# Forecast Reconciliation: Step Description

Forecast Reconciliation step is aimed at bringing ML and Statistical forecasts to the same granularity level (e.g. at product/location/day level). The result of this step is used to generate hybrid forecast value. Usually, the forecast is split to more granular level at the Reconciliation step.

This step may include some PL processing logic (e.g. phase-in and phase-out dates) as well as

This step is not needed if only one approach (ML or Stat) is used for forecasting.

## Code Realization Requirements

The code should be created in a form of a SAS macro set on SAS VIYA platform.

# Input Data

The initial data for the step are listed below. These tables should be present in the system before this step for example in DDS area.

• IN\_PRODUCT (see data requirements)

• IN\_LOCATION (see data requirements)

## VF\_FORECAST\_<target var postfix>

Forecast values from VF forecast step that is disaggregated to sku/location level and disaccumulated to daily granularity.

|  |  |
| --- | --- |
| **VF\_FORECAST\_<target\_var\_type>** | |
| Column Name | Description |
| **product\_lvl\_id<m>** | ID of product group for which specific ML model will be trained, m= **vf\_product\_lvl** |
| **location\_lvl\_id<n>** | ID of location group for which specific ML model will be trained, n= **vf\_location\_lvl** |
| **customer\_lvl\_id<l>** | ID of customer group for which specific ML model will be trained, l= **vf\_customer\_lvl** |
| **distr\_channel\_lvl\_id<k>** | ID of distr\_channel group for which specific ML model will be trained, k= **vf\_distr\_channel\_lvl** |
| **PERIOD\_DT** | Date of sales, granularity = **vf\_time\_lvl** |
| **FORECAST\_VALUE** | VF forecast value |

## ML\_FORECAST\_<target var postfix>

Forecast values from ML forecast step, this table include only those triples PRODUCT\_ID| LOCATION\_ID|PERIOD\_DT.

|  |  |
| --- | --- |
| **ML\_FORECAST\_<target\_var\_type>** | |
| Column Name | Description |
| **product\_lvl\_id<M>** | ID of product group for which specific ML model will be trained, M = **ml\_product\_lvl** |
| **location\_lvl\_id<N>** | ID of location group for which specific ML model will be trained, N = **ml\_location\_lvl** |
| **customer\_lvl\_id<L>** | ID of customer group for which specific ML model will be trained, L = **ml\_customer\_lvl** |
| **distr\_channel\_lvl\_id<K>** | ID of distr\_channel group for which specific ML model will be trained, K = **ml\_distr\_channel\_lvl** |
| **PERIOD\_DT** | Date of sales, **ml\_time\_lvl** |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | ‘new’ or ‘old’ |
| **FORECAST\_VALUE\_total** | ML forecasted value |
| **FORECAST\_VALUE\_baseline** |  |

## VF\_TS\_SEGMENTS\_<target var postfix>

Name of Segments for each TS in VF\_FORECAST\_TABLE

|  |  |
| --- | --- |
| **VF\_TS\_SEGMENTS\_<target\_var\_type>** | |
| Column Name | Description |
| **product\_lvl\_id<m>** | ID of product group for which specific ML model will be trained, m = **vf\_product\_lvl** |
| **location\_lvl\_id<n>** | ID of location group for which specific ML model will be trained, n = **vf\_location\_lvl** |
| **customer\_lvl\_id<l>** | ID of customer group for which specific ML model will be trained, l = **vf\_customer\_lvl** |
| **distr\_channel\_lvl\_id<k>** | ID of distr\_channel group for which specific ML model will be trained, k = vf**\_distr\_channel\_lvl** |
| **SEGMENT\_NAME** | VF segment name |

## CONFIG\_PARAMETERS

The following config parameters are used within the step.

