
INSTITUT DES
ACTUAIRES

Data Innovation Lab

Milliman

Predictive Analytics

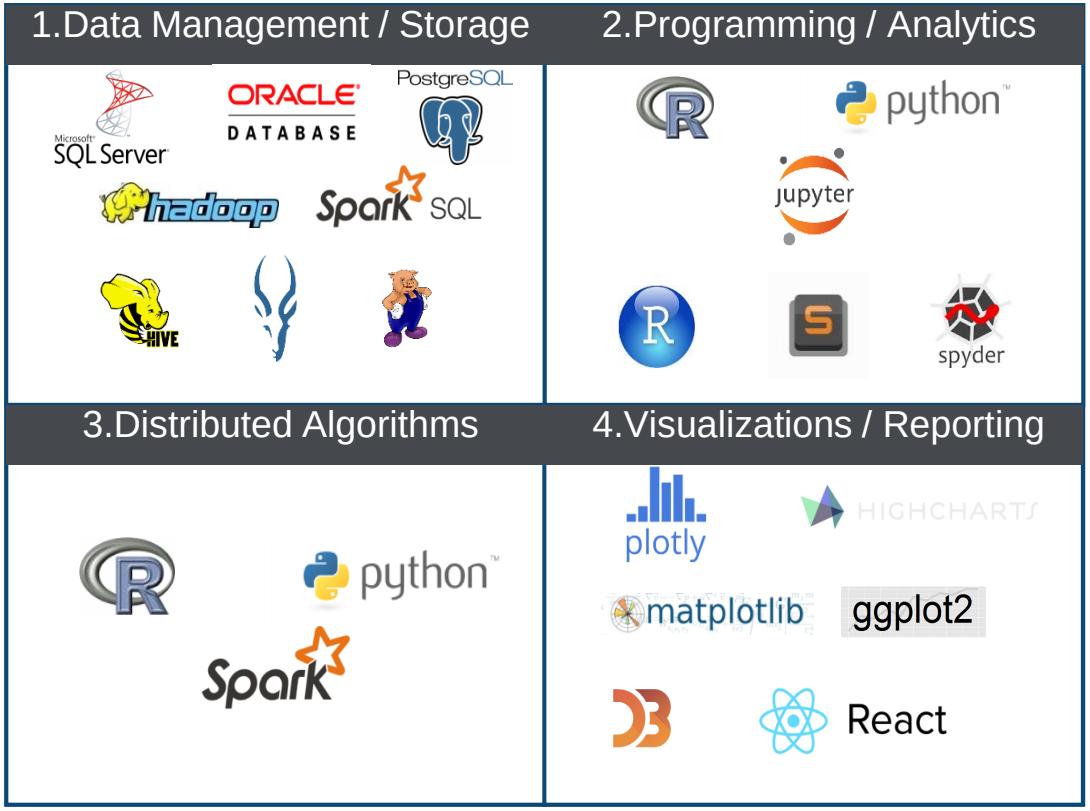
Which technologies? How to use them? Which applications to insurance?

