Data Engineering Applied to Retail

Final Defense
"Diplôme d'Ingénieur de l'EPITA", 2022 promotion
Presented by: Tristan BILOT

Apprenticeship supervised by:
Olivier MORILLOT (Carrefour)
Juliette CHANSARD (EPITA)

30/08/2022

Plan

1. Introduction
    - The company
    - My role
2. Missions
    - Training optimization of a Machine Learning model
    - Development of a system for user management
    - Containerization and deployment of a Machine Learning project
3. Acquired knowledge
4. Benefits for the company
5. Conclusion

Plan

1. Introduction
   - The company
   - My role

2. Missions
   - Training optimization of a Machine Learning model
   - Development of a system for user management
   - Containerization and deployment of a Machine Learning project

3. Acquired knowledge

4. Benefits for the company

5. Conclusion

Outline

1. Introduction
   - The company
   - My role

The company

Carrefour

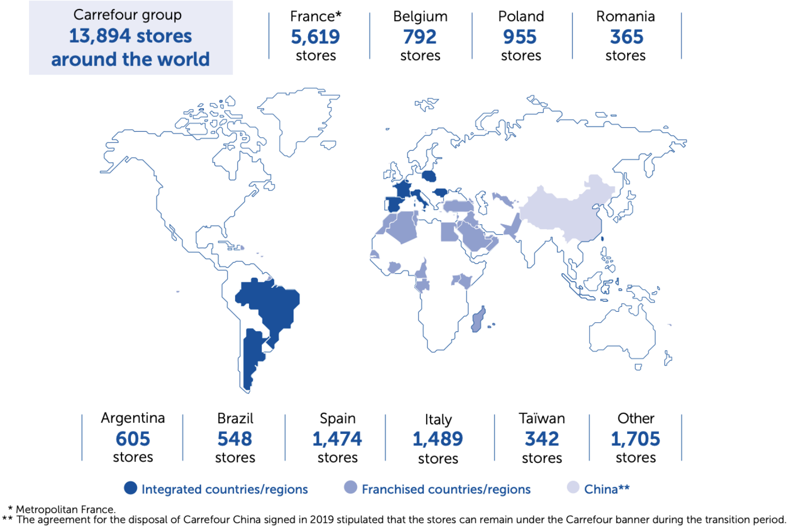
- One of the retail leaders in Europe and in the whole world
- Headquarters based in Massy (91), France

Numbers

- Created in : **1959**
- Present in : **30** countries
- Sales revenues in 2020 : **78.6 billion €**
- Employees in 2022 : **320.000**

Figure – Carrefour logo

The company



Carrefour group
13,894 stores around the world

France* **5,619** stores

Belgium **792** stores

Poland **955** stores

Romania **365** stores

Argentina **605** stores

Brazil **548** stores

Spain **1,474** stores

Italy **1,489** stores

Taïwan **342** stores

Other **1,705** stores

● Integrated countries/regions ● Franchised countries/regions ● China**

* Metropolitan France.
** The agreement for the disposal of Carrefour China signed in 2019 stipulated that the stores can remain under the Carrefour banner during the transition period.

Figure – Carrefour international coverage

The company

Carrefour & data

- In 2019, Carrefour is making digital and e-Commerce its priority
- The company released 2.8 billion € of investment over 5 years
- A partnership between Carrefour and Google was born in June 2018
- The **Carrefour-Google Data Lab** was created, where Data Scientists, Data Engineers and Data Analysts work jointly to leverage the data of Carrefour to make predictions with **Machine Learning** methods

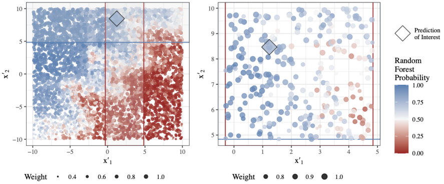


Figure – Data analysis illustration.

