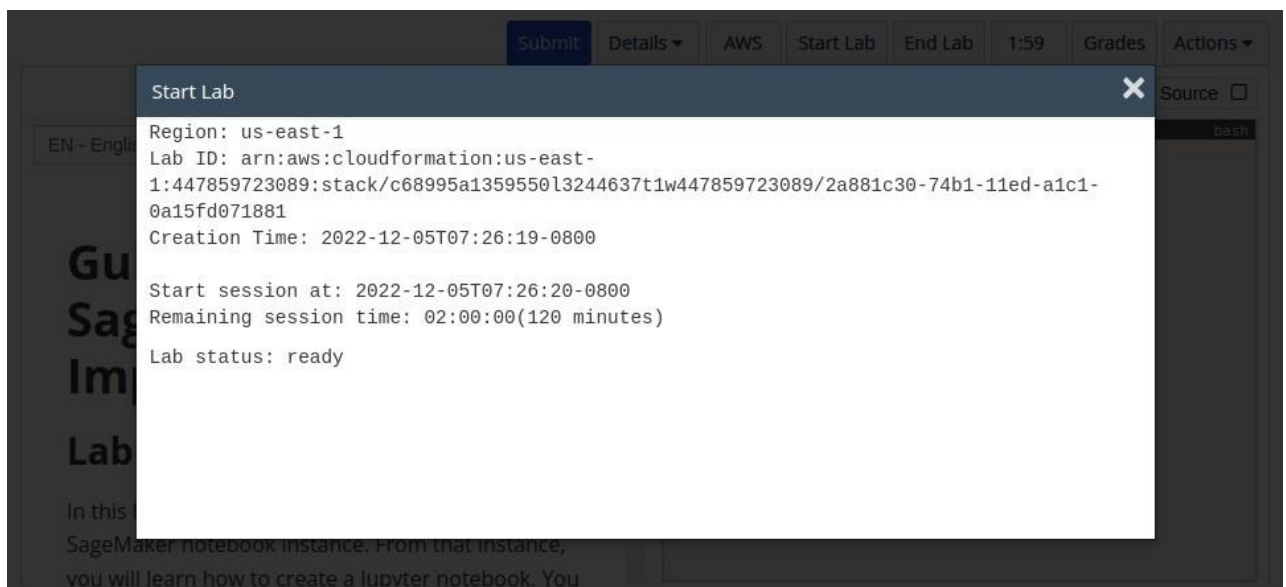


Atelier : Amazon SageMaker  
Professeur : Fahd KALLOUBI  
Année : 2023/2024

Réalisé par : **Mouad GUEDAD**



## 1. Traitement des données avec SageMaker



## 1. Nous allons commencer par le téléchargement et l'extraction de données

Name

bank-additional-full...

bank-additional-nam...

bank-additional.csv

```
[3]: %%sh
wget -N https://sagemaker-sample-data-us-west-2.s3-us-west-2.amazonaws.com/sagemaker-sample-data-us-west-2/bank-additional.zip
unzip -o bank-additional.zip

Archive: bank-additional.zip
  creating: bank-additional/
  inflating: bank-additional/bank-additional-names.txt
  inflating: bank-additional/bank-additional.csv
  inflating: bank-additional/bank-additional-full.csv
```

## 2. Nous allons ensuite le charger avec Pandas

```
import pandas as pd
data = pd.read_csv('./bank-additional/bank-additional-full.csv')
print(data.shape)
data[:5]
```

(41188, 21)

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	campaign	pdays	previous	poutcome	emp.var.rate
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1
1	57	services	married	high.school	unknown	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1
2	37	services	married	high.school		yes	no	telephone	may	mon	...	1	999	0	nonexistent	1.1
3	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	1.1
4	56	services	married	high.school	no	no	yes	telephone	may	mon	...	1	999	0	nonexistent	1.1

3. Maintenant, téléchargeons le dataset sur Amazon S3. Nous utiliserons un compartiment « Bucket » par défaut créé automatiquement par SageMaker dans la région dans laquelle nous exécutons le code

4. Comme SageMaker Processing prend en charge tous les problèmes d'infrastructure, nous pouvons nous concentrer sur le script lui-même. Nous n'avons pas non plus à nous soucier d'Amazon S3 : SageMaker Processing copiera automatiquement l'ensemble de données d'entrée de S3 dans le conteneur, et les ensembles de données traités du conteneur vers S3

Les chemins de conteneur sont fournis lorsque nous configurons le Job lui-même.