|  |  |  |
| --- | --- | --- |
| **CONFIG.TGT\_VAR\_CONFIG** | | |
| Column Name | Description | Example |
| **tgt\_type** | One of 3 types of the target variable:   * SELLIN – means CPG sales to its customer, * SELLOUT – means CPG’s customer sales to their clients, * POS – means sales in the point of sales, can be relevant for both Retailer and CPG | POS |
| **value\_src** | Name of the target variable from the source table. It should be quantity of sales.  Feasible values: INVOICE\_QTY, SALES\_QTY, SHIPMENT\_QTY, ORDER\_QTY. | SALES\_QTY |
| **act\_flag** | Activity flag, whether this target variable is needed to be forecasted. Feasible values: 0 or 1 | 1 |
| **vf\_product\_lvl** | Aggregation level for ML ABT by product hierarchy, default value is 8, which means PRODUCT\_ID | 7 |
| **vf\_location\_lvl** | Aggregation level for ML ABT by product hierarchy, default value is 6, which means LOCATION\_ID | 1 |
| **vf\_customer\_lvl** | Aggregation level for ML ABT by product hierarchy, default value is 6, which means CUSTOMER\_ID | 5 |
| **vf\_distr\_channel\_lvl** | Aggregation level for ML ABT by product hierarchy, default value is 3, which means DISTR\_CHANNEL\_ID | 1 |
| **vf\_time\_lvl** | Accumulation level for ML ABT by time hierarchy, default value is WEEK.2, which means weeks began from Monday | MONTH |
| **ml\_product\_lvl** | Aggregation level for ML ABT by product hierarchy, default value is 8, which means PRODUCT\_ID | 7 |
| **ml\_location\_lvl** | Aggregation level for ML ABT by product hierarchy, default value is 6, which means LOCATION\_ID | 5 |
| **ml\_customer\_lvl** | Aggregation level for ML ABT by product hierarchy, default value is 6, which means CUSTOMER\_ID | 4 |
| **ml\_distr\_channel\_lvl** | Aggregation level for ML ABT by product hierarchy, default value is 3, which means DISTR\_CHANNEL\_ID | 1 |
| **ml\_time\_lvl** | Accumulation level for ML ABT by time hierarchy, default value is WEEK.2, which means weeks began from Monday | WEEK.2 |
|  |  |  |

## INITIAL\_GLOBAL parameters

All parameters are listed in initial\_global file.

|  |  |
| --- | --- |
| **INITIAL\_GLOBAL parameters init** | |
| Column Name | Description |
| **IB\_HIST\_END\_DT** | Last known date (i.e. sales and stock information is known) |
| **IB\_FC\_HORIZ** | Horizon of forecast |

## Other Dependencies

There is assumption that the lowest level of any hierarchy is defined as PRODUCT\_LVL\_ID8 or LOCATION\_LVL\_ID6, or CUSTOMER\_LVL\_ID6, or DISTR\_CHANNEL\_ID2.

Assumption: ML\_FORECAST is provided on more granular or the same granularity level then VF forecast regarding product, location, customer, distr\_channel dimensions, i.e. .

All join-s with dictionaries (PRODUCT, LOCATIONS, DISTR\_CHANNELS, CUSTOMERS) must be performed via Forecast Flag table in order to prevent join with fake or obsolete elements of the dictionaries.

All steps of the algorithm below should be applied to all target variables **tgt\_type** which have **act\_flag** = 1.

# Algorithm Definition

## Join VF and ML forecasts

**Inputs:** VF\_FORECAST\_<target\_var\_type>, ML\_FORECAST\_<target\_var\_type>

**Transformation algorithm:**

1. Define mid-term reconciled forecast
   1. Calculate length of short-term forecasting period
2. 0. select sum(intnx(step\_interval, horizon\_length0, horizon\_length, ‘s’)\_) \_into:delays\_config\_length from DELAYS\_CONFIG;

if IB\_FC\_HORIZ > delays\_config\_length

then do;

* 1. then store the part of VF\_FORECAST table that is related to mid-term horizon periods to separate table;

create table mid\_reconciled\_forecast as

select \* except FORECAST\_VALUE,

PERIOD\_DT as PERIOD\_END\_DT,

FORECAST\_VALUE as VF\_FORECAST\_VALUE\_REC,

missing as ML\_FORECAST\_VALUE,

‘Regular’ as DEMAND\_TYPE,

‘old’ as ASSORTOMENT\_TYPE

from VF\_FORECAST; end;

else do;WHERE PERIOD\_DT > intnx(‘day’, IB\_HIST\_END\_DT, delays\_config\_length)[[1]](#footnote-2)

* 1. leave only short-term horizon dates in VF\_FORECAS\_TABLE

select \* from VF\_FORECAST

WHERE PERIOD\_DT <= intnx(‘day’, IB\_HIST\_END\_DT, delays\_config\_length)

1. Add PERIOD\_END\_DT to ML\_FORECAST (as ML) and to VF\_FORECAST0.c (as VF) tables
   1. ML.PERIOD\_END\_DT = intnx(ml\_time\_lvl, period\_dt, 1)-1
   2. VF.PERIOD\_END\_DT = intnx(vf\_time\_lvl, period\_dt, 1)-1

For example if PERIOD\_DT = 01.02.2021 then

If time\_lvl = ‘day’ then PERIOD\_END\_DT = 01.02.2021 (the same day)

If time\_lvl = ‘week.2’ then PERIOD\_END\_DT = 07.02.2021 (the end of week 07.02.2021)

If time\_lvl = ‘MONTH’ then PERIOD\_END\_DT = 28.02.2021 (the end of MONTH 02.2021)

1. ML left join PRODUCT, LOCATION, CUSTOMER, DISTR\_CHANNEL on product\_lvl\_id<M>

location\_lvl\_id<N>

customer\_lvl\_id<L>

distr\_channel\_lvl\_id<K>, rename FORECAST\_VALUE to ML\_FORECAST\_VALUE.

