PCM - Predictive Credit Management

Objectives of the document

This document provides a technical overview of the solution delivered by D3S. It explains the building blocks of the source code and the methodology followed to obtain insightful predictions for cash recovery.

General presentation of the solution

Business stake and approach

The main business stake is to increase overdue coverage with the existing task force. As of today, dunning and pre-dunning actions focus on the largest outstanding amounts, leaving aside smaller accounts (below a threshold). Pre-dunning include some additional rules applied by cash collection teams through a time-consuming manual process.
Cash collection is steered with End of Month KPIs. Although not necessarily representative of the cost of working capital, EOM metrics are relevant as they are fully aligned with other business steering indicators.
Predictive analytics are a way forward, especially to better address the smaller accounts on which the overdue rate is higher.
Wrap Up

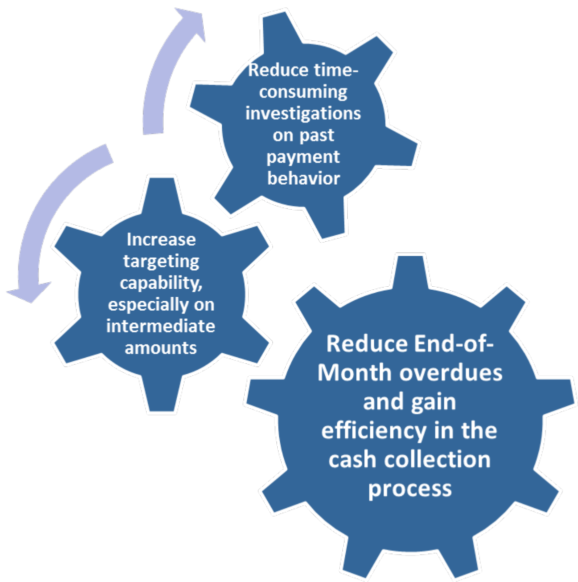


Figure 1: objectives and core principles

- Machine learning technology models deep correlation patterns from past behavior
- A model is trained on 2 years' history, including all documents paid in month or not
- The predictive engine generates risk scores allowing to focus on the main stakes
- Cash collectors follow the proposed priority, applying their expert judgment as required (known event at payer, dunning blocked, etc.)
- Model performance is monitored monthly through a dedicated report (QlikView)
- Statistic on each payer's past payment behavior are also made available without drilling down into SaP

Machine learning methodology

The predictive solution leverage machine learning technology. A model is first trained on payment history to learn customer behavior based on all available caracteristics. For new customers, the model infers behavior based on available data (country, currency, sector, invoice-caracteristics, etc)

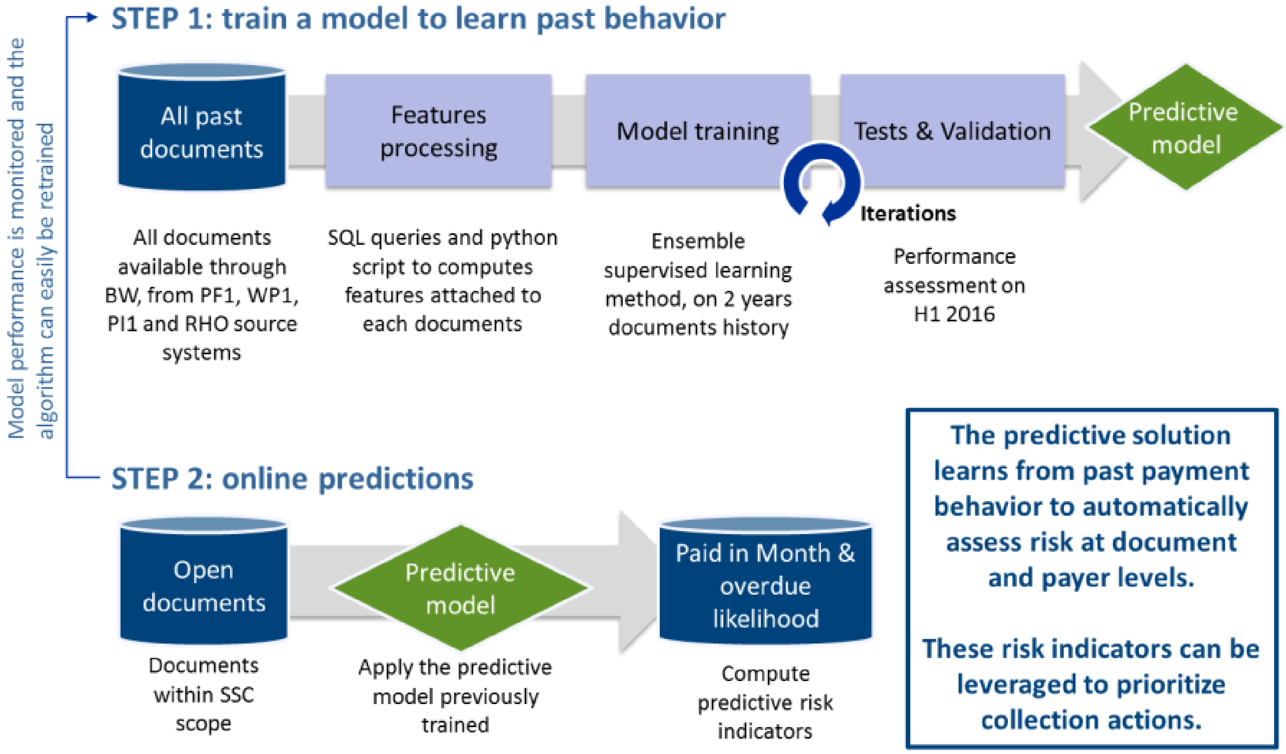


Figure 2:overview of the machine learning methodology

Outputs available to Cash collectors

The predictive solution generates cash recovery insights at payer and document levels. The illustration below shows the indicators computed at PRS Customer Zone/GBU/PRS Customer level. Similar information is available to drill down on underlying documents, either open or due in the current month.

STANDARD INSIGHTS	PRS Cust. Zone (FI)	Usual information from BW
	GBU	Usual information from BW
	GBU description	Usual information from BW
	PRS Customer	Usual information from BW
	Name	Usual information from BW
	Outstanding 04.01.2017	Usual information from BW
	Overdue 04.01.2017	Usual information from BW
	Overdue 0-4 days	Usual information from BW
	Overdue 5-10 days	Usual information from BW
	Overdue 11-30 days	Usual information from BW
	Overdue 31-60 days	Usual information from BW
	Overdue 61-90 days	Usual information from BW
	Overdue > 90 days	Usual information from BW
	pre-chasing amount (last three days of the month)	Total open amount for documents with a net due date in the last 3 days of the month
CM RISK	Risk rating	Rating score, taking into account both chasing and prechasing risk (thresholds to be fine-tuned)
	Risk amount not PIM	Amount considered at risk for payment before the end of the current month (PIM = Paid In Month)
	Average Probability Not PIM	Mean probability for payment before the end of the current month (PIM = Paid In Month)
Pre-chasing	pre-chasing amount	Total open amount for documents with a net due date between today and the end of month (PRECHASING)
	Risk not PIM prechasing	Amount to be prechased and considered at risk for payment before the end of the current month (PIM = Paid In Month)
	Overdue Risk	Expected amount to be paid after the net due date (overdue, disregarding payment in month or not)
	Per. Exp. Amt overdue	% of the above compared to total pre-chasing amount
Chasing	Chasing amount	Total open amount for documents with a net due date in the past
	Risk amount not PIM_chasing	Amount to be chased and considered at risk for payment before the end of the current month (PIM = Paid In Month)
Others	Nb order blocked (overdue)	Number of orders blocked because of an overdue (nota: this indicator is currently refreshed on a weekly basis in BW)
	Last Payment Method	Last observed payment method by the payer
Payer history	Rate not PIM (last 12 months)	Overall % of documents not paid in month over last 12 months history for the payer
	Rate not PIM (last 6 months)	Overall % of documents not paid in month over last 6 months history for the payer
	Rate not PIM (last 3 months)	Overall % of documents not paid in month over last 3 months history for the payer
	Average delay (last 12 months)	Average observed delay on overdue documents, taking into account last 12 months history for the payer (nota: this indicator takes into account cleared documents, in order to have a robust information on the delay)
	Average delay (last 6 months)	Average observed delay on overdue documents, taking into account last 6 months history for the payer
	Average delay (last 3 months)	Average observed delay on overdue documents, taking into account last 3 months history for the payer
	Pay cycle	Most frequent Day of payment in the month when a pay cycle is detected
	Nb GBU for the payer	Number of GBU for the PRS Customer with open amount

Figure 3: illustration of indicators computed by the solution (view per GBU)

Technical description

Solution Architecture

|

Overview

There are four building blocks in or interacting with the solution:

- BW/SAP server: interaction in/out from the SAP system
- SFTP/SSH server/client: file exchange (in/out) and remotely process execution
- Python server: predictive engine and working list generation
- MSSQL server: data storage, features engineering and processing

Figure 4 is an overview of the solution architecture. BW pushes new data onto the BW SFTP server (1) and launches a remote command to trigger all processing (2). Python server connects to the BW SFTP server (using a sftp client) to download data files (3). These files are inserted in SQL server tables where they are processed and pushed back to the Python server for predictive modeling. Results are then pushed back to the SQL server. Result tables are available through BW, connecting directly to the SQL server (4).

