femto-st SCIENCES & TECHNOLOGIES

M1 info., Initiation à la Recherche
Recherche Reproductibilité : introduction

Jean-François COUCHOT
Université de Franche-Comté

cnrs UBFC UNIVERSITÉ BOURGOGNE-FRANCHE-COMTÉ UNIVERSITÉ DE FRANCHE-COMTÉ SUP MICRO TECH ENSMM utbm université de technologie Belfort-Montbéliard

Plan

Crise de la reproductibilité

Notions de reproductibilité

Méthodes assurant cette reproductibilité

Inspirations

En plus des références ponctuelles, ce cours est inspiré de

- FUN MOOC : Recherche reproductible : principes méthodologiques pour une science transparente
- Livre : Antunes, B. A., & Hill, D. R. (2024). RECHERCHE REPRODUCTIBLE.

Plan

Crise de la reproductibilité

Notions de reproductibilité

Méthodes assurant cette reproductibilité

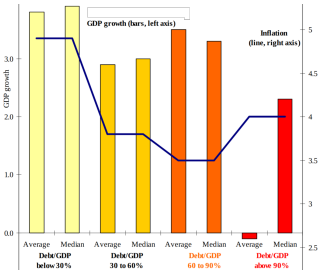
Un peu d'histoire de cette crise

2005 : "Most Research Findings Are False for Most Research Designs and for Most Fields" [1]

- Recherche scientifique est incertaine :
  - difficile de déterminer les causes de non-reproductibilité d'une expérience,
  - improbable d'atteindre un taux de validité des résultats de 100%

2010 : un ratio [2] entre dette publique et PIB à ne pas dépasser ?

- Justification de politiques d'austérité mondiales
- 2013 : Herndon, étudiant [3] (U. Massachusetts) : accède aux données et tableur
  - Des formules fausses dans le tableur
  - Des données disponibles (contredisant le message) écartées par les auteur · e · s
  - Après correction : croissance réduite à 2.2% (et pas -0.1%)



1. Ioannidis, J. P. (2005). Why most published research findings are false. PLoS medicine, 2(8), e124.
2. Reinhart, C. M., & Rogoff, K. S. (2010). Growth in a Time of Debt. American economic review, 100(2), 573-578.
3. Herndon, T., Ash, M., & Pollin, R. (2014). Does high public debt consistently stifle economic growth ? A critique of Reinhart and Rogoff. Cambridge journal of economics, 38(2), 257-279.

Quelques données chiffrées sur cette crise

2016 : Enquête générale [4] auprès de 1500 scientifiques

Discipline	% d'échec d'expé. de collègues	% d'échec de sa propre expé.
Chimie	90	60
Biologie	80	60
Physique et ingénierie	70	50
Médecine	70	60
Géologie et environnement	60	40

2016 : Reproductibilité en science informatique [5]

- 601 articles de conférences ACM de 2012 (de qualité ?)
- 217 résultats reproduits (seulement)

4. Baker, M. (2016). 1,500 scientists lift the lid on reproducibility. Nature, 533(7604).
5. Collberg, C., & Proebsting, T. A. (2016). Repeatability in computer systems research. Communications of the ACM, 59(3), 62-69.

Facteurs possibles à l'"origine de cette crise

Méthodologiques

- P-hacking : résultats artificiellement significatifs par manipulation des données/analyses statistiques
- Publication biaisée : tendance à publier principalement les résultats positifs
- Petites tailles d'échantillons : peuvent conduire à des résultats non robustes, difficiles à reproduire.

Contexte de la recherche

- Pression de publier dans des revues à fort impact : encourage à prendre des raccourcis méthodologiques
- Incitation à l'innovation : peut conduire à des études moins rigoureuses
- Complexité croissante des méthodes
- Le non-partage des données, des codes et des protocoles expérimentaux : limite la possibilité de vérifier les résultats.

Humains

- Biais cognitifs des chercheur · e · s : sujets à des biais cognitifs qui peuvent influencer leurs interprétations des résultats.

Plan

Crise de la reproductibilité

Notions de reproductibilité

Méthodes assurant cette reproductibilité

Complexité de la reproductibilité

Terminologie floue

- Multiplicité des termes utilisés : replicabilité, repétabilité,test, robustesse
- Diversité des sens selon les communautés scientifiques.

Questions liées à la reproductibilité

- Qui peut reproduire ? soi-même, un collègue, etc.
- Dans quel but ? valider, contredire, interpréter
- Comment reproduire ? même instrumentation, protocole, etc.
- Qu'est-ce qui doit être identique ? mesures,
- Quand la reproductibilité est-elle nécessaire ? vérification, démonstration, etc.