The company

Leveraging data to improve revenues

- Fortunately, Carrefour possesses a huge amount of data which can be used in predictive tasks
- Transaction histories or client behaviors can be leveraged to train models to make predictions on future similar events
- Such models could help decision making and increase sales and revenues

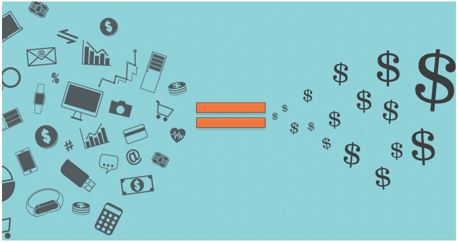


Figure – Data=money illustration. Source : Modern Dealership

Outline

1. Introduction
   - The company
   - My role

My role

Data Engineer

- The Data Engineer is responsible for the design and the development of systems for collecting, storing and analyzing data
- He transforms the raw data so that they are exploitable by the Data Scientists
- He also improves the reliability of the code, optimizes the complexity of models and deploy them in production

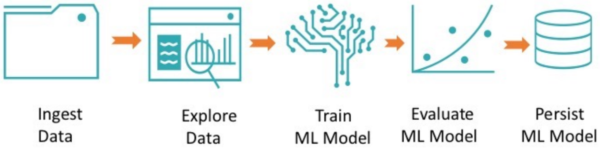


Figure – Data Engineering pipeline example

Plan

1. Introduction
   - The company
   - My role
2. Missions
   - Training optimization of a Machine Learning model
   - Development of a system for user management
   - Containerization and deployment of a Machine Learning project
3. Acquired knowledge
4. Benefits for the company
5. Conclusion

Outline

2 Missions

- Training optimization of a Machine Learning model
- Development of a system for user management
- Containerization and deployment of a Machine Learning project

Training optimization of a Machine Learning model

Team and context

Pricing Team

- **Goal :** to compute the price elasticity of product sold in Carrefour franchise stores
- **In brief :** to find, using a Machine Learning algorithm, the price of a product that will bring in the most revenue for the store
- **How :** the predictions done by our algorithms are given to store managers so that then can use them to guide their previsions

Constraints

- The model requires a huge amount of data to make accurate price predictions (235Go)
- These data are mostly the transaction history of every products in every stores and **need to be fetch from the cloud** to train the model locally
- The initial time taken by the algorithm to fetch the data was **very slow**, a new way to fetch these data had to be found

Mission

- To build an algorithm to fetch the data from the cloud via Internet
- This algorithm should :
  - **Be fast**
  - **Handle huge tables of data**

Training optimization of a Machine Learning model

Stack

Technical stack

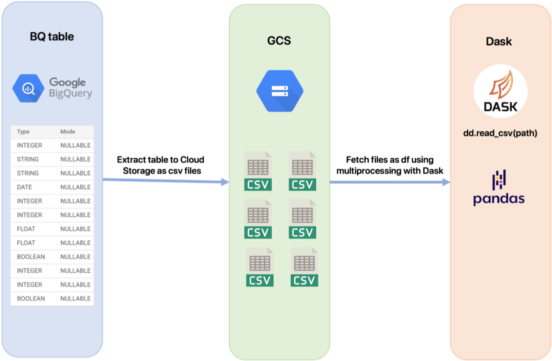
- **Cloud :** GCP
- **Language :** Python 3
- **Machine :** 30 vCPUs, 120Go RAM memory

Model training

- **Initial training time :** 15h
- **Execution recurrence :** monthly

Training optimization of a Machine Learning model

Initial implementation



BQ table

Type	Mode
INTEGER	NULLABLE
STRING	NULLABLE
STRING	NULLABLE
DATE	NULLABLE
INTEGER	NULLABLE
INTEGER	NULLABLE
FLOAT	NULLABLE
FLOAT	NULLABLE
BOOLEAN	NULLABLE
INTEGER	NULLABLE
INTEGER	NULLABLE
BOOLEAN	NULLABLE

Extract table to Cloud Storage as csv files

GCS

Fetch files as df using multiprocessing with Dask

Dask

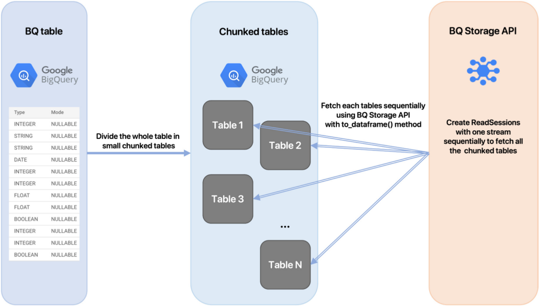
dd.read_csv(path)

Figure – Initial architecture used for data fetching

Originally, the SQL table was converted in hundreds of csv files and stored on Google Cloud Storage. Then, these files are fetched and loaded in memory from a Python library : dask.