Eric.O Lebigot, Yves-Richard Hong T.H, Rémi Bellina, Antoine Ly

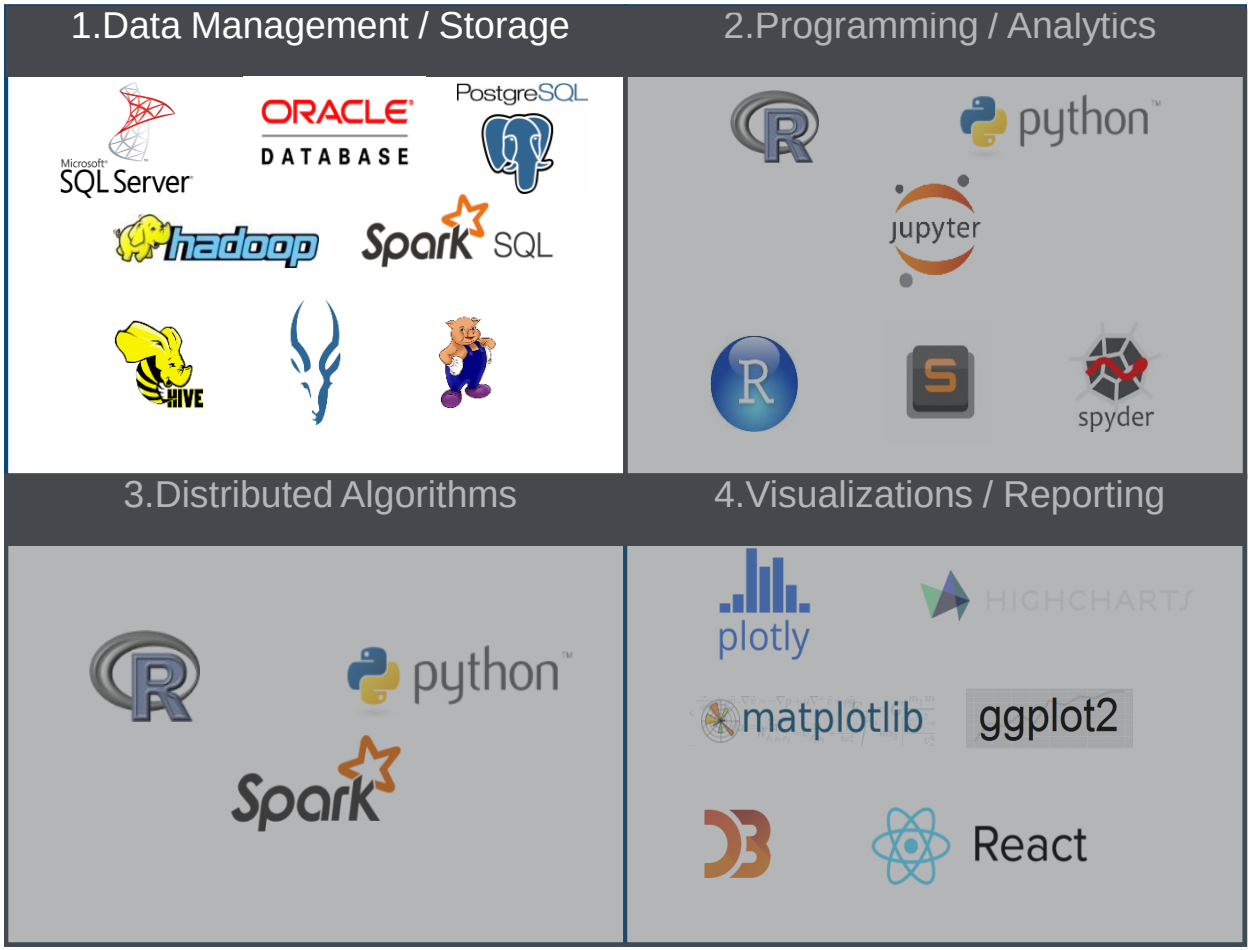
November, 8th 2016

Introduction

- Why those technologies ?
- Which one to use? When?



1. Data Management / Storage



1. Data Management / Storage

RDBMS vs distributed storage Hadoop

Query management	System management	Storage
SQL manager	System tables(meta data)	Data tables

RDBMS

- Everything integrated in one system
- Deals with relational databases

Technologies to set-up a RDBMS

Microsoft SQL Server · ORACLE DATABASE · PostgreSQL

How to manage your RDBM?

SQL (SQLite, Dbvisualizer etc.)

Query management	System management	Storage
Several devices to manage queries (SQL, noSQL)	Meta data on several devices	HDFS (redundant)

Hadoop

- Everything is distributed on several systems (quicker multi-accesses)
- Deals with different type of databases (structured or unstructured)
- Redundance, scalability, streaming

Technologies to set-up Hadoop

hadoop

How to manage Hadoop?

SQL like (Hive, Impala, Pig) or use API (Python, R)

1. Data Management / Storage

Data Lake

Storage technologies made for Big-Data volumetry and **fail-safe using redundant storage**

Hadoop HDFS:

Limitless size storage thanks to a distributed storage, and fail-safe thanks to a redundant storage

Parquet:

Compressed file format working with Hadoop HDFS, data is analyzed and optimally compressed to optimize performance.

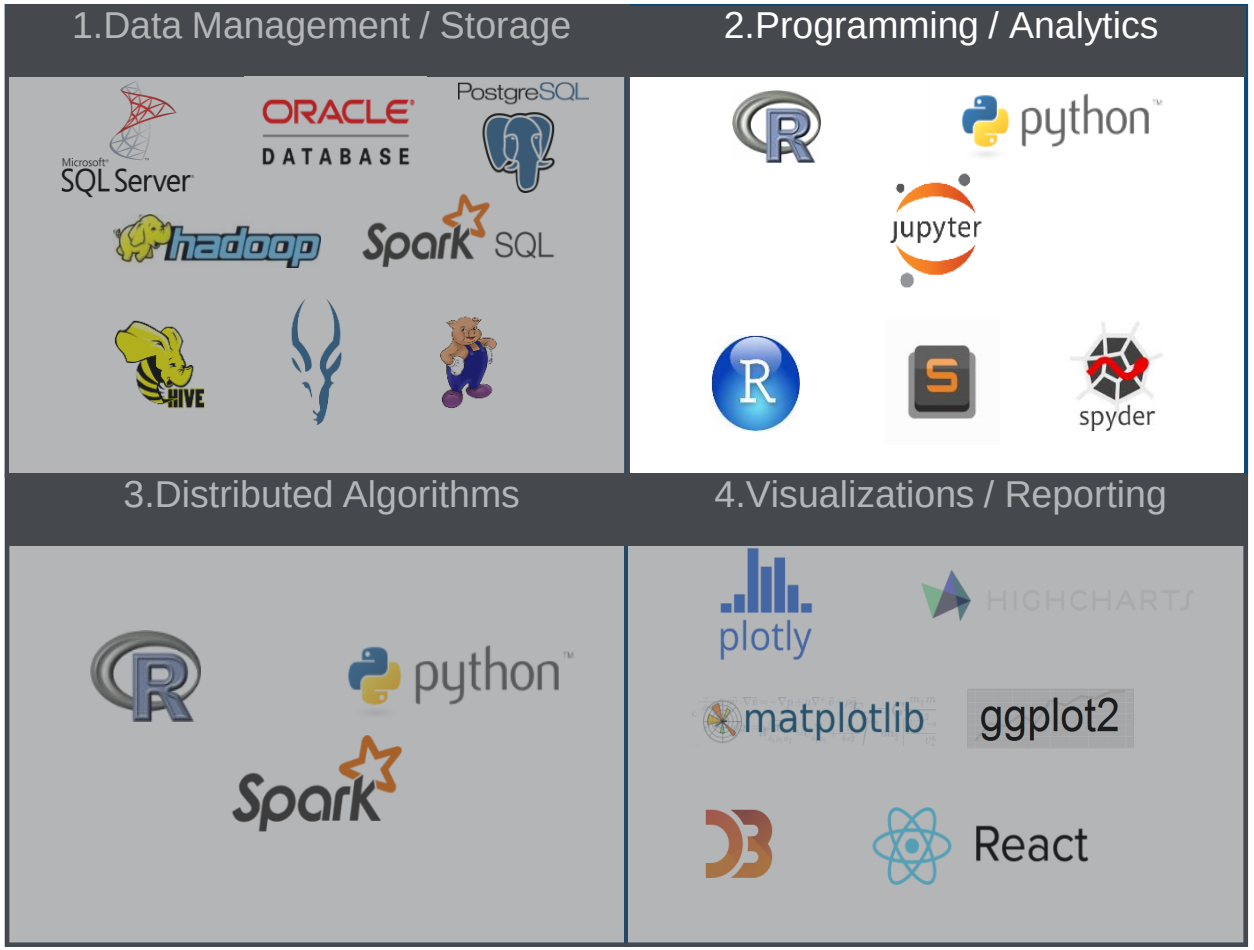
Data-sources

Data is imported in the Hadoop HDFS **Data-Lake**

Data-Lake

- All data-sources are merged into one single data storage system
- Unified storage support
- Prepare the data to be reshaped and enriched

2. Programming / Analytics



2. Programming / Analytics

R, Python

- What is R/Python compared to more traditional tools (SAS, SPSS etc.)? How to chose between these technologies?