Voici ce que nous allons utiliser :

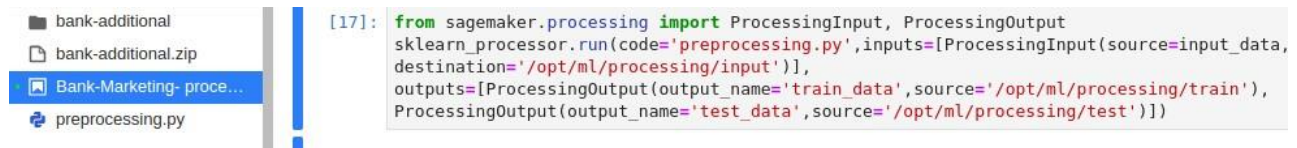
- L'ensemble de données d'entrée : /opt/ml/processing/input
- L'ensemble d'apprentissage traité : /opt/ml/processing/train
- L'ensemble de test traité : /opt/ml/processing/test

Dans le dossier de l'atelier, vous allez trouver le script de traitements nommé « preprocessing.py ». Ce script chargera le dataset, effectuera quelques prétraitements et enregistrera l'ensemble de données traitées.

5. Exécutions du script de traitements : Pour en revenir à notre notebook, nous utilisons l'objet `SKLearnProcessor` du SDK SageMaker pour configurer la tâche de traitement :

- a. Nous définissons quelle version de scikit-learn nous voulons utiliser et quelles sont nos exigences en matière d'infrastructure.
- b. Ensuite, nous lançons simplement le Job, en transmettant le nom du script, le chemin d'entrée du dataset dans S3, les chemins du dataset définis par

l'utilisateur dans l'environnement de traitement SageMaker.



6. Après quelques minutes, le job se termine et nous pouvons voir les outputs du script :

```
Job Name: sagemaker-scikit-learn-2022-12-07-18-35-47-522
Inputs: [{'InputName': 'input-1', 'AppManaged': False, 'S3Input': {'S3Uri': 's3://sagemaker-us-east-1-447859723089/sage
maker/DEMO-smprocessing/input/bank-additional-full.csv', 'LocalPath': '/opt/ml/processing/input', 'S3DataType': 'S3Prefi
x', 'S3InputMode': 'File', 'S3DataDistributionType': 'FullyReplicated', 'S3CompressionType': 'None'}}, {'InputName': 'co
de', 'AppManaged': False, 'S3Input': {'S3Uri': 's3://sagemaker-us-east-1-447859723089/sagemaker-scikit-learn-2022-12-07-
18-35-47-522/input/code/preprocessing.py', 'LocalPath': '/opt/ml/processing/input/code', 'S3DataType': 'S3Prefix', 'S3In
putMode': 'File', 'S3DataDistributionType': 'FullyReplicated', 'S3CompressionType': 'None'}}]
Outputs: [{'OutputName': 'train_data', 'AppManaged': False, 'S3Output': {'S3Uri': 's3://sagemaker-us-east-1-44785972308
9/sagemaker-scikit-learn-2022-12-07-18-35-47-522/output/train_data', 'LocalPath': '/opt/ml/processing/train', 'S3UploadM
ode': 'EndOfJob'}}, {'OutputName': 'test_data', 'AppManaged': False, 'S3Output': {'S3Uri': 's3://sagemaker-us-east-1-447
859723089/sagemaker-scikit-learn-2022-12-07-18-35-47-522/output/test_data', 'LocalPath': '/opt/ml/processing/test', 'S3U
ploadMode': 'EndOfJob'}}]
```

