

Association rules Project (Fabio Costantino)

My role in the Company

As a Content Analyst in the Localization Team, I am responsible for migrating data in and out of our Content Management System called STEP. These are mostly text data - a.k.a. content, which are then published daily in our website for all customers to consume.

As a Localization team, we mainly deal with translated text in 12 different languages, therefore also managing the flow of translation files in and out of the company. In order to translate our content, I have daily contact with external translation agencies and manage the use of a Translation Management System called XTM, which helps organize all files and workflows in a single tool.

Besides, we also provide our manager(s) and the rest of the team with reports of many kind: from tailored collection of data coming from different sources, to reports on products (or families of products) performance.

Some of the tools I use on a daily/weekly basis to perform my job are: **STEP Content Management System**, **XTM Translation Management System**, **Microsoft SQL Server Management Studio**, **Report Builder** from **Adobe Analytics**, **DOMO** as a BI dashboard tool, **Office package**, **Jupyter Notebook** (with **Python**).

Brief

Define which tasks performed by the Translation and Content Teams directly impact the performance of the products and what activities should the Teams implement to optimize performance.

Business problem

Our Line Manager in the Localization Team has received a report for the French market, concerning an attempt to score the quality of the content created for a large section of our products. I am requested to analyze these data to try and extract some insight that might not be immediately visible.

Current process

Currently, scoring content is an extremely difficult task, since there is no objective measurement. Many different approaches are taken, such as analyze the performance of the product page as a whole, performing A/B tests with different contents, scoring content based on parameters given by SEO tools like Search Metrics, etc.

Current issue

None of the above mentioned solution is definitive, but in my opinion the analysis of objective information (e.g. presence/absence of a video in the page, presence/absence of technical explanation, etc.) using data mining techniques can help the team to take more informed decisions.

Proposed solution

My **main goal** will be to establish the tasks that need to be prioritized because they appear to **directly affect the performance of the products**.

The followings are the file provided to me to start this task.

- 'quality_FR.csv' is the report for the French market that I am required to analyse.
- 'data_matrix' is the matrix that explains each section of the report to which a score was given (e.g. "primary_Image", "Additional_Images", etc.).

I will proceed as follows:

- I will import the information and clean the data.
- I will include data on the conversion rate for the same products (i.e. the percentage of visitors who brought a particular product out of all those who visited the page).
- I will apply the Association Rule algorithm to mine meaningful relations among variables.

Assumption: for the purpose of this analysis, I will assume that a conversion rate higher than the average for the given dataset constitutes an indication of good performance.

Pros and Cons

This process is easily repeatable and can be tailored over time adjusting parameters to include more or fewer relations among variables. Whilst flexibility is certainly a plus, we should never forget that the task of scoring a

0. Importing libraries

First of all I need to import all relevant Python libraries.

```
In [1]: # Used to Load and manipulate dataframes
import pandas as pd
import numpy as np

# Used to display screenshots in Jupyter notebook
from IPython.display import Image

# Used to create a sparse matrix from the dataset
from mlxtend.preprocessing import TransactionEncoder

# Used to find frequent itemsets with the apriori algorithm
from mlxtend.frequent_patterns import apriori

# Used to derive association rules from frequent itemsets
from mlxtend.frequent_patterns import association_rules
```

1. Sourcing the data

1.1 Importing the necessary files

First, I need to import the dataset originally given to me to analyse. I will do so, using the `read_csv()` function from the pandas library.

```
In [2]: # Importing the original report for France and visualizing the first 5 rows
report_FR = pd.read_csv('quality_FR.csv', encoding='utf-8')

report_FR.head()
```

Out[2]:

	ReportDate	Market	SN	Brand	Book	BookID	SuperSection	SuperSectionID
0	01-02-19	FR	1243127	Siemens	Electrical, Automation & Cables	PSB_346948	Automation & Control Gear	PSSS_421167
1	01-02-19	FR	1243140	Siemens	Electrical, Automation & Cables	PSB_346948	Automation & Control Gear	PSSS_421167
2	01-02-19	FR	1243142	Siemens	Electrical, Automation & Cables	PSB_346948	Automation & Control Gear	PSSS_421167
3	01-02-19	FR	1243145	Siemens	Electrical, Automation & Cables	PSB_346948	Automation & Control Gear	PSSS_421167
4	01-02-19	FR	145207	Merlin Gerin	Electrical, Automation & Cables	PSB_346948	Automation & Control Gear	PSSS_421167

5 rows × 26 columns

In order to understand the meaning of the different scores in the columns of this dataset, I need the matrix with the explanations, and I will show it here using the `Image()` function to read screenshots in Jupyter Notebook.