Reproductibilité : hypothétique ou effective ?

- (-) Tendance : la valeur d'une publication réside dans son originalité
- (-) Motivation très réduite à reproduire les expériences des autres
- (+) Citée comme "la moindre des choses" dans les bonnes pratiques scientifiques
- (+) Souvent brandie comme un principe moral indiscutable

Répétabilité, Reproductibilité, Réplicabilité

Un besoin de formalisation

- Jusqu'en 2006 : trois termes interchangeables, équivalents
- 2006 : distinction [6] entre réplicable et reproductible en épidémiologie
- 2009 : réplicabilité prise pour répétable [7]

Définition actuelle [8] en sciences informatiques

- **Répétabilité** : expérience réalisée par la même équipe dans le même environnement (logiciel/matériel) et avec le même code/algorithme
- **Reproductibilité** : expérience réalisée par une équipe différente dans un environnement et avec un équipement différents, mais en utilisant le même code/algorithme
- **Réplicabilité** : expérience réalisée par une équipe différente dans un environnement et avec un équipement différents, en utilisant un code différent, mais en conservant le même algorithme.

6. Peng, R. D., Dominici, F., & Zeger, S. L. (2006). Reproducible epidemiologic research. American journal of epidemiology, 163(9), 783-789.

7. Drummond, C. (2009, June). Replicability is not reproducibility : nor is it good science. In Proceedings of the Evaluation Methods for Machine Learning Workshop at the 26th ICML (Vol. 1). Montreal, Canada : National Research Council of Canada

8. Association For Computing Machinery `https://www.acm.org/publications/policies/artifact-review-and-badging-current`

Les 6 catégories de reproductibilié [9]

- Repro. **computationnelle** : obtenir les mêmes résultats finaux qu'une étude initiale en utilisant uniquement les données et les instructions fournies
- Repro. **expériment[a] directe** : dans des domaines aux conditions expérimentales très contrôlées, vise à reproduire des patterns de données ($\approx$ valeurs exactes) en utilisant la statistique pour validation
- Repro. **indirecte** : dans des domaines où le contrôle expérimental est limité (biologie, psychologie), repose sur la convergence de multiples lignes de preuve indépendantes, (pas nécessairement une répétition d'expériences standardisées)
- Repro. **par expertise** : en sciences du "rare", repose sur l'habileté d'un expert à obtenir des résultats similaires dans des conditions uniques et irréproductibles
- Repro. **de l'observation** : repose sur l'expertise du chercheur à obtenir des résultats similaires dans des contextes spécifiques, en utilisant des méthodes rigoureuses et des outils d'analyse adaptés
- Le reste : la recherche **non-reproductible** qui privilégie la subjectivité et la contextualisation

9. Leonelli, S. (2018, October). Rethinking reproducibility as a criterion for research quality. In Including a symposium on Mary Morgan : curiosity, imagination, and surprise (pp. 129-146). Emerald Publishing Limited.

Reproductibilité computationnelle difficile

- Interdépendance des logiciels : calculs très sensibles aux changements de versions ou d'environnements, aux nombreuses bibliothèques et outils utilisé · e · s
- Manque de reconnaissance : le travail de développement n'est généralement pas assez valorisé dans la publication scientifique
- Manque de compétences : de nombreux chercheurs ne disposent pas des compétences nécessaires en informatique pour gérer efficacement la complexité des environnements de calcul

Plan

Crise de la reproductibilité

Notions de reproductibilité

Méthodes assurant cette reproductibilité

- Document computationnel
- Gestion de l'aléa

Objectifs

- Code enrelacé de commentaires, de figure, d'equations mathématiques
  - programmation lettrée (Knuth 1984) ou document computationnel
- Gestion de l'aléa
- Suivi des versions pour pouvoir regénérer une version spécifique (celle d'un article p. ex.)
  - De Git (Github, Gitlab) à Zénodo ou Software Heritage pour l'archivage
- Mémoriser l'environnement d'exécution du code :
  - Conteneur ou machines virtuelles

Plan

Crise de la reproductibilité

Notions de reproductibilité

Méthodes assurant cette reproductibilité
- Document computationnel
- Gestion de l'aléa

Notebook Jupyter en image

jupyter reproductibilite Last Checkpoint: 4 minutes ago

File Edit View Run Kernel Settings Help — Trusted

Markdown — JupyterLab — Python 3 (ipykernel)

Classification des données du diabete à l'aide d'un arbre de décision basé sur l'entropie