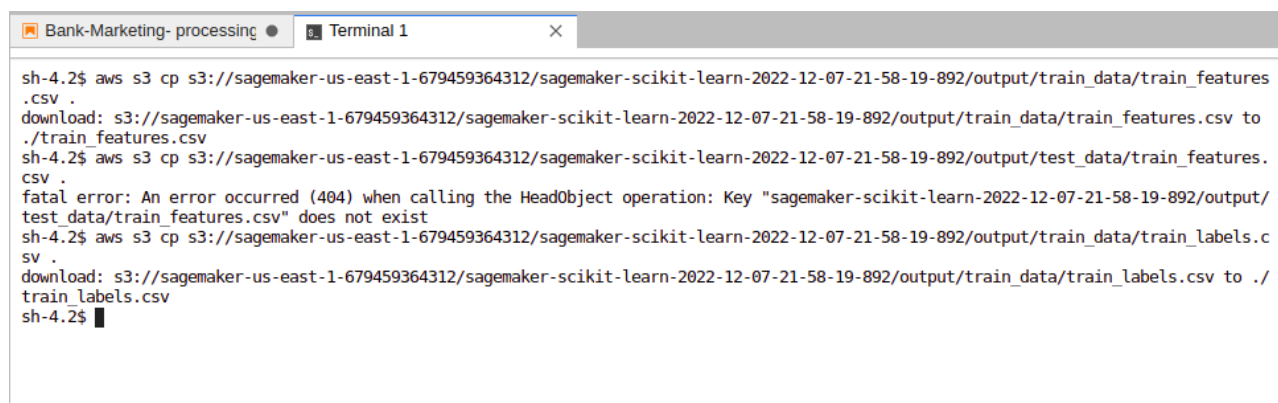
7. Finalement, nous pouvons afficher le job pour connaître l'emplacement des dataset traités

```
preprocessing_job_description = sklearn_processor.jobs[-1].describe()
output_config = preprocessing_job_description['ProcessingOutputConfig']
for output in output_config['Outputs']:
    print(output['S3Output']['S3Uri'])
```

```
s3://sagemaker-us-east-1-447859723089/sagemaker-scikit-learn-2022-12-07-18-35-47-522/output/train_data
s3://sagemaker-us-east-1-447859723089/sagemaker-scikit-learn-2022-12-07-18-35-47-522/output/test_data
```

8. Dans un terminal, nous pouvons utiliser l'AWS CLI pour récupérer les ensembles d'entraînement et de test traités situés au chemin précédent. Pour ce faire, accédez à l'onglet « Launcher » et cliquez sur « Terminal » et puis lancez les deux commandes suivantes pour copier les fichiers de S3 vers la machine locale ainsi que deux autres commandes pour afficher les premiers enregistrements :

- `aws s3 cp s3://sagemaker-us-east-1-679459364312/sagemaker-scikit-learn-2022-12-07-21-58-19-892/output/train_data/train_features.csv .`
- `aws s3 cp s3://sagemaker-us-east-1-679459364312/sagemaker-scikit-learn-2022-12-07-21-58-19-892/output/train_data/train_labels.csv .`