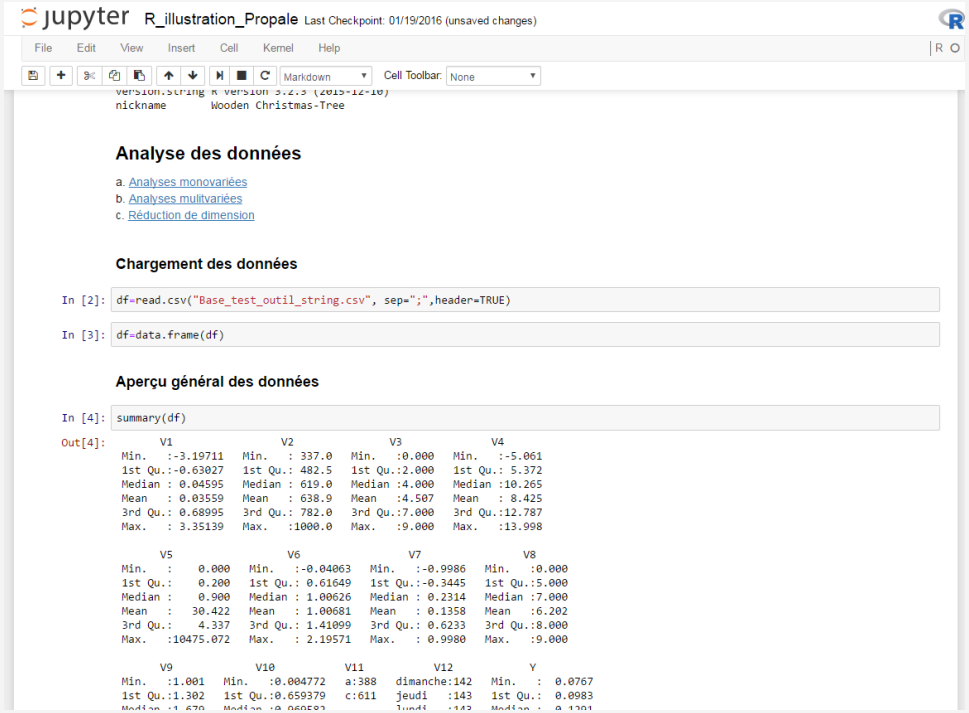
R		python
Designed for **statistical computing** and graphics. Not the easiest to start programming	**Learning curve**	Its design philosophy emphasizes **code readability** which makes Python a good starting point.
A large and rich diversity of statistical and machine learning modelling libraries.	**Modelling**	Less used for statistical modelling but largely adopted by the machine learning community, in particular in semantic and image analysis (deep neural network).
Some extensions as RShiny allow you to prototype dashboard. Huge development with R could be less easy.	**Prototyping**	Object oriented language, you can easily design your own application and integrate it in an operational process.
Not always optimized.	**Speed**	Python presents better performance than R.

2. Programming / Analytics

Powerful tool to save time in analysis

Jupyter: Web user application for Julia Python and R

- Save time: a unique tool to code, comment and plot
- Share your work with multi export format (pdf, HTML, markdown)
- Easily explore your data and test solutions



Notebook demo

2. Programming / Analytics

R, Python for Machine Learning

- Which one to chose to explore machine learning algorithm?

Each algorithm has its own libraries. You could then chose the implementation you prefer.

Libraries:

GLM: **stats**
CART: **tree, rpart**
Random Forest: **caret, randomForest**
Gradient boosting: **GBM, xgboost**
etc.

You not always know which one to chose. Few centralized libraries (caret) with everything.

Not recommended for neural network

Modelling

As in R many libraries exists. The one which makes reference in machine learning is **scikit-learn.** You can design quickly many algorithm and prepare your data with this package.

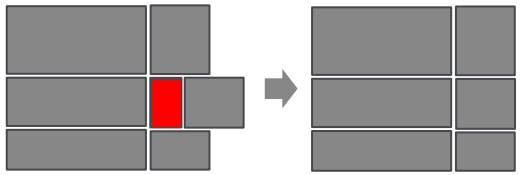
Packages:

Machine Learning (Preprocessing, clustering, Modelling, Dimensionality Reduction): **scikit-learn**
Statistical models: **statsmodels**
Neural Networks: **Keras, TensorFlow, Theano**

2. Programming / Analytics

Example of Python use for data-cleaning

1.1 Data structure



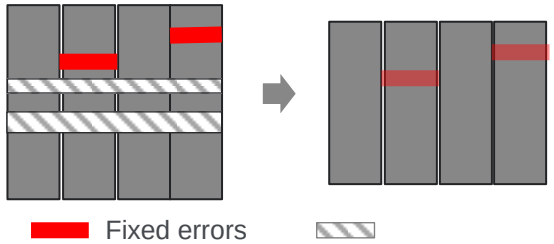
Identify and fix **data-structure** issues, such as **irregular column numbers:**

- This can occur during file merging phase if the initial files have different data-structures.
- Or mostly when working with CSV files due to CSV separators.

```
def clean_sep(line):
    …
    return clean_line
```

Application of the *clean_sep* function on each line

1.2 Data content



Identify **data-content** issues:

- Identify types for each variable.
- Identify anomalies in the values taken by each variables. (“#Values!” Errors for instance)

```
def detect_type(line):
    …
    return typelist
```

