IFT3700

Devoir 2

Rapport d’analyse

Xuanchen Liu – matricule : 20173286

Van Nam Vu – matricule : 20170148

23 décembre 2021

Question 1

Taille de chaque serveur :

-Avec toutes les données :

$$\frac{7300}{1200} * 10^9 * [24 * size(FLOAT32) + 3 * size(FLOAT64) + size(INT8)]$$

$$= 618GB$$

-Avec seulement les données position :

$$\frac{7300}{1200} * 10^9 * [3 * size(FLOAT64)] = 68GB$$

-Avec seulement les données catégories :

$$\frac{7300}{1200} * 10^9 * size(INT8) = 6GB$$

On a 7200 milliards / 1200 serveurs = 6.08 milliards étoiles par serveur

1.A

Idée : comme RAM = 128 Gb qui équivalent à 12 milliards étoiles, alors on garde 12 milliards/serveurs pour maximiser les travails en parallèle pour économiser le temps.

Étape 0 : Réduit les donnes d'étoiles en gardant seulement les données de position seulement. Temps = 618/3 + 68/3 = 229s

Étape 1 :

- Envoyer 6 milliards d'étoiles de 600 serveurs (devient **serveurs libres** qui sert à calculer parallèle) à 600 autres serveurs (**serveurs occupé**). Le temps de communication et transition est de 68Gb/1Gb par sec = 68 secs (car les serveurs travaillent parallèlement). + temps de lecture : 68Gb/3Gb par sec = 22.66 secs, pour un total de 90.66 secondes.
- On garde un serveur libre pour accumuler les résultats – **Appelons FINAL**

Étape 2 :

- Pour chaque serveurs (appelons S1 et S2), on divise les étoiles en 2 packages (P1-P2, 6 milliards étoiles par packages). On fait la même chose avec tous les 600 autres serveurs.
- Alors pour chaque paire de serveurs, on a 4 packages (P1-2-3-4), alors on a 6 combinaisons possibles (12,13,14,23,24,34). Donc on va louer 4 serveurs libres et envoyer les copies des packages pour faire exécuter l'algo de recherche 1000 paires d'étoiles les plus proches en termes de distance euclidienne (nlogn) parallèlement. Le temps de transition est de 4 transitions = 68sec*4 = 272sec/tour.

