET Fund\_set.Picking\_file\_stkret\_win;

**RUN**;

**PROC** **SUMMARY** DATA = Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESGSCORE >= **75**;

CLASS FUNDID YEAR MONTH;

VAR PICKING\_PER\_STOCK\_MED;

OUTPUT OUT = DATA\_GE75

SUM(PICKING\_PER\_STOCK\_MED) = PICKING\_GE\_75;

**RUN**;

### 79) Picking from <25 Score – ‘PICKING\_LE\_25’

Code logic – Calculating sum of picking returns yielded by stocks which have ESGscore >= 75 for specified fund portfolio per month and year.

**PROC** **SUMMARY** DATA = Picking\_file\_stkret\_win NWAY MISSING;

WHERE ESGSCORE <= **25**;

CLASS FUNDID YEAR MONTH;

VAR PICKING\_PER\_STOCK\_MED;

OUTPUT OUT = DATA\_LE25

SUM(PICKING\_PER\_STOCK\_MED) = PICKING\_LE\_25;

**RUN**;

### 80) ESG returns (Equal-weighted) – ‘EW\_ESG\_RETURNS’

Code logic :- The code computes equal-weighted returns yielded by stock with ESG ratings for the fund portfolio for a specified year and month by using the monthly returns of ESG holdings.

**PROC** **SUMMARY** DATA = NZD\_DATASET\_FINAL NWAY MISSING;

WHERE ESG = **1** AND MONTHLYRETURNS NE **.** ;

CLASS FUNDID YEAR MONTH;

VAR MONTHLYRETURNS;

OUTPUT OUT = ESG\_RETURNS\_EW

MEAN (MONTHLYRETURNS) = EW\_ESG\_RETURNS;

**RUN**;

### 81) Regression Equation for capm\_alpha generation

**gen modate = ym(YEAR, MONTH)**

**gen ex\_ret = Fundreturns\_NZD1 - RF**

**/\* ch\_4 alphas \*/**

**bys FundId : asreg ex\_ret Mkt\_RF SMB\_NZ HML\_NZ MOM\_NZ, wind(modate 24) min (12)**

### 82) Generating ch4\_alpha, size\_beta, momentum\_beta, market\_beta, value\_beta

**gen ch4\_alpha = \_b\_cons**

**gen size\_beta = \_b\_SMB\_NZ**

**gen momentum\_beta = \_b\_MOM\_NZ**

**gen market\_beta = \_b\_Mkt\_RF**

**gen value\_beta = \_b\_HML\_NZ**

### 83) Regression Equation for CAPM alpha beta variable generation

**/\* capm alphas \*/**

**bys FundId : asreg ex\_ret Mkt\_RF, wind(modate 24) min (12)**

**gen capm\_alpha = \_b\_cons**

### 84) Regression Equation for generating Idiosyncratic volatility

**bys FundId : asreg ex\_ret Mkt\_RF SMB\_NZ HML\_NZ MOM\_NZ,fit**

**bys FundId : asrol \_residuals, stat(sd) gen (IDIO\_4FF) win(modate 12) min(12)**

### 85) Regression Equation for generating Value at risk variable

**bys FundId: asrol Fundreturns\_NZD1, win(modate 12) min(12) stat(median) perc(0.05) gen (VaR\_rol)**

**gen V\_RISK = (-1)\* VaR\_rol**

### 86) Regression to generate beta which is used in timing and picking calculations

Code logic – This code is used to generate beta which is used in calculating timing and picking in the timing and picking equations. Following are the equations. The highlighted beta is calculated in the following stata code.

TIMING\_PER\_STOCK\_MED = (Percentholdings - MED\_HOLDINGS\_SECID) \* (BETA \* NZX50returns\_NZD1\_FRWRD);

PICKING\_PER\_STOCK\_MED = (Percentholdings - MED\_HOLDINGS\_SECID) \* (monthlyreturns\_FRWRD - (BETA\*NZX50returns\_NZD1\_FRWRD));

**sort SecID YEAR MONTH**

**gen modate = ym(YEAR,MONTH)**

**bys SecID: asreg NZX50returns\_NZD1\_FRWRD monthlyreturns, wind(modate 12) min(11)**

**gen beta = \_b\_monthlyreturns**

**saveold output\_dataset, version(12) replace**

**browse**

**save "R:\SAS - Research\SWARN\_RESEARCH\SAS\_DATASETS\FUND\_MONTH\_DATA\FUND\_NZD\Supporting\_files\beta\_file\_STKRET\_WIN.dta", replace**

### 87) Generalized stata code to be used before running any regression

**gen modate = ym(YEAR,MONTH)**

**gen log\_age = ln(FUND\_AGE\_MONTH)**

**gen log\_tna = ln(Netassets\_NZD)**

**gen log\_EPU = ln(EPU\_Scores)**

**gen SUM\_PICKING\_MED\_WIN\_scaled = SUM\_PICKING\_MED\_WIN/100**

**gen ESG\_MINUS\_NON\_ESG\_P\_WIN\_scaled = ESG\_MINUS\_NON\_ESG\_P/100**

**gen PICKING\_Q5\_MINUS\_Q1\_scaled = PICKING\_Q5\_MINUS\_Q1/100**

**gen SUM\_TIMING\_MED\_WIN\_scaled = SUM\_TIMING\_MED\_WIN/100**

**gen ESG\_MINUS\_NON\_ESG\_TIM\_WIN\_scaled = ESG\_MINUS\_NON\_ESG\_TIM/100**

**gen TIMING\_Q5\_MINUS\_Q1\_scaled = TIMING\_Q5\_MINUS\_Q1/100**

**gen NZD\_ESG\_WT\_OF\_FUND\_SCALED = NZD\_ESG\_WT\_OF\_FUND/100**

**gen ESGGE75\_SUM\_WT\_NZD\_scaled = ESGGE75\_SUM\_WT\_NZD/100**

**/\* The below regression checks for missing values in the dataset. The dataset from this dataset is the final dataset and no of obs in this dataset should be 12764. \*/**

**/\* Creating p1 values \*/**

**reg TIMING\_Q5\_MINUS\_Q1\_scaled log\_EPU log\_age log\_tna TOTAL\_COST\_RATIO TRACK\_ERROR FLOW\_PERCENT\_WIN\_NZD size\_beta value\_beta momentum\_beta**

**predict p1**