Checks description and error resolution

To see on which datasets the checks are applied on and the error level associated, please refer to the following documents :

- Novecare :
- SpP :

The python library code for all the checks and metrics presented here can be found in this file : dataiku_metrics_checks.py

> Note that if the project stops due to a failing check related to a manual file at the hands of the business, there is a WebApp in place to allow users to reload the project after having fixed the issue (see dedicated section here)

Row counts (*Check record count*)

Description :

This check compares the row count of two datasets to make sure we did not lose data or generate duplicates in the project Flow.

Applied to :

All_features_prices_ds_joined : with previous recipe (All_features_prices_dataset_prepared)
Checks that the join between all of our data sources (including manual inputs) does not generate duplicates

> **Error resolution**
>
> If this check fails, it probably means that one of the inputs before the join recipe contains duplicates. The duplicates generated in the output dataset should be identified (using the 'cpc' key) and the inputs reviewed to identify the one responsible.
> Then, it is a matter of understanding if it is coming from the data source itself or generated in the data preparation steps of the Dataiku Flow.
> If it is coming from the source itself, the owner of the data should be contacted to check if it is a normal situation. If yes, adaptations might have to be done to the aggregation of these data before joining it by a Data Scientist / Engineer (same as if the duplicates are generated in the data preparation steps.

Null values (*Check null values / Check null values by family*)

Description :

This check is to make sure we do not have missing values when it can make the project crash.

Applied to :

- Encoding_dataset on all columns family by family (this will be the input of similarity model that does not accept null values).

  > **Error resolution**
  >
  > A failure at this stage could potentially mean a lack of imputation on one of the features. It is important to assess if the missing value is coming from bad quality or if it is considered normal and should be managed by imputation or filtering in the Dataiku Flow. If it is the latest then a data scientist has to adapt the project variables to add said imputation or filter.

- Forecasts_prepared / Historical_prepared on all columns with a relative threshold to make sure the Data Lake data we use are complete.

  > **Error resolution**
  >
  > These are checks on the raw data coming from the Data Lake. If one fails it probably means a loading issue on the Data Lake itself, to be directly checked with the team managing it.

- Manual_regions on manual_region with a relative threshold to make sure the manual input data are exhaustive and to ensure no manipulation mistake has happened on the file.

  > **Error resolution**
  >
  > If it fails, the input Gsheet must be checked and either restored back to previous functional version or filled if several new regions appeared.

Unicity (*Check unicity*)

Description :

Checks unicity of the defined key to make sure we do not have duplicates in several parts of the project Flow.

Applied to :

- pricing_features_dataset on **cpc** key
- All_features_prices_ds_joined on **cpc** key
- Encoding_dataset on **cpc** key
- price_recommendations_dataset on **cpc** key

Error resolution

The recipe leading to the dataset containing the failure should be analyzed step-by-step to understand the reason of the duplicates. Previous datasets should also be checked in case the duplicates were generated earlier in the Flow.

Imputations (several checks)

Description :

Checks the percentage of imputation by CPC and by feature, as well as the impact of the imputation for the characteristics included in the first model using their average weights.

Applied to :

- Imputed_features_by_cpc (with a threshold of 0.5 for the imputation percentage and 0.4 for the imputation impact)
- Imputed_HBs_by_cp (with a threshold of 0.5 for the imputation percentage)
- Imputation_pct_by_feature (with a threshold of 0.5 for the imputation percentage)

Note that these thresholds can be modified in the check parameters.

Error resolution

A high ratio of imputed data for a column is correlated to a high ratio of null values. These columns should be analyzed to understand why we have a lot of null values.

If too many different columns are imputed for the same CPC, results for this CPC might not be inaccurate. If that happens to a high ratio of the CPCs of a family, said family should be reviewed by a Data Scientist and potentially adapted.

Number of CPCs (*Check nb CPCs variation*)

Description :

This check controls the variation in the number of CPCs between monthly updates of extracted data.

Applied to :

- Forecasts_prepared / Historical_prepared : checks for partial reloads of the main data sources from the Data Lake
- All_features_prices_ds_joined : checks that the final dataset of the data preparation has an expected volume of records.

Error resolution

This check failing is a sign of potential partial load of one of the data sources on one of the month (probably the latest one). The data sources should be checked to ensure every month is complete.

Features correlation (*Check correlated features*)

Description :

Returns the features with a high correlation between them on a family basis.

> **Error resolution**
>
> Families having highly-correlated features should be analyzed closely and their feature sets might have to be reviewed by a Data Scientist.

Weights variability *(Check weights variability)*

Description :

The purpose here is to make sure the models are not too volatile by looking at the evolution of the importance of each feature between the runs.

> **Error resolution**
>
> Families having high volatility should be analyzed closely and their feature sets might have to be reviewed by a Data Scientist.

Applied to :

"Weighting_folder" containing the weights for all the families and features.

> **Error resolution**
>
> If the check fails for some families, an analysis should be conducted by a data scientist to understand why the volatility is high on the involved family. It could either be due to external factors like market economics, in which case no action may be required, either due to some data quality or missing features that make usage of the family results not accurate enough. In this case, results should be enhanced if possible by adapting the used features or finetuning the model.

Data types *(Check data types)*

Description :

Checks that a column contains the expected type of data.

Applied to :

pricing_features_dataset : Applied to all the _log features that should be float type to not break the following code.

taxonomy datasets : taxonomy columns used as either HB or features and containing numerical values are checked.

Parameters_import_sql : note that this dataset is using a specific *check_type_based_on_type_column* function where the type to be checked is specified in a dedicated column instead of being a parameter.

> **Error resolution**
>
> In case of error, the computation of the _log features in the feature engineering recipes should be checked and potential edge-cases might have to be addressed.

Data Drift *(Check for data drift in input columns)*

Description :

Returns features whose data have drifted in the current run compared with the last three runs (can be modified by changing the "*n_versions"* parameter in the function).

> **Error resolution**
>
> If data drift is detected, it's probably a loading problem on the data lake itself, to be checked directly with the team managing it.

Applied to :

pricing_features_dataset : Applied to all specified features from the "*data_drift_cols*" parameter in Variables.

Error resolution

Failure of this check is an indication of a potential problem in the current month's data sources, based on historical data. Data sources need to be checked to understand where the drift is coming from.