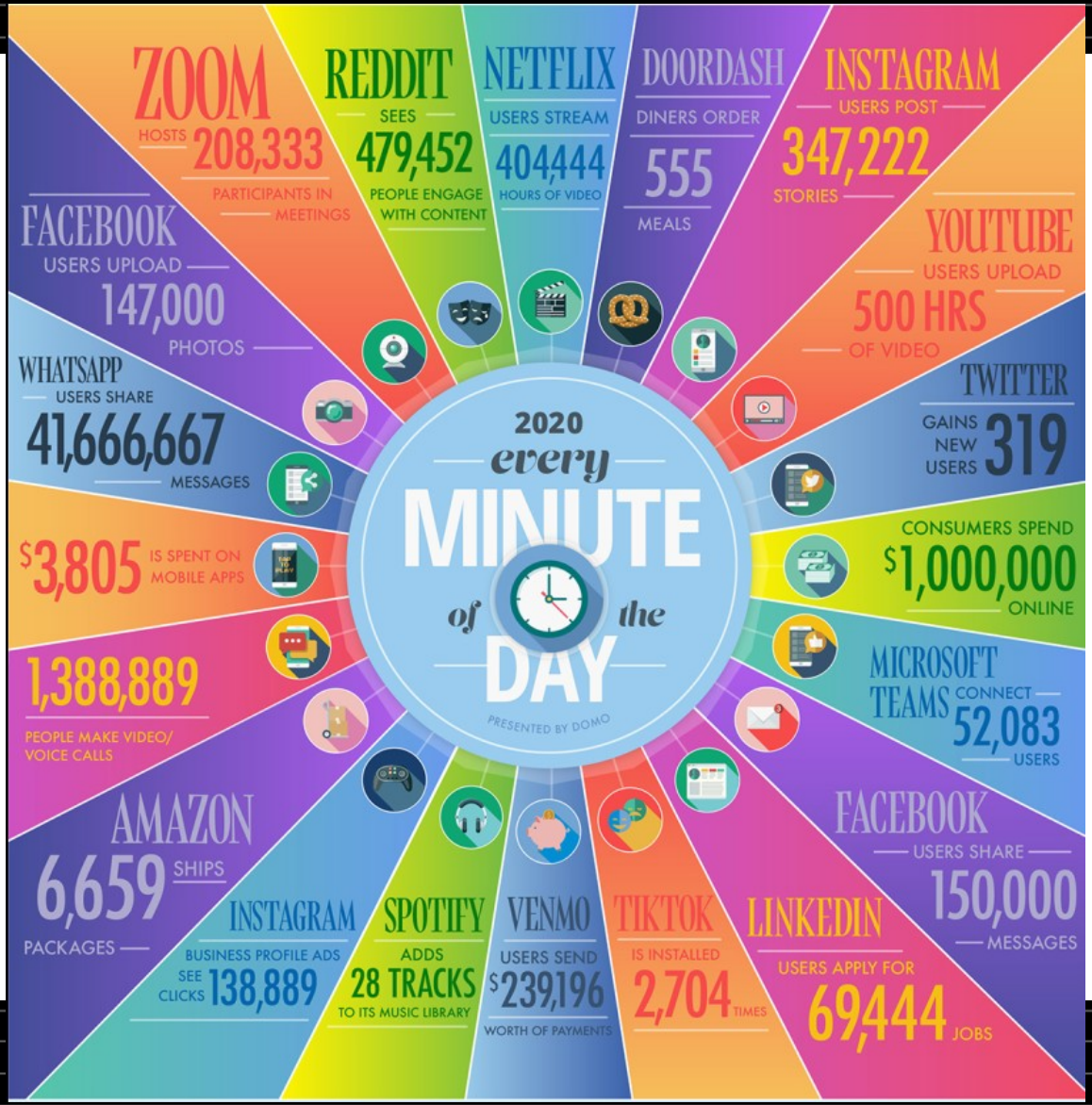
# Analyse de données et Data Science

Une intro

C. Fonlupt


2020
every
MINUTE
of the
DAY
PRESENTED BY DOMO
ZOOM
HOSTS 208,333
PARTICIPANTS IN MEETINGS
REDDIT
SEES 479,452
PEOPLE ENGAGE WITH CONTENT
NETFLIX
USERS STREAM 404,444
HOURS OF VIDEO
DOORDASH
DINERS ORDER 555
MEALS
INSTAGRAM
USERS POST 347,222
STORIES
FACEBOOK
USERS UPLOAD 147,000
PHOTOS
YOUTUBE
USERS UPLOAD 500 HRS
OF VIDEO
WHATSAPP
USERS SHARE 41,666,667
MESSAGES
TWITTER
GAINS NEW USERS 319
$3,805 IS SPENT ON MOBILE APPS
CONSUMERS SPEND $1,000,000 ONLINE
1,388,889
PEOPLE MAKE VIDEO/ VOICE CALLS
MICROSOFT TEAMS CONNECT 52,083 USERS
AMAZON
6,659 SHIPS
PACKAGES
FACEBOOK
USERS SHARE 150,000
MESSAGES
INSTAGRAM
BUSINESS PROFILE ADS SEE CLICKS 138,889
SPOTIFY
ADDS 28 TRACKS
TO ITS MUSIC LIBRARY
VENMO
USERS SEND $239,196
WORTH OF PAYMENTS
TIKTOK
IS INSTALLED 2,704 TIMES
LINKEDIN
USERS APPLY FOR 69,444 JOBS

# Data Science

- Après l’accumulation de données
- Exploitation de ces données
- Compréhension de ces données
- Visualisation de ces données
- Prédiction de données

# Définition de la data Science

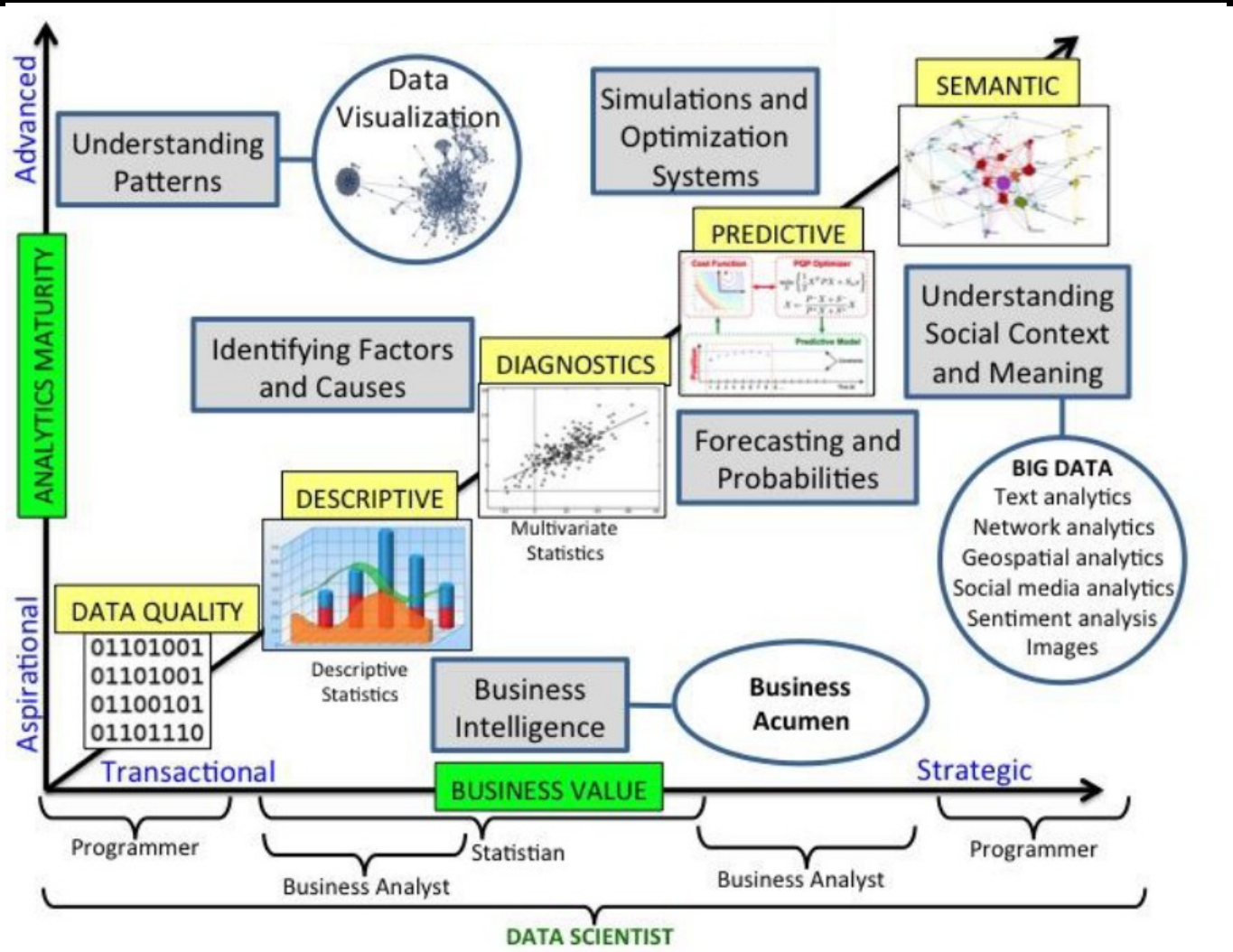
- S’appuyer sur des outils mathématiques, statistiques, informatiques et de visualisation des données pour transformer ces données brutes en informations utiles

# Le data scientist


Il doit tout savoir faire !