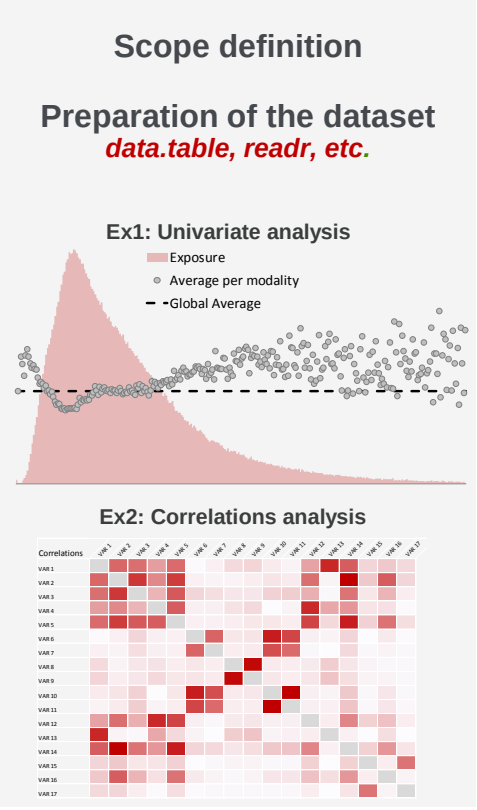
Application of the *detect_type* function on each line

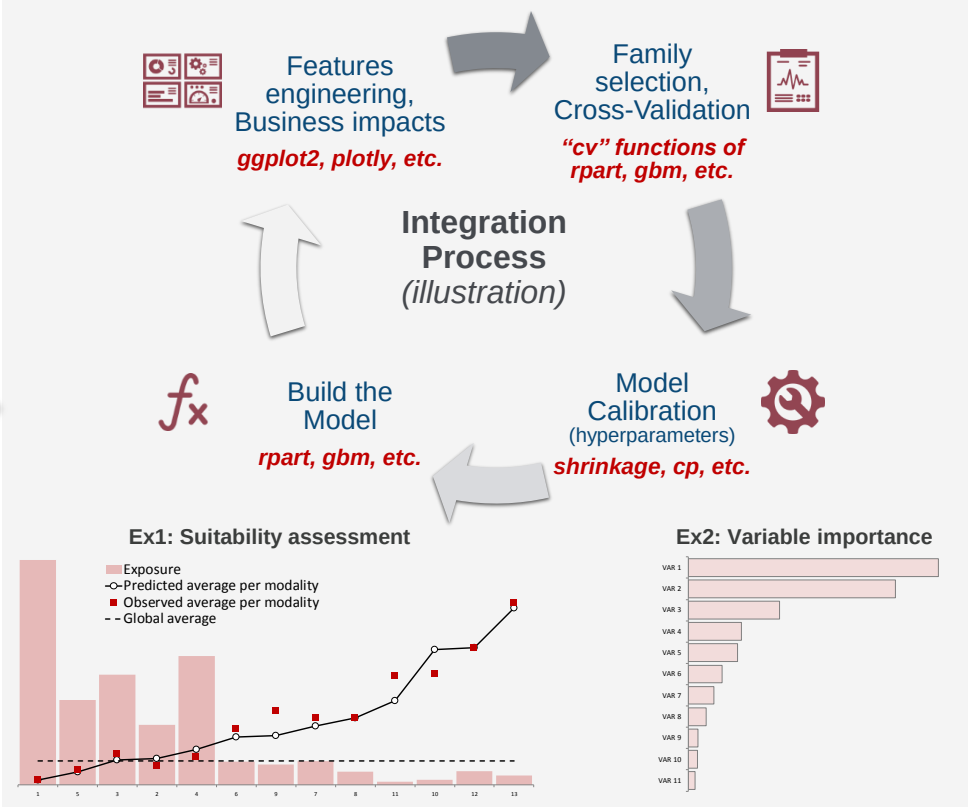
Data-cleaning

2. Programming / Analytics

Insurance case studies (1/2)

- How to choose and compute a model within **an insurance context using R**?
  - **Classical Statistics** (emphasize model) vs **Machine Learning** (distribution free, focused on making accurate predictions)
  - **Link between modeling and financial objectives**





2. Programming / Analytics

Insurance case studies (2/2)

- The **application domain** of Data Science is **very large within insurance**.

Binary Target

- Lapse Rate (churn, lapse between UC to Euro, etc.)
- Fraud analysis
- Etc.

Quantitative Target

- Pricing (pure premium)
- Customer Value
- Etc.

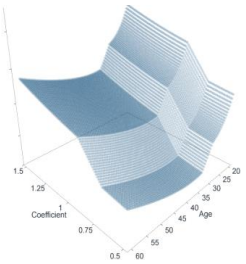
Specific Studies

- Multi equipment analysis
- Replace an existing pricing structure
- Claims development analysis
- Etc.

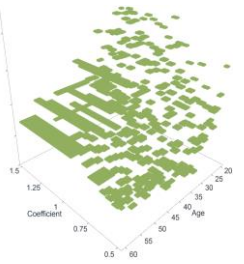
Illustration

Predict the **pure premium** within a **motor insurance** dataset (Property Damage Liability for instance)

DATABASE				IDEAL
X_1	X_2	W	Y_{obs}	Y_{theo}
COEF	AGE	Weight	Target	Theoretical Pure Premium
0.50	34	0.5	23.1	17.7
0.85	32	0.8	6.5	60.9
1.05	20	1	287.8	172.2
1.00	20	1	150.3	170.5
0.50	36	1	12.8	19.1
0.50	36	0.5	123.6	16.7
⋮	⋮	⋮	⋮	⋮
0.50	44	1	20.9	23.7