In [3]: *# Importing a screenshot of the matrix file*


Out[3]:






CONTENT ELEMENT	STATUS	0 POINT CONTENT	1 POINT CONTENT	2 POINT CONTENT	3 POINT CONTENT
Product Description - English	Active	Measurement • 40 character description on web	Measurement • 40 character description on web - but family rule completed	Measurement • No criteria for 2 points	Measurement • 100 characted description on web
Product Description - local market	Active	Measurement • 40 character description on web	Measurement • 40 character description on web - but family rule completed	Measurement • 100 character description but translation links not in place	Measurement • 100 character description; translation links in place
Primary Image	Active	Measurement • No image in STEP/No image on website	Measurement • Primary image present, image call number starts with any letter apart from F or R or • Image is not specific (regardless of prefix)	Measurement • Specific Image, R prefix	Measurement • Specific Image, F prefix
Additional Images	Active			Measurement • Secondary image(s) (colour photograph), non-F prefix	Measurement • Secondary image(s) (colour phorograph), F prefix
Datasheet - English	Active	Measurement • No datasheet logged against the SKU	Measurement • No criteria for 1 point	Measurement • SKU has local-language datasheet	Measurement • SKU has local-language datasheet which has been updated within the last 24 months
Datasheet - local market	Active	Measurement • No datasheet logged against the SKU	Measurement • SKU has non local-language datasheet	Measurement • SKU has local-language datasheet	Measurement • SKU has local-language datasheet which has been updated within the last 24 months
Overview/ Authored Content	Active	Measurement • No authored content	Measurement • Range content only present	Measurement • Product has sub-range content	Measurement • Product has line-level specific content
Translation	Active	Measurement • Neither product nor subrange have translation links in place (i.e. not translated)	Measurement • Either product or sub-range has translation link; status of link immaterial	Measurement • Both the product and the sub-range have translation links	Measurement • Both product and sub-range translation links in place - both links are up-to-date
Attribute Values	Active	Measurement • 0%	Measurement • 1% to 59% fill-rate	Measurement • 60% to 89% fill-rate	Measurement • 90%+ fill-rate
Video Link	Active	Measurement	Measurement	Measurement	Measurement • Product has a video linked

Since I am requested to judge the performance of the products in the report, I decided to find more data using our **Adobe Analytics** platform. I am using the latest **conversion rate** as a KPI to understand how appealing is the web page to our customers for each product.

In [4]: *# Importing the dataset created after mining for more data in Adobe Analytics*
france_data = pd.read_excel('data_FR.xlsx')

Showing the first 5 rows
france_data.head()

Out[4]:

	ID	Conversion_rate_FR
0	1243127	
1	1243140	
2	1243142	
3	1243145	
4	145207	

2. Exploring and transforming the data

Now that I have all necessary file at my disposal, I will need to prepare them for the model. I decided to divide this section into 4 sub-sections that will deal step by step with the preparation as follows:

- Trimming the dataset
- Merging the data in one dataset
- Exploring and transforming the new dataset
- Preparing the dataset for the model

2.1 Trimming the dataset

The original report has many columns that I will not be using for this analysis, therefore I decided to create a simpler version of the dataset leaving only the product ID and all columns that have a numerical score.

In [5]: *# First of all, I will obtain a list of all columns and what they contain*
`report_FR.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 188 entries, 0 to 187
Data columns (total 26 columns):
ReportDate                188 non-null object
Market                    188 non-null object
SN                         188 non-null int64
Brand                     188 non-null object
Book                      188 non-null object
BookID                    188 non-null object
SuperSection              188 non-null object
SuperSectionID            188 non-null object
Section                   188 non-null object
SectionID                 188 non-null object
Family                    188 non-null object
FamilyID                  188 non-null object
Descr                     188 non-null object
SCORES_CONTENT            188 non-null int64
Product_Description       188 non-null int64
Translation                188 non-null int64
Primary_Image             188 non-null int64
Additional_Images         188 non-null int64
Datasheets                188 non-null int64
Overview_Authored_Content 188 non-null int64
Attribute_Values          188 non-null int64
Video_Link                188 non-null int64
Category                  187 non-null object
Cell                      187 non-null object
Technology                187 non-null object
Global_Buyer_PM           187 non-null object
dtypes: int64(10), object(16)
memory usage: 38.3+ KB
```

Conveniently, all the information I need is stored in the dataset as a number (`int64`), therefore I can filter everything by the type of data to obtain quickly a trimmed dataset. Looking for a solution, I have found the `select_dtypes()` function, that filters the original dataset to end up only with those columns that have `int64` as data type.