- Informaticien
- Mathématicien
- Financier
- Analyste
- Visualiseur !

# Une autre vue



# Complexe mais

- Demande croissante du métier de Data Scientist
- Apprendre à prédire le futur c’est plutôt sympa comme job
- On va apprendre à manipuler en quelques lignes des centaines ou milliers de données
- Il y a déjà plein d’outils


ANATOMY OF A
DATA SCIENTIST
SALARY
Average salary of data scientists is
$120,000/year
EDUCATION
• 88% of all data scientists have at least a Master's degree
• 46% of data scientists have a PhD
BENEFITS
• Harvard Business Review called data science the "Sexiest Job of the 21st Century"
• One of the fastest growing careers in the United States
• 94% of data science graduates have found jobs since 2011
SKILLS
• Programming languages (R, Python, SQL, Hive, etc.)
• Statistics
• Multivariable calculus and linear algebra
• Machine learning
• Software engineering
• Wrangle, visualize, and communicate data to management
RESPONSIBILITIES
• Conduct research
• Extract, clean, and analyze data from varied sources
• Solve problems
• Build automation tools
• Communicate findings to management
CAREER POSSIBILITIES
• The majority of data scientists work in the technology industry.
• Other options include marketing, consulting, healthcare and pharmaceuticals, finance, government, gaming, and many more.
RESOURCES:
https://insidebigdata.com/2017/08/05/benefits-data-scientist-career/
https://www.glassdoor.com/Salaries/us-data-scientist-salary-SRCH_IL.0,2_IN1_KO3,17.htm
https://blog.udacity.com/2014/11/data-science-job-skills.html
https://online.rutgers.edu/resources/infographics/what-can-you-do-with-a-career-in-data-science/?program=mi
THE COMPUTER MERCHANT, LTD.
THE IT STAFFING COMPANY

# Que faire de ces données ?

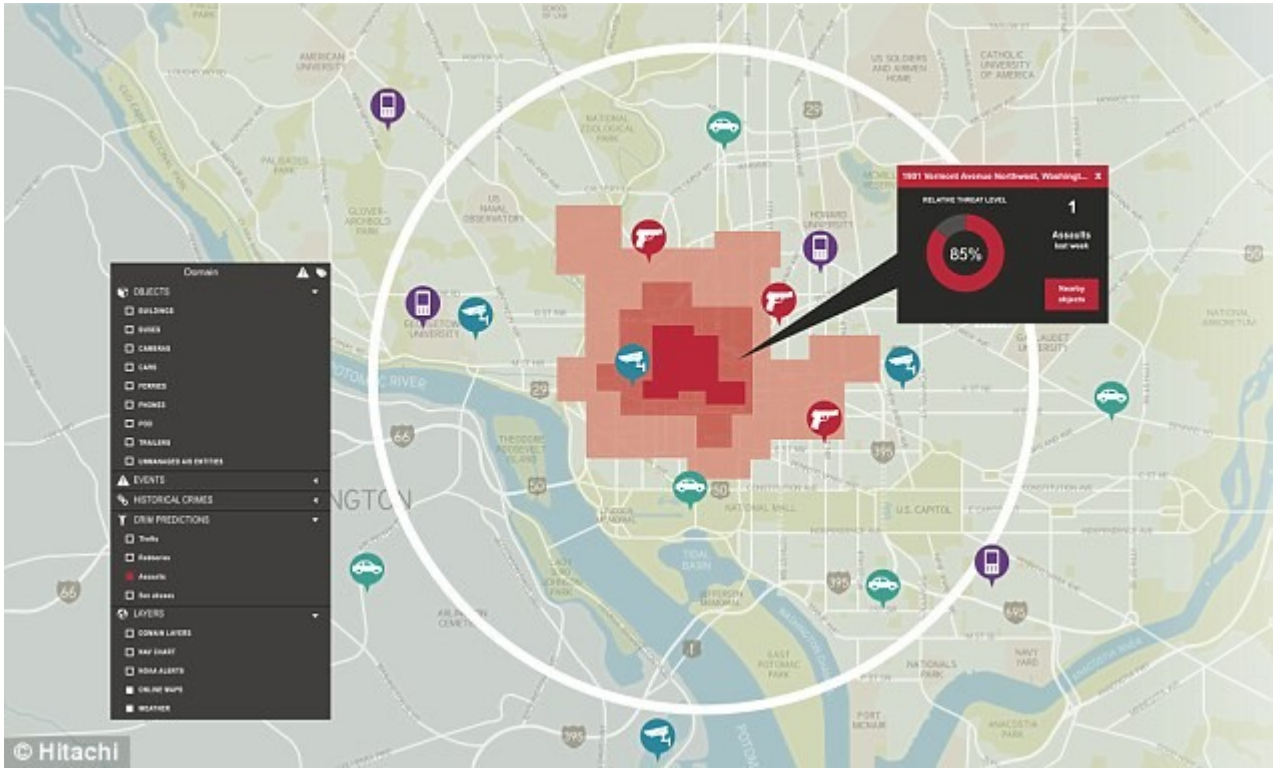
- Vendre
  - Mieux connaître son client (amazon, netflix, …)
  - Proposition d'achats supplémentaires
- Aider à la décision
  - Les banques, les assurances
  - Prévention des risques, de la délinquance,…
  - Médecine (anticiper la réponse d'un patient avant le début d'une chimiothérapie, ..)
  - Pharmacie (rapport coûts/bénéfices d'un traitement…)
  - Prévenir la pollution
- Science comportementale
  - comprendre des biais
  - analyse d'opinion sur le web

# Secteur bancaire

je collabore avec le département des ressources humaines sur un projet en lien avec le Machine Learning et les données analytiques. Il s'agit de développer un outil qui permette d'avoir des visions, à date et future, de nos besoins en compétences. Cela permet au département RH d'anticiper les recrutements sur les prochaines années et de faire ainsi en sorte que la banque ne manque pas d'experts, notamment dans les domaines porteurs d'avenir comme la Blockchain.

Source : https://group.bnpparibas/actualite/metiers-banque-data-scientist-senior

# Minority report




Source dailymail.co.uk

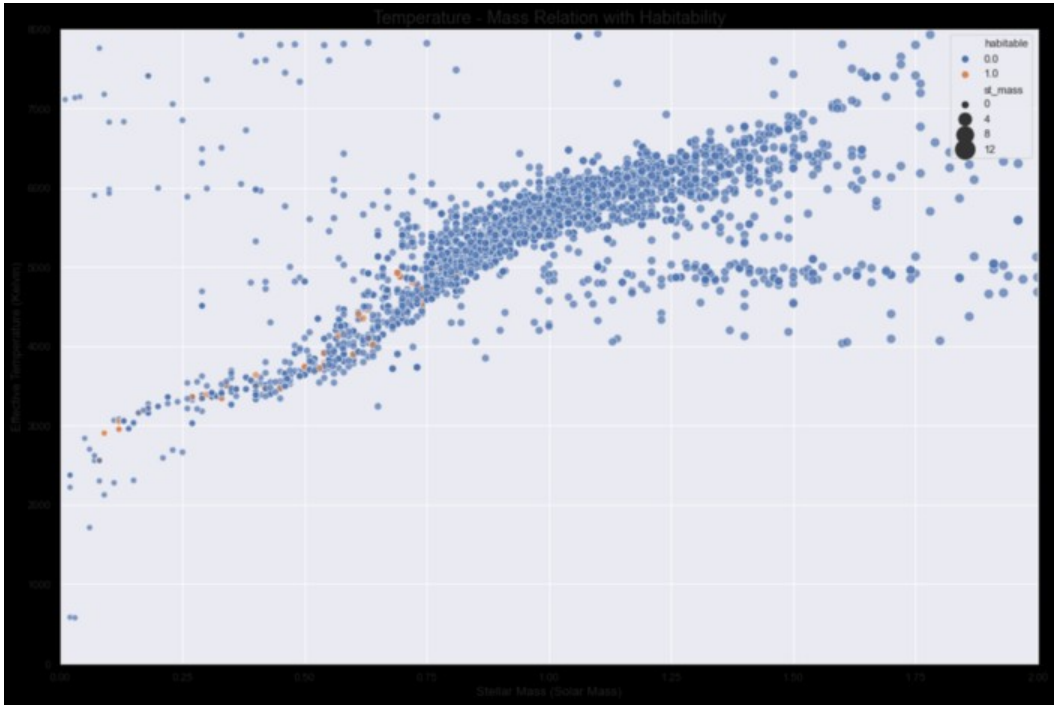
## WHAT DATA DOES PCA USE?

Hitachi's Predictive Crime Analytics blends 'real-time event data from public safety systems and sensors with historical and contextual crime data, social media and other sources.'