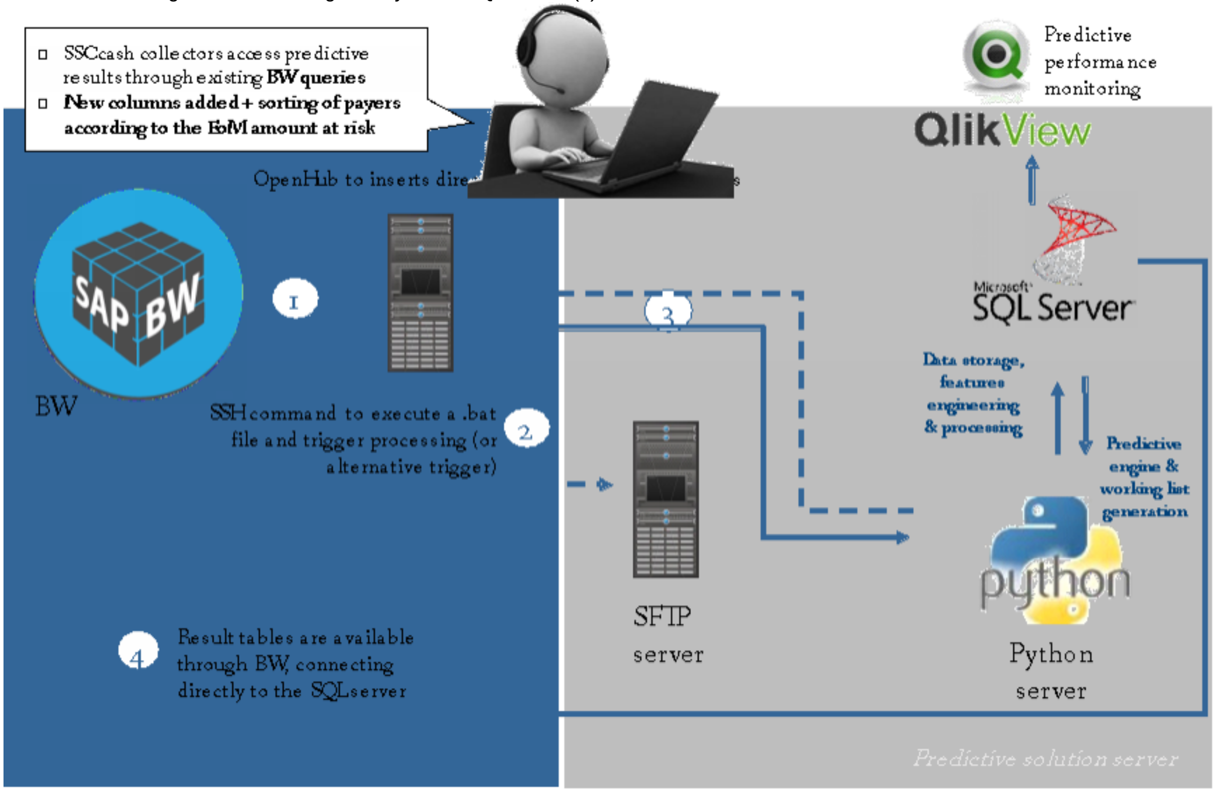


Figure 4. Solution architecture overview

Communication process between BW and the solution server

- Execution is lauched through a SSH command, running a .bat file on the sftp server
- End of processing is indicated through the .bat return code
- BW then read a specific status table in SQL server, with a code indicating the status and the actions required:
  - CODE = 0:
    - Results ready, to be uploaded BW reads result tables in the SQL server
  - CODE = 1
    - Error#1, data upload issue BW regenerates the input files and re-launch the SSH command
  - CODE = 2
    - Error#2, issue encountered specific manual action required (eg. Full reload of the data set)

Table 1 shows an example of the table status. Columns description are the following:

- TIMESTAMP: start timestamp of the command
- TIMESTAMP_END: end of the execution of the command
- COMAND: name of the command executed
- CODE: exist code of the application
- MESSAGE: description of the exit code

Table 1 Example of STATUS table

26/04/2017 13:16	26/04/2017 13:19	run	0	OK

Inbound connectors

For inbound connector, BW is interacting with the sftp/ssh server (see Figure 5).

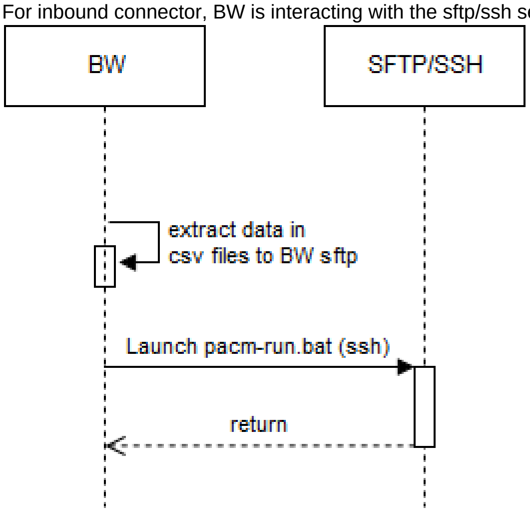


Figure 5. Inbound connectors sequence diagram

The BW server pushes csv files onto BW sftp server. The following csv file are expected:

- C_CST_CA2: credit code table (full replacement at each upload)
- C_COMPCDE: company code table (full replacement at each upload)
- C_CUSTID: customer code table (full replacement at each upload)
- GL_ACCOUNT: gl account code table (full replacement at each upload)
- G_CWWE01: sub activity table (full replacement at each upload)
- TCURR: exchange rate table (full replacement at each upload)
- COUNTRY: country table (full replacement at each upload)
- DBFIAR20: credit management table (incremental upload)
- DBFIAR21: order blocked table (incremental upload)

Then a ssh remote command launches the process called "**pacm-run.bat**"

Outbound connectors

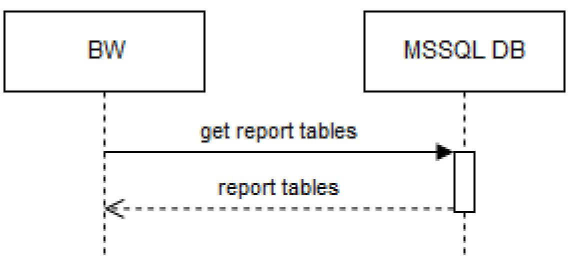


Figure 6. outbound connectors with MSSQL sequence diagram

For outbound connector, BW is interacting with the MSSQL database (Figure 6) to read the results tables (schema **cm_data**). The end of the name of the results tables are linked to the server:

- PRODUCTION SERVER: <SERVER> = PRD
- PRE-PRODUCTION/TEST SERVER: <SERVER>=TST

The results tables are the following:

- PRIORITY_PER_AMOUNT_<SERVER>: predictive indicators and statistics on each open document within prechasing and chasing scope for the month
- PRIORITY_PER_PAYER_<SERVER>: predictive indicators aggregated by payers, used for cash collection priorities setting
- PERF_MONITORING_<SERVER>: stores predictive performance of the past 4 months (predictive performance indicators are computed on the first day of each month, from previous months' results).

For example for the production server the name of the first result table will be: PRIORITY_PER_AMOUNT_PRD

Solution building blocks

Figure 7 describes all the interaction against the different components.

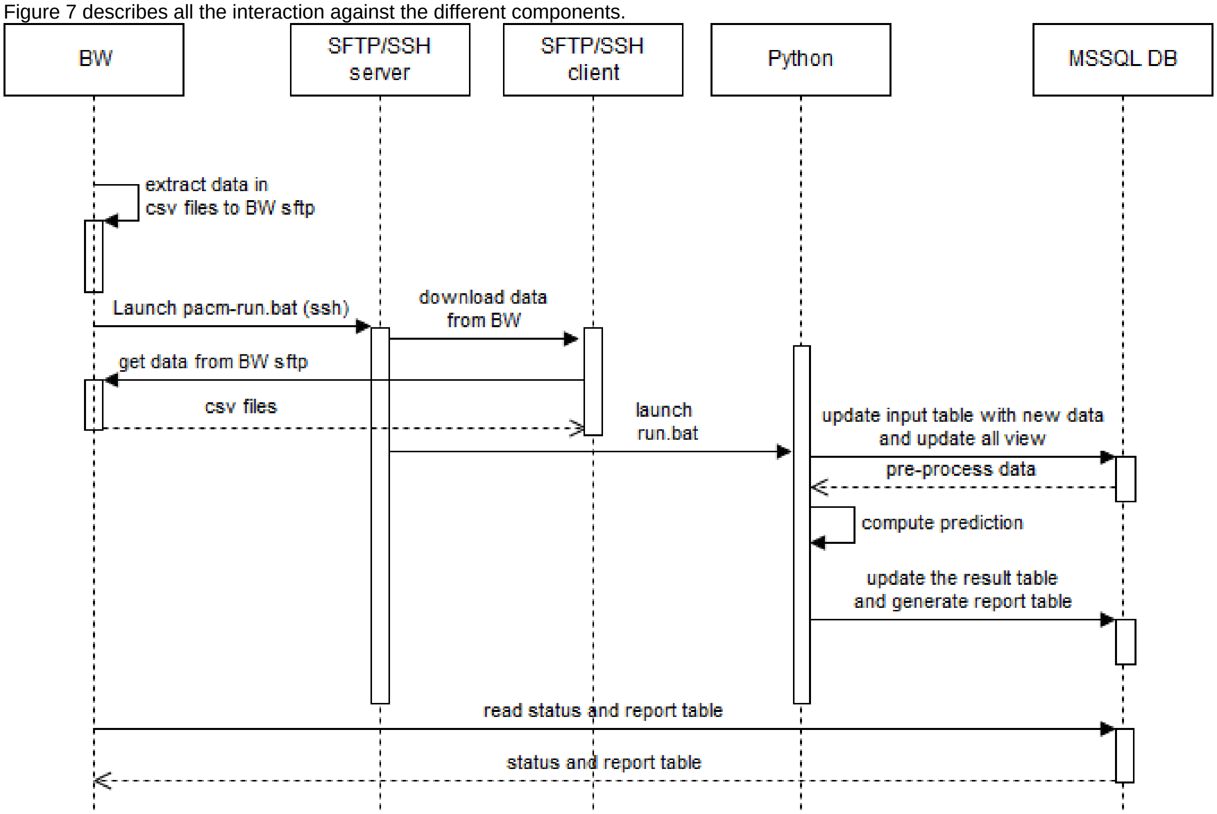


Figure 7. Sequence diagram

MS SQL server

SQL server is used for data storage, features engineering and processing.
Version : MSQL Server 2012

Connection

Type	Instance	DB
Pre-production	WDCPABP21\PACM_DBPRD	PACM_DBTST
Production	WDCPABP21\PACM_DBPRD	PACM_DBPRD

Inputs/outputs

The input tables are attached to the schema **cm_data**:

- C_CST_CA2: credit code table
- C_COMPCDE: company code table
- C_CUSTID: customer code table
- GL_ACCOUNT: gl account code table
- G_CWWE01: sub activity table
- TCURR: exchange rate table
- COUNTRY: country table
- DBFIAR20: credit management table
- DBFIAR21: order blocked table

Results tables are attached to the schema **cm_data**:

- PRIORITY_PER_AMOUNT_<SERVER>: list of priority per invoices
- PRIORITY_PER_PAYER_<SERVER>: list of priority per payer
- PERF_MONITORING_<SERVER>: follow the performance of the past 4 months

Python

Version :

- python 3.5.2
- scikit-learn 0.17.1
- pandas 0.18.1
- pymssql 2.1.3
- openpyxl 2.3.2
- xlrd 1.0.0

Connection

Type	Instance
Pre-production	WDCPAAT21
Production	WDCPAAP21

Inputs/outputs

All interactions with the sftp server go through the pacm-workspace ("**F:\pacm_workspace"**). The workspace folders are the following:

- **archive:** folder used to store, for each run, the value in the inbox folder
- **bin:** folder with all the executable (run, train, reset_db, ping)
- **etc:** folder with the configuration files for the loging (loging.config) and for the application (pacm.config)
- **inbox:** composed of one folder (**in**) used to get new data
- **logs:** folder to store the logs

SFTP server

Version : Bitvise SSH Server 7.16

Connection

Type	Instance
Pre-production	WDCPAAT21
Production	WDCPAAP21

Inputs/outputs

Inputs use the folder workspace inbox folder: **F:\pacm_workspace\inbox\in** to push new data in the application

SFTP client

Version : Bitvise SSH Client 7.22

Connection

Type	Instance
Dev Quality	wbdsapr3.ibm.be.solvay. com wbqsapr3.ibm.be.solvay.com
Production	wbpsapr3.ibm.be.solvay.com