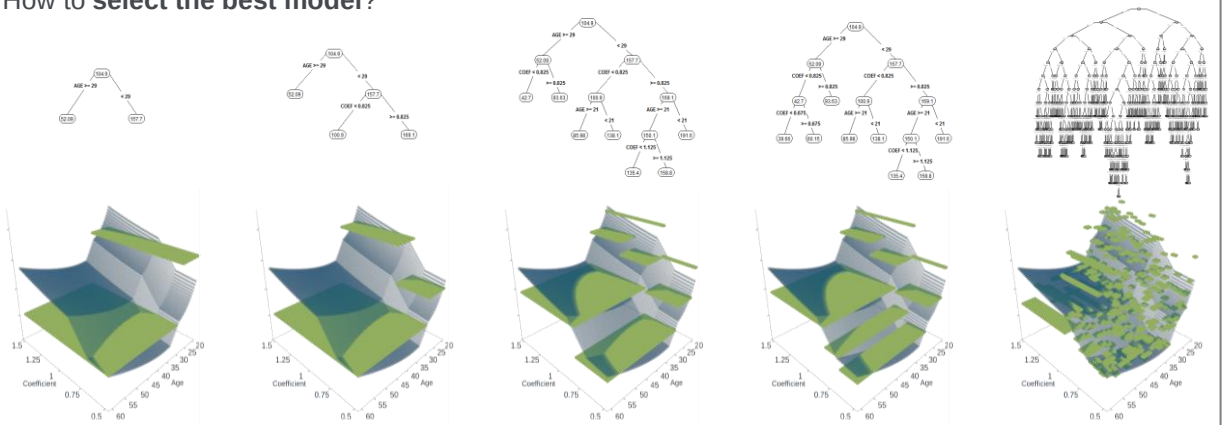


What is **wanted**



What is only **seen**

How to **select the best model**?



2. Programming / Analytics

In real life: R, Python

- How to install / try them ?
  - Python: Anaconda distribution https://www.continuum.io/downloads
  - R: CRAN https://www.r-project.org/

- Use an IDE (Integrated Development Environment)
  - Python: Spyder (provided with Anaconda), PyCharm
  - R: Rstudio https://www.rstudio.com/
  - Other: Visual Studio, Eclipse etc.

- Advantages of IDE:
  - Structure your projects and save time

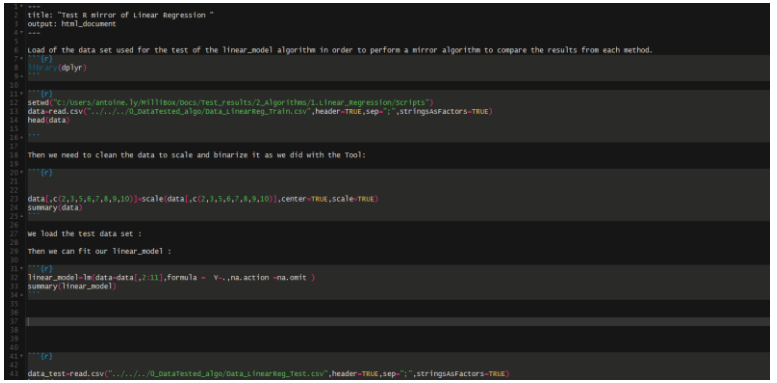
- Text editors + Terminal: Notepad++, Sublime Text

2. Programming / Analytics

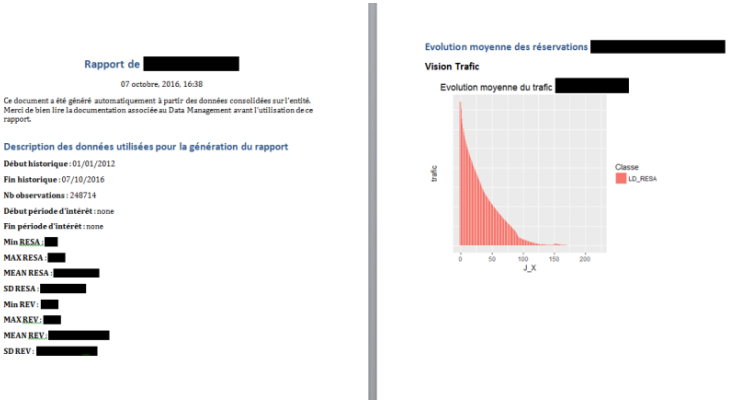
In real life: Case study

- Automatic reporting thanks to Rstudio Knit()

Script template (Rmarkdown)