  Le temps de l'algo =

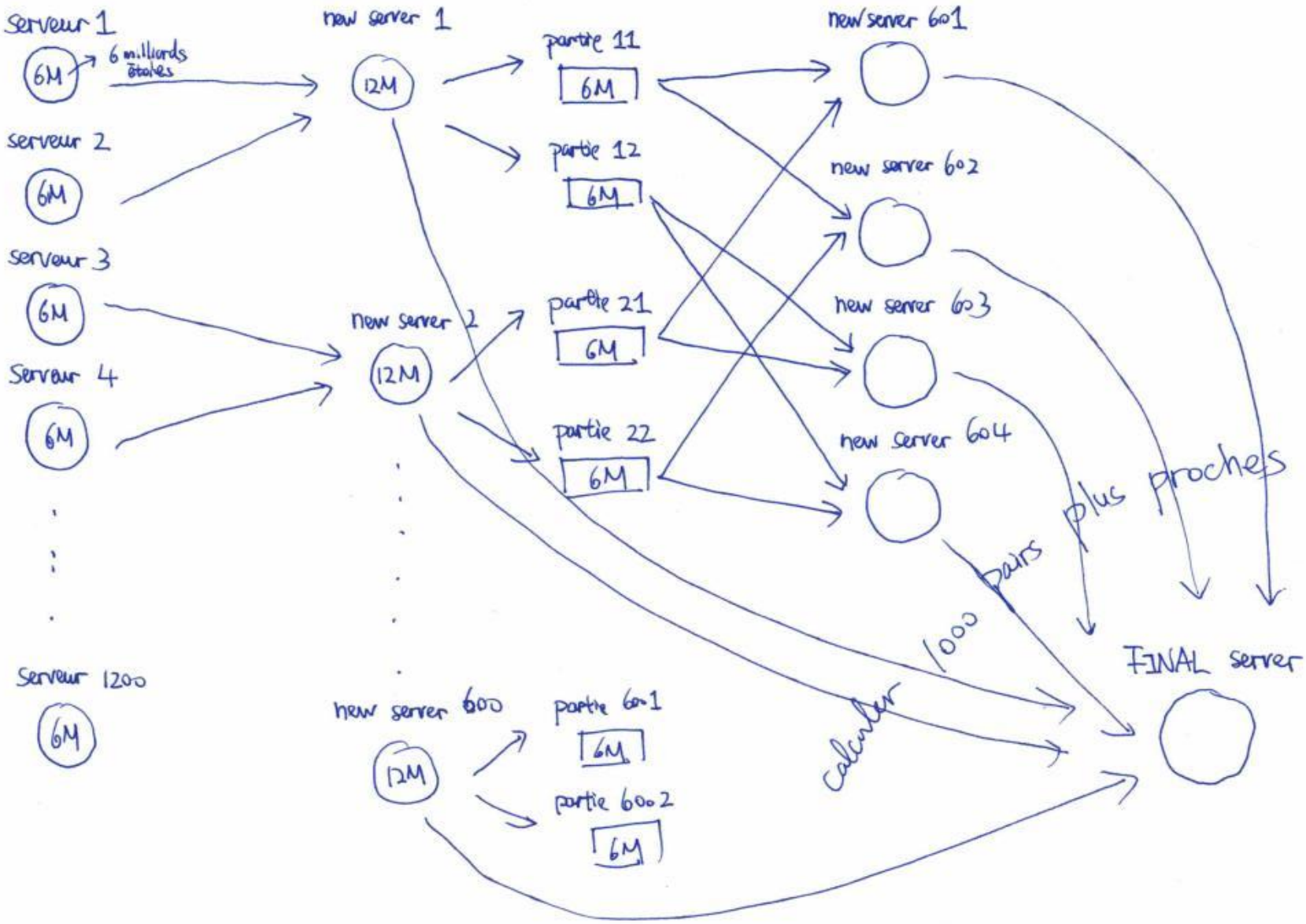
  12 milliards*log (12 milliards) /1milliards opération par sec = 120s
- Comme on a 499 serveurs libres, alors on peut louer en même temps **499/4 = 124 quadruples serveurs pourra se servir**, donc on peut exécuter pour 124 paires de serveurs en parallèle. Comme on a 600 serveurs, alors on a 600*599/2 = 179700 combinaison possible. Donc il nous faut 179700/124 = 1449 fois exécuter algo de recherche, qui donne 1449*120sec = 173904 secondes ≈ 48h + 1449 fois transition = 1449*272sec = 394128sec ≈ 109.48h => **Total = 157.48h**

Étape 3 :

- À chaque tour d'exécution d'algo de recherche, on envoie le résultat à FINAL pour combiner et garder les 1000 les plus proches (en comparant la distance entre les paires et garder les 1000 plus petites).

  La taille = 124 pairs *2000 étoiles*12 = 2976000Gb/3Gb par sec ≈ 1sec, et donc le temps de lectures pour 1449 fois = 1449*1 sec = 1449sec ≈ 0.4h

Temps d'exécution total : ≈ 160h ≈ 1 semaine


serveur 1
6M
6 milliards stokes
serveur 2
6M
serveur 3
6M
Serveur 4
6M
Serveur 1200
6M
new server 1
12M
new server 2
12M
new server 600
12M
partie 11
6M
partie 12
6M
partie 21
6M
partie 22
6M
partie 601
6M
partie 602
6M
new server 601
new server 602
new server 603
new server 604
1000 paires plus proches
calculer
FINAL server

1.B

Etape1 : Map

Lire les données sur les étoiles dans chaque serveur. Enregistrer la donnée qui porte sur la catégorie de l'étoile dans ce serveur. Ensuite on divise 1200 serveurs à 120 serveurs pour chaque catégorie (Pour ne pas dépasser la capacité du serveur). Et puis on envoie l'étoile avec la catégorie au serveur correspont à la catégorie. Le temps pour lire des données dans chaque serveur est de 618GB / 3GB/s = 206s, le temps pour écrire à un nouveau serveur est de 6GB / 3GB/s + 6GB/ 1GB/s = 8s. Le temps total pour étape 1 est de 214s.