Inputs/outputs

Download all the file from **/exploit/BW/PREDICTCM** in the workspace inbox folder **F:\pacm_workspace\inbox\in** using the following command :

```
sftpc -profile=F:\pacm-workspace\bin\bw_prod_sftp.tlp -hostKeyFile=F:\pacm-workspace\bin\bw_prod_sftp.pub -cmd="cd PREDICTCM; get * F:\pacm-workspace\inbox\in -o"
```

Deep-dive on solution building blocks

Table 1 describes the main step of the solution main steps. There is four main steps:

- Get data from BW
- Load data and compute features
- Predictive engine and working list generation
- Performance assessment

Table 2. Main steps

ID	Main step	Step	Description	Location
0	Get data from BW	From BW to python	Download the data from PW to python inbox folder	Batch script
1	Load data and compute features	From python to SQL	Push the raw data to SQL	Python function
		Features computation	Compute the customer features based on the raw data	SQL queries
2	Predictive engine and working list generation	Load the prediction models	Load the predictions models from the user workspace	Python function
		Get the data from the SQL	Load the data from SQL in python	Python function
		Predict the model	Apply the prediction model	Python function
		Chasing adjustement	Adjust the prediction for chasing invoices	Python function
		Push the predictive indicators to SQL	Write the prediction in SQL	Python function
		Generate result tables	Generate result tables in SQL	SQL queries
3	Performance assessment	Get the result of the past month	Compute the result of the past month	SQL queries
		Compute performance report	Compare the prediction and the reallity	SQL queries
4	Archive input data	Archive input data	Move the input data to the archive folder	Python function

MS SQL server

MS SQL server is used for data storage, features engineering and processing. All the queries are located in: *app\core\sql*
We focus on the online version (**app\core\sql\online**) of the sql queries in this section. The offline queries (**app\core\sql\offline**) are used to build-up the training dataset. They can be found in annex "**Off line sql queries**" (page). For each source table, all column names and types are described in annex DB table details (page 29).
Figure 8 describes the data model used in the MS SQL server. There are four main steps:

1. Create schema and table in the database
2. Preprocess data to compute predictive features (used by the Python predictive model)
3. Generate result tables, including predictive insights and statistics about past behavior
4. Evaluate predictive performance, comparing what was predicted and what was ultimately observed (monthly)

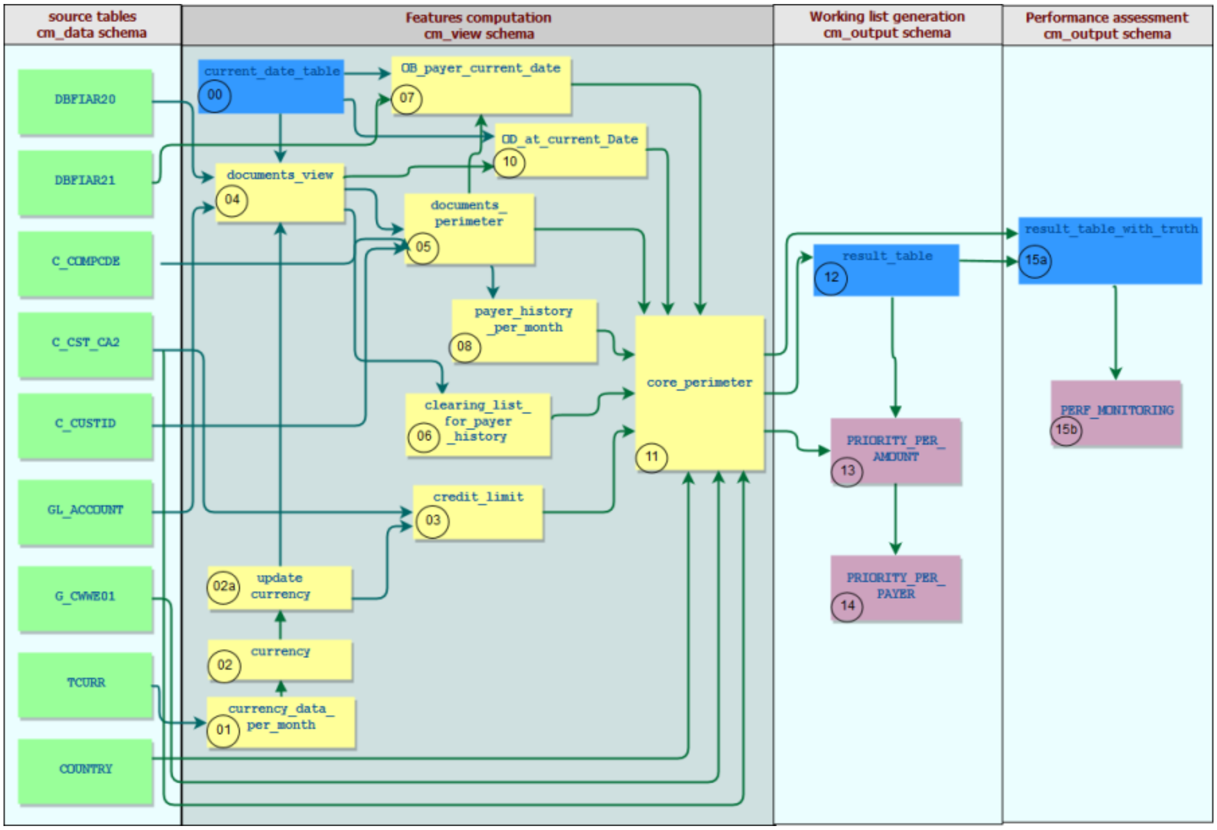


Figure 8. SQL Data model

Create database

Folder: app\core\sql\create_db
Description: clear and recreate the full schemas:

- cm : containing all input data
- cm_view : temporary views for online
- cm_view_training : temporary views for training
- cm_output: results
- cm_data: view used to interact with BW

List of files:

- 000_create_schema_cm_view_training: : create **cm_view_training** schema
- 0a_create_schema_cm.sql: create **cm** schema
- 0b_create_schema_cm_view.sql: create **cm_view** schema
- 0c_create_schema_cm_output.sql: create **cm_output** schema
- 1a_create_current_date_table.sql: create a table with the current date
- 1_create_table.sql: create input data tables
- 2_Solvay_functions.sql: define all the specific functions used in the queries
- 3_create_input_data_view.sql: create **cm_data** schema and view with the input data
- 4_create_payment_table.sql: create PAYMENT_DESC_TABLE with the description of the payment method

Preprocess data to compute predictive features

This section followings the graph numbering in Figure 8.

00 – current_data_table

Python function: app.simul.predictive_model.update_simulation_day()
This table contains the current timestamp in UTC, it is updated by python script after loading the data in the database.
Table 3. current_date_table preview

Current Date
2016-06-01 0:00:00.000

01 – currency_data_per_month

Folder: app\core\sql\create_view\online
Files: 01_transform_currency.sql
From: TCURR table
Where: -
Group by: -
Description: Fill the currency table based on the TCURR table with one exchange rate per month of the year. If for one month, it is not given, then use the last available
Note: K_FFACT and K_TFACT cannot be zero, zero values are replaced per 1

Table 4. currency_data_per_month table preview

C_FRMCUR	C_TOCURR	K_ECRATE	month	year	K_FFACT	K_TFACT	C_ECREDAT
PLN	USD	0.319423504	3	1967	1	1	30/11/2010
NZD	HKD	6.77448	6	2014	1	1	30/06/2014
GBP	HKD	15.31908	6	1978	1	1	20/03/2007
GBP	PLN	5.60821	9	1977	1	1	31/01/2006
HKD	GBP	0.072801769	1	1988	1	1	31/01/2006
HKD	INR	5.67652	4	1998	1	1	20/03/2007
HKD	PHP	6.2152	5	1955	1	1	20/03/2007
HUF	PLN	0.01519	8	1978	100	1	31/01/2006
CAD	EUR	0.699986	4	2016	1	1	29/04/2016
...	...	...	....	...	...	...	...

02 – currency

Folder: app\core\sql\create_view\online
Files: 02_Currency_preprocessing.sql
From: currency_data_per_month
Where: -
Group by: -
Description: Compute historical statistics for each currency

Table 5. currency table preview

C_FRMCUR	C_TOCUR	K_ECRAT	Month	Year	K_FFAC	K_TFAC	Avg Ex Rat	Dev Ex Rate	Min Ex Rat	Max Ex Rate
THB	EUR	0.02511	12	2016	1	1	0.02523	0.00024	0.02507	0.02576
AED	EUR	0.23925	12	2016	1	1	0.24168	0.00444	0.23925	0.24971
AUD	EUR	0.66899	12	2016	1	1	0.66727	0.00700	0.64986	0.67723
JPY	EUR	0.00817	12	2016	1	1	0.00806	0.00022	0.00756	0.00817
HUF	EUR	0.00320	12	2016	1	1	0.00320	0.00001	0.00316	0.00321
MXN	EUR	0.05113	12	2016	1	1	0.05114	0.00063	0.04999	0.05287
SGD	EUR	0.65313	12	2016	1	1	0.65196	0.00294	0.64309	0.65313
HRK	EUR	0.13314	12	2016	1	1	0.13261	0.00100	0.13055	0.13314
...	...	...	....	...	...	...	...	...	...	...

02a – update currency table

Folder: app\core\sql\create_view\online
Files: 02a_modif_currency_table_for_JPY_KRW
From: currency_data_per_month
Where: -
Group by: -
Description: Update CURRENCY table for currency JPY and KRW. SAP store JPY and KRW currency / 100. This table modify the currency table to add a multiplication per 100 (K_TFACT column).
Warning: Hard coded data modification on specific currency JPY and KRW

03 – credit_limit

Folder: app\core\sql\create_view\online
Files: 03_Credit_limit_preprocessing.sql
From: C_CST_CA2, currency
Where: -
Group by: -
Description: Convert the credit limit in EUR

Table 6. credit_limit table preview

C_CST_CA2	oCRED_LIMIT	oREC_LIM_CUR	oREC_CR_LIM	C_FRMCUR	K_ECRATE	K_TFACT	K_FFACT	Credit_Limit_EUR	Rec Credit_Limit_EUR
1	0	EUR	1000000	EUR	1	1	1	0	1000000
1000005	0	EUR	0	EUR	1	1	1	0	0
100001	0	EUR	0	EUR	1	1	1	0	0
1000016	0	EUR	0	EUR	1	1	1	0	0
1000026	0	EUR	0	EUR	1	1	1	0	0
1000030	0	EUR	0	EUR	1	1	1	0	0
...	...	...	....	...	...	...	...	...	...