These include:

- CCTV and video management systems such as Genetec and Pelco.
- Emergency call data
- Gunshot detection systems including Shotspotter
- Live weather radar
- Twitter feeds
- Traffic systems
- Crime and incident data

# Détection des exoplanètes en astronomie



# A vous de jouer ?

- La base de données des joueurs de la FIFA
- https://www.kaggle.com/thec03u5/fifa-18-demo-player-dataset

```
Entrée [1]: import pandas as pd

Entrée [2]: https://www.kaggle.com/thec03u5/fifa-18-demo-player-dataset

e:\langages\python3\lib\site-packages\IPython\core\interactiveshell.py:3146: DtypeWarning: Columns (23,35) have mixed type
s.Specify dtype option on import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,

Entrée [3]: données = pd.read_csv('c:/Temp/CompleteDataset.csv',sep=',',low_memory=False)

Entrée [4]: données

Out[4]:
```

| | Unnamed: 0 | Name | Age | Photo | Nationality | Flag | Overall | Potential | Club | Club Logo | ... | RB | RCB | RCM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | Cristiano Ronaldo | 32 | https://cdn.sofifa.org/48/18/players/20801.png | Portugal | https://cdn.sofifa.org/flags/38.png | 94 | 94 | Real Madrid CF | https://cdn.sofifa.org/24/18/teams/243.png | ... | 61.0 | 53.0 | 82.0 |
| 1 | 1 | L. Messi | 30 | https://cdn.sofifa.org/48/18/players/158023.png | Argentina | https://cdn.sofifa.org/flags/52.png | 93 | 93 | FC Barcelona | https://cdn.sofifa.org/24/18/teams/241.png | ... | 57.0 | 45.0 | 84.0 |
| 2 | 2 | Neymar | 25 | https://cdn.sofifa.org/48/18/players/190871.png | Brazil | https://cdn.sofifa.org/flags/54.png | 92 | 94 | Paris Saint-Germain | https://cdn.sofifa.org/24/18/teams/73.png | ... | 59.0 | 46.0 | 79.0 |

# Les données

- structurées / non structurées
- complètes / incomplètes / incorrectes
- textuelles / non textuelles

# Données structurées

- format normalisé permettant de fournir des informations sur une page et de classer le contenu de cette page

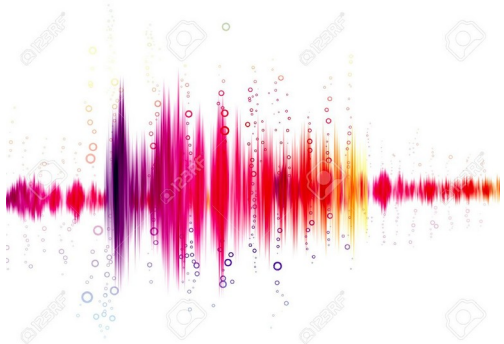
```
<html>
  <head>
    <title>Party Coffee Cake</title>
    <script type="application/ld+json">
    {
      "@context": "https://schema.org/",
      "@type": "Recipe",
      "name": "Party Coffee Cake",
      "author": {
        "@type": "Person",
        "name": "Mary Stone"
      },
      "datePublished": "2018-03-10",
      "description": "This coffee cake is awesome and perfect for parties.",
      "prepTime": "PT20M"
    }
    </script>
  </head>
  <body>
  <h2>Party coffee cake recipe</h2>
  <p>
    This coffee cake is awesome and perfect for parties.
  </p>
  </body>
</html>
```

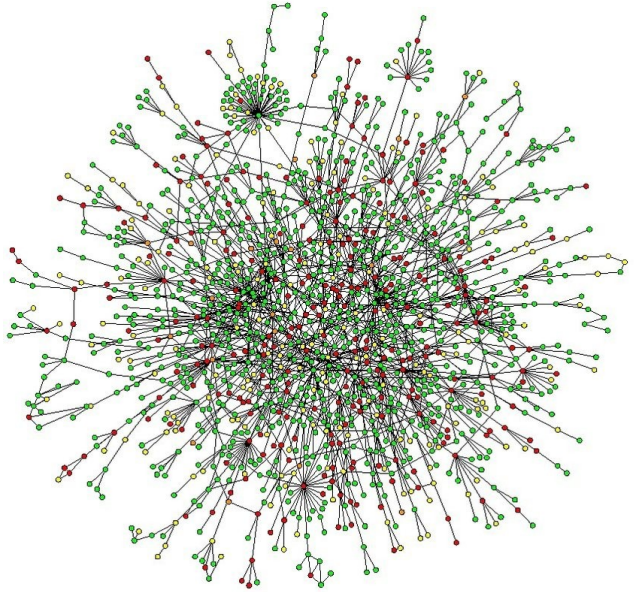
# Données semi-structurées

- Contiennent des éléments d'informations
- Par exemple fichier json, XML, texte brut

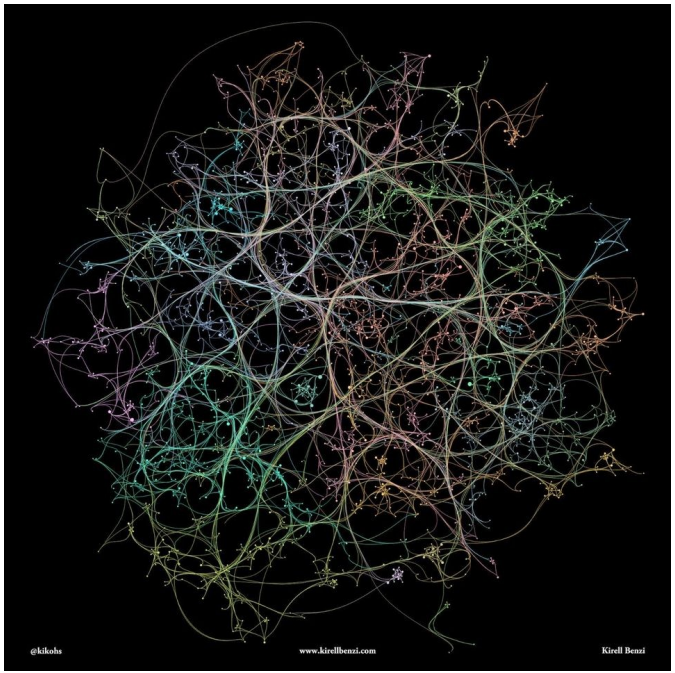
# Données non structurées

- Tous les autres !





# Data Science et Arts




startups de viva tech

https://actu.epfl.ch/news/
que-peut-nous-apprendre-wikipedia-sur-les-intera-4/

# Pour nous
# l'éco-système python

- python
- numpy
- pandas
- sci-kit (peut-être)
- matplotlib, seaborn





# Python

- *cf* résumé du cours
- un module pratique pour notre cours
- les expressions régulières (module re)
- utile pour analyser un fichier texte

# Les données pour ce cours

- données structurées
  - listes
  - tableaux
  - réseaux

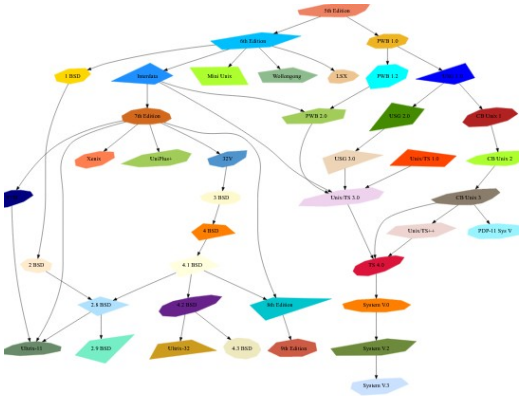
# Les données structurées

- CSV (comma separated values)

```
,Name,Age,Photo,Nationality,Flag,Overall,Potential,Club,Club
Logo,Value,Wage,Special,Acceleration,Aggression,Agility,Balance,Ball
control,Composure,Crossing,Curve,Dribbling,Finishing,Free kick accuracy,GK
diving,GK handling,GK kicking,GK positioning,GK reflexes,Heading
accuracy,Interceptions,Jumping,Long passing,Long
shots,Marking,Penalties,Positioning,Reactions,Short passing,Shot power,Sliding
tackle,Sprint speed,Stamina,Standing
tackle,Strength,Vision,Volleys,CAM,CB,CDM,CF,CM,ID,LAM,LB,LCB,LCM,LDM,LF,LM,LS,LW
,LWB,Preferred Positions,RAM,RB,RCB,RCM,RDM,RF,RM,RS,RW,RWB,ST
0,Cristiano
Ronaldo,32,https://cdn.sofifa.org/48/18/players/20801.png,Portugal,https://cdn.so
fifa.org/flags/38.png,94,94,Real Madrid
CF,https://cdn.sofifa.org/24/18/teams/243.png,€95.5M,€565K,2228,89,63,89,63,93,
95,85,81,91,94,76,7,11,15,14,11,88,29,95,77,92,22,85,95,96,83,94,23,91,92,31,80,8
5,88,89.0,53.0,62.0,91.0,82.0,20801,89.0,61.0,53.0,82.0,62.0,91.0,89.0,92.0,91.0,
66.0,ST LW ,89.0,61.0,53.0,82.0,62.0,91.0,89.0,92.0,91.0,66.0,92.0
1,L.
```