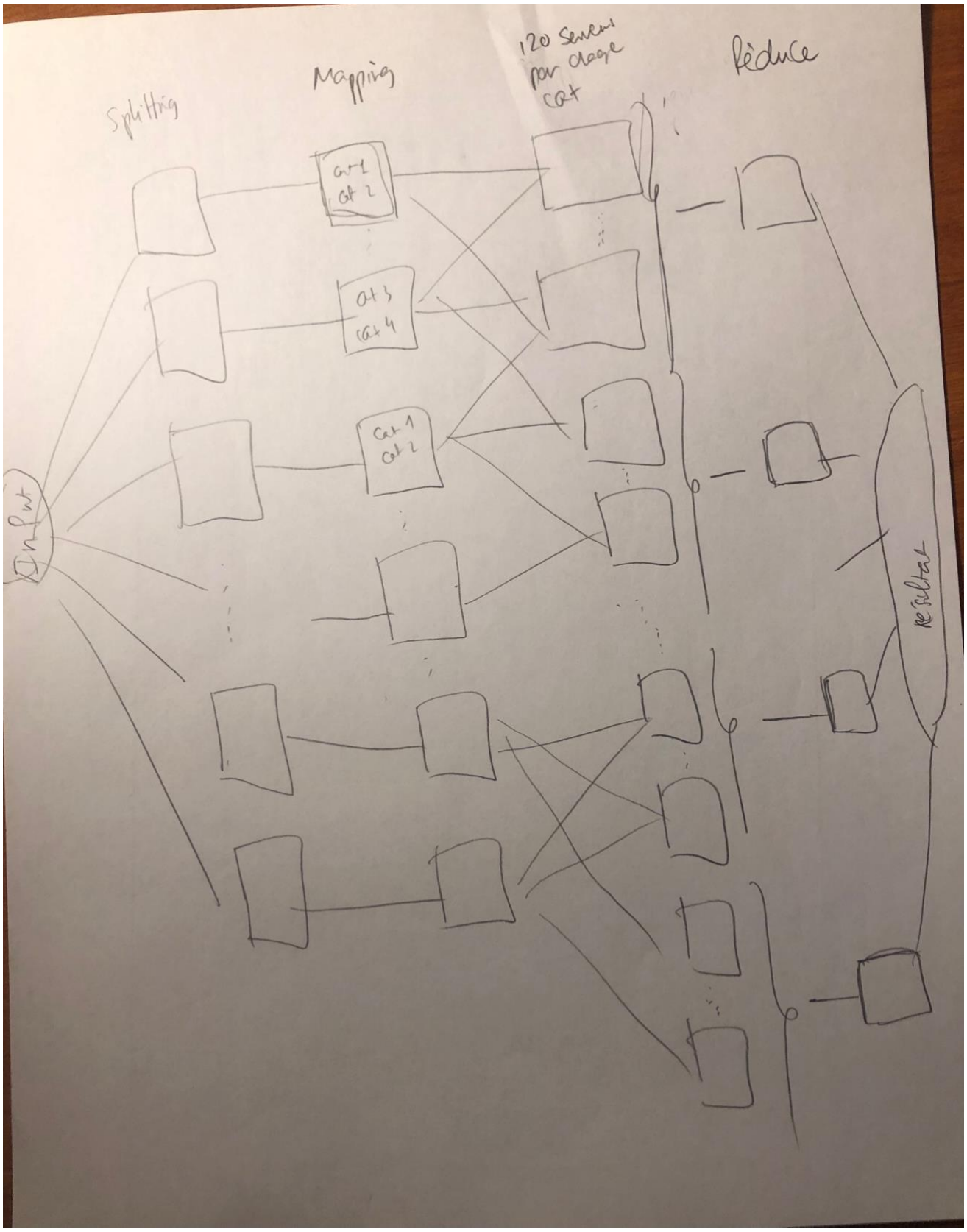
Etape2 : Reduce

Le nombre d'étoiles dans chaque catégorie est donne directement par la taille du serveur correspondant divise par taille de chaque donnée dans le serveur, soit le INT8 avec une taille de 1 byte. Et on combine 120 serveurs pour chaque catégorie pour avoir le nombre total d'étoiles dans chaque catégorie.

```
map(key,value):
//key: serveur, value: etoile
//enregistrer le categorie d'etoile dans le dossier correspondant [a1,a2...a10], ai indique le categorie i
    switch(etoile.categorie):
        case(1...10): a1...a10 <-- add etoile.categorie

reduce(key,value):
//key: dossier, value:size du dossier
    nombre = value / size de INT8
```

Comme la taille de 6milliard est 6Gb donc le temps total d'exécution est moins de 30 minutes.