```
In [6]: # Applying the filtering function to the original dataset
trimmed_data_FR = report_FR.select_dtypes(include=[np.int64])

# Checking that the function has worked properly, I can see that now the dataset only contains
# the information I will need from here forward
trimmed_data_FR.head()
```

Out[6]:

	SN	SCORES_CONTENT	Product_Description	Translation	Primary_Image	Additional_Ima
0	1243127	12	3	3	2	
1	1243140	12	3	3	2	
2	1243142	12	3	3	2	
3	1243145	12	3	3	2	
4	145207	14	3	3	2	

2.2 Merging the data in one dataset

To perform my analysis, I want the conversion rate to be included into my `trimmed_data_FR` dataset.

However, before I can successfully merge the two dataframes, I need to rename the column that will be pivotal for the function to be used properly. Since our products have a unique stock number, this is clearly the best choice, but in the two datasets they are labelled differently.

```
In [7]: # The first code applies the rename function to the dataframe
trimmed_data_FR.rename(columns={'SN': 'ID'}, inplace=True)

#The second shows the result being successful
trimmed_data_FR.head()
```

C:\Users\E0658269\AppData\Local\Continuum\anaconda3\lib\site-packages\pandas\core\frame.py:4025: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: <http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy>
return super(DataFrame, self).rename(**kwargs)

Out[7]:

	ID	SCORES_CONTENT	Product_Description	Translation	Primary_Image	Additional_Ima
0	1243127	12	3	3	2	
1	1243140	12	3	3	2	
2	1243142	12	3	3	2	
3	1243145	12	3	3	2	
4	145207	14	3	3	2	

Now I can proceed to merge the two dataset on the column **ID**. To do so, I will use the `merge()` function.

```
In [8]: # Applying the merge function to the two datasets
final_dataset_FR = pd.merge(left=trimmed_data_FR, right=france_data, on='ID',
                             how='left')

# Checking that all data in the new dataset are correctly implemented
final_dataset_FR.head()
```

Out[8]:

	ID	SCORES_CONTENT	Product_Description	Translation	Primary_Image	Additional_Ima
0	1243127	12	3	3	2	
1	1243140	12	3	3	2	
2	1243142	12	3	3	2	
3	1243145	12	3	3	2	
4	145207	14	3	3	2	

2.3 Exploring and transforming the new dataset

In this step, I will do all necessary preparations for the data to be fit into the model.

```
In [9]: # In preparation for the modelling, I will also double-check again that no value is missing
final_dataset_FR.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 188 entries, 0 to 187
Data columns (total 11 columns):
ID                                188 non-null int64
SCORES_CONTENT                    188 non-null int64
Product_Description                188 non-null int64
Translation                       188 non-null int64
Primary_Image                     188 non-null int64
Additional_Images                 188 non-null int64
Datasheets                       188 non-null int64
Overview_Authored_Content         188 non-null int64
Attribute_Values                  188 non-null int64
Video_Link                       188 non-null int64
Conversion_rate_FR                188 non-null float64
dtypes: float64(1), int64(10)
memory usage: 17.6 KB
```


At this point, I only need to have the columns that will be used in the model. A quick look at the data I am left with makes me understand that the column *SCORES_CONTENT* is useless for our final steps, since it simply represent the sum of all other numerical columns.

I will therefore proceed eliminating it using the `drop()` function.

```
In [10]: # Dropping the SCORES_CONTENT column
model_dataset_FR = final_dataset_FR.drop(final_dataset_FR.columns[1], axis=1)

# Inspecting the results with the head() function
model_dataset_FR.head()
```

Out[10]:

	ID	Product_Description	Translation	Primary_Image	Additional_Images	Datasheets	Over
0	1243127		3	3	2	0	0
1	1243140		3	3	2	0	0
2	1243142		3	3	2	0	0
3	1243145		3	3	2	0	0
4	145207		3	3	2	0	3

The column `Conversion_rate_FR` also need my attention, since it must be transformed in a binary list of values for the model to work properly. I decided to look at the **mean** of the data and simply assign to each row the value of *higher* or *lower* depending on where they fall with respect to the mean itself. This was already anticipated in the **assumption** contained in the paragraph **Proposed solution**.