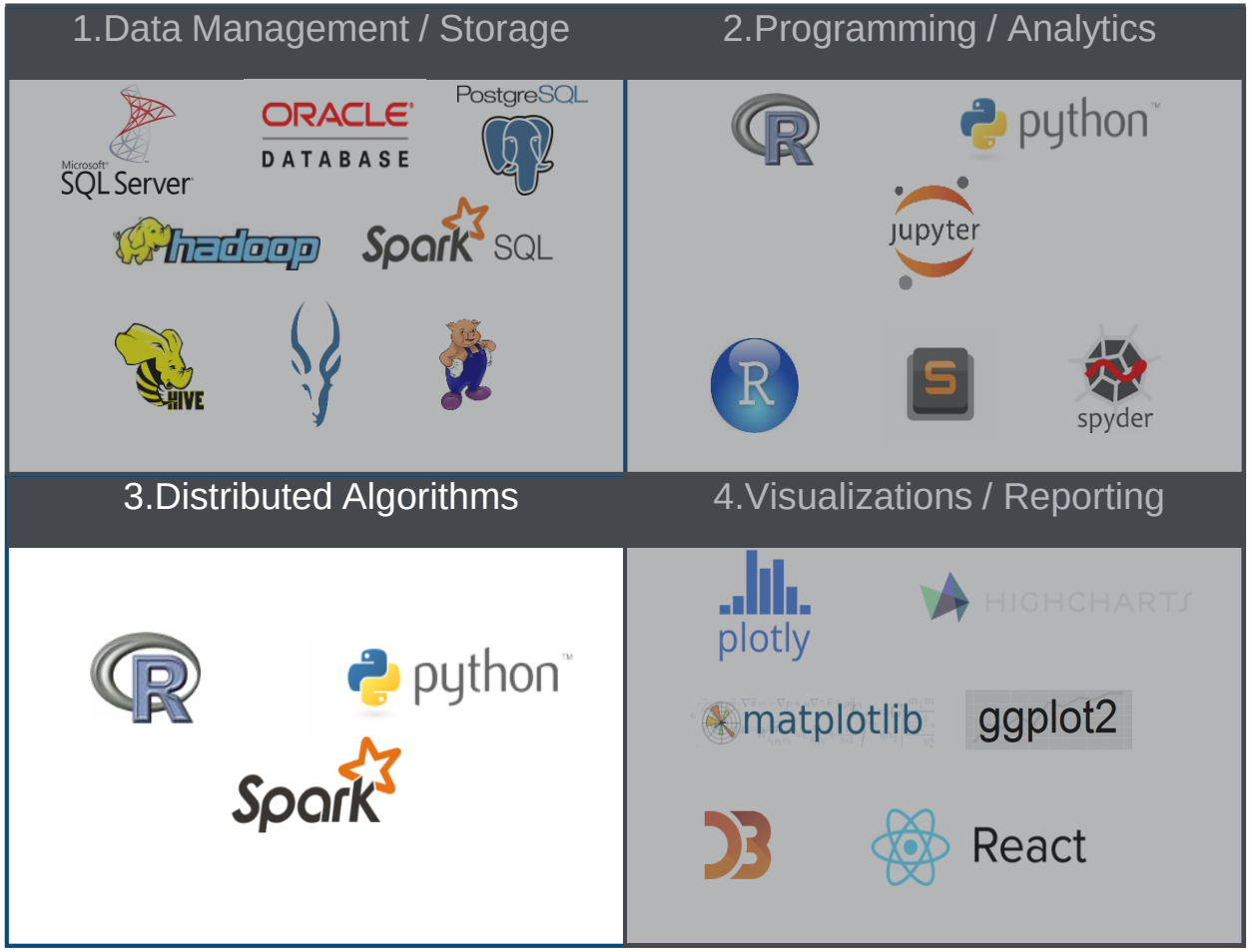


Daily reports Word



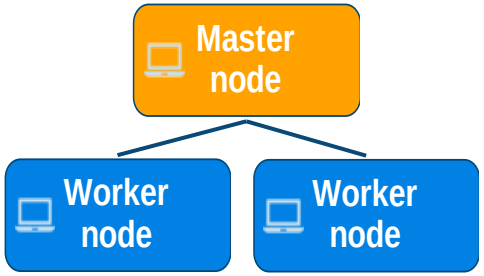
- Weekly/monthly reporting on triangles in mortgage (credit default)

3. Distributed Algorithms

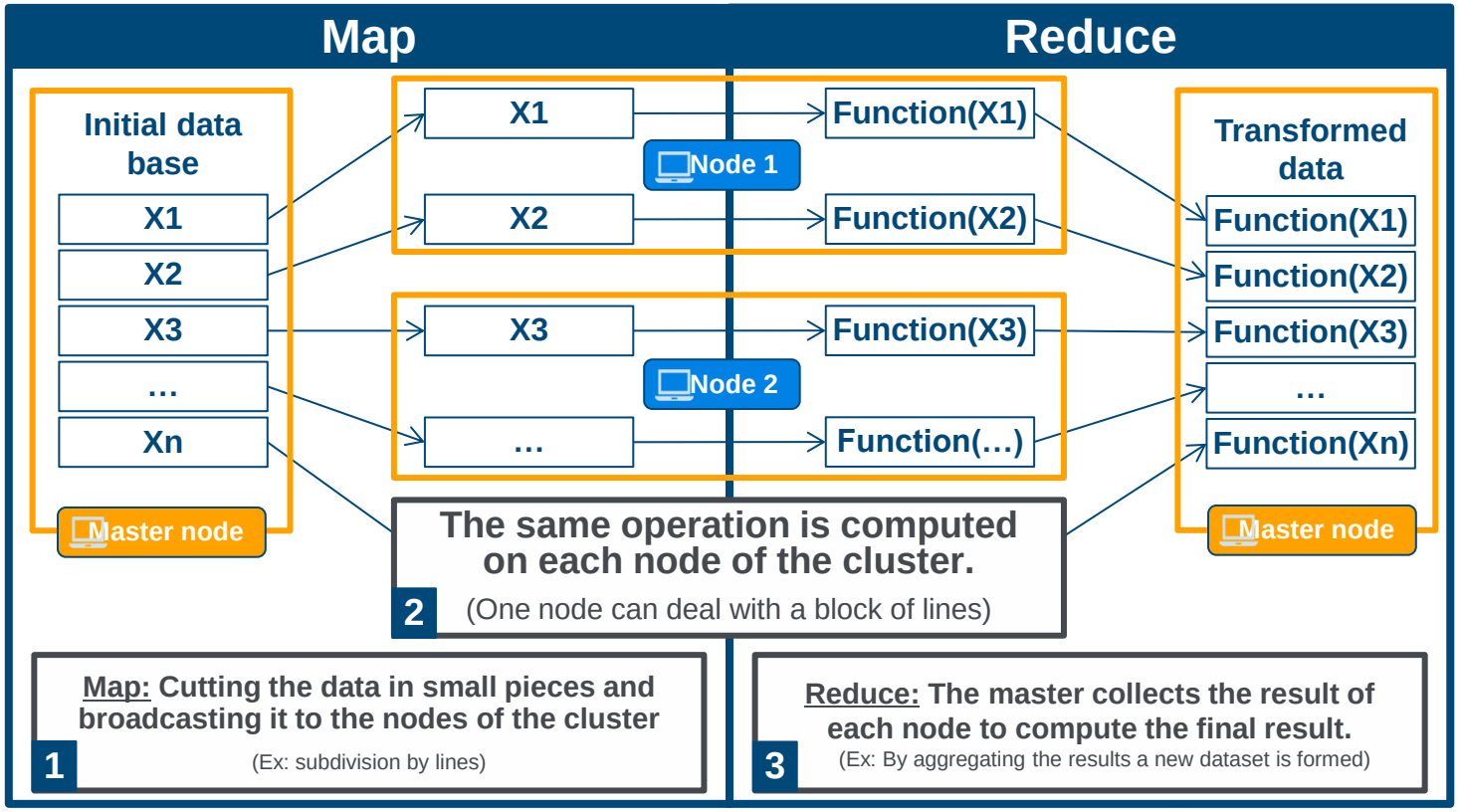


3. Distributed Algorithms

Analytics - Map-Reduce: How does it work ?



