

TP3 : Régression linéaire par Mini batch

1^{ère} étape :

Nous savons que la 1^{ère} étape consiste à importer un dataset, nous avons déjà fait ça par 2 méthodes :

- 1.Importer un fichier Excel ou CSV en utilisant la librairie [pandas](#),
- 2.Importer le jeu de données [make regression](#) from [sklearn.datasets](#)

Ici, on va générer des nombres aléatoires depuis une loi normale multidimensionnelle en passant par [numpy.random.multivariate_normal\(\)](#) :

Prélever des échantillons aléatoires à partir d'une distribution normale multivariée.

La distribution normale multivariée, multinormale ou gaussienne est une généralisation de la distribution normale unidimensionnelle à des dimensions supérieures. Une telle distribution est spécifiée par sa matrice de moyenne et de covariance. Ces paramètres sont analogues à la moyenne (moyenne ou "centre") et à la variance (écart-type, ou "largeur", au carré) de la distribution normale unidimensionnelle.

`numpy.random.multivariate_normal(mean, cov[, size])`

Paramètres :

mean : tableau de dimension 1 « 1-D array », de longueur N : Moyenne de la distribution N-dimensionnelle.

cov : tableau de dimension 2 « 2-D array », de taille (N, N) : Matrice de covariance de la distribution.

Elle doit être symétrique et semi-définie-positive pour un échantillonnage correct.

taille : int ou tuple d'ints, « facultatif »

Returns :

out : `out[i,j,..., :]` est une valeur à N dimensions tirée de la distribution.

La moyenne est une coordonnée dans l'espace N-dimensionnel, qui représente l'endroit où les échantillons sont le plus susceptibles d'être générés. Cette coordonnée est analogue au sommet de la courbe en cloche pour la distribution normale unidimensionnelle ou univariée.

La covariance indique le niveau auquel deux variables varient ensemble. À partir de la distribution normale multivariée, nous tirons des échantillons à N dimensions, $X = [x_1, x_2, \dots, x_N]$. L'élément de matrice de covariance C_{ij} est la covariance de x_i et x_j . L'élément C_{ii} est la variance de x_i (c'est-à-dire sa "propagation").

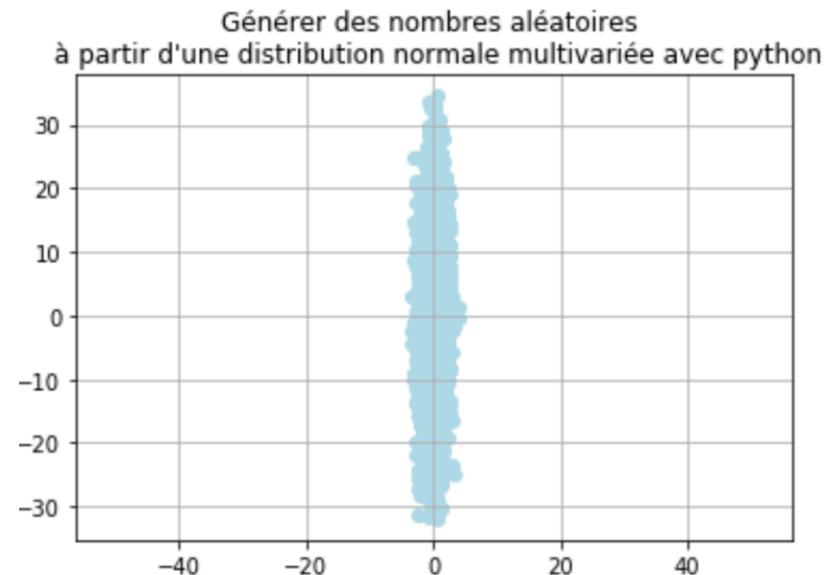
Exemple

```
import numpy as np
import matplotlib.pyplot as plt
```

```
mean = [0, 0]
cov = [[1, 0], [0, 100]]
```

```
x, y = np.random.multivariate_normal(mean, cov, 6000).T
plt.scatter(x,y,c='lightblue')
```

```
plt.grid()
plt.title("Générer des nombres aléatoires \n à partir d'une distribution normale multivariée avec python")
plt.axis('equal')
plt.show()
```



```
import numpy as np
import matplotlib.pyplot as plt

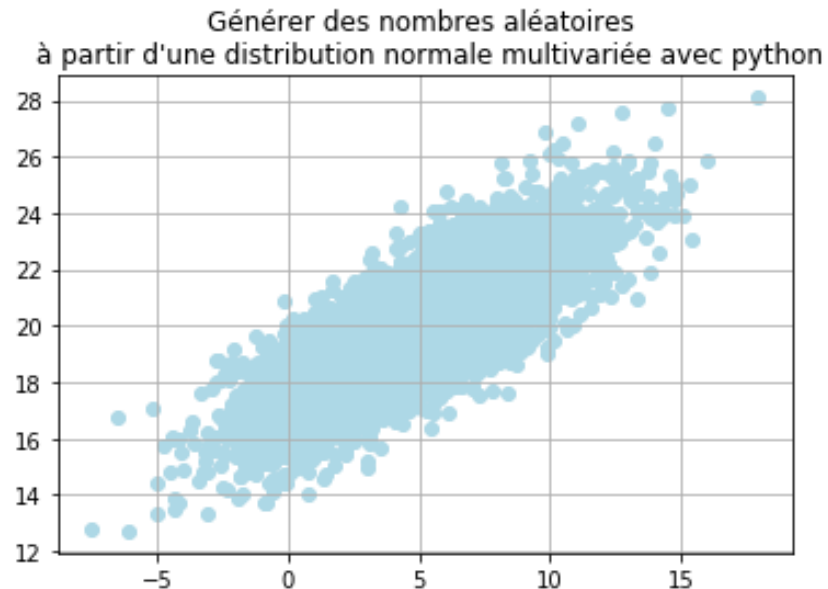
mean = [5,20]
cov = [[10,5],[5,4]]

x,y = np.random.multivariate_normal(mean,cov,5000).T

plt.scatter(x,y,c='lightblue')

plt.grid()
plt.title("Générer des nombres aléatoires \n à partir d'une distribution normal")

plt.show()
```