```
In [11]: # First of all, I calculate the mean of the data included in the column
mean = model_dataset_FR['Conversion_rate_FR'].mean()

# Now, using a lambda function, the whole column is checked and a different value is assigned according
# to whether the value x is higher or lower than the mean
model_dataset_FR['Conversion_mean'] = model_dataset_FR['Conversion_rate_FR'].apply(lambda x: 'higher' if x > mean else 'lower')

# Finally, I drop the original column from the dataset, in order not to duplicate data
model_dataset_FR.drop(columns='Conversion_rate_FR', inplace=True)
model_dataset_FR.head()
```

Out[11]:

	ID	Product_Description	Translation	Primary_Image	Additional_Images	Datasheets	Over
0	1243127		3	3	2	0	0
1	1243140		3	3	2	0	0
2	1243142		3	3	2	0	0
3	1243145		3	3	2	0	0
4	145207		3	3	2	0	3

All other columns in the dataset cannot be used to create a sparse matrix (i.e. a matrix containing only 1's and 0's for the purpose of classifying each row) as they are, therefore after considering and trying out few different methods, I decided that using the function `stack()` on the data will be the best option. This function allows me to associate all different columns to one **ID**.

```
In [12]: # First of all, I need to set 'ID' as index of the dataframe
model_dataset_FR = model_dataset_FR.set_index('ID')

# Checking that the transformation has worked
model_dataset_FR.head()
```

Out[12]:

	ID	Product_Description	Translation	Primary_Image	Additional_Images	Datasheets	Overvi
	1243127		3	3	2	0	0
	1243140		3	3	2	0	0
	1243142		3	3	2	0	0
	1243145		3	3	2	0	0
	145207		3	3	2	0	3

```
In [13]: # Now I apply the 'stack' function. At the same time, I reset the index so that
# 'ID' is again a column in the dataframe, that I can use later
model_dataset_FR = pd.DataFrame(model_dataset_FR.stack()).reset_index()

# Visualizing the results
model_dataset_FR.head()
```

Out[13]:

	ID	level_1	0
0	1243127	Product_Description	3
1	1243127	Translation	3
2	1243127	Primary_Image	2
3	1243127	Additional_Images	0
4	1243127	Datasheets	0

At the moment, the information we need is divided in 2 columns:

- Column **level_1** contains the type of score associated with each ID
- Column **0** contains the actual score associated with each ID

Since we need these information together, the two columns should simply be concatenated.

```
In [14]: # The data in column '0' are not strings, therefore I need to add 'astype(str)' before concatenating
# them to the data in column 'level_1'
model_dataset_FR['scored_element'] = model_dataset_FR.iloc[:,1]+"_"+model_dataset_FR.iloc[:,2].astype(str)
```

```
In [15]: # Before visualizing the dataset again, the columns that are not needed any longer are dropped
model_dataset_FR = model_dataset_FR.drop(model_dataset_FR.columns[[1,2]], axis=1)
model_dataset_FR.head()
```

Out[15]:

	ID	scored_element
0	1243127	Product_Description_3
1	1243127	Translation_3
2	1243127	Primary_Image_2
3	1243127	Additional_Images_0
4	1243127	Datasheets_0

2.4 Preparing the dataset for the model

To use the `apriori()` algorithm on `model_dataset_FR`, I need to have all IDs corresponding to a *basket* of scores.

```
In [16]: # First I need to group all scores by the unique ID that is in the column 'ID'
model_dataset_FR = model_dataset_FR.groupby(['ID'])

# Then, only the score column is left
model_dataset_FR = model_dataset_FR['scored_element']

# An array is created with the score column and the index are reset to include
# 'ID' in the dataset again
model_dataset_FR = model_dataset_FR.apply(np.array).reset_index()
model_dataset_FR.head()
```

Out[16]:

	ID	scored_element
0	131040	[Product_Description_3, Translation_1, Primary...
1	131056	[Product_Description_3, Translation_3, Primary...
2	144282	[Product_Description_3, Translation_3, Primary...
3	144298	[Product_Description_3, Translation_3, Primary...
4	144305	[Product_Description_3, Translation_3, Primary...

The final transformation I need is to create a sparse matrix from this dataset. I will do so using the `TransactionEncoder()` function that I imported from `mlxtend` in **0. Importing libraries**.