04– documents_view

Folder: app\core\sql\create_view\online
Files: 04_Documents_list.sql
From: DBFIAR20, C_CUSTID, GL_ACCOUNT, current_date_table, currency, C_COMPCDE, G_CWWE01, PAYMENT_DESC_TABLE
Where: GL_ACCOUNT."C_GL_TYPE" = 'RECEIVABLES' and (GL_ACCOUNT."C_GL_STYP" = 'PRODUCT AND SERVICES' or GL_ACCOUNT."C_GL_STYP" = 'SERVICES NON DOUBT' or GL_ACCOUNT."C_GL_STYP" = 'PRODUCT NON DOUBT') and DBFIAR20."0C_CTR_AREA" = 'SOLV'
Group by: -
Description: Preliminary filter, to reduce dimensions and transform key datas used in other views. Convert and consolidate the amount in EUR.
Note: For converting the amount in euro, the rate associated with the "0CLEAR_DATE" is used. If the document is still open, then the most recent exchange rate is used.
Warning: Hard coded document filtering is implemented here

05– documents_perimeter

Folder: app\core\sql\create_view\online
Files: 05_Documents_filtered_perimeter.sql
From: documents_view
Where: (documents_view."0CREATEDON" < documents_view."0CLEAR_DATE" or documents_view."0CLEAR_DATE" is null) and documents_view."Amount EUR" is not null
Group by: -
Description: Full data sample on which customer features are computed. Targets to be predicted are computed, as well as most features directly related to the document or static customer attributes. Condition on documents."Amount EUR" allow to remove the documents outside the scope. Table 7 explains the WHERE clause.

Table 7 Explanation of the where clause for documents_perimeter

documents_view."0CREATEDON" < documents_view."0CLEAR_DATE"	remove documents create and clear the same day (automatic clearing)
documents_view."0CLEAR_DATE" is null	keep open documents
documents_view."Amount EUR" is not null	remove document out of the scope (see Amount EUR calculation from documents_view)

06– clearing_list_for_payer_history

Folder: app\core\sql\create_view\online
Files: 06a_Clearing_list_by_payer_for_payer_history.sql
From: documents_view
Where: -
Group by: "Payer ID"
Description: this table computes the average payment day and week of a customer for the last 12 months

07– OB_payer_current_date

Folder: app\core\sql\create_view\online
Files: 07_OB_payer_per_current_date.sql
From: DBFIAR21
Where: -
Group by: "Payer ID "
Description: compute the block orders information per payer (number of blocked order, average resolution time)

08– payer_history_per_month

Folder: app\core\sql\create_view\online
Files: 08_Payer_history_per_month.sql
From: documents_perimeter
Where: ((documents_perimeter.C_FCONNUM is not null and documents_perimeter.[0LOGSYS] = 'PI1_020' and ([0CLEAR_DATE] >= '2017-02-01' or [0PSTNG_DATE] >= '2017-02-01')) or ([0CLEAR_DATE] < '2017-02-01') or (documents_perimeter.C_FCONNUM is null))
Group by: "Payer ID"
Description: past payer behavior for the last 12, 6 and 3 month, accounting for all previous documents keeping only the legal document for the document with a factoring contract number.

10– OD_at_current_Date

Folder: app\core\sql\create_view\online
Files: 10_Open_Documents_at_current_date.sql
From: documents, current_date_table
Where: -
Group by: -
Description: for each payer and current date, this view computes contextual indicators accounting for all open documents

11– core_perimeter

Folder: app\core\sql\create_view\online
Files: 11_Core_perimeter.sql
From: documents, current_date_table, currency, payer_history, payer_history_distinct, OD_at_Due_Date, credit_limit, OB_payer_due_date, DBFIAR20, C_COMPCDE, C_CST_CA2, C_CUSTID, GL_ACCOUNT, G_CWWE01, COUNTRY
Where:

- documents_perimeter."0FI_DOCSTAT" = 'O' : keep only open document
- (current_date_table."Current Date" < C_COMPCDE."C_MERGDAT" or C_COMPCDE."C_MERGDAT" is null): remove company merged
- documents_perimeter."0NETDUEDATE" <= EOMONTH(current_date_table."Current Date"): keep only open document with a net due date in the current month

Group by: -
Description: aggregate all previously computed information into one singe table. This table will then be used to apply the predictive model in Python.

12–result_table

Python function:

- *app.core.model.predictive_model_pckg.get_open_documents_from_db* (): extract from core_perimeter all the data where "0NETDUEDATE" is before or in the current month + (Apply predictive model)
- *app.core.model.predictive_model_pckg. push_pred_in_db()*: push the prediction in the result table

13–PRIORITY_PER_AMOUNT

Folder: app\core\sql\report
Files: 0a_outstanding_amount.sql, 0b_priority_per_amount.sql
From: DBFIAR20, result_table
Where: -
Group by: -
Description: for each document, this view computes contextual indicators accounting for all open documents

14–PRIORITY_PER_PAYER

Folder: app\core\sql\report
Files: 00_Clearing_list_by_payer_for_payer_history_per_GBU.sql, 00_Payer_history_per_month_per_zone_logsys_c_custid.sql, priority_payer.sql, 1_priority_C_CUSTID_payer.sql
From: PRIORITY_PER_AMOUNT
Where: -
Group by: [Payer ID], [0LOGSYS], [C_CUSTID], [PRS Cust. Zone (FI)], [GBU]
Description: for each GBU/payer, this view computes contextual indicators accounting for all open documents

15a–result_table_with_truth

Folder: app\core\sql\evaluate_perf
Files: 01_create_truth_table.sql
Description: Merge the result table of the prediction with what was observed during the previous month.

15b–PERF_MONITORING

Use for the performance evaluation – see User guide/Monitoring
Folder: app\core\sql\evaluate_perf
File: 02_perf_per_day.sql
Description: Performance of the algorithm is computed for each day and each zone from the previous month information. This query is executed at the beginning of each month.

Python

Table 8 describes the main python function called during the generation of the working list.
Table 8. Main python function for working list generation

0	Get data from BW	Download the data from PW to python inbox folder	app.process.get_data_from_bw_sftp()
1	Load data	Push the raw data to SQL	app.core.predictive_model.PredictiveModel.insert_data_in_db()
		Compute the customer features based on the raw data	app.predictive_model.create_all_view()
2	Predictive engine and working list generation	Load the predictions models from the user workspace	app.core.predictive_model.PredictiveModel.load()
		Load the data from SQL in python	app.core.predictive_model.PredictiveModel. get_open_documents_from_db()
		Apply the prediction model	app.core.predictive_model_pckg.load_and_apply_model()

		Adjust the prediction for chasing invoices	app.core.predictive_model_pckg.apply_reguralize_ratio_model()
		Write the prediction in SQL	app.core.predictive_model.PredictiveModel.push_pred_in_db()
		Execute SQL query to generate the result tables	app.predictive_model.create_report()
3	Performance assessment	Select results of the previous month	app.predictive_model.evaluate_perf()
		Compare past prediction with observed payment behavior	app.predictive_model.evaluate_perf()
4	Write reports and archive	Move the input data and report results to the archive folder	app.process.archive_inbox()

The main parts of the python code are in:

- "app/core": main function for model prediction
- "app/simul": function used for simulation
- "app/sklearn_ext": function used to preprocess the data
- "app/io": function to read or write a files
- "app/utils": useful functions like sql connection, ….

User guide

Installation

Two zipped files are needed to install the solution:

- pacm-0.8.0.zip: pacm application
- pacm-model.zip: predictive model

Step-by-step installation guide:

1. Unzip the application file in the folder "E:\PACM<*version*>"
2. Copy the folder "E:\PACM<*version*>\workspace_template" to "F:\pacm_workspace"
3. Update the BIN_FOLDER in : "F:\pacm_workspace\bin\ _env.bat"

    ```
    set BIN_FOLDER= E:\PACM<version>\bin
    ```

4. Configure the "F:\pacm_workspace\etc\pacm.config":

<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="858e273f-7ca0-4621-9072-46520c8699d8"><ac:plain-text-body><![CDATA[	[python server]]]></ac:plain-text-body></ac:structured-macro>

5.

server wher the application is run TST or PRD
Server=TST

[sql_server]
server=***
user=***
password=***
database=***
port=***

6.

folder used to bulk insert the result in the table
base_bulk_insert_folder=\\WDCPAAT21\bulk_insert

[model]

to skip the import_data step in case data is inserted directly in db
#skip_import_data=True

[predict]

8.

model folder to use
model_folder= F:\pacm_workspace\models\20161111

[train]

9. base model folder where the new trained model folder will be created,
10.

named according to current timestamp
model_base_folder=F:\pacm_workspace\models\
training_period_in_month = 36
testing_period_in_month = 6

[dynamic_threshold]

11.

dynamic threshold use to adjust the risk for customer with large outstanding amount
amount_threshold_list = [(300000, 0.5), (600000, 0.25), (1200000, 0)]

|

12. To test the connection to the db run: "F:\pacm_workspace\bin\ping.bat"
13. To create the db run: "F:\pacm_workspace\bin\create_db.bat"
14. Unzip the model (pacm-model.zip) in: "F:\pacm_workspace\models\20161111"
15. Launch the executable: "F:\pacm_workspace\bin\run.bat"

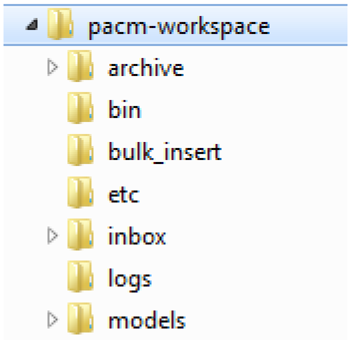


Figure 9. pacm workspace
Figure 9 shows the folder in the pacm-workspace

Tunning of the dynamic model

One parameter see the dynamic threshold section in the configuration file. This threshold allows to adapt the risk based on the outstanding amount of a customer. This dynamic threshold is used to compute the Risk Amount for the documents with a positive amount with the following formula:
RiskAmount= Outstanding Amount* Risk Probability
with Risk Probability=if Probability not PIM<threshold then 1if Probability not PIM>threshold then Probability not PIM
Table 9 shows the dynamic threshold.
Table 9 Dynamic probability threshold

0 - 300 K€	100%
300 K€ - 600 K€	50%
600 K€ - 1 200 K€	25%
1 200 K€ -	0%

These thresholds can be adjust in the configuration file in the "Dynamic treshold" section:

1.

dynamic threshold use to adjust the risk for customer with large outstanding amount
amount_threshold_list = [(300000, 0.5), (600000, 0.25), (1200000, 0)]
For example:

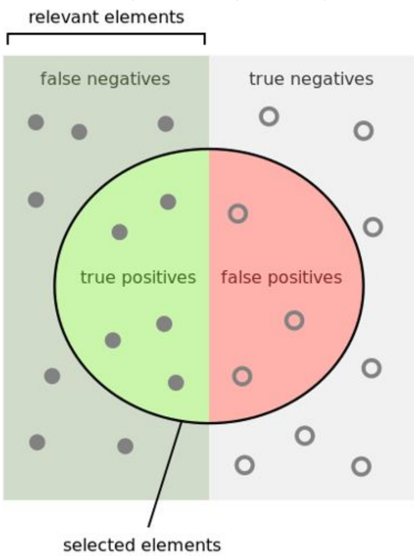
- If a customer has an outstanding amount of 1 500 K€, then the probability threshold is set to 0%. It means that all documents of the customer with a positive amount will have a risk probability set to 1. The other documents will keep the value of probability computed by the model.
- If a customer has an outstanding amount of 700 K€, then the probability threshold is set to 25%. It means that all documents of the customer with a positive amount and with a probability not PIM over 25% will have a risk probability set to 1. The other documents will keep the value of probability computed by the model.

- If a customer has an outstanding amount of 100 K€, then the probability threshold is set to 100%. It means that all documents of the customer will keep the value of probability computed by the model.

