# Devoir 3

Distribution des points:

| Section | Fichiers requis | Score |
|---|---|---|
| Reddit Weekends | `reddit_weekends.py` | 35 |
| + graphiques, réponses courtes | `reddit_weekends.ipynb` | 10 |
| Chess Ratings | `chess_ratings.py` | 35 |
| + graphiques, réponses courtes | `chess_ratings.ipynb` | 20 |

En général vos devoirs seront auto-notés, c'est-à-dire que vous ne devez **pas modifier la signature des fonctions définies** (mêmes entrées et sorties). Pour les questions impliquant Numpy et Pandas, vous **devez** gérer toute itération via des appels aux fonctions pertinentes dans les bibliothèques. Cela signifie que vous ne pouvez pas utiliser de Python natif pour les boucles, les compréhensions de liste, etc. lorsque quelque chose aurait pu être fait avec un appel natif pandas/numpy. Les boucles sont autrement autorisées.

### Soumission

Pour soumettre les fichiers, veuillez soumettre **uniquement les fichiers requis** (énumérés dans le tableau ci-dessus) que vous avez complétés dans **gradescope**. N'incluez pas de données ou d'autres fichiers divers. Vous n'avez pas besoin de soumettre tous les fichiers en même temps pour exécuter l'autograder. Par exemple, si vous n'avez terminé que `reddit_weekends.py`, vous n'avez pas besoin de soumettre le reste des fichiers pour obtenir des commentaires sur `reddit_weekends.py`.

## 1. Week-ends Reddit

Cette question utilise des données dérivées de l'archive Reddit Comment, qui est une collection de chaque commentaire Reddit, distribué sous forme de 150 Go de JSON compressé.

Le fichier fourni `reddit-counts.json.gz` contient un décompte du nombre de commentaires publiés quotidiennement dans chaque subreddit de province canadienne et dans r/canada lui-même. Encore une fois, le format est du JSON compressé ligne par ligne. Il s'avère que Pandas (≥0.21) peut gérer la compression automatiquement et nous n'avons pas besoin de décompresser explicitement : ``

```
counts = pd.read_json(path, lines=True)
```

Nous nous posons la question suivante: y a-t-il un nombre différent de commentaires Reddit publiés les jours de semaine que les jours de fin de semaine ?

Pour cette question, nous examinerons uniquement les valeurs en 2012 et 2013 et dans le subreddit r/canada. Commencez par créer un DataFrame avec les données fournies et séparez les jours de la semaine des week-ends.

Indice : utilisez `datetime.date.weekday`. Les valeurs égalent à 5 sont le samedi et 6 sont le dimanche.

Complétez le programme `reddit_weekends.py` pour cette question et suivez le notebook `reddit_weekends.ipynb`. Des notes vous seront attribuées pour votre travail dans le fichier python, ainsi que pour les chiffres et les réponses courtes dans le notebook. Le fichier python a également une section de code à exécuter. Vous pouvez l'exécuter via la ligne de commande avec la commande suivante :

```
python3 reddit_weekends.py ./data/reddit-counts.json.gz
```

Notez que la sortie produite par la commande ci-dessus ne sera pas marquée. Nous testerons vos fonctions directement via des tests unitaires.

### 1.1 Test T de Student

- Complétez `reddit_weekends.py:process_data()`
- Complétez `reddit_weekends.py:tests()`

Utilisez scipy.stats pour effectuer un test T sur les données afin d'obtenir une valeur p. Pouvez-vous conclure qu'il y a un nombre différent de commentaires en semaine par rapport au week-end ? Essayez `stats.normaltest` pour voir si les données sont distribuées normalement, et `stats.levene` pour voir si les deux ensembles de données ont des variances égales. Pensez-vous maintenant pouvoir tirer une conclusion ? (Indice : non, nous obtenons un "0,0438".)

### 1.2 Solution 1: transformation des données

- Complétez les graphiques dans `reddit_weekends.ipynb`

Jetez un oeil à l'histogramme des données. Vous remarquerez qu'il est biaisé : c'est la raison pour laquelle il n'était pas distribué normalement dans la dernière partie. Transformez les décomptes afin que les données n'échouent pas au test de normalité. Options probables pour les transformations : np.log, np.exp, np.sqrt, counts**2. Choisissez celle qui se rapproche le plus d'une distribution normale. (Indice: Aucun d'entre elles ne passera le test de normalité.)

### 1.3 Solution 2 : théorème central limite

- Complétez `reddit_weekends.py:central_limit_theorem()`
- Complétez les graphiques dans `reddit_weekends.ipynb`

Le théorème central limite dit que si nos nombres sont suffisamment grands et que nous examinons les moyennes de l'échantillon, alors le résultat devrait être normal. Nous allons combiner tous les jours de semaine et de fin de semaine de chaque paire année/semaine et nous allons prendre la moyenne de leurs décomptes (non transformés).

Indice : vous pouvez obtenir une "année" et un "numéro de semaine" à partir des deux premières valeurs renvoyées par date.isocalendar(). Cette année et ce numéro de semaine vous donneront un identifiant pour la semaine. Utilisez Pandas pour regrouper par cette valeur et agréger en prenant la moyenne. Par contre, l'année renvoyée par isocalendar n'est pas toujours la même que l'année de la date. Utilisez l'année de l'isocalendar, qui est correcte dans ce cas.