Splitting
Mapping
120 Servers par étape cat
Réduce
Input
cat 1 cat 2
cat 3 cat 4
cat 1 cat 2
Résultat

1.C

Étape 1

Dans chaque serveur, on collecte la catégorie de toutes les étoiles qui ont une valeur de hachage égal à la valeur de hachage de l'étoile à prédire. La valeur de hachage est obtenue par la méthode **Locally-sensitive Hashing**. Par exemple, une étoile dans catégorie 1 et une étoile dans catégorie 2 ont tous la même valeur de hachage que celle de l'étoile à prédire, on ajoute 1 et 2 dans un fichier qui va contenir à la fin tous les catégorie conformes (pour le but de prendre la majorité). Le temps pour lire les données est de 206s, pour écrire est de (m/3 + m) s ou m est le nombre d'étoile qui a la même valeur de hachage que celle à prédire. Le temps total pour étape 1 est de temps pour lire + temps pour lire + temps de LSH.

Étape 2

Combiner les résultats obtenus dans chaque serveur, est identifier la catégorie qui porte la fréquence le plus grande comme la catégorie de l'étoile à prédire.

```
same_LSH = []
for 1200 serveur:
    for etoile in serveur:
        if(LSH(etoile) == LSH(predict_etoile)):
            same_LSH <-- add etoile.categorie

predict_categorie <-- categorie with most frequency in same_LSH
```

Le temps total est < 45min.

Question 2

3.a) Les colonnes peuvent donner des impacts sur la prédiction de chaque colonne ci-dessous :

- **GPA** : La prédiction de GPA pourra bien fonctionner en utilisant les colonnes comme Minimum wage, life expectancy ou homeless population ou health Expenditure.
- **Internet connection speeds** : La prédiction de GPA pourra bien fonctionner en utilisant les colonnes comme InternetUser ou Median Age.
- **Alcohol consumption** : La prédiction avec démocracy index, Tertiary education atteinment et Importance religion.
- **Intentional homicide rate** : Externel debt, Income equality, health expenditure ou minimun wage
- **Military expenditures**: Oil production, Median Age ou Economie Freedom.
- **Human development Index** : Homeless population, Spending Education, Age criminal responsibility, or importance religion.
- **Democracy Index :** Importance religion, Internet User
- **Tertiary education attainment** : Spending Education, importance religion, GDP, Human development index
- **Importance religion :** GDP, Women Age First Mariage, or Democracy index
- **Christianity** : Tertiary education attainment, Importance religion,
- **Islam :** Les autres indices comme Christianity, Buddhism, Jewish, importance religion
- **Buddhism** : Les autres indices comme Christianity, Islam, Jewish, importance religion
- **Jewish :** Economie Freedom, Tertiary education attainment, Journal Articles
- **Under-five mortality** : Suicide rate, Population Growth rate, Women age first marriage

- **Age criminal responsibility :** Importance religion, Suicide rate, Alcohol consumption

- **Minimum wage :** Life expectancy, Homeless population, meat consumption, milk consumption

- **External debt :** Alcohol consumption, Homeless Population, milk consumption, meat consumption, GDP

- **Income equality** : GDP, Importance religion, Homeless population

- **Health expenditure** : GDP, Tertiary education attainment,

- **Suicide rate** : Tertiary education attainment, Population growth rate, external debt

- **Fertility rate** : Tertiary education attainment, Importance religion, GDP

- **Tobacco Consumption** : Alcohol consumption, Food Energy, Heath expenditure,

- **Obesity rate** : Alcohol consumption, Food Energy, Heath expenditure,

- **Internet User** : Internet connection speeds, Democracy index, Journal articles, book published

- **Median age :** Women age first marriage, GDP

- **Economie Freedom** : GDP, Human Dev Index, Tertiary education attainment

- **Oil production** : GDP, External debt,

- **Population growth rate** : GDP, Tertiary education attainment, Human Dev Index, Internet user

- **Life expectancy** : Health expenditure, Suicide rate, GDP, income equality

- **Meat consumption** : GDP, Obesity rate

- **Incarceration rate** : Military expenditures, democracy index

- **Literacy rate** : Spending education. GDP, Tertiary education attainment, Journal articles, Books published.