Monitoring

Performance

The solution generates a performance monitoring table at the beginning of each month, comparing predictive insights generated with realized payment behavior. Description of the performing monitoring columns are in annex "Description of the performance monitoring table" (page)



The usual predictive performance monitoring metrics are:

- Recall Score: it quantifies the true positive rate (value between 0 and 1).

- Precision Score: it quantifies how many documents are flagged as positive (depending on the target this will mean 'not paid in month' or 'overdue') are actually so (value between 0 and 1).

- AUC: Area under the Curve, which plots Recall and precision for different decision threshold.

As the main objectif is to prioritize cash collection actions, the solution also generates Lorenz curves. They sort customers with different ordering criteria (simulating the working list ranking) and compare the total amount not paide in month for the first N customers. The curves allow to compare three different criteria and monitor the predictive ranking gain:
Customer number

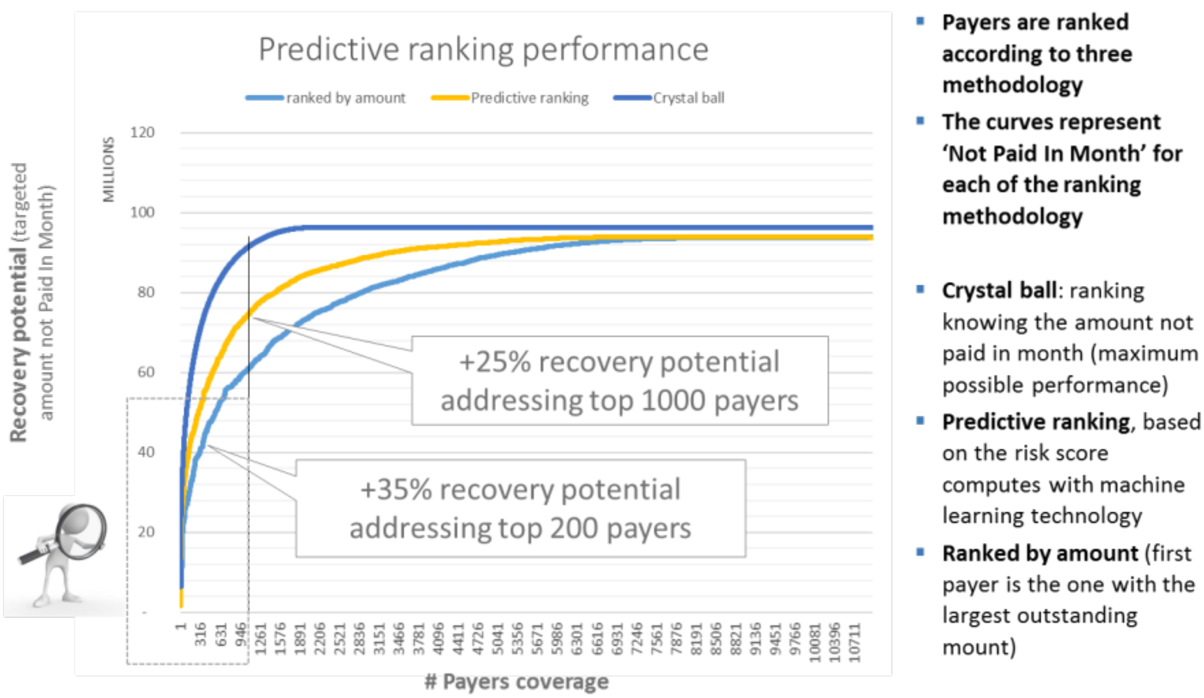


Figure 10. Ranking performance

Technical

All application logs are stored in the folder "F:\pacm_workspace\logs".
All the input data are archived in the volume "F:\pacm_workspace\archive". If the folder size exceed the current volume size (100Go), the volume size should be increase.
If the size of the data base execeed the SQL database size, the data base should be increase.

Procedure to retrain the model

Attention point: model training is a long process (around 12 hours)
Three models are used:

- Model for prediction of the documents not paid in the current month (**prediction_model_training_calibrated_1_PIM.pk**)
- Model for prediction of the documents overdue (**prediction_model_training_calibrated_1_overdue.pk**)
- Regularisation model to correct the probability for long overdue documents (**reg_model_All_norm.pk**)

When to retrain the model?

The model should be retrained when one of the following happens:

- Major point-in-time degradation of the predictive performance, as shown with Lorentz curve metrics (see above)
- Smaller but progressive drift during two months, assessed through either:
  - Lorentz curve metrics (see above)
  - Expected not PIM amount ratio (comparing the expected not PIM amount with the realized amount not Paid In Month). Point-in-time delta can happen due to very large amounts on some overdue documents, but a continuous drift requires attention.

How to retrain the model (step-by-step)?

Retrain model for prediction of not paid documents in the current month / documents overdue

1. Set the training period in the configuration file (train section)

[train]

1. base model folder where the new trained model folder will be created,
2. named according to current timestamp
   model_base_folder=F:\pacm_workspace\models\
   training_period_in_month = 30
   testing_period_in_month = 6
3. To train the model run: "F:\pacm_workspace\bin\train.bat"
4. A new folder is created in the "model_base_folder"
5. Change in the predict section of the configuration file the new folder path

[predict]

1. model folder to use
   model_folder= F:\pacm_workspace\models\20161111
   This procedure uses lots of memory and is time consuming (~12h). It is advices to retrain the model on a local computer. To re-train localy the model:

- Copy the online DataBase on your local environment
- Install the application on your computer
- Run the train process

Retrain model for regularization model

To retrain the regularization model the data of the past four month are used (in table result_table_with_truth).

1. To train the model run: "F:\pacm_workspace\bin\ train_regularization_model.bat"
2. A new folder is created in the "model_base_folder"
3. Change in the predict section of the configuration file the new folder path

[predict]

1. model folder to use
   model_folder= F:\pacm_workspace\models\20161111

What are the checkpoints?

Check the model validity using the performance assessment file stored in the folder with the trained model. For the overdue model the file is : **prediction_model_training_calibrated_1_overdue.csv** and for the not paid in month model the file is : **prediction_model_training_calibrated_1_PIM.csv**.
Model information stored in the file are the following:

- **Total_observation** : total number of observation
- **Total_observation_Train** : number of observations used to train the model
- **Total_observation_Test** : number of observations user to test the model
- **AUC_score** : Indicator of the global model performance. This score corresponds to the area under the precision-recall curve. This score should be over 0.85 (see monitoring performance section).
- **Confusion matrix** : precision, recall and support on the test data. The first line is for the document paid in month, the second line is for document not paid in month.
- **Features importances of the variable in the model** : List of the feature importance in the variable in the model

You will find below an example of prediction_model_training_calibrated_1_PIM.csv file.
prediction_model_training_calibrated_1_PIM
Total_observation : 8083490
Total_Observations_Train : 7028867
Total_Observations_Test : 1054623
AUC_score : 0.909
Scores_Matrix :
"precision,recall,support"
"0.889,0.939,807373.000"
"0.756,0.616,247250.000"
Features importances :
"variable_name,feature_importance"
"scenario,0.208"
"Rate_Not_PIM_last_12_month,0.087"
"AVG_delay_over_remaining_days_last_12_month,0.072"
"AVG_delay_over_remaining_days_last_3_month,0.052"
"AVG_delay_over_remaining_days_last_6_month,0.045"
"Ratio_remaining_days_in_month,0.041"
"C_COMPCAF,0.036"
"Amount_Rate_Not_PIM_last_12_month,0.033"
"Ratio_AVG_day_of_month,0.031"
"Rate_Not_PIM_last_6_month,0.029"
"Payment Term,0.029"
"0POST_KEY,0.028"
"Nb_docs_last_12_month,0.026"
"Nb_docs_last_3_month,0.025"
"Amount EUR,0.025"
"Nb_docs_last_6_month,0.024"
"Relative_Amount_vs_OD,0.020"
"Amount_ratio_Late_OD,0.019"
"Ratio_AVG_week_number,0.019"
"Rate_Not_PIM_last_3_month,0.017"
"nb_distinct_clear_date,0.016"
"STDEV_week_number,0.016"
"STDEV_day_of_month,0.015"
"Amount_Rate_Not_PIM_last_6_month,0.014"
"C_DOCTYP,0.014"
"LAST_C_PM_MTHD,0.012"
"GBU_group,0.012"
"Amount_Rate_Not_PIM_last_3_month,0.012"
"Create_post_lag,0.010"
"OD_Due_Date_vs_CL_EUR,0.005"
"OB_nb_overdue,0.004"
"C_COMPCDE__K_INTRAT,0.002"

Procedure for Exec builder

If some modification is made to the source code, then the application executable must be re-build. The steps to package the python source code in an application are :

1. Check the version of the application to build in the file "version.py"
2. Run "build.bat"

The compress application named "pacm-<*version*>.zip" is created in the folder "dist"
Then follow the installation steps of the user guide.

APPENDIX

Off line sql queries

For training/offline, all the function works in **cm_view_training** schema.
These offlines queries are run to compute the dataset used to train the model. To simulate the offline behavior for each day in the training period, a view of DBFIAR20 is created with only the data availaible at the simulation date. Based on this view, all the customer features are computed (same queries as online on **cm_view_training** schema). To reduce the size of the training data, only the 1, 5 ,10 ,15, 20, 25 and last day of month are used. Figure 11 shows the training data generation mechanism.

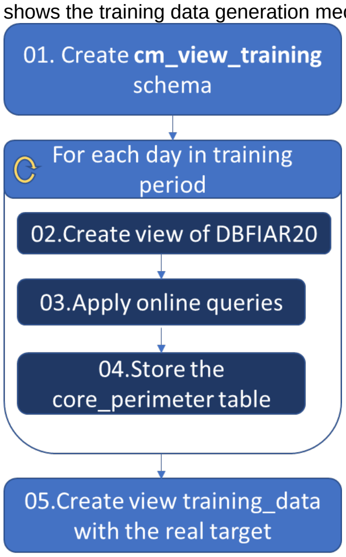


Figure 11 Training data generation

01 – create schema cm_view_training

Folder: app\core\sql\create_view\offline
Files: 000_create_schema_cm_view_training.sql
From: -
Where: -
Group by: -
Description: create the schema cm_view_training

02 – create credit_mgt_view

Folder: app\core\sql\create_view\offline
Files: 001a_modif_input_data_to_keep_doc.sql
From: DBFIAR20
Where: [0PSTNG_DATE] <= cm_view_training.current_date_table."Current Date"
Group by: -
Description: This view is a preliminary filter, to simulate the DBFIAR20 data availaible at a given date. This view is then used to apply all the online queries.