Vérifiez ces valeurs pour voir s'ils ont une distribution gaussienne et une variance égale. Appliquez un test T si cela a du sens.

Nous devrions noter que nous modifions subtilement la question ici. C'est maintenant quelque chose comme "le nombre de commentaires le week-end et les jours de semaine pour chaque semaine diffère-t-il?"

### 1.4 Solution 3 : test non paramétrique

- Complétez `reddit_weekends.py:mann_whitney_u_test()`

Une autre option est un test statistique qui ne se soucie pas autant de la forme de son entrée. Le test U de Mann-Whitney ne suppose pas de valeurs distribuées normalement ni que la variance des deux distributions sont égales.

Effectuez un test U sur les décomptes (initiaux non transformés, non agrégés). Notez que nous devrions faire un test bilatéral. Assurez-vous que les arguments de la fonction sont corrects.

Notez que nous modifions subtilement la question à nouveau. Si nous parvenons à une conclusion à cause d'un test U, c'est quelque chose comme "il n'est pas également probable qu'il y ait un plus grand nombre de commentaires la fin de semaine par rapport aux jours de semaine".

### 1.5 Questions

Répondez à ces questions dans la section spécifiée dans `reddit_weekends.ipynb` :

- Laquelle des quatre transformations suggérées vous rapproche le plus de satisfaire les hypothèses d'un test T ?
- Nous avons donné des explications imprécises en mots de ce que le test hebdomadaire et le test de Mann-Whitney testaient réellement. Faites de même pour le test T d'origine et pour le test T des données transformées. Autrement dit, décrivez quelle serait la conclusion si vous pouviez rejeter l'hypothèse nulle de ces tests.
- Parmi les quatre approches, laquelle, selon vous, réussit le mieux à obtenir une réponse à la question initiale : "Y a-t-il un nombre différent de commentaires Reddit publiés la semaine et la fin de semaine ?" Expliquez brièvement pourquoi. (Il n'est pas clair qu'il y a une seule réponse correcte à cette question, mais il y en a des mauvaises !)
- Quand y a-t-il plus de commentaires Reddit publiés dans r/canada en moyenne, les jours de semaine ou les week-ends ?

## 2. Classement aux Échecs

Lors de la comparaison des performances entre deux groupes, une erreur courante consiste à les comparer en fonction des meilleurs de chaque groupe, en particulier dans les scénarios dans lesquels l'un des groupes a un nombre d'échantillons significativement plus élevé que l'autre (surreprésenté). Un tel exemple est aux échecs, où les gens ont essayé incorrectement de prétendre que les joueuses sont pires que les joueurs parce que si on considère les meilleurs joueurs, aucune femme n'a été championne du monde, ou parce que la différence d'Elo (une méthode pour calculer la compétence relative niveaux de joueurs) entre la meilleure femme et le meilleur homme est grande.

Pourquoi est-ce une erreur? Puisque ça néglige le taux de participation de ces groupes. Naïvement, si deux groupes sont tirés de la même distribution, le groupe surreprésenté aura plus de chances d'être dans les queues de la distribution. Par exemple, disons qu'il y a deux groupes de personnes : A et B. Le groupe A compte 10 personnes, le groupe B en compte 2. Chacune des 12 personnes se voit attribuer au hasard un numéro entre 1 et 100 (avec remplacement). Ensuite, prenez le maximum du groupe A comme score du groupe A et le maximum du groupe B comme score du groupe B. En moyenne, le groupe A obtiendra un score d'environ 91 et le groupe B d'environ 67, où la seule différence entre les deux groupes est la taille. Le plus grand groupe a plus de chances d'avoir un score élevé, donc obtiendra en moyenne un score plus élevé. La manière équitable de comparer ces groupes de taille inégale consiste à comparer leurs moyennes (moyennes), et non leurs valeurs maximales. Bien sûr, dans cet exemple, ce serait 50 pour les deux groupes – pas de différence ! Notez que cela ne fait que souligner l'effet statistique de la simple comparaison des groupes surreprésentés et sous-représentés et néglige tout problème potentiel ou autre biais dans les données.

Dans cette section, nous explorerons cela dans le contexte des classements d'échecs mentionnés précedemment. Aucun code pour vous aider n'est fourni dans le fichier python ; à la place, vous pouvez suivre dans le cahier `chess_ratings.ipynb`.

### 2.1 Charger et nettoyer les données

- Complétez `chess_ratings.py:parse_xml()`
- Complétez `chess_ratings.py:clean_data()`

Un fichier XML (zippé) vous est fourni :

```
./data/standard_oct22frl_xml.zip
```

Il a été obtenu à partir de la base de données des joueurs FIDE qui peut être trouvée ici. Notez que les notes sont mises à jour assez régulièrement, nous utilisons donc l'horodatage fixe du 06 octobre 2022, qui vous est fourni dans le répertoire de données. Vous ne devriez pas avoir besoin de télécharger quoi que ce soit.