# Les données structurées

- Excel
- réseaux GraphViz (dot)



```
digraph "unix" {
        graph [ fontname = "Helvetica-Oblique",
                fontsize = 36,
                label = "\n\n\n\nObject Oriented Graphs\nStephen North, 3/19/93",
                size = "6,6" ];
        node [  shape = polygon,
                sides = 4,
                distortion = "0.0",
                orientation = "0.0",
                skew = "0.0",
                color = white,
                style = filled,
                fontname = "Helvetica-Outline" ];
        "5th Edition" [sides=9, distortion="0.936354", orientation=28, skew="-0.126818", color=salmon2];
```

de graphviz.org

# Les données semi-structurées

- XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<breakfast_menu>
<food>
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>
   Two of our famous Belgian Waffles with plenty of real maple syrup
   </description>
    <calories>650</calories>
</food>
<food>
    <name>Strawberry Belgian Waffles</name>
    <price>$7.95</price>
    <description>
    Light Belgian waffles covered with strawberries and whipped cream
    </description>
    <calories>900</calories>
</food>
```

de w3schools.com

# Les données semi-structurées

- JSON semblable à XML mais plus simple

```
{
    "glossary": {
        "title": "example glossary",
                "GlossDiv": {
            "title": "S",
                        "GlossList": {
                "GlossEntry": {
                    "ID": "SGML",
                                        "SortAs": "SGML",
                                        "GlossTerm": "Standard Generalized Markup Language",
                                        "Acronym": "SGML",
                                        "Abbrev": "ISO 8879:1986",
                                        "GlossDef": {
                        "para": "A meta-markup language, used to create markup languages such as DocBook.",
                                                "GlossSeeAlso": ["GML", "XML"]
                    },
                                        "GlossSee": "markup"
                }
            }
        }
    }
}
```

de json.org

# Ce ne sont que quelques exemples

- On peut cependant remarquer la multiplicité des données
- Leur hétérogénéité
- Parfois des données incomplètes

# En deuxième partie

- Analyse en composantes principales
- Algorithmes simples d'apprentissage
  - la classification supervisée
  - la régression
  - le regroupement (clustering)

# Exemple prédire si un passager du Titanic va survivre ou non

- TP dessus prédiction en 2ème temps

Out[2]:

| | PassengerId | Survived | Pclass | Name | Sex | Age | SibSp | Parch | Ticket | Fare | Cabin | Embarked |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 0 | 3 | Braund, Mr. Owen Harris | male | 22.0 | 1 | 0 | A/5 21171 | 7.2500 | NaN | S |
| **1** | 2 | 1 | 1 | Cumings, Mrs. John Bradley (Florence Briggs Th... | female | 38.0 | 1 | 0 | PC 17599 | 71.2833 | C85 | C |
| **2** | 3 | 1 | 3 | Heikkinen, Miss. Laina | female | 26.0 | 0 | 0 | STON/O2. 3101282 | 7.9250 | NaN | S |
| **3** | 4 | 1 | 1 | Futrelle, Mrs. Jacques Heath (Lily May Peel) | female | 35.0 | 1 | 0 | 113803 | 53.1000 | C123 | S |
| **4** | 5 | 0 | 3 | Allen, Mr. William Henry | male | 35.0 | 0 | 0 | 373450 | 8.0500 | NaN | S |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |

de w3schools.com

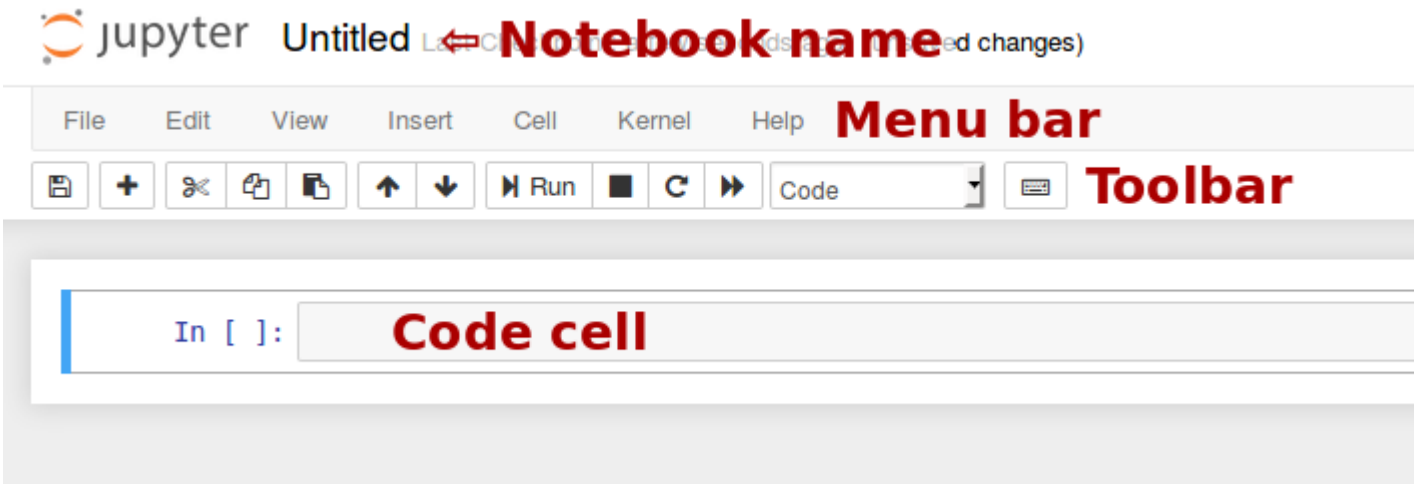
# Notre environnement de travail

- Jupyter notebooks
- Julia Python R
- https://jupyter.org/

# Étend la console vers le web

- Permet d'éditer du code dans le navigateur
- Exécution du code et résultat dans le navigateur
- Affiche les résultats de plusieurs sources HTML, LaTeX, PNG, SVG, matplotlib…
- Formatage du etxte en utilisant le langage markdown
- Formules mathématiques manipulées nativement grâce à MathJax

# L'interface des notebook



# Description

- Nom du notebook → sauvegardé automatiquement au format .ipynb
- Les cellules :
  - du code (pour nous en python) qui dépend du Kernel → shift + Enter pour l'exécution
  - du code Markdown pour la présentation
  - des cellules de texte non évaluées

# Du code

- voir le notebook Démonstration notebook

1. Basic navigation: `enter`, `shift-enter`, `up/k`, `down/j`
2. Saving the notebook: `s`
3. Change Cell types: `y`, `m`, `1-6`, `t`
4. Cell creation: `a`, `b`
5. Cell editing: `x`, `c`, `v`, `d`, `z`
6. Kernel operations: `i`, `0` (press twice)

et H

# Markdown

- Sur-ensemble de HTML
- Permet de faire des présentations, d'écrire du texte, d'inclure des images…
- https://daringfireball.net/projects/markdown/

# Markdown (suite)

- Un lien s' écrit comme ça [Google](http://google.fr)
- On peut mettre également des images ![alt text](/path/to/img.jpg "Title")
- *cf* notebook Démonstration notebook

# Visualisation : Seaborn

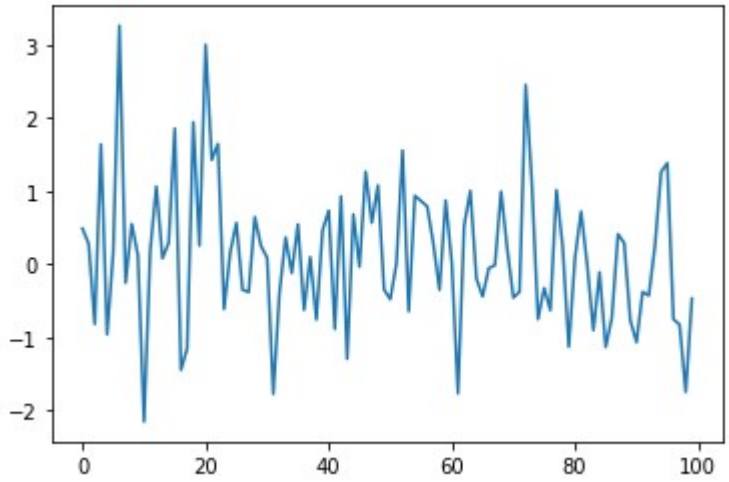
- surcouche de matplotlib (https://matplotlib.org/) bibliothèque standard de python
- nous allons l'utiliser avec les jupyter notebook

# Matplotlib

- Principe général
- import matplotlib.pyplot as plt
- plt.commande(x=abscisses,y=ordonnées)
- ou plt.commande(données) dans ce cas les ordonnées, les abscisses étant implicites

# Exemple

plt.plot(np.random.randn(100))



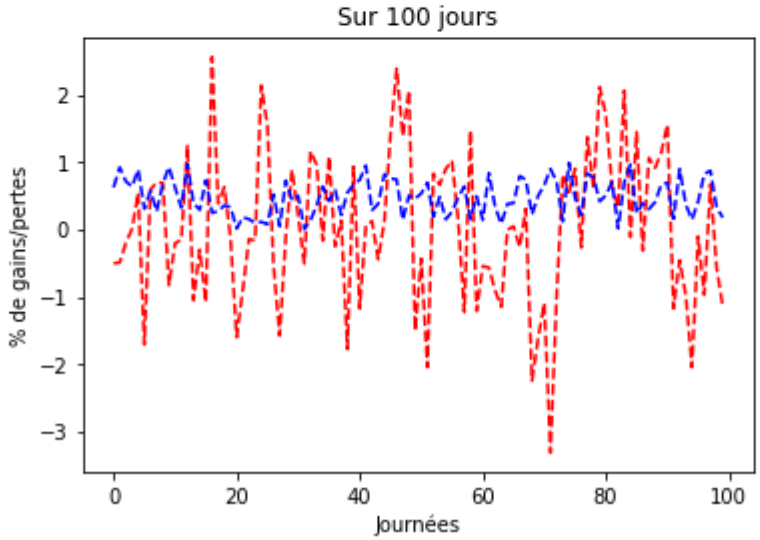

randn distribution normale

# Améliorations

**De manière plus jolie**

Entrée [14]:
```
plt.plot(np.random.randn(100),'r--') # red pour rouge
plt.plot(np.random.rand(100),'b--') # b pour blue
plt.title("Sur 100 jours")
plt.xlabel('Journées')
plt.ylabel("% de gains/pertes")
```

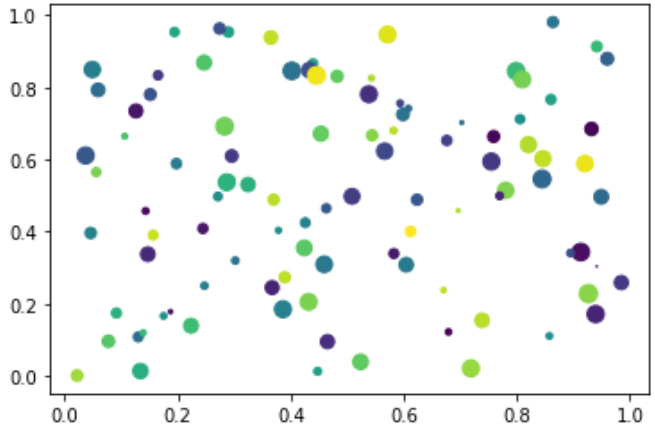
Out[14]: `Text(0, 0.5, '% de gains/pertes')`



# Nuage de points

Entrée [16]:
```
x = np.random.random(size = 100) # par défaut random.ran
y = np.random.random(size = 100)
taille = np.random.random(size = 100)*100
couleurs = np.random.random(size = 100)*100
# s représente la taille des points
#c représente les couleurs
plt.scatter(x,y, s=taille, c=couleurs)
```

Out[16]: <matplotlib.collections.PathCollection at 0x1de7dffea00>



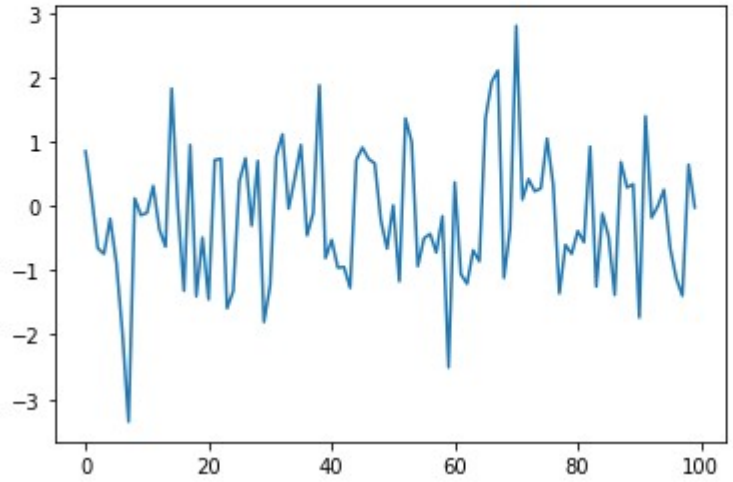
# Seaborn

- représentation plus élaborée
- est basée sur matplotlib
- import seaborn as sns
- bien adapté aux dataframes de pandas

# Avec seaborn