Training optimization of a Machine Learning model

Solution



BQ table

Google BigQuery

Type	Mode
INTEGER	NULLABLE
STRING	NULLABLE
STRING	NULLABLE
DATE	NULLABLE
INTEGER	NULLABLE
INTEGER	NULLABLE
FLOAT	NULLABLE
FLOAT	NULLABLE
BOOLEAN	NULLABLE
INTEGER	NULLABLE
INTEGER	NULLABLE
BOOLEAN	NULLABLE

Divide the whole table in small chunked tables

Chunked tables

Google BigQuery

Table 1

Table 2

Table 3

...

Table N

Fetch each tables sequentially using BQ Storage API with to_dataframe() method

BQ Storage API

Create ReadSessions with one stream sequentially to fetch all the chunked tables

Figure – New architecture used for data fetching

In this approach, we divide the whole table in chunks, which are groups of lines extracted based on multiple indexes. To avoid bottlenecks and limitations imposed by GCP, new tables are created via SQL for each chunk and are then fetched sequentially

Training optimization of a Machine Learning model

Results

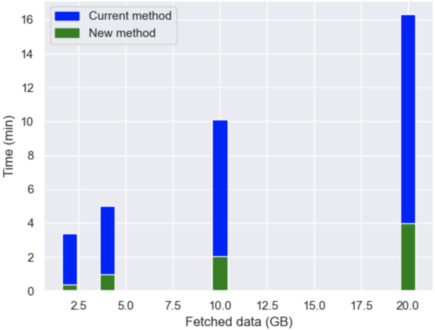


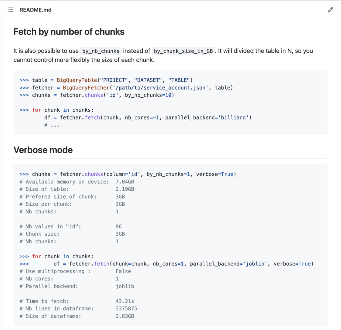
Figure – Results comparison between the initial fetching method and our new solution

Mission fulfillment

- Be fast : **4x** to **10x faster**
- Handle huge tables of data : tables from **arbitrary size** could be used
- The whole training took **initially 15h** to complete and **now** takes only **5h** (/3 !)

Training optimization of a Machine Learning model

Results



README.md

Fetch by number of chunks

It is also possible to use `by_nb_chunks` instead of `by_chunk_size_in_GB`. It will divided the table in N, so you cannot control more flexibly the size of each chunk.

```
>>> table = BigQueryTable("PROJECT", "DATASET", "TABLE")
>>> fetcher = BigQueryFetcher('/path/to/service_account.json', table)
>>> chunks = fetcher.chunks('id', by_nb_chunks=10)

>>> for chunk in chunks:
        df = fetcher.fetch(chunk, nb_cores=-1, parallel_backend='billiard')
        # ...
```

Verbose mode

```
>>> chunks = fetcher.chunks(column='id', by_nb_chunks=1, verbose=True)
# Available memory on device:  7.04GB
# Size of table:               2.19GB
# Prefered size of chunk:      3GB
# Size per chunk:              3GB
# Nb chunks:                   1

# Nb values in "id":           96
# Chunk size:                  3GB
# Nb chunks:                   1

>>> for chunk in chunks:
>>>         df = fetcher.fetch(chunk=chunk, nb_cores=1, parallel_backend='joblib', verbose=True)
# Use multiprocessing :        False
# Nb cores:                    1
# Parallel backend:            joblib

# Time to fetch:               43.21s
# Nb lines in dataframe:       3375875
# Size of dataframe:           2.83GB
```

Figure – Fast fetching module shared with other teams at Carrefour

Fast fetching Python module

A Python module for BigQuery table fast fetching has been developed and shared to other teams so that they can use it to improve their fetching time too

Training optimization of a Machine Learning model

Failures

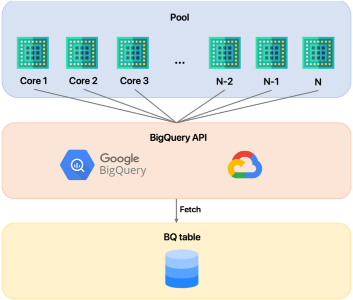


Figure – Architecture of fetching using multiprocessing

Could we be even faster with multiprocessing ?