Import des bibliothèques suffisantes

+ 1 cell hidden

Télechargement du fichier et conversion en un dataframe

```
[2]: df = pd.read_csv('https://members.femto-st.fr/jf-couchot/sites/femto-st.fr.jf-couchot/files/content/diabetes.txt')
```

Prétraitements sur les données

+ 4 cells hidden

Mise en place du modèle de ML

La profondeur est arbitrairement fixée à 5

```
[5]: clf = DecisionTreeClassifier(criterion="entropy",max_depth=5)
```

Première évaluation du modèle

Pour une profondeur donnée **égale à 5**, on construit un StratifiedShuffleSplit qui va proposer **20 découpages 0.8/0.2** avec une graine originale égale à **42**.

Sur ces 20 évaluations, la moyenne et l'écart type du **score F1** (voir wikipedia) sont calculés puis affichés.

Rappel : le F1 score est donné par la formule $\frac{2TP}{2TP+FP+FN}$

où:

- TP : nombre de True Positive

jupyter reproductibilite Last Checkpoint: 5 minutes ago

File Edit View Run Kernel Settings Help — Trusted

Markdown — JupyterLab — Python 3 (ipykernel)

Recherche de paramètre optimisant la classification selon le F1 Score

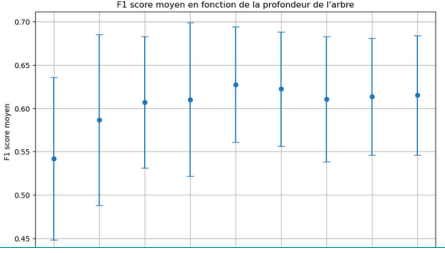
Pour chaque profondeur entre 2 et 10, on calcule cette moyenne et cet écart type, puis on affiche ceci sous la forme de points.

```
[7]: depths = range(2, 11)
mean_accuracies = []
std_deviations = []

for depth in depths:
    clf = DecisionTreeClassifier(criterion="entropy", max_depth=depth)
    scores = cross_val_score(clf, X, y, cv=cv, scoring='f1')
    mean_accuracies.append(np.mean(scores))
    std_deviations.append(np.std(scores))

# Create the boxplot
plt.figure(figsize=(10, 6))
plt.errorbar(depths, mean_accuracies, yerr=std_deviations, fmt='o-', linestyle='None', capsize=5)
plt.xlabel("Profondeur de l'arbre")
plt.ylabel("F1 score moyen")
plt.title("F1 score moyen en fonction de la profondeur de l'arbre")
plt.grid(True)
plt.show()
```

```
/tmp/ipykernel_30649/3394767657.py:13: UserWarning: linestyle is redundantly defined by the 'linestyle' keyword a
rgument and the fmt string "o-" (-> linestyle='-'). The keyword argument will take precedence.
  plt.errorbar(depths, mean_accuracies, yerr=std_deviations, fmt='o-', linestyle='None', capsize=5)
```



Document computationnel

Entrelacement de cellules

- contenant du texte qui
  - définit la structure du documents (les sections, sous-sections...)
  - définit/rappelle des notions (eventuellement avec des formules mathématiques)
  - fournit des analyses des résultats numériques
- contenant du code (naturellement commenté) qui
  - implante une nouveauté (algorithme)
  - effectue des analyses, construit des figures (à insérer dans l'article)
- contenant des images

Formatage du texte à l'aide de la syntaxe Markdown

Structure du document

```
# Un  titre de section 1
## Un titre de sous-section
### Un titre de sous-sous-section
```

Styles de texte

```
_italique_, **gras**, **_gras-italique_**, ~~barré~~
```

Liens

```
la moyenne et l'écart type du score [F1](https://en.wikipedia.org/wiki/F-score) sont calculés
```

Listes

Une liste non ordonnée :

```
* une élément
* un autre
  * un sous élément
  * un autre sous élément
* un dernier élément
```

Une liste ordonnée :

```
1. élément un
2. élément deux
```

Formules mathématiques

`Cette expression $\sum_{i=1}^n X_i$ est inlinée.` $\rightsquigarrow$ Cette expression $\sum_{i=1}^{n} X_i$ est inlinée.