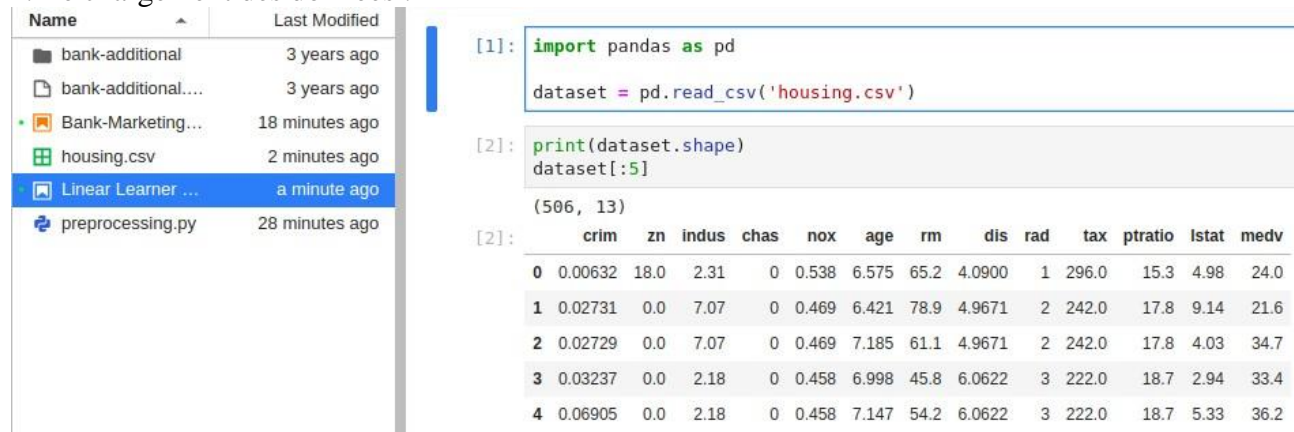


## 2. L'entraînement d'un modèle en utilisant SageMaker SDK avec les algorithmes intégrés

### 2.1 La préparation de données

Les algorithmes intégrés dans Amazon SageMaker s'attendent à ce que l'ensemble de données soit dans un certain format, tel que CSV, protobuf ou libsvm. Les formats pris en charge sont répertoriés dans la documentation de l'algorithme. Dans le dossier de l'atelier vous allez trouver le dataset « housing.csv » et le notebook « Linear Learner on Boston Housing.ipynb », importez les dans votre environnement. Dans ce qui suit, nous allons expliquer chaque cellule à part.

#### 1. Le chargement des données :



The screenshot shows a Jupyter Notebook interface. On the left, a file explorer lists files: bank-additional, bank-additional..., Bank-Marketing..., housing.csv, Linear Learner..., and preprocessing.py. The 'Linear Learner...' notebook is selected. The main area shows three code cells:

```
[1]: import pandas as pd
dataset = pd.read_csv('housing.csv')
```

```
[2]: print(dataset.shape)
dataset[:5]
```

The output of the second cell is:

```
(506, 13)
```

	crim	zn	indus	chas	nox	age	rm	dis	rad	tax	ptratio	lstat	medv
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	5.33	36.2

#### 2. En lisant la documentation officielle de SageMaker

#### 3. Divisons notre dataset en 90% d'entraînement et 10% de test

4. Nous allons sauvegarder ces deux partitions dans deux fichiers csv sans entête et sans indices. Remarquez que les deux fichiers sont créés dans le workspace à gauche.

5. Maintenant, nous devons uploader ces deux fichiers dans S3 dans deux dossier distincts. Nous allons utiliser le compartiment « Bucket » par défaut crée par SageMaker dans la région où nous exécutons notre notebook. Nous pouvons connaître le nom en exécutons « sagemaker.Session.default\_bucket() »

```
[7]: import sagemaker

print(sagemaker.__version__)

sess = sagemaker.Session()
bucket = sess.default_bucket()

prefix = 'boston-housing'
training_data_path = sess.upload_data(path='training_dataset.csv', key_prefix=prefix + '/input/training')
validation_data_path = sess.upload_data(path='validation_dataset.csv', key_prefix=prefix + '/input/validation')

print(training_data_path)
print(validation_data_path)
```

2.119.0  
s3://sagemaker-us-east-1-679459364312/boston-housing/input/training/training\_dataset.csv  
s3://sagemaker-us-east-1-679459364312/boston-housing/input/validation/validation\_dataset.csv

### 2.2 Configurer le job d'entraînement

L'objet Estimator (sagemaker.estimator.Estimator) est la pierre angulaire de l'entraînement



d'un modèle dans SageMaker. Il vous permet de sélectionner l'algorithme approprié, de définir vos besoins en matière d'infrastructure d'entraînement, etc.

1. Les algorithmes SageMaker sont conditionnés dans des conteneurs Docker. En utilisant boto3 et l'API `image_uris.retrieve()`, nous pouvons facilement trouver le nom de l'algorithme Linear Learner dans la région dans laquelle nous exécutons notre job

```
382416733822.dkr.ecr.us-east-1.amazonaws.com/linear-learner:1
```

2. Maintenant que nous connaissons le nom du conteneur, nous pouvons configurer notre Job d'entraînement avec l'objet Estimator

```
ll_estimator = Estimator(container,
    role=role,
    instance_count=1,
    instance_type='ml.m5.large',
    output_path='s3://{}/{}'.format(bucket, prefix)
)
```

3. Ensuite, nous devons définir les hyper paramètres : afin de régler les paramètres nous devons lire la documentation de l'algorithme et respecter les paramètres obligatoires uniquement, ensuite on doit vérifier rapidement les paramètres facultatifs pour les valeurs par défaut qui pourraient entrer en conflit avec votre ensemble de données.