- Further improvements based on multiprocessing have been tried to be applied to the fetching technique
- However, GCP applies many constraints regarding the requests and we could not achieve using multiprocessing properly in this case

Outline

2 Missions

- Training optimization of a Machine Learning model
- Development of a system for user management
- Containerization and deployment of a Machine Learning project

Development of a system for user management

Team and context

Platform Team

- **Goal :** to develop a global platform to deploy projects based on data pipelines
- **In brief :** migrate every ML projects on a centralized platform
- **How :** by using the Airflow pipeline orchestrator managed by Google (Cloud Composer)

Development of a system for user management

Mission

Constraints

- The Airflow software is used as a web server hosting the platform directly in Google Cloud
- By default, each user visiting the platform has access to every projects, which is not secure and scalable for multiple projects

Mission

- To implement a system responsible of the management of the projects and their users
- This system should :
  - **Isolate every projects only to their members**
  - **Admin accounts should access every projects**

Development of a system for user management

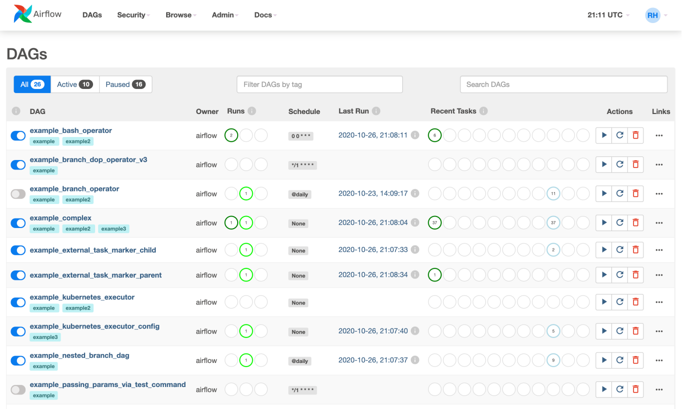
Stack

Technical stack

- **Cloud :** GCP
- **Languages :** Python 3, Bash
- **Software :** Airflow

Development of a system for user management

Stack



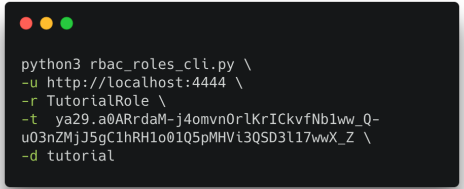
DAGs

	DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links
●	example_bash_operator example example2	airflow	2 ○ ○	0 0 * * *	2020-10-26, 21:08:11	6 ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ ↻ 🗑	...
●	example_branch_dop_operator_v3 example	airflow	○ ○ ○	*/1 * * * *		○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ ↻ 🗑	...
○	example_branch_operator example example2	airflow	○ 1 ○	@daily	2020-10-23, 14:09:17	○ ○ ○ ○ ○ ○ ○ ○ 11 ○ ○	▶ ↻ 🗑	...
●	example_complex example example2 example3	airflow	1 1 ○	None	2020-10-26, 21:08:04	37 ○ ○ ○ ○ ○ ○ ○ 37 ○ ○	▶ ↻ 🗑	...
●	example_external_task_marker_child	airflow	○ 1 ○	None	2020-10-26, 21:07:33	○ ○ ○ ○ ○ ○ ○ ○ 2 ○ ○	▶ ↻ 🗑	...
●	example_external_task_marker_parent	airflow	○ 1 ○	None	2020-10-26, 21:08:34	1 ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ ↻ 🗑	...
●	example_kubernetes_executor example example2	airflow	○ ○ ○	None		○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ ↻ 🗑	...
●	example_kubernetes_executor_config example3	airflow	○ 1 ○	None	2020-10-26, 21:07:40	○ ○ ○ ○ ○ ○ ○ ○ 5 ○ ○	▶ ↻ 🗑	...
●	example_nested_branch_dag example	airflow	○ 1 ○	@daily	2020-10-26, 21:07:37	○ ○ ○ ○ ○ ○ ○ ○ 9 ○ ○	▶ ↻ 🗑	...
○	example_passing_params_via_test_command example	airflow	○ ○ ○	*/1 * * * *		○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ ↻ 🗑	...