03 – APPLY THE ONLINE QUERIES ON CREDIT_MGT_VIEW in shema cm_view_training

Folder: app\core\sql\create_view\offline
Files: 01_transform_currency.sql, 02a_modif_currency_table_for_JPY_KRW.sql, 02_Currency_preprocessing.sql, 03_Credit_limit_preprocessing.sql, 04_Documents_list.sql, 05_Documents_filtered_perimeter.sql, 06a_Clearing_list_by_payer_for_payer_history.sql, 07_OB_payer_per_current_date.sql, 08_Payer_history_per_month.sql, 10_Open_Documents_at_current_date.sql
Description: Same queries than online apply on **cm_view_training schema** and based on the **credit_mgt_view** (modification in 04_Documents_list.sql in the FROM clause)

04– Save the core_perimeter table in core_perimeter_concat

Folder: app\core\sql\create_view\offline
Files: 12_Core_perimeter_concat.sql
From: core_perimeter
Where: -
Group by: -
Description: Save the previous core_perimeter table

05- Create training data with the updated target

Folder: app\core\sql\create_view\offline\final_table
Files: 0_create_result_view.sql
From: core_perimeter_concat
Where: ((C_FCONNUM is not null and 0LOGSYS] = 'PI1_020' and ([0CLEAR_DATE] >= '2017-02-01' or [0PSTNG_DATE] >= '2017-02-01')) or ([0CLEAR_DATE] < '2017-02-01') or (C_FCONNUM is null))
Group by:
Description: Generate the training data keeping only the legal document for the document with a factoring contract number

DB table details

Table 11 describes the input format of the table in the database.
Table 11. Table column definition

cm_data	STATUS_TST	TIMESTAMP	datetime
cm_data	STATUS_TST	TIMESTAMP_END	datetime
cm_data	STATUS_TST	COMMAND	varchar
cm_data	STATUS_TST	CODE	int
cm_data	STATUS_TST	MESSAGE	varchar
cm_data	PRIORITY_PER_PAYER_TST	0LOGSYS	char
cm_data	PRIORITY_PER_PAYER_TST	C_CUSTID	char
cm_data	PRIORITY_PER_PAYER_TST	PRS Cust. Zone (FI)	varchar
cm_data	PRIORITY_PER_PAYER_TST	GBU	varchar
cm_data	PRIORITY_PER_PAYER_TST	PRS Customer	varchar
cm_data	PRIORITY_PER_PAYER_TST	outstanding amount	numeric
cm_data	PRIORITY_PER_PAYER_TST	overdue amount	numeric
cm_data	PRIORITY_PER_PAYER_TST	overdue amount_0_4	numeric
cm_data	PRIORITY_PER_PAYER_TST	overdue amount_5_10	numeric
cm_data	PRIORITY_PER_PAYER_TST	overdue amount_11_30	numeric
cm_data	PRIORITY_PER_PAYER_TST	overdue amount_31_60	numeric
cm_data	PRIORITY_PER_PAYER_TST	overdue amount_61_90	numeric
cm_data	PRIORITY_PER_PAYER_TST	overdue amount_90+	numeric
cm_data	PRIORITY_PER_PAYER_TST	Prechasing amt lst 3 dom	numeric
cm_data	PRIORITY_PER_PAYER_TST	Risk amount not PIM	numeric
cm_data	PRIORITY_PER_PAYER_TST	probability_Not_PIM	decimal
cm_data	PRIORITY_PER_PAYER_TST	Prechasing amount	numeric
cm_data	PRIORITY_PER_PAYER_TST	Risk not PIM prechasing	numeric
cm_data	PRIORITY_PER_PAYER_TST	Risk overdue prechasing	numeric
cm_data	PRIORITY_PER_PAYER_TST	Per. Exp. Amt overdue	numeric
cm_data	PRIORITY_PER_PAYER_TST	chasing amount	numeric
cm_data	PRIORITY_PER_PAYER_TST	Risk not PIM chasing	numeric
cm_data	PRIORITY_PER_PAYER_TST	Risk ranking	int
cm_data	PRIORITY_PER_PAYER_TST	OB_nb_overdue	int
cm_data	PRIORITY_PER_PAYER_TST	LAST_C_PM_MTHD	char

cm_data	PRIORITY_PER_PAYER_TST	Nb_docs_last_12_month	int
cm_data	PRIORITY_PER_PAYER_TST	Nb_docs_Not_PIM_last_12_month	int
cm_data	PRIORITY_PER_PAYER_TST	Nb_docs_last_6_month	int
cm_data	PRIORITY_PER_PAYER_TST	Nb_docs_Not_PIM_last_6_month	int
cm_data	PRIORITY_PER_PAYER_TST	Nb_docs_last_3_month	int
cm_data	PRIORITY_PER_PAYER_TST	Nb_docs_Not_PIM_last_3_month	int
cm_data	PRIORITY_PER_PAYER_TST	Nb_late_docs_12_month	int
cm_data	PRIORITY_PER_PAYER_TST	Cumulated_Delay_last_12_month	int
cm_data	PRIORITY_PER_PAYER_TST	Nb_late_docs_6_month	int
cm_data	PRIORITY_PER_PAYER_TST	Cumulated_Delay_last_6_month	int
cm_data	PRIORITY_PER_PAYER_TST	Nb_late_docs_3_month	int
cm_data	PRIORITY_PER_PAYER_TST	Cumulated_Delay_last_3_month	int
cm_data	PRIORITY_PER_PAYER_TST	Pay cycle_dom	numeric
cm_data	PRIORITY_PER_PAYER_TST	Pay_cycle_week	numeric
cm_data	PRIORITY_PER_PAYER_TST	Nb GBU for the payer	int
cm_data	PRIORITY_PER_AMOUNT_TST	simulation_ts	datetime
cm_data	PRIORITY_PER_AMOUNT_TST	0LOGSYS	char
cm_data	PRIORITY_PER_AMOUNT_TST	C_COMPCDE	char
cm_data	PRIORITY_PER_AMOUNT_TST	C_CUSTID	char
cm_data	PRIORITY_PER_AMOUNT_TST	0FISCPER	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	0AC_DOC_NO	char
cm_data	PRIORITY_PER_AMOUNT_TST	0ITEM_NUM	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	0FI_DSBITEM	char
cm_data	PRIORITY_PER_AMOUNT_TST	PRS Cust. Zone (FI)	varchar
cm_data	PRIORITY_PER_AMOUNT_TST	GBU	varchar
cm_data	PRIORITY_PER_AMOUNT_TST	PRS Customer	varchar
cm_data	PRIORITY_PER_AMOUNT_TST	Payer ID	varchar
cm_data	PRIORITY_PER_AMOUNT_TST	outstanding amount	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	overdue amount	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	overdue amount_0_4	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	overdue amount_5_10	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	overdue amount_11_30	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	overdue amount_31_60	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	overdue amount_61_90	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	overdue amount_90+	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	pre-chasing amount last three	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	0COUNTRY	char
cm_data	PRIORITY_PER_AMOUNT_TST	Amount EUR	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	0POST_KEY	char
cm_data	PRIORITY_PER_AMOUNT_TST	0NETDUEDATE	date
cm_data	PRIORITY_PER_AMOUNT_TST	scenario	varchar
cm_data	PRIORITY_PER_AMOUNT_TST	scenario_last_3_days	varchar
cm_data	PRIORITY_PER_AMOUNT_TST	Number of days past due date	int
cm_data	PRIORITY_PER_AMOUNT_TST	Risk amount not PIM	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	probability_Not_PIM	decimal
cm_data	PRIORITY_PER_AMOUNT_TST	probability_Not_PIM_Risk	decimal
cm_data	PRIORITY_PER_AMOUNT_TST	proba_overdue	decimal

cm_data	PRIORITY_PER_AMOUNT_TST	Prechasing amount	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	Risk not PIM prechasing	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	Risk overdue prechasing	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	chasing amount	numeric
cm_data	PRIORITY_PER_AMOUNT_TST	Risk not PIM chasing	numeric
cm_data	PERF_MONITORING_TST	evaluation day	datetime
cm_data	PERF_MONITORING_TST	simulation_day	datetime
cm_data	PERF_MONITORING_TST	C_CUSTID__C_ZONEFI	char
cm_data	PERF_MONITORING_TST	precision_Paid_In_Month	numeric
cm_data	PERF_MONITORING_TST	recall_Paid_In_Month	numeric
cm_data	PERF_MONITORING_TST	precision_Not_Paid_In_Month	numeric
cm_data	PERF_MONITORING_TST	recall_Not_Paid_In_Month	numeric
cm_data	PERF_MONITORING_TST	Expected not PIM amount	numeric
cm_data	PERF_MONITORING_TST	Not PIM amount	numeric
cm_data	PERF_MONITORING_TST	Expected not PIM amount ratio	numeric
cm_data	PERF_MONITORING_TST	NOT_PIM	numeric
cm_data	PERF_MONITORING_TST	CB_NOT_PIM_50	numeric
cm_data	PERF_MONITORING_TST	CB_NOT_PIM_100	numeric
cm_data	PERF_MONITORING_TST	CB_NOT_PIM_200	numeric
cm_data	PERF_MONITORING_TST	CB_NOT_PIM_300	numeric
cm_data	PERF_MONITORING_TST	CB_NOT_PIM_400	numeric
cm_data	PERF_MONITORING_TST	CB_NOT_PIM_500	numeric
cm_data	PERF_MONITORING_TST	CB_NOT_PIM_1000	numeric
cm_data	PERF_MONITORING_TST	A_NOT_PIM_50	numeric
cm_data	PERF_MONITORING_TST	A_NOT_PIM_100	numeric
cm_data	PERF_MONITORING_TST	A_NOT_PIM_200	numeric
cm_data	PERF_MONITORING_TST	A_NOT_PIM_300	numeric
cm_data	PERF_MONITORING_TST	A_NOT_PIM_400	numeric
cm_data	PERF_MONITORING_TST	A_NOT_PIM_500	numeric
cm_data	PERF_MONITORING_TST	A_NOT_PIM_1000	numeric
cm_data	PERF_MONITORING_TST	EA_NOT_PIM_50	numeric
cm_data	PERF_MONITORING_TST	EA_NOT_PIM_100	numeric
cm_data	PERF_MONITORING_TST	EA_NOT_PIM_200	numeric
cm_data	PERF_MONITORING_TST	EA_NOT_PIM_300	numeric
cm_data	PERF_MONITORING_TST	EA_NOT_PIM_400	numeric
cm_data	PERF_MONITORING_TST	EA_NOT_PIM_500	numeric
cm_data	PERF_MONITORING_TST	EA_NOT_PIM_1000	numeric
cm_data	PERF_MONITORING_TST	REA_NOT_PIM_50	numeric
cm_data	PERF_MONITORING_TST	REA_NOT_PIM_100	numeric
cm_data	PERF_MONITORING_TST	REA_NOT_PIM_200	numeric
cm_data	PERF_MONITORING_TST	REA_NOT_PIM_300	numeric
cm_data	PERF_MONITORING_TST	REA_NOT_PIM_400	numeric
cm_data	PERF_MONITORING_TST	REA_NOT_PIM_500	numeric
cm_data	PERF_MONITORING_TST	REA_NOT_PIM_1000	numeric
cm_data	PERF_MONITORING_TST	S_NOT_PIM	numeric
cm_data	PERF_MONITORING_TST	S_CB_NOT_PIM_50	numeric
cm_data	PERF_MONITORING_TST	S_CB_NOT_PIM_100	numeric