### Linear learner hyperparameters

- **num\_classes** : Le nombre de classes pour la variable de réponse.  
L'algorithme suppose que les classes sont étiquetées 0, ..., num\_classes – 1.  
Obligatoire lorsque predictor\_type est multiclass\_classifier. Sinon, l'algorithme l'ignore.
- **predictor\_type** : Spécifie le type de variable cible comme étant une classification binaire, une classification multiclass ou une régression.  
Obligatoire.  
Valeurs valides : binary\_classifier, multiclass\_classifier ou regressor.
- Les paramètres : **accuracy\_top\_k**, **balance\_multiclass\_weights**, **beta1**, **beta2**, **bias\_lr\_mult**, **bias\_wd\_mult**, **binary\_classifier\_model\_selection\_criteria**, **early\_stopping\_patience**, **early\_stopping\_tolerance**, **epochs**, **f\_beta**, **feature\_dim**, **huber\_delta**, **init\_bias**, **init\_method**, **init\_scale**, **init\_sigma**, **l1**, **learning\_rate**, **loss**, **loss\_insensitivity**, **lr\_scheduler\_factor**, **lr\_scheduler\_minimum\_lr**, **lr\_scheduler\_step**, **margin**, **mini\_batch\_size**, **momentum**, **normalize\_data**, **normalize\_label**, **num\_calibration\_samples**, **num\_models**, **num\_point\_for\_scaler**, **optimizer**, **positive\_example\_weight\_mult**, **quantile**, **target\_precision**, **target\_recall**, **unbias\_data**, **unbias\_label**, **use\_bias**, **use\_lr\_scheduler** et enfin **wd** sont optionnels .

Tâches d'entraînement [Infos](#)

Actions ▼

[Créer une tâche d'entraînement](#)

&lt; 1 &gt;

	Nom ▼	Heure de création ▼	Durée	Statut de la tâche ▼	Statut du groupe d'instances pré-initialisées
<input type="radio"/>	linear-learner-2022-12-11-00-37-29-567	Dec 11, 2022 00:37 UTC	3 minutes	✔ Completed	-
<input type="radio"/>	linear-learner-2022-12-11-00-27-40-391	Dec 11, 2022 00:27 UTC	3 minutes	✘ Failed	-
<input type="radio"/>	linear-learner-2022-12-10-23-34-20-821	Dec 10, 2022 23:34 UTC	4 minutes	✔ Completed	-
<input type="radio"/>	linear-learner-2022-12-07-23-15-51-934	Dec 07, 2022 23:15 UTC	4 minutes	✘ Failed	-
<input checked="" type="radio"/>	linear-learner-2022-12-07-22-34-33-834	Dec 07, 2022 22:34 UTC	4 minutes	✔ Completed	-

## Paramètres de tâche

Nom de la tâche

linear-learner-2022-12-07-22-34-33-834

Séries chronologiques des métriques SageMaker

Désactivé

ARN

arn:aws:sagemaker:us-east-1:679459364312:training-job/linear-learner-2022-12-07-22-34-33-834

Durée d'entraînement (en secondes)

132

Statut

✔ Completed

[Afficher l'historique](#)

Temps facturable (en secondes)

132

Économies en coûts d'entraînement spot géré

0%

Heure de création

Dec 07, 2022 22:34 UTC

Source/parent de tâche de réglage

-

Heure de la dernière modification

**Dans cette partie on va essayer de trouver les hyperparameters optimaux pour le modèle.**

```
ll_estimator.set_hyperparameters(feature_dim = 13,
                                predictor_type = 'regressor',
                                mini_batch_size = 20,
                                epochs = 5,
                                num_models = 10,
                                loss = 'absolute_loss')
```

382416733822.dkr.ecr.us-east-1.amazonaws.com/linear-learner:1