Figure – Airflow pipeline orchestrator

Development of a system for user management

Solution



```
python3 rbac_roles_cli.py \
-u http://localhost:4444 \
-r TutorialRole \
-t  ya29.a0ARrdaM-j4omvnOrlKrICkvfNb1ww_Q-
uO3nZMjJ5gC1hRH1o01Q5pMHVi3QSD3l17wwX_Z \
-d tutorial
```

Figure – Python program responsible of user management

- Leverages Google Groups to fetch users from every projects in a synchronized way
- Makes a difference between users from Google Groups and users from Airflow and update them consequently

Mission fulfillment

- Isolate every projects only to their members : **done**
- Admin accounts should access every projects : **done**

Development of a system for user management

Failures

Could we trigger our program optimally ?

- There is currently no way to handle user creation/deletion from Google Groups
- Thus, it is needed to run the program periodically in order to maintain the list of users updated

Outline

2 Missions

- Training optimization of a Machine Learning model
- Development of a system for user management
- Containerization and deployment of a Machine Learning project

Containerization and deployment of a Machine Learning project

Team and context

Carrefour Promo Optimizer Team

- **Goal :** to evaluate the performance of future/past promotions
- **In brief :** provide to the analysts an algorithm which can make predictions on the revenues made by new promotions
- **How :** by using Machine Learning models trained on past transactions and promotions

Containerization and deployment of a Machine Learning project

Constraints

- This is a **very old project**, created roughly 5 years ago
- The code was **not versioned** (no Git used), the libraries was **obsolete**, the code was **legacy** and **coupled** to the virtual machine (VM) running it

Mission

- To make the project maintainable and deployable on other machines by isolating its components and using best coding practices
- The project should :
  - **Run on whatever VM using Docker containers**
  - **Use updated libraries and formatted code**
  - **Be well structured so that future modifications can be done easily**

Containerization and deployment of a Machine Learning project

Stack

Technical stack

- **Cloud :** GCP
- **Languages :** Python 2 & 3, Bash, SQL
- **Software :** Airflow, Jenkins, Docker, Kubernetes, Git

Model training

- **Initial training time :** 1h
- **Execution recurrence :** weekly, on Saturday night

Containerization and deployment of a Machine Learning project

Solution

Harmonization of Python versions and dependencies

- The project initially used two different versions of Python with two sets of libraries
- Every libraries have been updated using only one upgraded version of Python
- This step broke numerous parts of the code so refactoring was inevitable

Figure – Data Science libraries used in CPO

Containerization and deployment of a Machine Learning project

Solution

Containerization

- The project was initially dependent of the machine running it, so deleting the machine would also delete the whole project
- Using Docker containers make possible to run the project on different VMs or Kubernetes clusters deployed inside Carrefour
- The project was divided in 3 distinct services which are : Analytics, Predictive API and DAGs (Directed Acyclic Graphs)

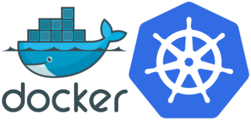


Figure – Docker and Kubernetes

Containerization and deployment of a Machine Learning project

Solution

Pre-processing and training optimization

- Data pre-processing and training was really slow
- Prallelization algorithms (multiprocessing) have been used to improve the compute time, leveraging the 16 cores available on the VM
- This optimization ended up by dividing the whole training time by 2, from 1h to 30min

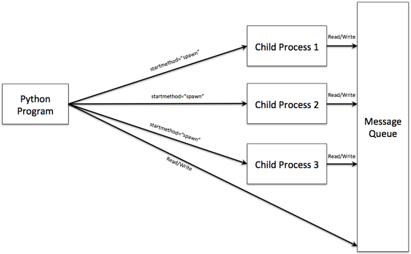


Figure – Multiprocessing parallelization

Additional work

- Improvement of AES encryption security using CBC algorithm with HMAC hashing
- Implementation of a granular logging system based on Cloud
- Migration of training and API on Kubernetes
- Writing of Python unit tests
- Improvement of model execution reliability
- Automatic code formatting
- Continuous Delivery

Mission fulfillment

- Run on whatever VM using Docker containers : **done**
- Use updated libraries and formatted code : **done**
- Be well structured so that future modifications can be done easily : **done**

Containerization and deployment of a Machine Learning project

Failures

Could we make a full refactoring of the model ?