cm_data	PERF_MONITORING_TST	S_CB_NOT_PIM_200	numeric
cm_data	PERF_MONITORING_TST	S_CB_NOT_PIM_300	numeric
cm_data	PERF_MONITORING_TST	S_CB_NOT_PIM_400	numeric
cm_data	PERF_MONITORING_TST	S_CB_NOT_PIM_500	numeric
cm_data	PERF_MONITORING_TST	S_CB_NOT_PIM_1000	numeric
cm_data	PERF_MONITORING_TST	S_A_NOT_PIM_50	numeric
cm_data	PERF_MONITORING_TST	S_A_NOT_PIM_100	numeric
cm_data	PERF_MONITORING_TST	S_A_NOT_PIM_200	numeric
cm_data	PERF_MONITORING_TST	S_A_NOT_PIM_300	numeric
cm_data	PERF_MONITORING_TST	S_A_NOT_PIM_400	numeric
cm_data	PERF_MONITORING_TST	S_A_NOT_PIM_500	numeric
cm_data	PERF_MONITORING_TST	S_A_NOT_PIM_1000	numeric
cm_data	PERF_MONITORING_TST	S_EA_NOT_PIM_50	numeric
cm_data	PERF_MONITORING_TST	S_EA_NOT_PIM_100	numeric
cm_data	PERF_MONITORING_TST	S_EA_NOT_PIM_200	numeric
cm_data	PERF_MONITORING_TST	S_EA_NOT_PIM_300	numeric
cm_data	PERF_MONITORING_TST	S_EA_NOT_PIM_400	numeric
cm_data	PERF_MONITORING_TST	S_EA_NOT_PIM_500	numeric
cm_data	PERF_MONITORING_TST	S_EA_NOT_PIM_1000	numeric
cm_data	PERF_MONITORING_TST	S_REA_NOT_PIM_50	numeric
cm_data	PERF_MONITORING_TST	S_REA_NOT_PIM_100	numeric
cm_data	PERF_MONITORING_TST	S_REA_NOT_PIM_200	numeric
cm_data	PERF_MONITORING_TST	S_REA_NOT_PIM_300	numeric
cm_data	PERF_MONITORING_TST	S_REA_NOT_PIM_400	numeric
cm_data	PERF_MONITORING_TST	S_REA_NOT_PIM_500	numeric
cm_data	PERF_MONITORING_TST	S_REA_NOT_PIM_1000	numeric
cm_data	DBFIAR21	0LOGSYS	char
cm_data	DBFIAR21	0DOC_NUMBER	char
cm_data	DBFIAR21	C_BLOCKTY	char
cm_data	DBFIAR21	0CSM_CRTI	numeric
cm_data	DBFIAR21	0RECORDMODE	char
cm_data	DBFIAR21	0CSM_CLTI	numeric
cm_data	DBFIAR21	0DATEFROM	date
cm_data	DBFIAR21	0DATETO	date
cm_data	DBFIAR21	C_COMPPRS	char
cm_data	DBFIAR21	C_CRDACC	char
cm_data	DBFIAR21	0C_CTR_AREA	char
cm_data	DBFIAR21	C_CUSTPR	char
cm_data	DBFIAR21	0G_CWWE01	char
cm_data	DBFIAR21	0CRED_GROUP	char
cm_data	DBFIAR21	0REPR_GROUP	char
cm_data	DBFIAR21	K_COUNTER	decimal
cm_data	DBFIAR21	C_NBHOURS	numeric
cm_data	DBFIAR21	C_MTHFROM	numeric
cm_data	DBFIAR21	C_MTHTO	numeric
cm_data	DBFIAR21	C_CUSTID	char
cm_data	DBFIAR21	C_CUSTPRS	char

cm_data	DBFIAR21	C_PAYERID	char
cm_data	DBFIAR21	C_CRD_AC	char
cm_data	DBFIAR21	C_CST_CA2	char
cm_data	DBFIAR21	CDOC_TYPE	char
cm_data	DBFIAR21	C_TIMESTP	numeric
cm_data	GL_ACCOUNT	0CHRT_ACCTS	char
cm_data	GL_ACCOUNT	0GL_ACCOUNT	char
cm_data	GL_ACCOUNT	0BAL_FLAG	char
cm_data	GL_ACCOUNT	0INCST_FLAG	char
cm_data	GL_ACCOUNT	0LOGSYS	char
cm_data	GL_ACCOUNT	0SEM_POSIT	char
cm_data	GL_ACCOUNT	0SOURSYSTEM	char
cm_data	GL_ACCOUNT	C_GL_TYPE	char
cm_data	GL_ACCOUNT	C_GL_STYP	char
cm_data	GL_ACCOUNT	C_MGN_ACC	char
cm_data	GL_ACCOUNT	C_EXTRFLG	char
cm_data	GL_ACCOUNT	C_TEMPRES	char
cm_data	GL_ACCOUNT	C_STKACCT	char
cm_data	GL_ACCOUNT	C_INTRFLG	char
cm_data	GL_ACCOUNT	C_GLFAMIL	char
cm_data	C_CST_CA2	0C_CTR_AREA	char
cm_data	C_CST_CA2	C_CST_CA2	char
cm_data	C_CST_CA2	0CRED_GROUP	char
cm_data	C_CST_CA2	0CURRENCY	char
cm_data	C_CST_CA2	0CRED_LIMIT	decimal
cm_data	C_CST_CA2	0RC_LIM_CUR	char
cm_data	C_CST_CA2	0REC_CR_LM	decimal
cm_data	C_CST_CA2	0RISK_CATEG	char
cm_data	C_CST_CA2	C_GARAM	numeric
cm_data	C_CST_CA2	0CRED_ACCNT	char
cm_data	C_CST_CA2	0REPR_GROUP	char
cm_data	C_CST_CA2	0RATING	char
cm_data	C_CST_CA2	0PMNT_INDEX	char
cm_data	C_CST_CA2	0NXT_REVIEW	decimal
cm_data	C_CST_CA2	0LST_REVIEW	decimal
cm_data	C_CST_CA2	0LST_INT_RV	decimal
cm_data	C_CST_CA2	0LOGSYS	char
cm_data	C_CST_CA2	0CUST_GR_CM	char
cm_data	C_CST_CA2	C_CRDSTAT	char
cm_data	C_CST_CA2	C_RISKMAN	char
cm_data	C_CST_CA2	C_CRDACC	char
cm_data	C_CST_CA2	C_ACTDAT	date
cm_data	C_CST_CA2	C_ACTMONT	numeric
cm_data	C_CST_CA2	C_LRE_DAT	date
cm_data	C_CST_CA2	C_NRE_DAT	date
cm_data	C_CUSTID	0LOGSYS	char
cm_data	C_CUSTID	C_CUSTID	char

cm_data	C_CUSTID	C_CUSTPRS	char
cm_data	C_CUSTID	C_CUSTMPR	char
cm_data	C_CUSTID	C_CUSTPR	char
cm_data	C_CUSTID	0ACCNT_GRP	char
cm_data	C_CUSTID	0ADDR_NUMBR	char
cm_data	C_CUSTID	0AF_CUSTDC	char
cm_data	C_CUSTID	0AF_CUSTID	char
cm_data	C_CUSTID	0APO_LOCNO	char
cm_data	C_CUSTID	0BPARTNER	char
cm_data	C_CUSTID	0CITY_2	char
cm_data	C_CUSTID	0COUNTRY	char
cm_data	C_CUSTID	0CUST_CLASS	char
cm_data	C_CUSTID	0CUST_MKT	char
cm_data	C_CUSTID	0DBDUNS_NUM	numeric
cm_data	C_CUSTID	0FISCVARNT	char
cm_data	C_CUSTID	0ID_XCPD	char
cm_data	C_CUSTID	0INDUSTRY	char
cm_data	C_CUSTID	0NAME	varchar
cm_data	C_CUSTID	0TAX_NUMB	char
cm_data	C_CUSTID	0TAX_NUMB2	char
cm_data	C_CUSTID	0VISIT_RYT	char
cm_data	C_CUSTID	C_INT_GRP	char
cm_data	C_CUSTID	C_DUNSHQ	numeric
cm_data	C_CUSTID	C_DUNSPA	numeric
cm_data	C_CUSTID	C_DUNSDU	numeric
cm_data	C_CUSTID	C_DUNSGU	numeric
cm_data	C_CUSTID	0ACCNT_GRPV	char
cm_data	C_CUSTID	0DEL_INDIC	char
cm_data	C_CUSTID	C_STCEG	char
cm_data	C_CUSTID	C_STAT_SL	char
cm_data	C_CUSTID	0LANGU	char
cm_data	C_CUSTID	0TRANSPZONE	char
cm_data	C_CUSTID	C_LOCATIO	numeric
cm_data	C_CUSTID	C_ZONEFI	char
cm_data	C_CUSTID	C_FLGINT	char
cm_data	C_CUSTID	C_CSMETH	char
cm_data	C_CUSTID	C_ENTRP	numeric
cm_data	C_CUSTID	C_ZONE	char
cm_data	C_CUSTID	C_MZONE	char
cm_data	C_CUSTID	C_PCOMPAN	char
cm_data	C_CUSTID	0CLM_CLGR	char
cm_data	C_CUSTID	0CLM_CLSP	char
cm_data	C_CUSTID	C_GZONE	char
cm_data	TCURR	KURST	char
cm_data	TCURR	FCURR	char
cm_data	TCURR	TCURR	char
cm_data	TCURR	GDATU	date

cm_data	TCURR	UKURS	decimal
cm_data	TCURR	FFACT	decimal
cm_data	TCURR	TFACT	decimal
cm_data	DBFIAR20	0AC_DOC_NO	char
cm_data	DBFIAR20	0FISCPER	numeric
cm_data	DBFIAR20	0FISCVARNT	char
cm_data	DBFIAR20	0FI_DSBITEM	char
cm_data	DBFIAR20	0ITEM_NUM	numeric
cm_data	DBFIAR20	C_CUSTID	char
cm_data	DBFIAR20	C_COMPCDE	char
cm_data	DBFIAR20	0LOGSYS	char
cm_data	DBFIAR20	0DOC_CURRCY	char
cm_data	DBFIAR20	0LOC_CURRCY	char
cm_data	DBFIAR20	0CHRT_ACCTS	char
cm_data	DBFIAR20	0CLEAR_DATE	date
cm_data	DBFIAR20	0CLR_DOC_NO	char
cm_data	DBFIAR20	0COUNTRY	char
cm_data	DBFIAR20	0CREATEDON	date
cm_data	DBFIAR20	0DOC_DATE	date
cm_data	DBFIAR20	0FI_DOCSTAT	char
cm_data	DBFIAR20	0G_CWWE01	char
cm_data	DBFIAR20	0G_CWWE13	char
cm_data	DBFIAR20	0GL_ACCOUNT	char
cm_data	DBFIAR20	0LAST_DUNN	date
cm_data	DBFIAR20	0NETDUEDATE	date
cm_data	DBFIAR20	0POST_KEY	char
cm_data	DBFIAR20	0PSTNG_DATE	date
cm_data	DBFIAR20	C_CST_CA2	char
cm_data	DBFIAR20	C_CUSTPRS	char
cm_data	DBFIAR20	C_DOCTYP	char
cm_data	DBFIAR20	C_DUNN_BL	char
cm_data	DBFIAR20	C_PM_MTHD	char
cm_data	DBFIAR20	0C_CTR_AREA	char
cm_data	DBFIAR20	C_SALEMP	char
cm_data	DBFIAR20	0DEB_CRE_DC	decimal
cm_data	DBFIAR20	0DEB_CRE_LC	decimal
cm_data	DBFIAR20	C_FCONNUM	char
cm_data	DBFIAR20	C_TIMESTP	numeric
cm_data	DBFIAR20	C_LGSYSAF	char
cm_data	DBFIAR20	C_COMPCAF	char
cm_data	COUNTRY	0COUNTRY	char
cm_data	COUNTRY	C_ZONE	char
cm_data	COUNTRY	C_GZONE	char
cm_data	COUNTRY	C_MZONE	char
cm_data	COUNTRY	C_ZREACH	char
cm_data	COUNTRY	C_ZONEFI	char
cm_data	COUNTRY	C_PZONE	char

