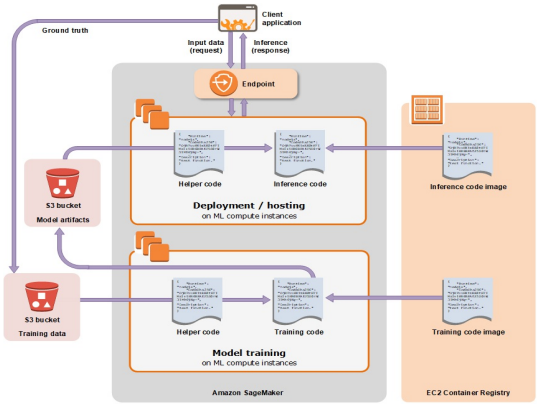
AWS SageMaker工作坊

AWS SageMaker是一個全受管的機器學習服務。從監督式學習到非監督式學習，從決策樹到深度學習，資料科學家可以都輕鬆的將自己的分析流程及演算法實現在上面，並可以使用AWS最佳化過的演算法來進行分析。特色為貼近目前開發者的生態體系，例如：使用Jupyter Notebook開發、可以使用Docker客製化環境、整合SDK可以跨語言使用提供其他團隊使用，目前提供AWS Cli、.NET、C++、Go、Java、JavaScript、PHP、Python、Ruby等SDK。

Created by Hans (github: https://github.com/ogckw)