- The model (pre-processing + training) contains thousands of complex lines of code, written since years
- It would take months with multiple Data Scientists/Engineers to fully refactor the code
- This refactoring will be necessary in the future, when complex features will be implemented

Plan

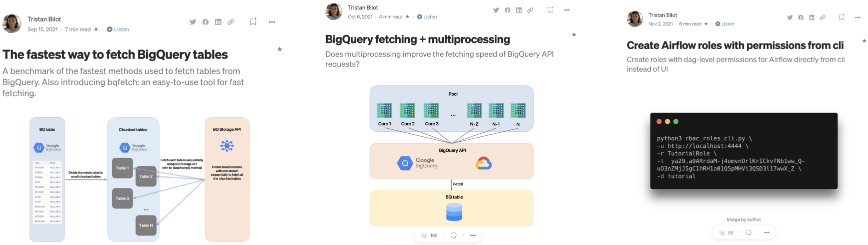
1. Introduction
   - The company
   - My role
2. Missions
   - Training optimization of a Machine Learning model
   - Development of a system for user management
   - Containerization and deployment of a Machine Learning project
3. Acquired knowledge
4. Benefits for the company
5. Conclusion

Acquired knowledge

The pluridisciplinarity of the missions gave me various knowledge :

- **Programming :** Python, Bash, SQL, software architecture, scientific programming
- **Machine Learning :** linear regression, random forest, sklearn, scipy
- **Cloud :** BigQuery, Compute Engine, Cloud Composer, Cloud Storage, Cloud Logging, IAM
- **Devops :** Jenkins, Airflow, Docker, Kubernetes

Published articles



Tristan Bilot
Sep 15, 2021 · 7 min read ★ · Listen

The fastest way to fetch BigQuery tables

A benchmark of the fastest methods used to fetch tables from BigQuery. Also introducing bqfetch: an easy-to-use tool for fast fetching.

Tristan Bilot
Oct 6, 2021 · 4 min read ★ · Listen

BigQuery fetching + multiprocessing

Does multiprocessing improve the fetching speed of BigQuery API requests?

Tristan Bilot
Nov 2, 2021 · 6 min read ★ · Listen

Create Airflow roles with permissions from cli

Create roles with dag-level permissions for Airflow directly from cli instead of UI

```
python3 rbac_roles_cli.py \
-u http://localhost:4444 \
-r TutorialRole \
-t  ya29.a0ARrdaM-j4omvnOrlKrICkvfNb1ww_Q-
uO3nZMjJ5gC1hRH1o01Q5pMHVi3QSD3l17wwX_Z \
-d tutorial
```

Image by author

Figure – Medium articles related to topics seen during my missions

Articles based on related topics and issues have been written in order to help the Data Engineering community

Plan

1. Introduction
   - The company
   - My role
2. Missions
   - Training optimization of a Machine Learning model
   - Development of a system for user management
   - Containerization and deployment of a Machine Learning project
3. Acquired knowledge
4. **Benefits for the company**
5. Conclusion

Benefits for the company

Benefits of my work for Carrefour :

- Pricing model 3 times faster to compute and cheaper to execute
- A fast fetching module which can be used by other teams at Carrefour (presented during 2 *Engineering Reviews* (technical meetings talking about Data Science))
- Make multi-projects possible on the shared pipeline platform
- Carrefour Promo Optimizer (CPO) project fully refactored and updated for production
- CPO model 2 times faster to compute and cheaper to execute

Plan

1. Introduction
   - The company
   - My role
2. Missions
   - Training optimization of a Machine Learning model
   - Development of a system for user management
   - Containerization and deployment of a Machine Learning project
3. Acquired knowledge
4. Benefits for the company
5. Conclusion

Conclusion

- Working as a Data Engineer was the perfect mix of Computer Science learned at EPITA and Data Science learned at Carrefour
- This apprenticeship allowed me to better targeting what my future work will be
- I acquired huge technical, scientific and business knowledge
- A perfect "first step" in the Data Science domain and a good starting point for a future PhD

End

Thank you for your attention

Do you have any questions ?