```
In [17]: # The first code is used to setup the Transaction Encoder
encoder = TransactionEncoder()

# The second code transforms the dataset into a sparse matrix
encoder_array = encoder.fit(model_dataset_FR['scored_element']).transform(model_dataset_FR['scored_element'])

# Finally, I visualize the matrix
encoder_array
```

```
Out[17]: array([[ True, False, False, ..., False,  True, False],
 [ True, False, False, ...,  True,  True, False],
 [ True, False, False, ...,  True,  True, False],
 ...,
 [ True, False, False, ...,  True,  True, False],
 [ True, False, False, ...,  True,  True, False],
 [ True, False, False, ...,  True,  True, False]])
```

```
In [18]: # In this last step, I transform the matrix in a dataframe and visualize it
dataset_modelling_FR = pd.DataFrame(encoder_array, columns = encoder.columns_)

dataset_modelling_FR.head()
```

Out[18]:

	Additional_Images_0	Additional_Images_2	Additional_Images_3	Attribute_Values_1	Attribute_1
0	True	False	False	True	
1	True	False	False	True	
2	True	False	False	True	
3	False	False	True	True	
4	True	False	False	True	

5 rows × 24 columns

Now all columns are in a binary form and the model can be applied.

3. Applying the model

In order to mine information from the dataset just prepared, I will use the `apriori()` algorithm to extract the most frequent combinations of the different scores. The following codes have been tried multiple times to adjust the `min_support` and `min_threshold` values. This flexibility of the model has been already noted in the paragraph **Pros and Cons**.

The values that appear here allowed me to mine a few meaningful rules from an initial large number of combinations. When I tried to have less combinations found by the algorithm, the information resulting from my analysis appeared to have no meaning.

I believe that, given the nature of the dataset (i.e. not a set of purchased item), this phase was a bit more problematic and benefited from a trial-and-error approach.

```
In [19]: # This code will apply the apriori model to find the frequent combinations
frequent_combinations = apriori(dataset_modelling_FR, min_support=0.1, use_col
names=True)

# Inspecting the results with the tail() function gives me an idea of
# how many combinations were found
frequent_combinations.tail()
```

Out[19]:

	support	itemsets
1030	0.388298	(Datasheets_0, Overview_Authored_Content_2, Co...
1031	0.287234	(Attribute_Values_1, Datasheets_0, Overview_Au...
1032	0.101064	(Datasheets_0, Overview_Authored_Content_2, Co...
1033	0.287234	(Attribute_Values_1, Datasheets_0, Overview_Au...
1034	0.101064	(Datasheets_0, Overview_Authored_Content_2, Co...

```
In [20]: # Finally, here the actual association rules are found  
rules = association_rules(frequent_combinations, metric='confidence', min_threshold=0.5)  
  