```
ntrée [34]: sns.lineplot(x = np.arange(0,100,1) ,y = np.random.randn(100))
```

```
Out[34]: <AxesSubplot:>
```



# Premiers éléments

- Notebook sur les actions
  - cours du CAC40 du 01/12/2019 au 01/12/2020
  - et des actions Société Générale, Carrefour, Air Liquide et de l'or
- import seaborn as sns

# Dessinez les données

- commande lineplot pour afficher sous formes de lignes
- *cf* Notebooks
- Échelles incompatibles, pas beaucoup de sens

# Une meilleure visualisation

- Dessinez chacune des séries de manière individuelle
- Ou n'en gardez qu'une ou deux
- Le principe général est de lui indiquer l'axe des x et l'axe des y (data=…, x =..., y = ....)

# Plus précisément

- data → le DataFrame ou la série sur laquelle on travaille
- x → axe des x par exemple x = data.index (pour utiliser l'index)
- y → axe des ordonnées (en général une Series)

# Exemple

```
Entrée [73]: données
```

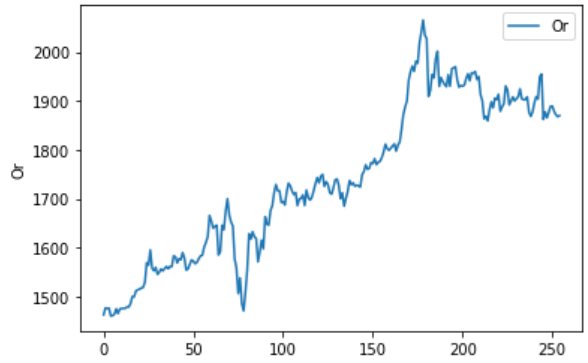
Out[73]:

| | CAC40 | SG | Carrefour | Air Liquide | Or |
|---|---|---|---|---|---|
| **0** | 5787.14 | 28.630 | 14.845 | 120.45 | 1462.71 |
| **1** | 5725.97 | 27.980 | 14.810 | 118.60 | 1476.62 |
| **2** | 5808.26 | 28.900 | 14.685 | 120.80 | 1475.57 |
| **3** | 5815.75 | 29.240 | 14.680 | 121.80 | 1476.07 |
| **4** | 5865.73 | 29.300 | 15.040 | 122.45 | 1460.06 |
| **...** | ... | ... | ... | ... | ... |
| **250** | 5577.97 | 17.524 | 13.630 | 135.20 | 1889.15 |
| **251** | 5589.38 | 17.470 | 13.900 | 136.45 | 1878.60 |
| **252** | 5563.04 | 17.080 | 13.625 | 137.75 | 1871.44 |
| **253** | 5564.16 | 17.148 | 13.750 | 137.95 | 1867.60 |
| **254** | 5542.46 | 16.786 | 13.780 | 138.75 | 1869.84 |

255 rows × 5 columns

```
Entrée [75]: sns.lineplot(data=données, x = données.index, y = 'Or', label = 'Or')
```

Out[75]: <AxesSubplot:ylabel='Or'>