- **Concept** created by Google in order to deal with Big Data using a cluster architecture.
- **2 functions Map() and Reduce()**



3. Distributed Algorithms

Map Reduce in practice

How to compute quadratic error of prediction on a billion of lines?

We define for all $Y, \hat{Y} \in \mathcal{Y}^N$,

$$WSSSE(Y, \hat{Y}) = \sum_{i=1}^{N} error_2\ (Y_i, \hat{Y}_i)$$

reducer

mapper

Example with PySpark

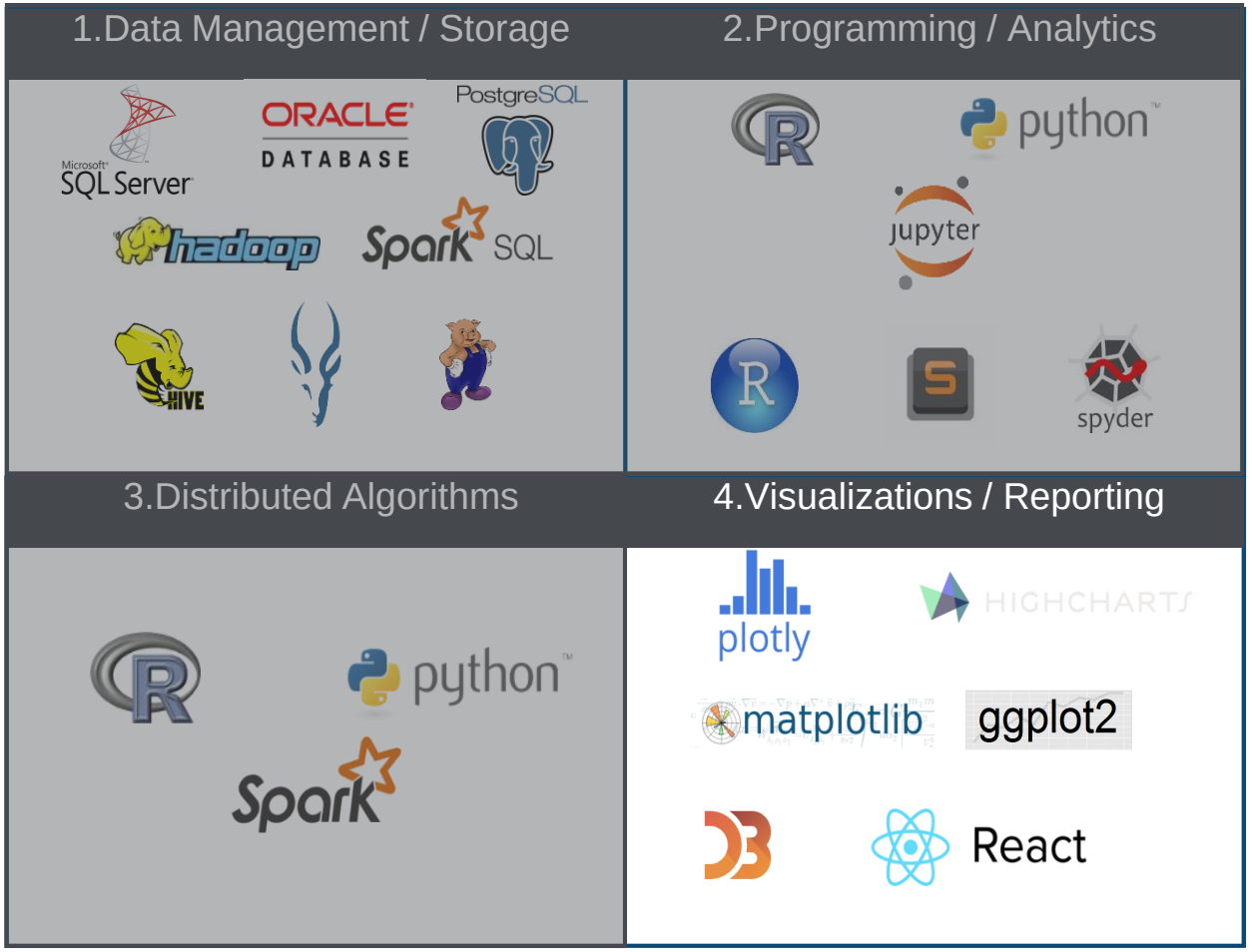
```
# Evaluate clustering by computing Within Set Sum of Squared Errors
def error(point):
    center = clusters.centers[clusters.predict(point)]
    return sqrt(np.sum([x**2 for x in (point - center)]))
```

mapper

```
WSSSE = parsedData.map(lambda point: error(point)).reduce(lambda x, y: x + y)
print("Within Set Sum of Squared Error = " + str(WSSSE))
```

reducer

4. Visualizations / Reporting



4. Visualizations / Reporting

Visualization for exploration

Dynamic and modern visualization techniques in order to build **business oriented visualizations**

Static

Matplotlib and ggplot:

Python and R graphic library, allowing graphical data export

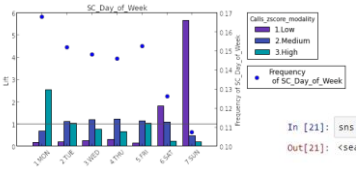
matplotlib ggplot2

Dynamic

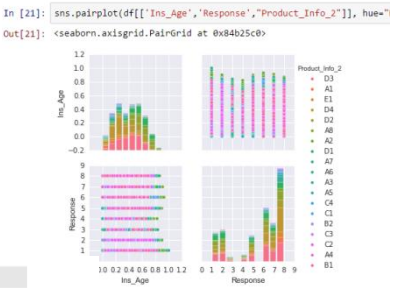
C3.js/D3.js:

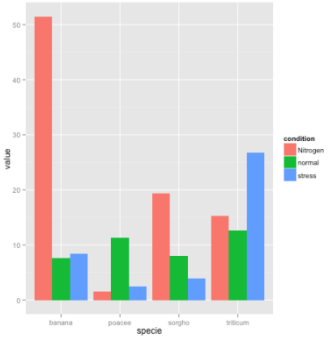
Web-technology based on javascript allowing dynamic data-visualization, useful from the exploring to the reporting phase.

matplotlib



Seaborn (python)



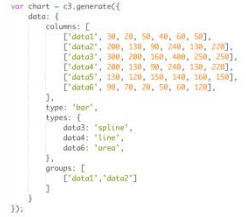


ggplot2

Usage:

- Matplotlib/ggplot works perfectly with Python/R and contains defaults plot styles with built-in code.