# The results are visualized sorted by the lift in descending order  
rules.sort_values(by='lift', ascending=False)
```

Out[20]:

	antecedents	consequents	antecedent support	consequent support	supl
2095	(Additional_Images_3)	(Attribute_Values_1, Product_Description_3, Pr...	0.106383	0.148936	0.106
6531	(Attribute_Values_1, Primary_Image_3)	(Additional_Images_3, Product_Description_3, O...	0.148936	0.106383	0.106
2094	(Attribute_Values_1, Primary_Image_3)	(Additional_Images_3, Product_Description_3)	0.148936	0.106383	0.106
361	(Attribute_Values_1, Primary_Image_3)	(Additional_Images_3)	0.148936	0.106383	0.106
362	(Additional_Images_3)	(Attribute_Values_1, Primary_Image_3)	0.106383	0.148936	0.106
6529	(Additional_Images_3, Product_Description_3)	(Attribute_Values_1, Overview_Authored_Content...	0.106383	0.148936	0.106
6528	(Additional_Images_3, Overview_Authored_Conten...	(Attribute_Values_1, Product_Description_3, Pr...	0.106383	0.148936	0.106
6525	(Attribute_Values_1, Product_Description_3, Pr...	(Additional_Images_3, Overview_Authored_Conten...	0.148936	0.106383	0.106
2078	(Additional_Images_3)	(Overview_Authored_Content_2, Primary_Image_3,...	0.106383	0.148936	0.106
2090	(Additional_Images_3, Product_Description_3)	(Attribute_Values_1, Primary_Image_3)	0.106383	0.148936	0.106
6521	(Additional_Images_3, Product_Description_3, O...	(Attribute_Values_1, Primary_Image_3)	0.106383	0.148936	0.106
6524	(Attribute_Values_1, Overview_Authored_Content...	(Additional_Images_3, Product_Description_3)	0.148936	0.106383	0.106
2089	(Attribute_Values_1, Product_Description_3, Pr...	(Additional_Images_3)	0.148936	0.106383	0.106
2075	(Attribute_Values_1, Primary_Image_3)	(Overview_Authored_Content_2, Additional_Image...	0.148936	0.106383	0.106
6534	(Additional_Images_3)	(Attribute_Values_1, Product_Description_3, Ov...	0.106383	0.148936	0.106
2073	(Overview_Authored_Content_2, Additional_Image...	(Attribute_Values_1, Primary_Image_3)	0.106383	0.148936	0.106
6517	(Attribute_Values_1, Product_Description_3, Ov...	(Additional_Images_3)	0.148936	0.106383	0.106
2069	(Overview_Authored_Content_2, Primary_Image_3,...	(Additional_Images_3)	0.148936	0.106383	0.106
2104	(Product_Description_3, Primary_Image_3)	(Additional_Images_3, Overview_Authored_Conten...	0.154255	0.106383	0.106
2093	(Product_Description_3, Primary_Image_3)	(Additional_Images_3, Attribute_Values_1)	0.154255	0.106383	0.106
6535	(Primary_Image_3)	(Additional_Images_3, Product_Description_3, O...	0.154255	0.106383	0.106
2096	(Primary_Image_3)	(Additional_Images_3, Product_Description_3, A...	0.154255	0.106383	0.106
2100	(Overview_Authored_Content_2, Product_Descript...	(Additional_Images_3)	0.154255	0.106383	0.106

	antecedents	consequents	antecedent support	consequent support	supl
363	(Primary_Image_3)	(Additional_Images_3, Attribute_Values_1)	0.154255	0.106383	0.106
6533	(Product_Description_3, Primary_Image_3)	(Additional_Images_3, Overview_Authored_Content_2)	0.154255	0.106383	0.106
6532	(Overview_Authored_Content_2, Primary_Image_3)	(Additional_Images_3, Product_Description_3, A...	0.154255	0.106383	0.106
369	(Overview_Authored_Content_2, Primary_Image_3)	(Additional_Images_3)	0.154255	0.106383	0.106
371	(Primary_Image_3)	(Additional_Images_3, Overview_Authored_Content_2)	0.154255	0.106383	0.106
6526	(Overview_Authored_Content_2, Product_Descript...	(Additional_Images_3, Attribute_Values_1)	0.154255	0.106383	0.106
24	(Primary_Image_3)	(Additional_Images_3)	0.154255	0.106383	0.106
...
3	(Conversion_mean_higher)	(Additional_Images_0)	0.218085	0.872340	0.127
3629	(Attribute_Values_1, Product_Description_3, Co...	(Overview_Authored_Content_2, Additional_Image...	0.212766	0.824468	0.117
942	(Attribute_Values_1, Conversion_mean_higher)	(Overview_Authored_Content_2, Additional_Image...	0.212766	0.824468	0.117
2534	(Attribute_Values_1, Datasheets_3)	(Overview_Authored_Content_2, Video_Link_0)	0.265957	0.781915	0.138
7453	(Attribute_Values_1, Product_Description_3, Da...	(Overview_Authored_Content_2, Video_Link_0)	0.265957	0.781915	0.138
5348	(Conversion_mean_higher)	(Overview_Authored_Content_2, Translation_3, P...	0.218085	0.771277	0.111
1479	(Conversion_mean_higher)	(Translation_3, Product_Description_3, Additio...	0.218085	0.771277	0.111
7452	(Attribute_Values_1, Overview_Authored_Content...	(Product_Description_3, Video_Link_0)	0.250000	0.835106	0.138
116	(Attribute_Values_1, Conversion_mean_higher)	(Additional_Images_0)	0.212766	0.872340	0.122
944	(Attribute_Values_1, Product_Description_3, Co...	(Additional_Images_0)	0.212766	0.872340	0.122
3683	(Attribute_Values_1, Product_Description_3, Co...	(Video_Link_0, Additional_Images_0)	0.212766	0.760638	0.106
960	(Attribute_Values_1, Conversion_mean_higher)	(Video_Link_0, Additional_Images_0)	0.212766	0.760638	0.106
6335	(Additional_Images_0, Datasheets_3)	(Overview_Authored_Content_2, Product_Descript...	0.223404	0.765957	0.111
7448	(Attribute_Values_1, Product_Description_3, Ov...	(Video_Link_0)	0.250000	0.851064	0.138
2530	(Overview_Authored_Content_2, Datasheets_3, At...	(Video_Link_0)	0.250000	0.851064	0.138
3501	(Overview_Authored_Content_2, Datasheets_3)	(Product_Description_3, Video_Link_0)	0.255319	0.835106	0.138

	antecedents	consequents	antecedent support	consequent support	supl
3676	(Attribute_Values_1, Conversion_mean_higher)	(Translation_3, Product_Description_3, Additio...	0.212766	0.771277	0.106
8895	(Attribute_Values_1, Conversion_mean_higher)	(Overview_Authored_Content_2, Translation_3, P...	0.212766	0.771277	0.106
1461	(Conversion_mean_higher)	(Overview_Authored_Content_2, Translation_3, A...	0.218085	0.797872	0.111
198	(Conversion_mean_higher)	(Translation_3, Additional_Images_0)	0.218085	0.797872	0.111
5346	(Product_Description_3, Conversion_mean_higher)	(Overview_Authored_Content_2, Translation_3, A...	0.218085	0.797872	0.111
1477	(Product_Description_3, Conversion_mean_higher)	(Translation_3, Additional_Images_0)	0.218085	0.797872	0.111
6332	(Product_Description_3, Additional_Images_0, D...	(Overview_Authored_Content_2, Video_Link_0)	0.223404	0.781915	0.111
1884	(Additional_Images_0, Datasheets_3)	(Overview_Authored_Content_2, Video_Link_0)	0.223404	0.781915	0.111
3498	(Overview_Authored_Content_2, Product_Descript...	(Video_Link_0)	0.255319	0.851064	0.138
832	(Overview_Authored_Content_2, Datasheets_3)	(Video_Link_0)	0.255319	0.851064	0.138
3649	(Attribute_Values_1, Conversion_mean_higher)	(Overview_Authored_Content_2, Translation_3, A...	0.212766	0.797872	0.106
954	(Attribute_Values_1, Conversion_mean_higher)	(Translation_3, Additional_Images_0)	0.212766	0.797872	0.106
3671	(Attribute_Values_1, Product_Description_3, Co...	(Translation_3, Additional_Images_0)	0.212766	0.797872	0.106
8887	(Attribute_Values_1, Product_Description_3, Co...	(Overview_Authored_Content_2, Translation_3, A...	0.212766	0.797872	0.106

24656 rows × 9 columns

The sheer number of rules (a staggering 24655!) found might look useless at first sight, however in the evaluation phase I plan to shed some light on my findings.

4. Evaluating the results

Strictly speaking, most results coming out of this evaluation are not useful. I am not interested in knowing that e.g. a *Primary_Image* with a score of 3 will lead to a *Additional_Image* with a score of 3 because there is no actionable meaning coming out of this obvious correlation.

However, if I filter the resulting association rules by those that have a conversion rate higher than the **mean** (as per my **assumption**), then I will discover information directly applicable to my brief.

I create a new subset, filtering the rules only for those **consequents** (right part of the rules) that have a conversion rate higher than the mean. Also, I sort these values by the lift (measure of the importance of the rule).