Code dans un notebook–1

Classification des données du diabete à l'aide d'un arbre de décision basé sur l'entropie

Import des bibliothèques suffisantes

[1]:
```
import matplotlib
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import cross_val_score, StratifiedShuffleSplit, train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
import numpy as np
```

Télechargement du fichier et conversion en un dataframe

[2]:
```
df = pd.read_csv('https://members.femto-st.fr/jf-couchot/sites/femto-st.fr.jf-couchot/files/cont
```

Prétraitements sur les données

Binarisation de la sortie

[3]:
```
df['Outcome'] = df['Outcome'].map({'NO': 0, 'YES': 1})
X = df.iloc[:, 0:8]
y = df.iloc[:, 8]
```

Traitement des valeurs abusivement nulles ou manquantes

Pour chaque attribut où il manque des données et où certaines données sont nulles alors qu'elles ne devraient pas l'être, remplacement de ces valeur par la moyenne des valeurs de l'attribut.

Lorsque la donnée est entrière, on tronque en entier (la troncature est une source potentielle d'erreur)

Organisation et Structure

- ordre logique : cellules structurées (section, sous-section. . .) suivant un flux logique
- une cellule, une tâche
- commentaires clairs et explicites : but de chaque bloc de code, choix algorithmiques, résultats intermédiaires
- Markdown pour la documentation

Code dans un notebook–2

Recherche de paramètre optimisant la classification selon le F1 Score

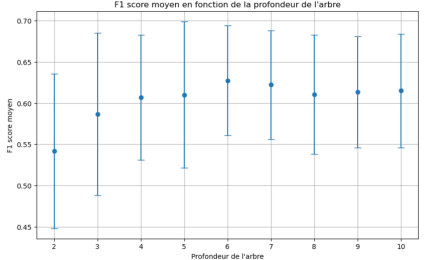
Pour chaque profondeur entre 2 et 10, on calcule cette moyenne et cet écart type, puis on affiche ceci sous la forme de points.

```
[7]: depths = range(2, 11)
     mean_accuracies = []
     std_deviations = []

     for depth in depths:
         clf = DecisionTreeClassifier(criterion="entropy", max_depth=depth)
         scores = cross_val_score(clf, X, y, cv=cv, scoring='f1')
         mean_accuracies.append(np.mean(scores))
         std_deviations.append(np.std(scores))

     # Create the boxplot
     plt.figure(figsize=(10, 6))
     plt.errorbar(depths, mean_accuracies, yerr=std_deviations, fmt='o-', linestyle='None', capsize=5)
     plt.xlabel("Profondeur de l'arbre")
     plt.ylabel("F1 score moyen")
     plt.title("F1 score moyen en fonction de la profondeur de l'arbre")
     plt.grid(True)
     plt.show()
```

```
/tmp/ipykernel_30649/3394767657.py:13: UserWarning: linestyle is redundantly defined by the 'linestyle' ke
yword argument and the fmt string "o-" (-> linestyle='-'). The keyword argument will take precedence.
  plt.errorbar(depths, mean_accuracies, yerr=std_deviations, fmt='o-', linestyle='None', capsize=5)
```



Sur la courbe précédente, le f1 score semble croissant jusqu'à une profondeur de 7, pour une décroissance ensuite. Cela semble ocrrespondre à un sur-apprentissage.

Synthèses et analyses

- Figures : construites dans le document maître (pas dans libreoffice)
- Analyse des résultats : détaillées, en relation avec les figures

Plan

Crise de la reproductibilité

Notions de reproductibilité

Méthodes assurant cette reproductibilité
- Document computationnel
- Gestion de l'aléa

Pseudo-aléa et tests

- Tests unitaires : génération de données d'entrée pseudo-aléatoires pour tester la robustesse/fiabilité de parties logicielles (code à droite)
- Tests de performance : génération de données d'entrée pseudo-aléatoires pour mesurer les performances d'algorithmes (tri, recherche,...)

```python
import unittest
from numpy import random

def ma_fonction(x, y):
    """Fonction à tester: une addition"""
    return x + y

class TestMaFonction(unittest.TestCase):
    def test_addition(self):
        for _ in range(100):  # 100 tests aléatoires
            a = random.randint(0, 100)
            b = random.randint(0, 100)
            self.assertEqual(ma_fonction(a, b), a + b)

if __name__ == '__main__':
    unittest.main()
```

Pseudo-aléa et apprentissage machine

- Génération de données d'entraînement "fictives" pour l'entraînement
- Initialisation aléatoire des poids d'un Réseau de Neurones, de centroïdes pour des clusters
- Partitionnement pseudo-aléatoire des datasets (code à droite)

```python
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

iris = load_iris()
X, y = iris.data, iris.target
X_train,X_test,y_train,y_test = train_test_split(X,y,
                                                 test_size=0.2)

model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
#Accuracy: 0.9666666666666667
```