Complétez la méthode `parse_xml()`. Notez que la première partie de la méthode contient du code pour extraire automatiquement le fichier zip pour vous. Assurez-vous d'obtenir les données suivantes à partir du XML (et de les utiliser comme noms de colonne) dans votre dataframe :

```
["nom", "classement", "sexe", "anniversaire", "pays", "drapeau", "titre"]
```

Ensuite, complétez `clean_data()`: supprimez les joueurs avec des anniversaires NaN, convertissez les types numériques en type approprié et filtrez les anniversaires `<=2002` (car les scores Elo pour les personnes de < 20 ans peuvent ne pas être fiables).

### 2.2 Comparer les distributions de notes

- Complétez `chess_ratings.py:bin_counts()`
- Complétez les graphiques dans `chess_ratings.ipynb`

Comparons maintenant la répartition des notes des joueurs masculins et féminins. Étant donné que les données sont assez fines, nous devrons regrouper les notes. Complétez les données de `bin_counts()`, qui devrait gérer le binning pour les données arbitraires et le choix des bins. En plus de renvoyer les décomptes bruts, renvoyez également les décomptes normalisés (`count_norm`).

Ensuite, vous utiliserez cette méthode pour compléter les graphiques dans `chess_ratings.ipynb`. Pour les graphiques, choisissez une *largeur* de bin de 50, une plage de 1000 à 2900 (assurez-vous que votre plage s'étend jusqu'à 2900 !), et le *centre* des bins comme point médian de chaque bin (c'est-à-dire pour un bin entre [1500, 1550], le centre serait 1525). Notez que lors de la définition de vos bins, vous constaterez que `len(bin_centers) = len(bins) - 1`.

À l'aide des données regroupées, tracez deux graphiques linéaires l'un à côté de l'autre. Un graphique devrait contenir les décomptes bruts (`count`) et l'autre les décomptes normalisés (`count_norm`). M/F devraient être de deux couleurs différentes. Assurez-vous d'inclure ces graphiques dans votre cahier lorsque vous soumettez.

### 2.3 Tests de permutation

- Compléte `chess_ratings.py:PermutationTests.job()`
- Complétez `chess_ratings.py:sample_two_groups()`

Finalement, nous effectuerons les tests de permutation indiqués dans l'introduction. Prenez l'ensemble de données nettoyé complet (hommes et femmes) et échantillonnez au hasard deux groupes sans remplacement. La taille des groupes doit refléter la différence du monde réel que nous souhaitons étudier, c'est-à-dire la taille du groupe masculin et féminin. Terminez `PermutationTests.job()`, qui implémente la partie échantillonnage de cette expérience, et renvoie la valeur maximale des groupes surreprésentés et sous-représentés respectivement.

Ensuite, complétez la méthode `sample_two_groups()`, qui exécute cette expérience `n_iter` de fois. Une fois terminé, exécutez cette expérience dans le bloc-notes avec au moins `n_iter=1000`. Exécutez la cellule qui imprime la différence moyenne obtenue à partir des tests de permutation, ainsi que les différences réelles. Assurez-vous d'inclure ces résultats imprimés dans votre notebook lorsque vous les soumettez.

### 2.4 Questions

Répondez à ces questions (1 à 3 phrases chacune) dans la section spécifiée dans `chess_ratings.ipynb` :

1. Interprétez les résultats - pouvez-vous tirer une conclusion ? Rappelez-vous que l'affirmation discutée dans l'introduction de cette question était "les hommes sont meilleurs que les femmes aux échecs parce que la plupart des meilleurs joueurs sont des hommes". (*Remarque : probablement une partie de votre réponse ici sera liée à votre réponse à la question suivante.*)
2. Pensez-vous que les chiffres obtenus ici racontent toute l'histoire ? Quels pourraient être les problèmes avec l'analyse menée ici ? Les données avec lesquelles nous travaillons sont-elles biaisées d'une quelconque manière (autre qu'un biais de surreprésentation) ? L'ELO est-il une bonne mesure et peut-il être utilisé pour répondre à la question initiale ? Existe-t-il des différences dans le traitement social, culturel et systémique des hommes et des femmes qui peuvent empêcher le groupe sous-représenté d'obtenir des résultats similaires ?

Le but de ces questions est de souligner que les données sont une représentation limitée du monde réel. Il est essentiel pour nous, en tant que scientifiques des données, de prendre du recul lorsque nous examinons un résultat et de réfléchir à la façon dont il est lié au monde réel, plutôt que de simplement supposer naïvement que les données et la configuration expérimentale sont bonnes, ce qui entraîne souvent des conclusions erronées ou incorrectes. Il pourrait y avoir plusieurs facteurs de causalité qui expliquent une relation qui sont indépendants de l'hypothèse d'origine: utilisation de données qui ne reflètent pas vraiment l'hypothèse que vous souhaitez tester, données biaisées (y compris les groupes surreprésentés), différences systémiques réelles entre les groupes, etc.

# Références

- **Reddit Weekends** est basé sur le cours de science des données de Greg Baker à SFU.
- **Chess Ratings** est basé sur l'analyse de cet article de blog.