cm_data	COUNTRY	C_STDPTRM	numeric
cm_data	C_COMPCDE	0LOGSYS	char
cm_data	C_COMPCDE	C_COMPCDE	char
cm_data	C_COMPCDE	0CHRT_ACCTS	char
cm_data	C_COMPCDE	0COMPANY	char
cm_data	C_COMPCDE	0COUNTRY	char
cm_data	C_COMPCDE	0CURRENCY	char
cm_data	C_COMPCDE	0C_CTR_AREA	char
cm_data	C_COMPCDE	0FISCVARNT	char
cm_data	C_COMPCDE	0SOURSYSTEM	char
cm_data	C_COMPCDE	0OFYEAR	numeric
cm_data	C_COMPCDE	0OFPER	numeric
cm_data	C_COMPCDE	0OFPER3	numeric
cm_data	C_COMPCDE	0RETROPOST	char
cm_data	C_COMPCDE	C_ZONE	char
cm_data	C_COMPCDE	C_STAT_SL	char
cm_data	C_COMPCDE	C_CONTRIB	numeric
cm_data	C_COMPCDE	C_DEFAREA	char
cm_data	C_COMPCDE	C_DEFPCTR	char
cm_data	C_COMPCDE	C_FGPTOB	char
cm_data	C_COMPCDE	0CO_AREA	char
cm_data	C_COMPCDE	C_COMPPRS	char
cm_data	C_COMPCDE	C_LANDSCP	char
cm_data	C_COMPCDE	C_FLGINT	char
cm_data	C_COMPCDE	C_CSMETH	char
cm_data	C_COMPCDE	C_ENTRP	numeric
cm_data	C_COMPCDE	C_ZONEPUR	char
cm_data	C_COMPCDE	K_INTRAT	decimal
cm_data	C_COMPCDE	C_MNGAREA	char
cm_data	C_COMPCDE	C_MNGCTRY	char
cm_data	C_COMPCDE	C_PZONE	char
cm_data	C_COMPCDE	C_PSCOPE	char
cm_data	C_COMPCDE	0VALIDTO	date
cm_data	C_COMPCDE	0VALIDFROM	date
cm_data	C_COMPCDE	C_MERBK	char
cm_data	C_COMPCDE	C_PMERBK	char
cm_data	C_COMPCDE	C_BFCCOMP	char
cm_data	C_COMPCDE	C_MERGDAT	date
cm_data	G_CWWE01	0G_CWWE01	char
cm_data	G_CWWE01	0G_CWWE02	char
cm_data	G_CWWE01	C_PFCTR2	char
cm_data	G_CWWE01	C_MAGNITU	char
cm_data	G_CWWE01	C_PFCTR1	char
cm_data	G_CWWE01	CPFCTR1_2	char
cm_data	G_CWWE01	CPFCTR2_2	char
cm_data	G_CWWE01	C_PFCTR_3	char

Description of the performance monitoring table

PERF_MONITORING	evaluation day	evalution day of the performance
PERF_MONITORING	simulation_day	day of the prediction
PERF_MONITORING	C_CUSTID__C_ZONEFI	zone of the payer
PERF_MONITORING	precision_Paid_In_Month	precision score of Paid In Month (PIM) documents
PERF_MONITORING	recall_Paid_In_Month	recall score of Paid In Month documents
PERF_MONITORING	precision_Not_Paid_In_Month	precision score of Not Paid In Month documents
PERF_MONITORING	recall_Not_Paid_In_Month	recall score of Not Paid In Month documents
PERF_MONITORING	Expected not PIM amount	expected amount Not Paid In Month
PERF_MONITORING	Not PIM amount	amount Not Paid In Month
PERF_MONITORING	Expected not PIM amount ratio	ratio beween Expected not PIM amount and Not PIM amount
PERF_MONITORING	CB_NOT_PIM_50	Lorenz curve point with first 50 customers sorted by the true not PIM amount
PERF_MONITORING	CB_NOT_PIM_100	Lorenz curve point with first 100 customers sorted by the true not PIM amount
PERF_MONITORING	CB_NOT_PIM_200	Lorenz curve point with first 200 customers sorted by the true not PIM amount
PERF_MONITORING	CB_NOT_PIM_300	Lorenz curve point with first 300 customers sorted by the true not PIM amount
PERF_MONITORING	CB_NOT_PIM_400	Lorenz curve point with first 400 customers sorted by the true not PIM amount
PERF_MONITORING	CB_NOT_PIM_500	Lorenz curve point with first 500 customers sorted by the true not PIM amount
PERF_MONITORING	A_NOT_PIM_50	Lorenz curve point with first 50 customers sorted by the amount
PERF_MONITORING	A_NOT_PIM_100	Lorenz curve point with first 100 customers sorted by the amount
PERF_MONITORING	A_NOT_PIM_200	Lorenz curve point with first 200 customers sorted by the amount
PERF_MONITORING	A_NOT_PIM_300	Lorenz curve point with first 300 customers sorted by the amount
PERF_MONITORING	A_NOT_PIM_400	Lorenz curve point with first 400 customers sorted by the amount
PERF_MONITORING	A_NOT_PIM_500	Lorenz curve point with first 500 customers sorted by the amount
PERF_MONITORING	EA_NOT_PIM_50	Lorenz curve point with first 50 customers sorted by the Expected not PIM amount
PERF_MONITORING	EA_NOT_PIM_100	Lorenz curve point with first 100 customers sorted by the Expected not PIM amount
PERF_MONITORING	EA_NOT_PIM_200	Lorenz curve point with first 200 customers sorted by the Expected not PIM amount
PERF_MONITORING	EA_NOT_PIM_300	Lorenz curve point with first 300 customers sorted by the Expected not PIM amount
PERF_MONITORING	EA_NOT_PIM_400	Lorenz curve point with first 400 customers sorted by the Expected not PIM amount
PERF_MONITORING	EA_NOT_PIM_500	Lorenz curve point with first 500 customers sorted by the Expected not PIM amount
PERF_MONITORING	REA_NOT_PIM_50	Lorenz curve point with first 50 customers sorted by the Risk amount not PIM
PERF_MONITORING	REA_NOT_PIM_100	Lorenz curve point with first 100 customers sorted by the Risk amount not PIM
PERF_MONITORING	REA_NOT_PIM_200	Lorenz curve point with first 200 customers sorted by the Risk amount not PIM
PERF_MONITORING	REA_NOT_PIM_300	Lorenz curve point with first 300 customers sorted by the Risk amount not PIM
PERF_MONITORING	REA_NOT_PIM_400	Lorenz curve point with first 400 customers sorted by the Risk amount not PIM
PERF_MONITORING	REA_NOT_PIM_500	Lorenz curve point with first 500 customers sorted by the Risk amount not PIM
PERF_MONITORING	S_CB_NOT_PIM_50	Lorenz curve point with first 50 customers (under 1 million €) sorted by the true not PIM amount
PERF_MONITORING	S_CB_NOT_PIM_100	Lorenz curve point with first 100 customers (under 1 million €) sorted by the true not PIM amount

PERF_MONITORING	S_CB_NOT_PIM_200	Lorenz curve point with first 200 customers (under 1 million €) sorted by the true not PIM amount
PERF_MONITORING	S_CB_NOT_PIM_300	Lorenz curve point with first 300 customers (under 1 million €) sorted by the true not PIM amount
PERF_MONITORING	S_CB_NOT_PIM_400	Lorenz curve point with first 400 customers (under 1 million €) sorted by the true not PIM amount
PERF_MONITORING	S_CB_NOT_PIM_500	Lorenz curve point with first 500 customers (under 1 million €) sorted by the true not PIM amount
PERF_MONITORING	S_A_NOT_PIM_50	Lorenz curve point with first 50 customers (under 1 million €) sorted by the amount
PERF_MONITORING	S_A_NOT_PIM_100	Lorenz curve point with first 100 customers (under 1 million €) sorted by the amount
PERF_MONITORING	S_A_NOT_PIM_200	Lorenz curve point with first 200 customers (under 1 million €) sorted by the amount
PERF_MONITORING	S_A_NOT_PIM_300	Lorenz curve point with first 300 customers (under 1 million €) sorted by the amount
PERF_MONITORING	S_A_NOT_PIM_400	Lorenz curve point with first 400 customers (under 1 million €) sorted by the amount
PERF_MONITORING	S_A_NOT_PIM_500	Lorenz curve point with first 500 customers (under 1 million €) sorted by the amount
PERF_MONITORING	S_EA_NOT_PIM_50	Lorenz curve point with first 50 customers (under 1 million €) sorted by the Expected not PIM amount
PERF_MONITORING	S_EA_NOT_PIM_100	Lorenz curve point with first 100 customers (under 1 million €) sorted by the Expected not PIM amount
PERF_MONITORING	S_EA_NOT_PIM_200	Lorenz curve point with first 200 customers (under 1 million €) sorted by the Expected not PIM amount
PERF_MONITORING	S_EA_NOT_PIM_300	Lorenz curve point with first 300 customers (under 1 million €) sorted by the Expected not PIM amount
PERF_MONITORING	S_EA_NOT_PIM_400	Lorenz curve point with first 400 customers (under 1 million €) sorted by the Expected not PIM amount
PERF_MONITORING	S_EA_NOT_PIM_500	Lorenz curve point with first 500 customers (under 1 million €) sorted by the Expected not PIM amount
PERF_MONITORING	S_REA_NOT_PIM_50	Lorenz curve point with first 50 customers (under 1 million €) sorted by the Risk amount not PIM
PERF_MONITORING	S_REA_NOT_PIM_100	Lorenz curve point with first 100 customers (under 1 million €) sorted by the Risk amount not PIM
PERF_MONITORING	S_REA_NOT_PIM_200	Lorenz curve point with first 200 customers (under 1 million €) sorted by the Risk amount not PIM
PERF_MONITORING	S_REA_NOT_PIM_300	Lorenz curve point with first 300 customers (under 1 million €) sorted by the Risk amount not PIM
PERF_MONITORING	S_REA_NOT_PIM_400	Lorenz curve point with first 400 customers (under 1 million €) sorted by the Risk amount not PIM
PERF_MONITORING	S_REA_NOT_PIM_500	Lorenz curve point with first 500 customers (under 1 million €) sorted by the Risk amount not PIM