```
]: sns.lineplot(data=données, x = données.index, y = données['Or'], label = 'Or')
```

<AxesSubplot:ylabel='Or'>

# Format général de seaborn

seaborn.**lineplot** (*, x=None, y=None, hue=None, size=None, style=None, data=None, palette=None, hue_order=None, hue_norm=None, sizes=None, size_order=None, size_norm=None, dashes=True, markers=None, style_order=None, units=None, estimator='mean', ci=95, n_boot=1000, seed=None, sort=True, err_style='band', err_kws=None, legend='auto', ax=None, **kwargs)

Draw a line plot with possibility of several semantic groupings.

# Suite

**Parameters:** **x, y** : ***vectors or keys in*** `data`

Variables that specify positions on the x and y axes.

**hue** : ***vector or key in*** `data`

Grouping variable that will produce lines with different colors. Can be either categorical or numeric, although color mapping behave differently in latter case.

**size** : ***vector or key in*** `data`

Grouping variable that will produce lines with different widths. Can be either categorical or numeric, although size mapping behave differently in latter case.

**style** : ***vector or key in*** `data`

Grouping variable that will produce lines with different dashes and/or markers. Can have a numeric dtype but will always be treated as categorical.

**data** : `pandas.DataFrame` , `numpy.ndarray` , ***mapping, or sequence***

Input data structure. Either a long-form collection of vectors that can be assigned to named variables or a wide-form dataset will be internally reshaped.

**palette** : ***string, list, dict, or*** `matplotlib.colors.Colormap`

Method for choosing the colors to use when mapping the `hue` semantic. String values are passed to `color_palette()` . L dict values imply categorical mapping, while a colormap object implies numeric mapping.

**hue_order** : ***vector of strings***

Specify the order of processing and plotting for categorical levels of the `hue` semantic.

**sizes** : ***list, dict, or tuple***

An object that determines how sizes are chosen when `size` is used. It can always be a list of size values or a dict mapping levels of the `size` variable to sizes. When `size` is numeric, it can also be a tuple specifying the minimum and maximum size to use such that other values are normalized within this range.

**size_order** : ***list***

Specified order for appearance of the `size` variable levels, otherwise they are determined from the data. Not relevant when the `size` variable is numeric.

**size_norm** : ***tuple or Normalize object***

Normalization in data units for scaling plot objects when the `size` variable is numeric.

**dashes** : ***boolean, list, or dictionary***

Object determining how to draw the lines for different levels of the `style` variable. Setting to `True` will use default dash codes, or you can pass a list of dash codes or a dictionary mapping levels of the `style` variable to dash codes. Setting to `False` will use solid lines for all subsets. Dashes are specified as in matplotlib: a tuple of `(segment, gap)` lengths, or an empty string to draw a solid line.

**markers** : ***boolean, list, or dictionary***

Object determining how to draw the markers for different levels of the `style` variable. Setting to `True` will use default markers, or you can pass a list of markers or a dictionary mapping levels of the `style` variable to markers. Setting to `False` will draw marker-less lines. Markers are specified as in matplotlib.

**style_order** : ***list***

Specified order for appearance of the `style` variable levels otherwise they are determined from the data. Not relevant when the `style` variable is numeric.