C3.js



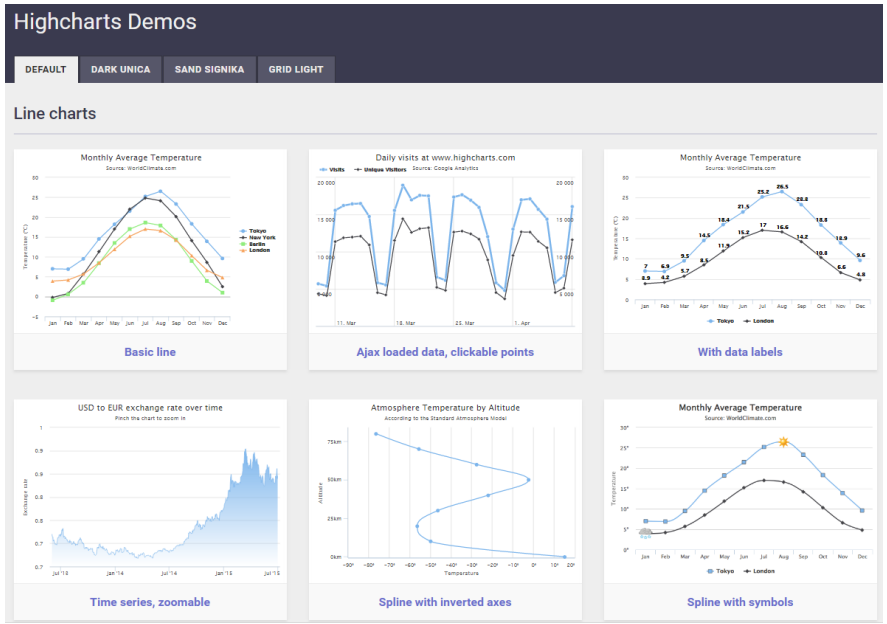
Usage:

- Use of C3.js and D3.js is recommended when dynamic interaction is required or when a web-integration is required.

4. Visualizations / Reporting

Build your own interactive dashboard (1/2)

- Hightcharts: SaaS to generate interactive graphics then integrate them in a web page
  - http://www.highcharts.com/

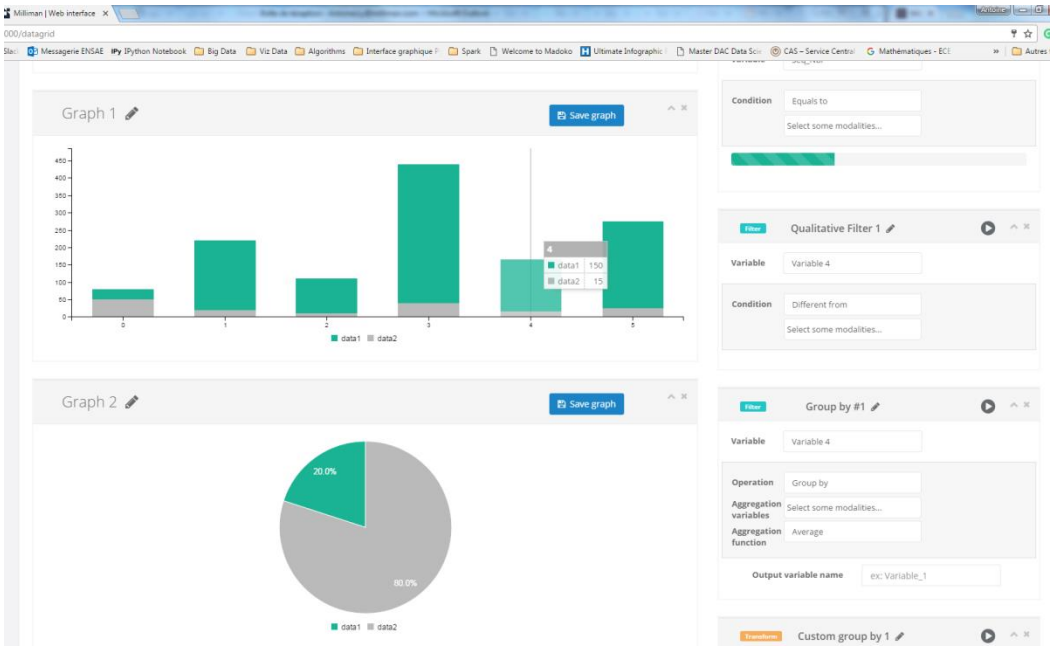


- Plotly: library available for R/Python to produce interactive graphics
  - https://plot.ly/python/offline/

4. Visualizations / Reporting

Build your own interactive dashboard (2/2)

C3.js D3-based reusable chart library



Integrating different technologies in one solution

An example of workflow used by Milliman

Data-Lake / Storage

Clustered Computing

Notebook interface

Web interface

- Spark/Hadoop Users across all industries and Big-Data sofware vendors:

(source : https://cwiki.apache.org/confluence/display/SPARK/Powered+By+Spark)

(…)

Data
Innovation Lab

Thank you

Eric. O Lebigot – eric.lebigot@axa.com

Rémi Bellina – remi.bellina@milliman.com

Yves-Richard Hong Tuan Ha – yves.hong@milliman.com

Antoine Ly – antoine.ly@milliman.com