```
In [21]: # Filtering by 'consequents' and sorting by 'Lift'
results = rules[rules.loc[:, "consequents"] == {'Conversion_mean_higher'}].sort_values(by='lift', ascending=False)
```

In order to visualize the antecedent (left part of the rule) completely and not in the truncated form, I will use the `set_option()` function that allows me to visualize the complete text.

```
In [22]: # Setting up the correct visualization
pd.set_option('display.max_colwidth', -1)

# Finally, I visualize the complete list
results
```

Out[22]:

	antecedents	consequents	antecedent support	consequent support	support
386	(Attribute_Values_1, Primary_Image_3)	(Conversion_mean_higher)	0.148936	0.218085	0.101064
2116	(Overview_Authored_Content_2, Primary_Image_3, Attribute_Values_1)	(Conversion_mean_higher)	0.148936	0.218085	0.101064
2145	(Attribute_Values_1, Product_Description_3, Primary_Image_3)	(Conversion_mean_higher)	0.148936	0.218085	0.101064
6539	(Attribute_Values_1, Product_Description_3, Overview_Authored_Content_2, Primary_Image_3)	(Conversion_mean_higher)	0.148936	0.218085	0.101064
54	(Primary_Image_3)	(Conversion_mean_higher)	0.154255	0.218085	0.101064
630	(Overview_Authored_Content_2, Primary_Image_3)	(Conversion_mean_higher)	0.154255	0.218085	0.101064
644	(Product_Description_3, Primary_Image_3)	(Conversion_mean_higher)	0.154255	0.218085	0.101064
3011	(Overview_Authored_Content_2, Product_Description_3, Primary_Image_3)	(Conversion_mean_higher)	0.154255	0.218085	0.101064

Filtering the results this way, we are down from 24655 to 8 rules. And all of them have a very high **lift**, which means that definitely the chance of having a conversion rate higher than the average is strongly connected with the antecedents.