**units** : ***vector or key in*** `data`

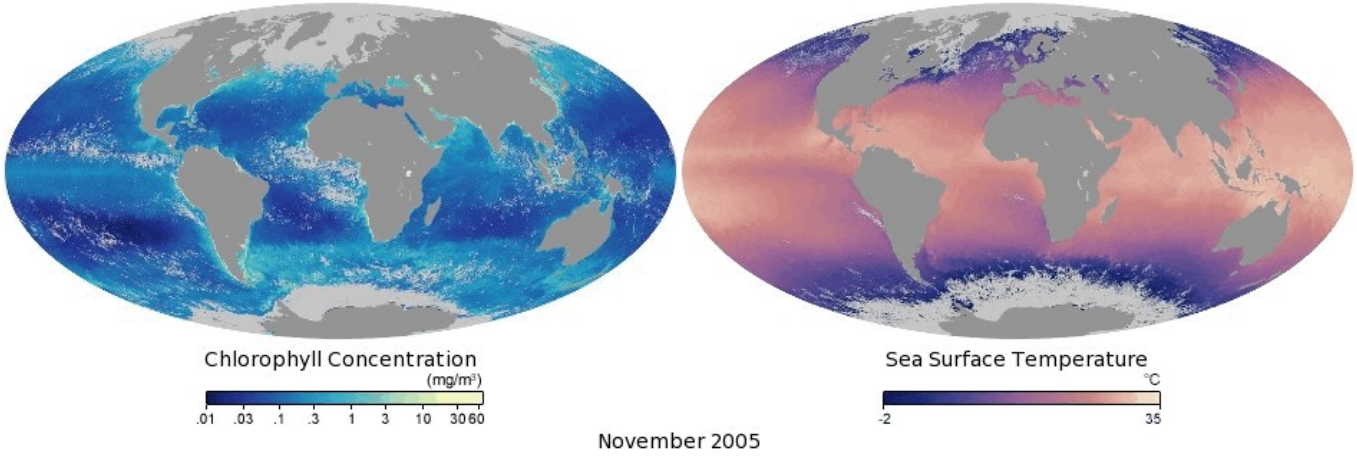
Grouping variable identifying sampling units. When used, a separate line will be drawn for each unit with appropriate semantics, but no legend entry will be added. Useful for showing distribution of experimental replicates when exact identities are not needed.

# Le Titanic

- Visualisation des informations sur le Titanic

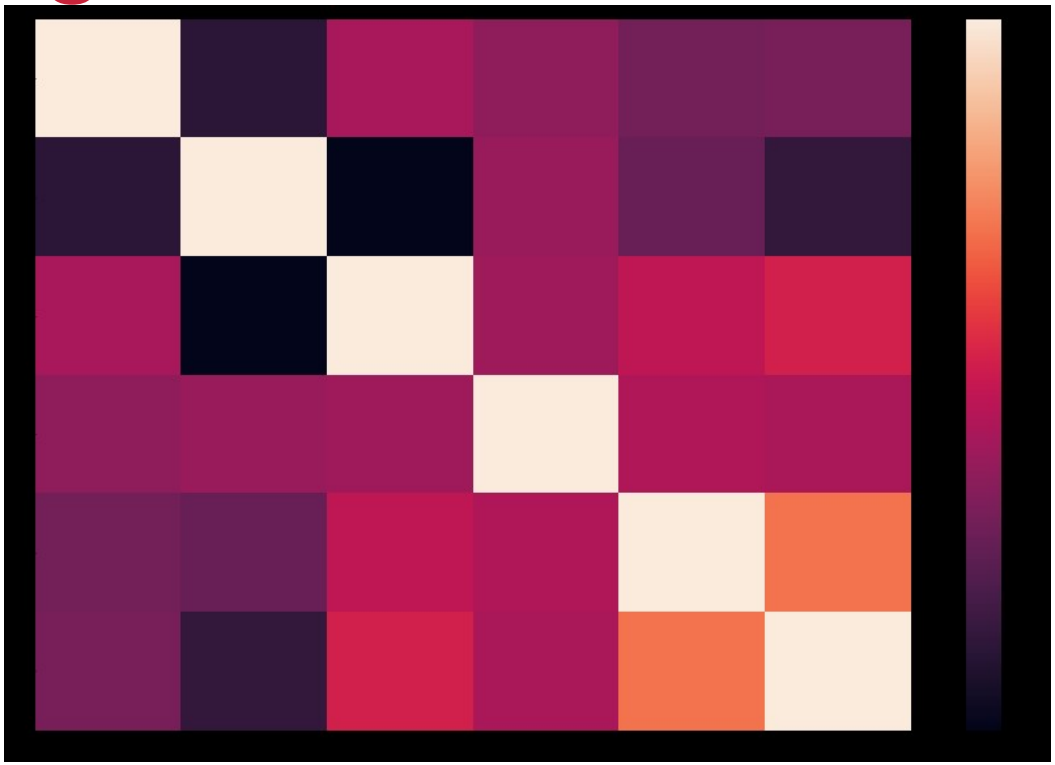
# Les Heatmaps (cartes de températures)

- Vient du domaine géographique afin de représenter des données en 2D




© Nasa

# Visualisation des données en 2D plutôt qu'en ligne



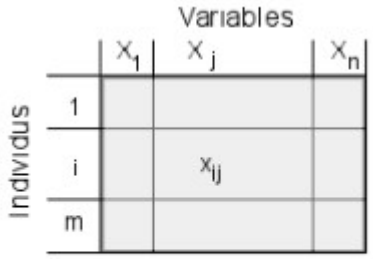
# Représentation matricielle des données

- Utilisée de manière implicite par exemple avec les DataFrame de pandas
- Outil mathématique très performant et très souple (rotation, translation,...)

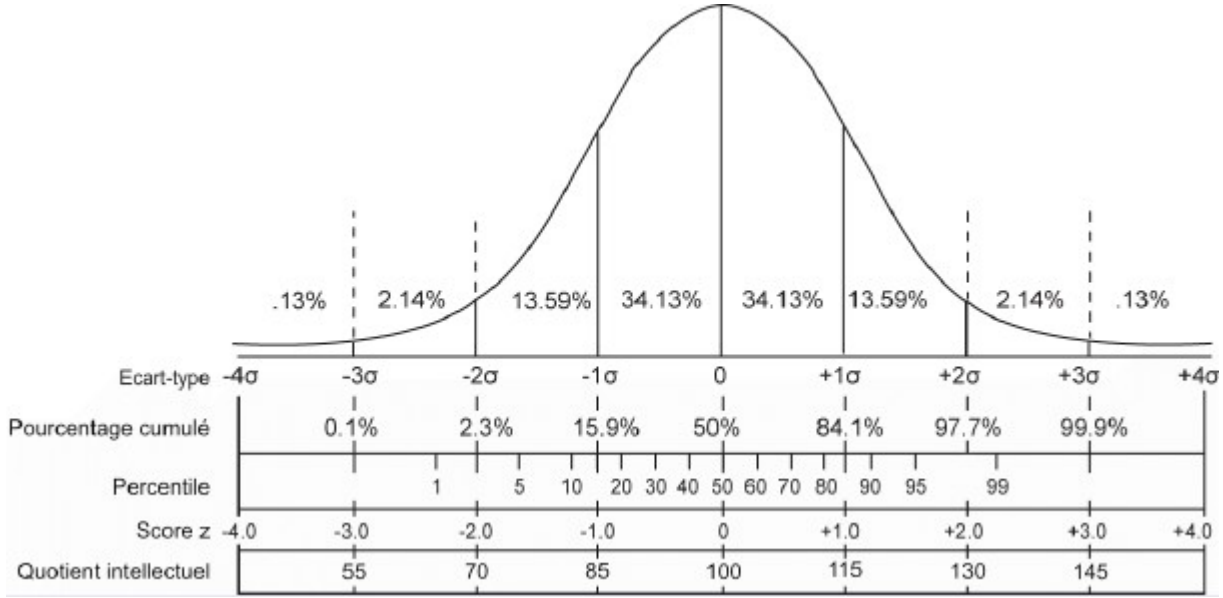
# Principe

- Plusieurs variables X1, X2, …, Xj pour j de 1 à n pour décrire le même individu/objet/observation
- La valeur de la variable j sur un individu se note $X_{ij}$

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{pmatrix}$$



# Quelques statistiques « classiques »



# Pour une variable x

- Échantillon {Xj} pour j de 1 à n
- Moyenne
- Variance de x
- Ecart-type

# Sur un échantillon avec 2 variables

- Étude des corrélations entre deux variables X et Y
  - prédire une variable en fonction de l'autre si elles sont corrélées
  - meilleure compréhension des données
  - la covariance est un outil permettant de calculer la corrélation

# Covariance

- Xi échantillon de la variable X
- X moyenne des observations de X
- Yi échantillon de la variable Y
- Y moyenne des observations de Y
- n nombre d'observations

# Covariance (II)

$$COV(X,Y) = \frac{\sum_{i=1}^{n}\left(X_i - \overline{X}\right)\left(Y_i - \overline{Y}\right)}{n-1}$$

# Exemple

| Température | Nombre de clients |
|---|---|
| 36° | 15 |
| 30° | 11 |
| 31,5° | 9 |
| 29° | 9 |
| 34,5° | 15 |
| 23° | 7 |

cf notebook

# Corrélation et covariance

- Ici la covariance est positive (21,5) ce qui indique que plus la température augmente plus le nombre de clients augmente
- La corrélation indique la force de la relation
  - σx écart-type de X
  - σy écart type de Y

$$Correlation = \frac{Cov\,(x,y)}{\sigma x * \sigma y}$$

# Plusieurs coefficients de corrélation

- Le coefficient le plus classique celui de Pearson, il calcule la force et la direction de l'association linéaire entre deux varianmes

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[\, n\sum x^2 - (\sum x)^2 \,][\, n\sum y^2 - (\sum y)^2 \,]}}$$

méthode corr dans pandas (cf Notebook)

# Dans pandas (corr)

- coefficient de pearson entre -1,0 et +1,0 (0 → aucune corrélation)
- mais aussi kendall, spearman..

# La matrice de covariance

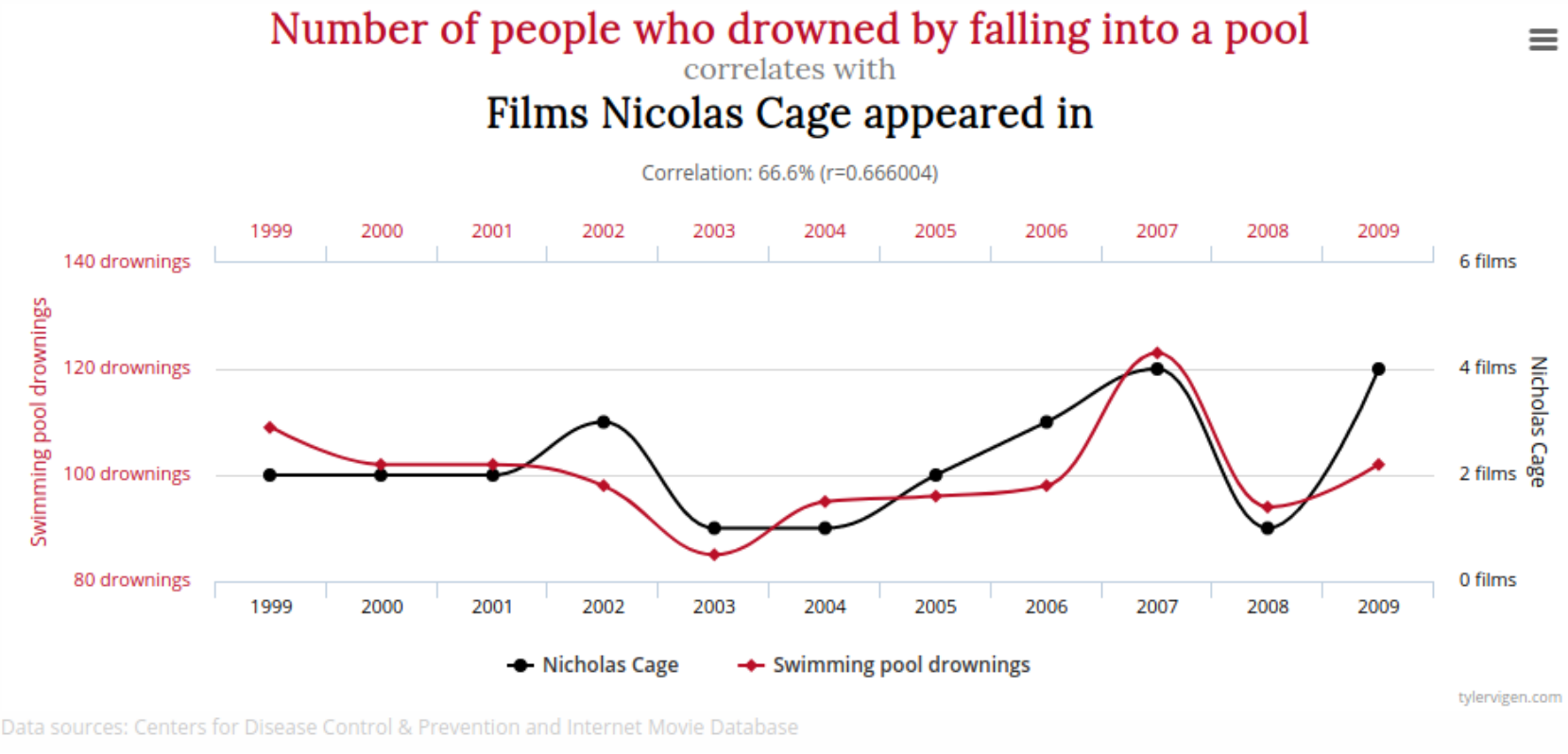
- On étend la notion de covariance avec un ensemble de mesures
- Par exemple, supposons que l'on ait 3 ensembles de mesures X, Y et Z
- La matrice de covariance serait obligatoirement symétrique et serait :

$$S = \begin{matrix} COV(X,X) & COV(X,Y) & COV(X,Z) \\ COV(Y,X) & COV(Y,Y) & COV(Y,Z) \\ COV(Z,X) & COV(Z,Y) & COV(Z,Z) \end{matrix}$$

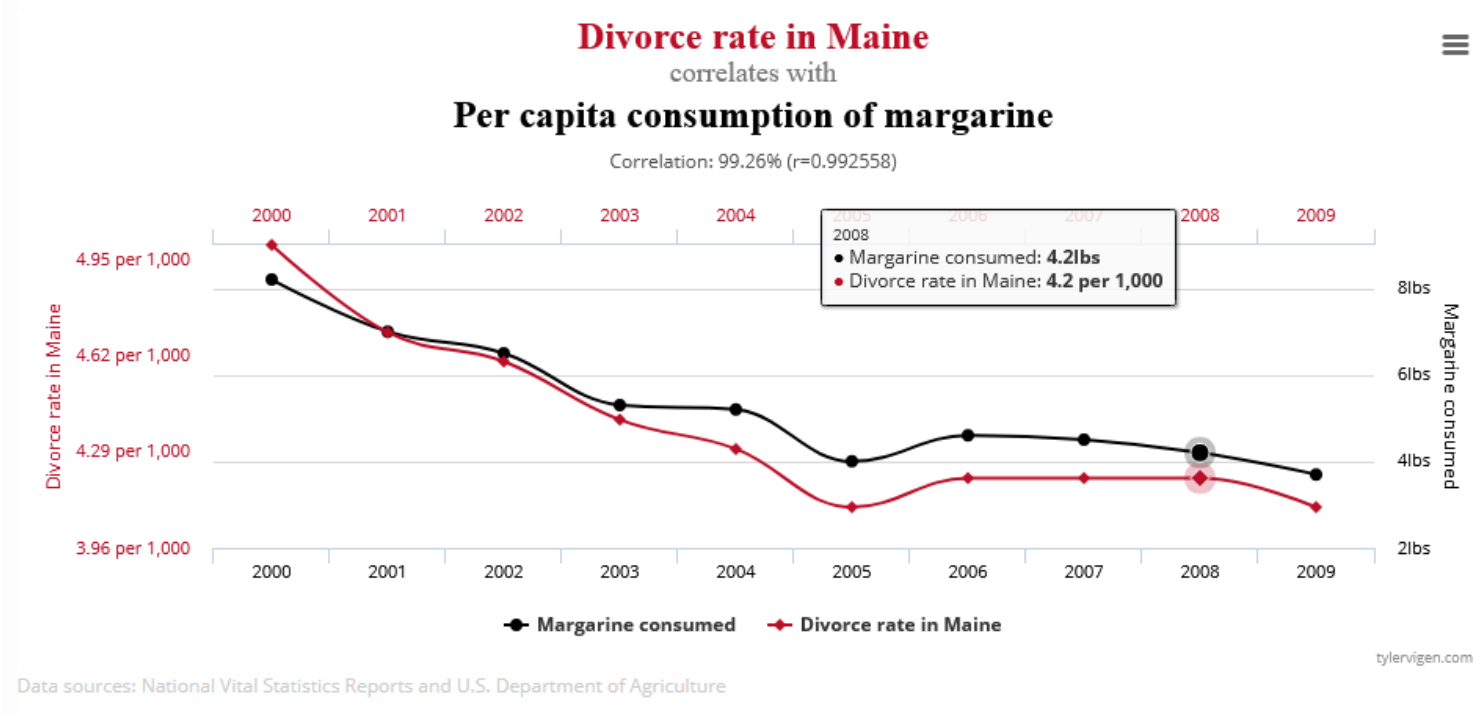
# La matrice de covariance (II)

- La diagonale est obligatoirement correspond à la variance de chaque variance
- Les autres lignes la covariance 2à2 entre les variables
- Cf notebook

# Des vraies corrélations



# Ou encore

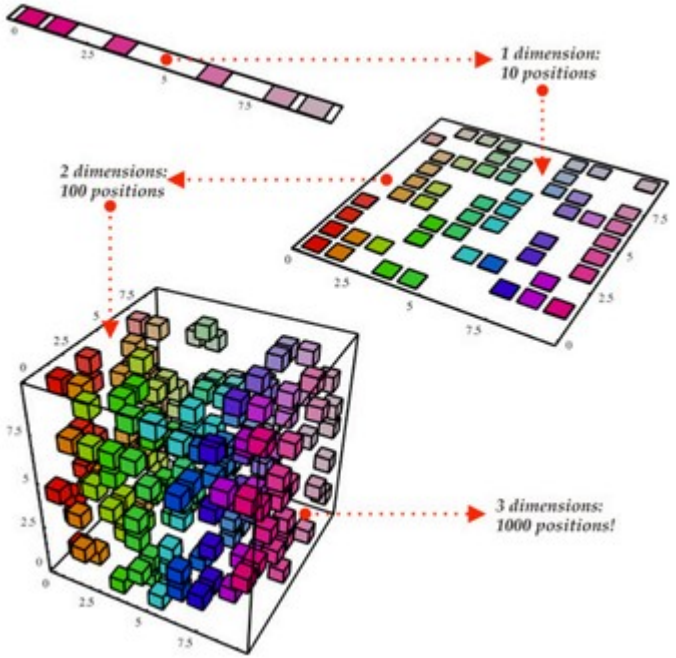

Divorce rate in Maine
correlates with
Per capita consumption of margarine
Correlation: 99.26% (r=0.992558)
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2008