Problèmes avec le pseudo-aléa

```python
import tensorflow as tf
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
iris = load_iris()
X, y = iris.data, iris.target
encoder = LabelEncoder()
y = encoder.fit_transform(y)
X_train,X_test,y_train,y_test=train_test_split(X,y,
                                               test_size=0.2,
                                               random_state=42)
for i in range(5):
    model = tf.keras.Sequential([
        tf.keras.layers.Dense(16, activation='relu', input_shape=(4,)),
        tf.keras.layers.Dense(3, activation='softmax')])
    model.compile(optimizer='adam',
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])
    model.fit(X_train, y_train, epochs=10, batch_size=32, verbose=0)
    test_loss, test_acc = model.evaluate(X_test, y_test, verbose=2)
    print(f"Essai {i+1}: Accuracy = {test_acc}")
```

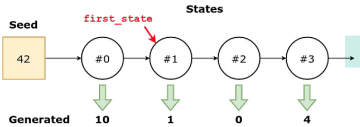
- Exécutions successives avec des résultats différents (sensibilité aux conditions initiales)

```
1/1 - 0s - 252ms/step - accuracy: 0.3000 - loss: 1.4606
Essai 1: Accuracy = 0.30000001192092896
1/1 - 0s - 253ms/step - accuracy: 0.3000 - loss: 3.1048
Essai 2: Accuracy = 0.30000001192092896
1/1 - 0s - 243ms/step - accuracy: 0.2333 - loss: 1.1109
Essai 3: Accuracy = 0.23333333432674408
1/1 - 0s - 250ms/step - accuracy: 0.3667 - loss: 1.3953
Essai 4: Accuracy = 0.36666667461395264
1/1 - 0s - 251ms/step - accuracy: 0.4333 - loss: 1.7283
Essai 5: Accuracy = 0.4333333373069763
```

Pseudo Random Number Generator

Intuition sur les PRNG

Etant donné une configuration interne, un PRNG fournit une sortie et modifie sa configuration interne.



Source [10].

Illustration au moyen de code

```
import random                                          654
def eval_random_state():                               114
    random.seed(42)                                    654
    state0 = random.getstate()
    print(random.randint(0,1000))                      654
    state1 = random.getstate()                         114
    print(random.randint(0,1000))                      654
    random.setstate(state0)
    print(random.randint(0,1000),"\n")

eval_random_state()
eval_random_state()
```

10. https://towardsdatascience.com/random-seeds-and-reproducibility-933da79446e3

Exemple complet avec gestion de l'aléa

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score, StratifiedShuffleSplit
from sklearn.datasets import load_wine
import matplotlib.pyplot as plt
import numpy as np

data = load_wine()
X, y = data.data , data.target

clf = DecisionTreeClassifier(criterion="entropy",random_state=0,max_depth=5)
cv = StratifiedShuffleSplit(n_splits=5, test_size=0.2, random_state=0)
scores = cross_val_score(clf, X, y, cv=cv, scoring='accuracy')
mean_score, std_deviation = np.mean(scores), np.std(scores)

print(f"Scores de valid. croisée: {scores}")
print(f"précision moyenne: {mean_score:.4f}")
print(f"Écart type des précisions: {std_deviation:.4f}")

clf = DecisionTreeClassifier(criterion="entropy",max_depth=5)
clf.fit(X, y)

index = 10 # ligne arbitraire
xpl = X[index:index+1]
prediction = clf.predict(xpl)
print("Exemple :", xpl, ", Prédiction :", prediction)
print("Vraie classe :", y[index])
```

```
$ python3 decision2.py
Scores de valid. croisée: [0.91666667 0.9444
0.91666667 0.86111111 0.97222222]
précision moyenne: 0.9222
Écart type des précisions: 0.0369
Exemple : [[1.41e+01 2.16e+00 2.30e+00 1.80e
1.05e+02 2.95e+00 3.32e+00 2.20e-01
2.38e+00 5.75e+00 1.25e+00 3.17e+00 1.51e+03
Vraie classe : 0
```


```
$ python3 decision2.py
Scores de valid. croisée: [0.91666667 0.9444
0.91666667 0.86111111 0.97222222]
précision moyenne: 0.9222
Écart type des précisions: 0.0369
Exemple : [[1.41e+01 2.16e+00 2.30e+00 1.80e
1.05e+02 2.95e+00 3.32e+00 2.20e-01
2.38e+00 5.75e+00 1.25e+00 3.17e+00 1.51e+03
```

- Dans chaque partie du code, essayer de retrouver toutes les parties aléatoires
- `random_state`, `seed`,...