And add columns product\_lvl\_id<m>, location\_lvl\_id<n>, customer\_lvl\_id<l>, distr\_channel\_lvl\_id<k> columns

rename FORECAST\_VALUE to ML\_FORECAST\_VALUE.

1. VF left join PRODUCT, LOCATION, CUSTOMER, DISTR\_CHANNEL on product\_lvl\_id<m>

location\_lvl\_id<n>

customer\_lvl\_id<l>

distr\_channel\_lvl\_id<k>,

And add columns product\_lvl\_id<M>, location\_lvl\_id<N>, customer\_lvl\_id<L>, distr\_channel\_lvl\_id<K>

rename FORECAST\_VALUE to VF\_FORECAST\_VALUE.

1. 2. (As ML) Left join 3 (as VF) on

ML.PERIOD\_DT<=VF.PERIOD\_END\_DT and ML.PERIOD\_END\_DT>=VF.PERIOD\_DT and PRODUCT\_LVL\_ID<max(m, M)>, LOCATION\_LVL\_ID<max(n, N)>, customer\_lvl\_id<max(l, L)>, distr\_channel\_lvl\_id<max(k, K)>

(where PERIOD\_DT > **IB\_HIST\_END\_DT**)

Transform columns

PERIOD\_DT = MAX(VF.PERIOD\_DT, ML\_PERIOD\_DT)

PERIOD\_END\_DT = MIN(VF.PERIOD\_END\_DT, ML\_PERIOD\_END\_DT)

And

Replace missing values VF\_FORECAST\_VALUE with zero.

ASSUMPTION: ML\_FORECAST is built for active quadruples product/location/customer/distr\_channel from Forecast\_Flag. Otherwise, alignment with forecast flag will be provided in disaggregation step.

1. Calculate forecast share and volume of VF\_FORECAST\_VALUE and ML\_FORECAST\_VALUEproportionaly to number of day in interval [PERIOD\_DT, PERIOD\_END\_DT]:
   1. For each row

VF\_FORECAST\_VALUE = VF\_FORECAST\_VALUE \* ,

ML\_FORECAST\_VALUE = ML\_FORECAST\_VALUE \*

Where is a function that calculate days number in a interval that contains PERIOD\_DT date, for example

Assumption: we count all calendar days in a period, exclusion of weekend is not needed.

Assumption: no need to consider more complex logic when splitting forecast volumes to the interval [PERIOD\_DT, PERIOD\_END\_DT]

Note, that VF\_FORECAST\_VALUE\_REC will be zero for all PRODUCT\_LVL\_ID<m> elements, that have no demand history in the past (e.g. new PRODUCT\_LVL\_ID<m>).

Output: As a result of this step, a table with mid-term forecast values is created at step 0.b (see description in 3.4 section, and a table for short-term reconciled forecast of the following structure is constructed, *T1.*

|  |  |  |
| --- | --- | --- |
| Column Name | | Description |
| **product\_lvl\_id<m>** | ID of product group for which specific ML model will be trained, m = **vf\_product\_lvl** | |
| **location\_lvl\_id<n>** | ID of location group for which specific ML model will be trained, m = **vf\_location\_lvl** | |
| **customer\_lvl\_id<l>** | ID of customer group for which specific ML model will be trained, l = vf**\_customer\_lvl** | |
| **distr\_channel\_lvl\_id<k>** | ID of distr\_channel group for which specific ML model will be trained, k = vf**\_distr\_channel\_lvl** | |
| **product\_lvl\_id<M>** | ID of product group for which specific ML model will be trained, M = **ml\_product\_lvl** | |
| **location\_lvl\_id<N>** | ID of location group for which specific ML model will be trained, N = **ml\_location\_lvl** | |
| **customer\_lvl\_id<L>** | ID of customer group for which specific ML model will be trained, L = **ml\_customer\_lvl** | |
| **distr\_channel\_lvl\_id<K>** | ID of distr\_channel group for which specific ML model will be trained, K = **ml\_distr\_channel\_lvl** | |
| **PERIOD\_DT** | Date of sales (calendar day) | |
| **PERIOD\_END\_DT** | End date of the sales period | |
| **VF\_FORECAST\_VALUE** | VF forecast value, related to the period ‘PERIOD\_START\_DT’ | |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast | |
| **ASSORTMENT\_TYPE** | new, or old | |
| **ML\_FORECAST\_VALUE** | ML forecasted value | |