```
In [23]: # In order to see it even more clearly, I can create a new array that only has
the antecedents
results_final = results['antecedents']

results_final
```

```
Out[23]: 386      (Attribute_Values_1, Primary_Image_3)
2116     (Overview_Authored_Content_2, Primary_Image_3, Attribute_Values_1)
2145     (Attribute_Values_1, Product_Description_3, Primary_Image_3)
6539     (Attribute_Values_1, Product_Description_3, Overview_Authored_Content
_2, Primary_Image_3)
54       (Primary_Image_3)
630      (Overview_Authored_Content_2, Primary_Image_3)
644      (Product_Description_3, Primary_Image_3)
3011     (Overview_Authored_Content_2, Product_Description_3, Primary_Image_3)
Name: antecedents, dtype: object
```

5. Answer to the brief

Now let us have again a look at the score matrix and see what conclusions we might draw for the French market from this analysis.

```
In [24]: # Showing again the matrix here for ease of consultation.
Image('data_matrix.PNG')
```

Out[24]:

CONTENT ELEMENT	STATUS	0 POINT CONTENT	1 POINT CONTENT	2 POINT CONTENT	3 POINT CONTENT
Product Description - English	Active	Measurement • 40 character description on web	Measurement • 40 character description on web - but family rule completed	Measurement • No criteria for 2 points	Measurement • 100 character description on web
Product Description - local market	Active	Measurement • 40 character description on web	Measurement • 40 character description on web - but family rule completed	Measurement • 100 character description but translation links not in place	Measurement • 100 character description; translation links in place
Primary Image	Active	Measurement • No image in STEP/No image on website	Measurement • Primary image present, image call number starts with any letter apart from F or R or • Image is not specific (regardless of prefix)	Measurement • Specific image, R prefix	Measurement • Specific image, F prefix
Additional Images	Active			Measurement • Secondary image(s) (colour photograph), non-F prefix	Measurement • Secondary image(s) (colour photograph), F prefix
Datasheet - English	Active	Measurement • No datasheet logged against the SKU	Measurement • No criteria for 1 point	Measurement • SKU has local-language datasheet	Measurement • SKU has local-language datasheet which has been updated within the last 24 months
Datasheet - local market	Active	Measurement • No datasheet logged against the SKU	Measurement • SKU has non local-language datasheet	Measurement • SKU has local-language datasheet	Measurement • SKU has local-language datasheet which has been updated within the last 24 months
Overview/ Authored Content	Active	Measurement • No authored content	Measurement • Range content only present	Measurement • Product has sub-range content	Measurement • Product has line-level specific content
Translation	Active	Measurement • Neither product nor sub-range have translation links in place (i.e. not translated)	Measurement • Either product or sub-range has translation link; status of link immaterial	Measurement • Both the product and the sub-range have translation links	Measurement • Both product and sub-range translation links in place - both links are up-to-date
Attribute Values	Active	Measurement • 0%	Measurement • 1% to 59% fill-rate	Measurement • 60% to 89% fill-rate	Measurement • 90%+ fill-rate
Video Link	Active	Measurement	Measurement	Measurement	Measurement • Product has a video linked

From the observation of the results, we can answer our brief as follows.

- The presence of a high resolution **Primary Image** is absolutely **fundamental** to achieve good results (this data is present in 100% of our rule set). I would suggest to the Content team to develop an automatic alert in the system when a product is launched without this standard in place.
- In 50% of the results, a product description **fully translated** and with the correct technical **translation link** in place helps to achieve a higher-than-average conversion rate. Naturally, translation is a key activity of the Translation Team, and the data show that it is directly driving a better performance, at least in the French market. This might seem not so useful a conclusion, but in many occasions translations is only perceived as a cost and not as an asset. Beside, I would suggest the Content Team to liase with the technical SEO team - a process can be put in place to check that the translation links are correctly implemented in the code of the website, especially for high-value products.
- In 50% of our results, a thorough **technical explanation** (called internally *authored content* or *overview*) at sub-range level appears to drive performance. However, it seems that the creation of this content is only important at the level of a family of products and not at the level of the single products. It appears that too much information is not necessary. I would suggest to the Content Team not to include many product level explanations and to the Translation Team to devote a limited budget to these texts.
- The data seem to suggest that **Attribute Values** can (and probably *should*) stay below 60% fill-rate. This could be linked to too much information being distractive for the customer. I would therefore suggest the Content Team to keep only attributes that are strictly necessary when launching a new product.

This concludes my analysis on the original report and the results will be present to my line manager.