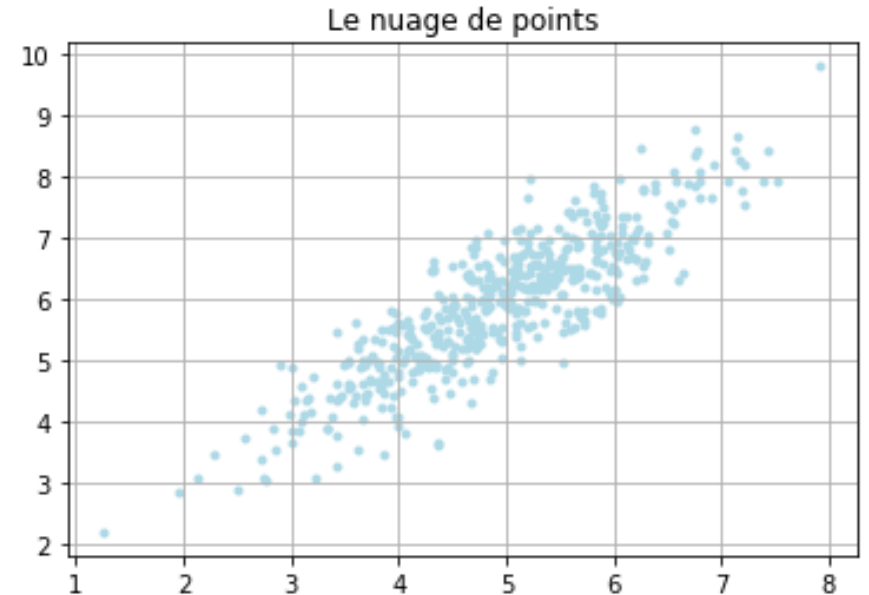


Régression linéaire par Mini batch

```
import numpy as np
import matplotlib.pyplot as plt

# créer Le dataset
mean = np.array([5.0, 6.0])
cov = np.array([[1.0, 0.95], [0.95, 1.2]])
data = np.random.multivariate_normal(mean, cov, 8000)

# visualiser Le dataset
plt.scatter(data[:500, 0], data[:500, 1], c='lightblue', marker = '.')
plt.grid()
plt.title("Le nuage de points")
plt.show()
```



- Ecrire le code pour mettre en œuvre la régression linéaire en utilisant la descente de gradient Mini batch.