## Reconcile ML and VF Forecast

**Inputs:** T1

**Transformation algorithm:**

1. Reconcile VF\_FORECAST\_VALUE to ML\_FORECAST\_VALUEwithin each group product\_lvl\_id<M> (as P<M>), location\_lvl\_id<N> (as L<N>), customer\_lvl\_id<L> (as C<L>), distr\_channel\_lvl\_id<K> (as D<K>), PERIOD\_DT

Note, that VF\_FORECAST\_VALUE\_REC will be zero for all P<M>,L<N>,C<L>,D<K> elements, that have no demand history in the past (e.g. new quadruple).

Output: As a result of this step, a table of the following structure is constructed, *T2.*

|  |  |
| --- | --- |
| Column Name | Description |
| **product\_lvl\_id<M>** | ID of product group for which specific ML model will be trained, M = **ml\_product\_lvl** |
| **location\_lvl\_id<N>** | ID of location group for which specific ML model will be trained, N = **ml\_location\_lvl** |
| **customer\_lvl\_id<L>** | ID of customer group for which specific ML model will be trained, L = **ml\_customer\_lvl** |
| **distr\_channel\_lvl\_id<K>** | ID of distr\_channel group for which specific ML model will be trained, K = **ml\_distr\_channel\_lvl** |
| **product\_lvl\_id<m>** | ID of product group for which specific ML model will be trained, m = **vf\_product\_lvl** |
| **location\_lvl\_id<n>** | ID of location group for which specific ML model will be trained, m = **vf\_location\_lvl** |
| **customer\_lvl\_id<l>** | ID of customer group for which specific ML model will be trained, l = **vf\_customer\_lvl** |
| **distr\_channel\_lvl\_id<k>** | ID of distr\_channel group for which specific ML model will be trained, k = **vf\_distr\_channel\_lvl** |
| **PERIOD\_DT** | Date of sales (start day of the period) |
| **PERIOD\_END\_DT** | End date of the period |
| **VF\_FORECAST\_VALUE\_REC** | VF forecast value |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | new, or old |
| **ML\_FORECAST\_VALUE** | ML forecasted value |

## Add Segment Name

**Inputs:** T2, VF\_TS\_SEGMENTS\_<target\_var\_type>

**Transformation algorithm:**

1. T2 left join VF\_TS\_SEGMENTS on PRODUCT\_LVL\_ID<m>, LOCATION\_LVL\_ID<n>, CUSTOMER\_LVL\_ID<l>, DISTR\_CHANNEL\_LVL\_ID<k> and add **SEGMENT\_NAME** (and exclude PRODUCT\_LVL\_ID<m>, LOCATION\_LVL\_ID<n>, CUSTOMER\_LVL\_ID<l>, DISTR\_CHANNEL\_LVL\_ID<k>fields)

product\_lvl\_id<m>, location\_lvl\_id<n>, customer\_lvl\_id<l>, distr\_channel\_lvl\_id<k>

Output: As a result of this step, the RECONCILED\_FORECAST table is constructed (detailed description see in section 3.4)*.*

## Output from the Algorithm

1. MID\_RECONCILED\_FORECAST the same structure ad for RECONCILED\_FORECAST (see below)
2. RECONCILED\_FORECAST

A table with all types of reconciled forecast

|  |  |
| --- | --- |
| **RECONCILED\_FORECAST** | |
| Column Name | Description |
| **product\_lvl\_id<M>** | ID of product group for which specific ML model will be trained, M = **ml\_product\_lvl** |
| **location\_lvl\_id<N>** | ID of location group for which specific ML model will be trained, N = **ml\_location\_lvl** |
| **customer\_lvl\_id<L>** | ID of customer group for which specific ML model will be trained, L = **ml\_customer\_lvl** |
| **distr\_channel\_lvl\_id<K>** | ID of distr\_channel group for which specific ML model will be trained, K = **ml\_distr\_channel\_lvl** |
| **PERIOD\_DT** | Date of sales (calendar day) |
| **PERIOD\_END\_DT** | End date of the period |
| **VF\_FORECAST\_VALUE\_REC** | VF forecast value |
| **SEGMENT\_NAME** | Name of segment that was linked to a pair product/location within VF Project (can be missing) |
| **DEMAND\_TYPE** | ‘promo’ or ‘regular’ – type ML model is used to forecast |
| **ASSORTMENT\_TYPE** | new, or old |
| **ML\_FORECAST\_VALUE** | ML forecasted value |
|  |  |

1. It means that if there is no dates more than [↑](#footnote-ref-2)