- **Women age first marriage** : Importance religion, Population growth rate, life expectancy

- **Spending Education** : Tertiary education attainment, GDP

- **Homeless population** : GDP, Human development index, income equality

- **Milk consumption** : GDP, Health expenditure, Food Energy

- **Journal articles :** Literacy rate, Tertiary education attainment, Spending education

- **Book Published** : Literacy rate, Tertiary education attainment, Spending education

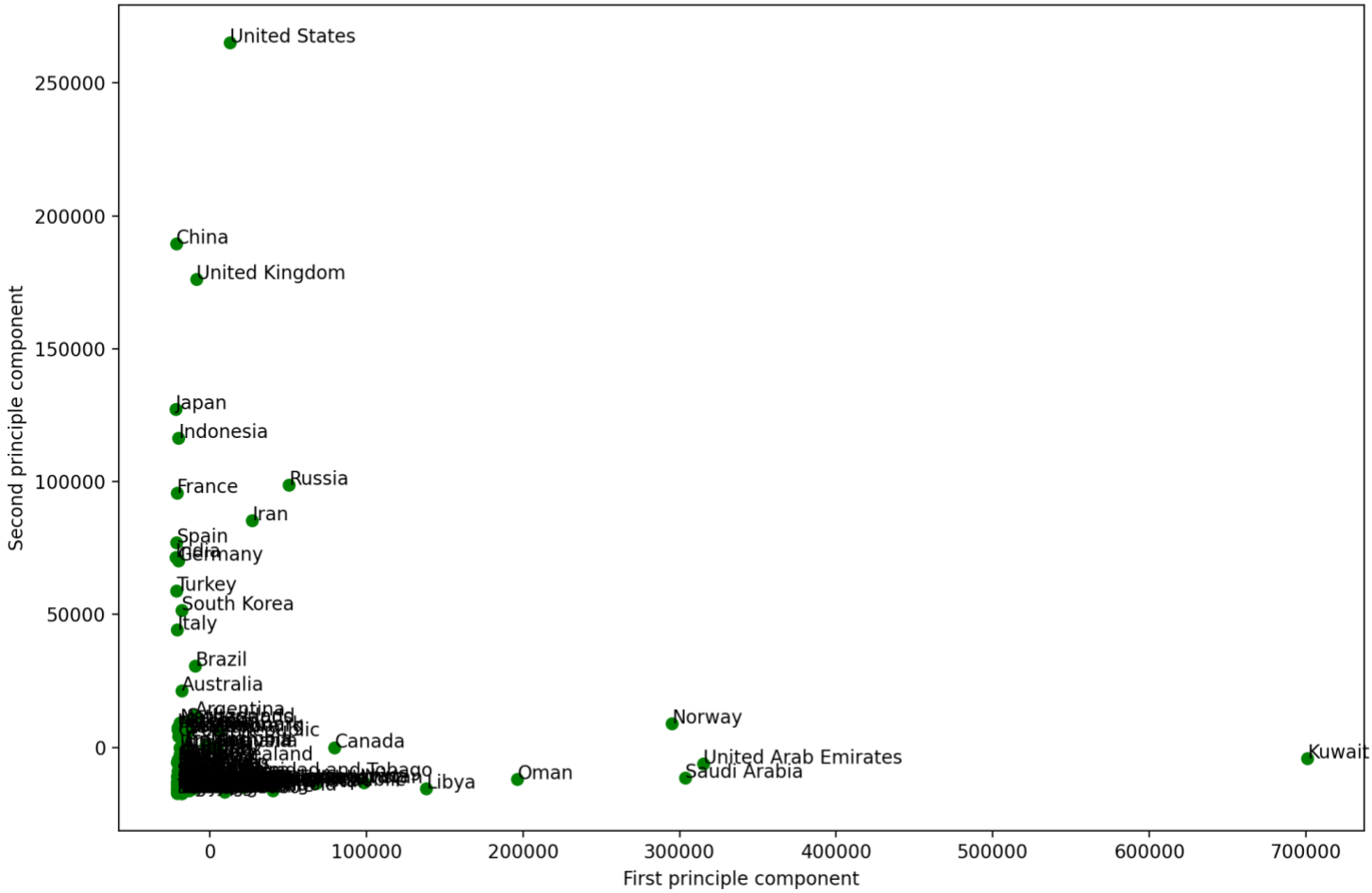
- **Food energy** : Meat consumption, Tobacco consumption, Health expenditure, Yearly temperature