Margarine consumed: 4.2lbs
Divorce rate in Maine: 4.2 per 1,000
4.95 per 1,000
4.62 per 1,000
4.29 per 1,000
3.96 per 1,000
Divorce rate in Maine
8lbs
6lbs
4lbs
2lbs
Margarine consumed
Margarine consumed
Divorce rate in Maine
tylervigen.com
Data sources: National Vital Statistics Reports and U.S. Department of Agriculture

# Analyse en composantes principales ou PCA

- Le principe de l'ACP (ou PCA) est une technique qui est souvent utilisée afin de diminuer la dimension d'ensemble de données de grande taille
- Les ensembles plus simples sont en général plus faciles à comprendre et analyser
- Parfois perte d'une précision
- Simplification vs Précision

# Historique

- Méthode relativement ancienne
  - Pearson (1901)
- Transformation de variables corrélées entre elles en nouvelles composantes décorrélées que l'on appellera les composantes principales


1 dimension:
10 positions
2 dimensions:
100 positions
3 dimensions:
1000 positions!

# L’ACP

- Chercher une représentation alternative vers un espace de dimension plus petit (idéal un espace de taille 2 ou 3)
- Les nouvelles variables dans cet espace sont des combinaisons linéaires des variables initiales
  - Composantes principales : les nouvelles variables
  - Axes principaux : les nouveaux axes de l’espace d’arrivée
  - Facteurs principaux : les formes linéaires

# Principe

- On va utiliser la fameuse matrice de covariance !
- Les étapes
  - Standardisation des données
  - Calcul de la matrice de covariance !
  - Calcul des valeurs propres pour identifier les composants principaux
  - Calcul du vecteur caractéristique
  - Repositionner les données sur les axes principaux

# Standardisation

- Phase critique (mais pas obligatoire)
- Pour éviter qu'une variable dont les valeurs varient entre 500 et 1000 masquent celles qui oscillent entre 0,5 et 1 !
-

$$z = \frac{value - mean}{standard\ deviation}$$

# Le calcul de la matrice de covariance !

- Comprendre les relations entre les variables (cf infra)
- Par exemple pour 3 variables

$$\begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix}$$

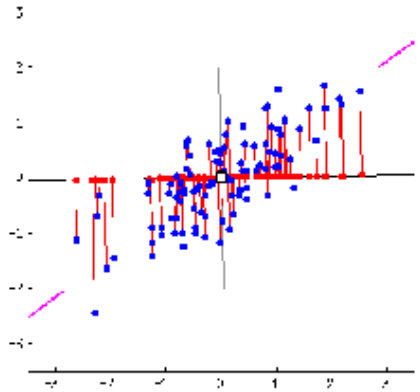
Note Cov(x,x) = Var(X)

# Calcul des vecteurs et valeurs propres de la matrice

- Permet de déterminer les composantes principales (combinaison linéaire des anciennes variables)
- Ces nouvelles variables sont décorrélées
- Le maximum d'informations dans la première composante

# Explications

- Trouvez l'axe qui capture le plus d'informations en minimisant la dispersion des points sur la droite
  - Distance euclidienne
  - et ainsi de suite pour axes



Source: stats.stackexchange.com

# Mathématiquement

- Les vecteurs propres de la matrice de covariance sont les composants principaux
- Les valeurs propres sont les coefficients des composants principaux (variance prise en charge)
- En ordonnant les valeurs propres par ordre décroissant, on obtient les composantes principales par ordre d'importance

# Une dernière étape

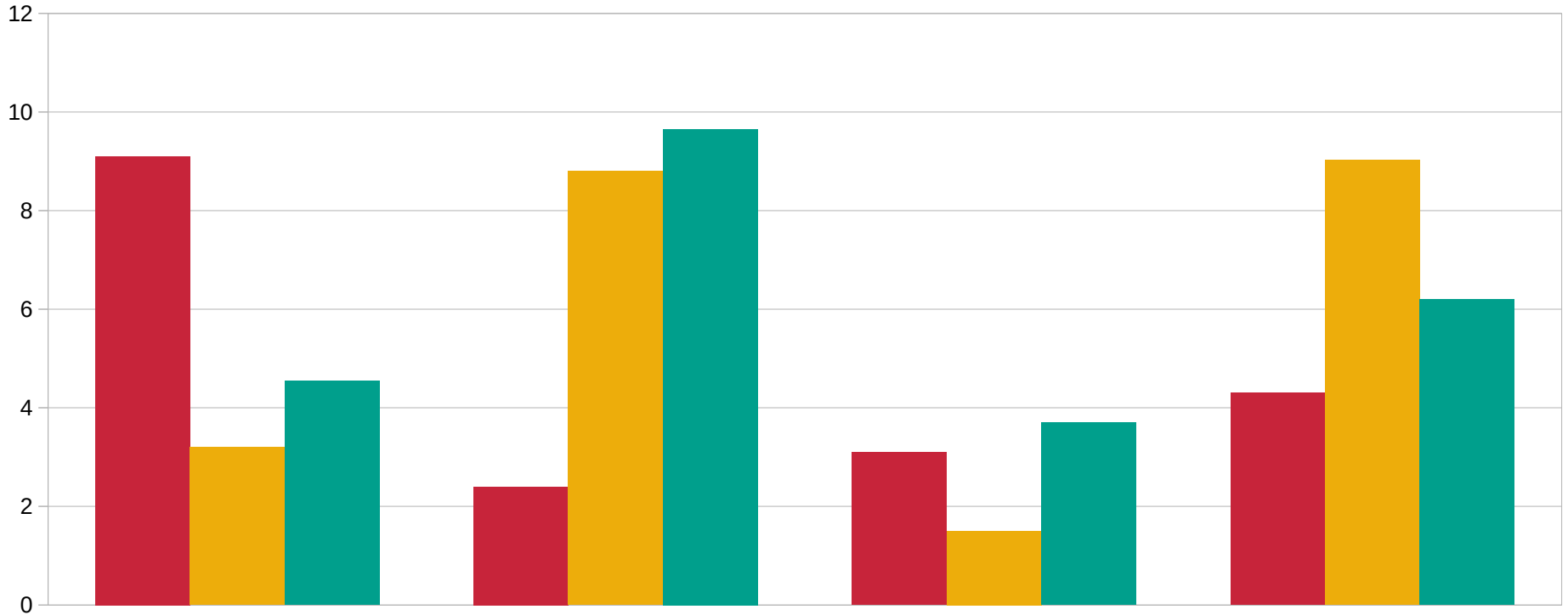
- Recalculer les données dans la nouvelle représentation
- Construire la matrice P dont les colonnes sont les vecteurs propres calculés
- La matrice ZP (où Z est la matrice normalisée) est la nouvelle représentation (les colonnes sont indépendantes des unes des autres)

# Résumé

Soit S un échantillon de données n individus (lignes) d variables colonnes

1) Normaliser les données
2) Calculer la matrice co-variance
3) Calculer les valeurs propres et les vecteurs propres
4) Prendre les k plus grandes valeurs propres et les axes associés
5) Calculer la nouvelle représentation de S

| | | | | |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |


12
10
8
6
4
2
0