- **Yearly temperature** : Health expenditure, Oil production

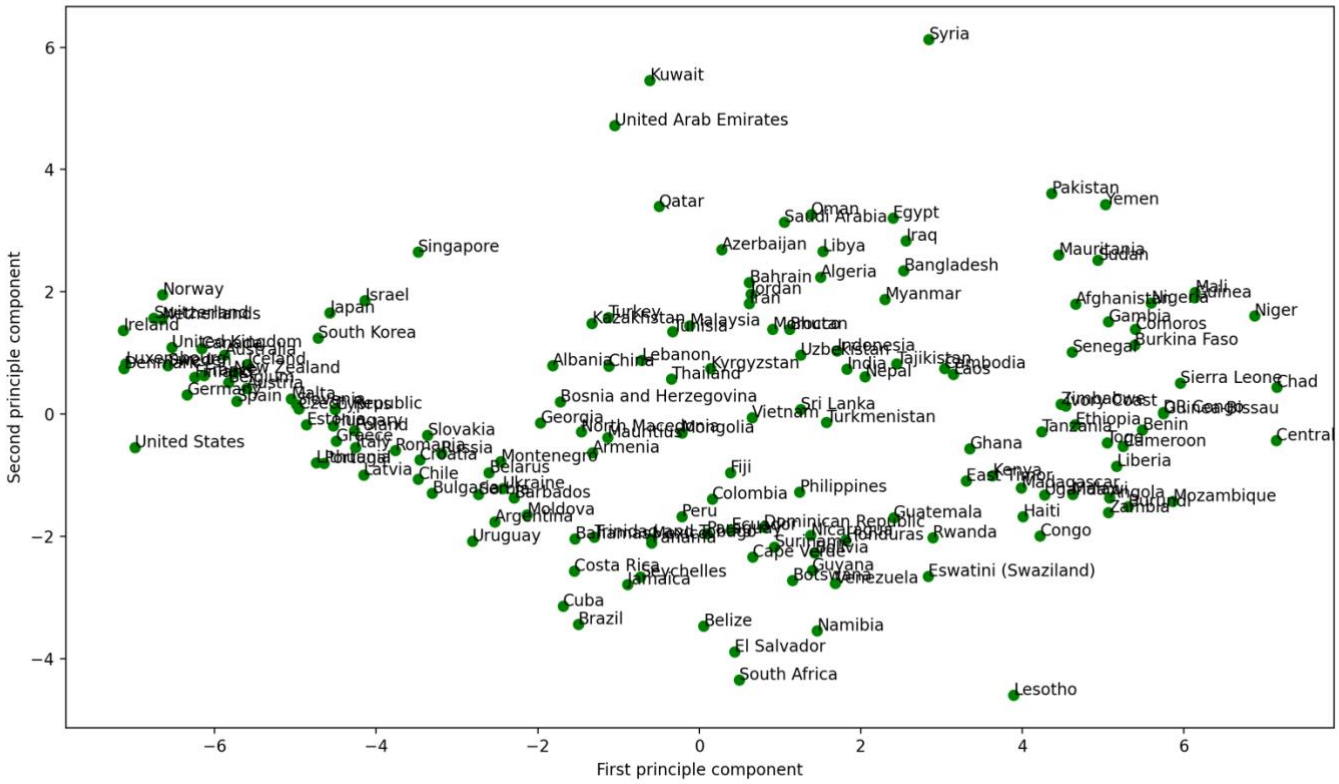
3.c)

1. **Regression linéaire : GDP et Internet connection speeds**
2. **Classifieur bayésien : GDP et Homeless population**

4.a) PCA avec dataB sans normaliser



PCA avec dataB normalisé



La majoritaire des pays sont distribués dans la région (0,0) → (50000,100000), et certains pays comme United States, China, United Kingdom domine dans le sens du second principal component. Et les autres comme Kuwait, United Arab Emirates, Saudi Arabia domine dans le sens du first principale component. Après la normalisation, on voit que les pays sont compacts sur la ligne y=-|x|-3, et les autres reste au-dessus de cette ligne. La majoritaire des pays ont au moins un des principle component positif. Le graph ne nous montre pas une distribution clairement identifiée. On a choisi PCA comme méthode de réduction dimensionnelle car on trouve que cette méthode nous aide à garder les components les plus importants, pour dimension 2, c'est peut-être le GDP qui est un des deux composantes principales. Avec 5 dimensions, on aura plus d'information sur le dataset mais on ne peut pas visualiser facilement, et aussi pour dimension 5, on croit que c'est toujours GDP qui est un des 5 composantes principales.

5.b)