SageMaker元件說明

SageMaker架構



- S3 Bucket

  提供SageMaker儲存及讀取訓練資料、存放模型

- Container Registry

  提供程式碼執行環境的image

- Notebook Instance

  用於資料科學家互動式分析服務，可以使用資料科學家常使用的`Jupyter Notebook`分析並分享給合作團隊。

- ML Training Service

  用於訓練及分析資料運算服務，可以利用分散式演算法來進行分散式運算，不用管理叢集的基礎設施，讓資料科學家只用專注在算法上而不是管理叢集。

- ML Hosting Service

  用於最終提供預測或分類算演法結果服務，不用管理叢集的基礎設施，可以輕鬆的整合任何語言呼叫演算法使用，就像在使用習慣語言中的SDK一樣。

Workshop 流程教學指引

預先要求

- AWS帳號並有Admin權限
- 能連上網路且足夠順暢的電腦

- 瀏覽器為Chrome或Firefox
- 基礎的IAM、EC2、S3知識

Module0 Prepare Environment 約10分鐘

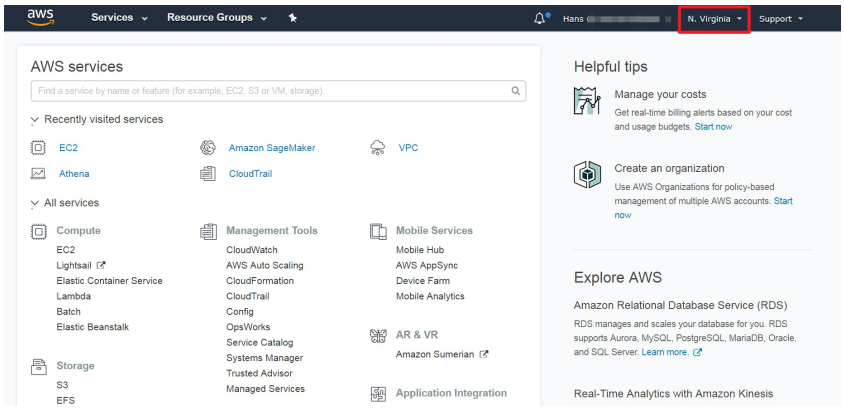
準備基礎環境

1. 登入AWS

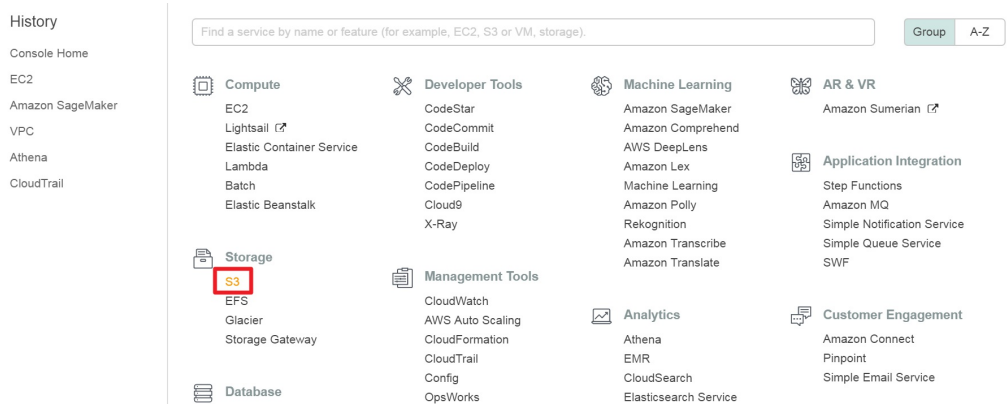
   本Lab可選擇的Region如下列，請確保接下來開的服務都在同個Region，這邊範例使用`US East (N. Virginia)`

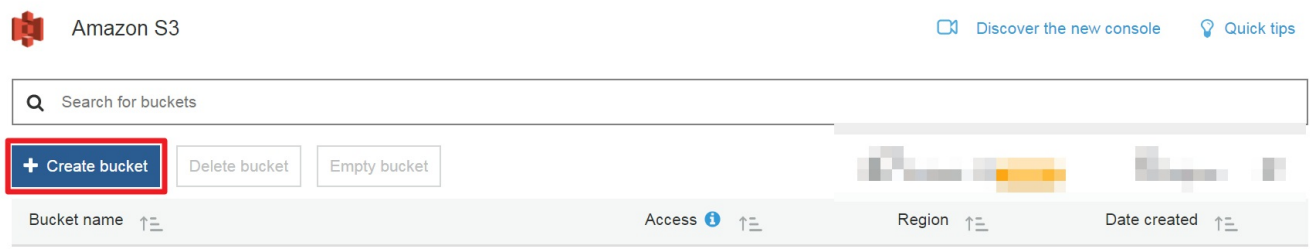
   1. US East (N. Virginia)
   2. US East (Ohio)
   3. US West (Oregon)
   4. EU (Ireland)

   確認自己所在的Region為US East (N. Virginia)



2. 建立S3 Bucket提供之後進行Module使用





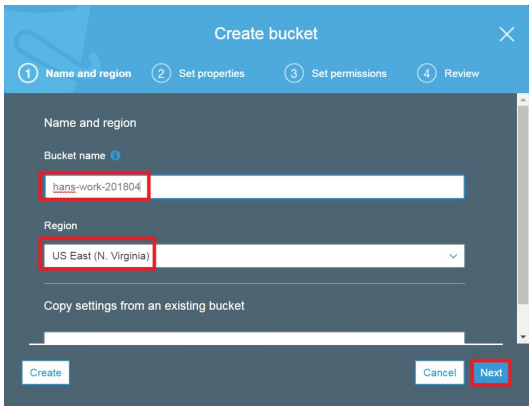
   輸入S3 名稱及區域

   Bucket Name: 請填入想要的名稱

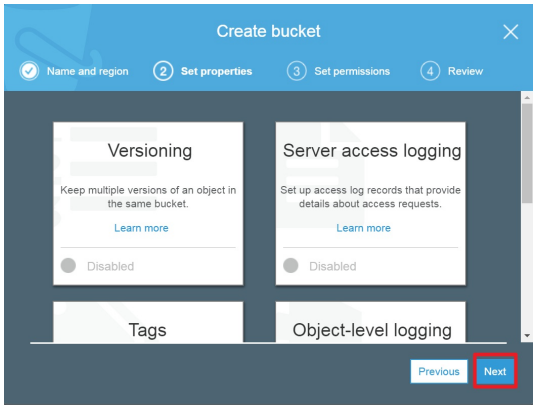
   這邊的範例使用`hans-work-201804`

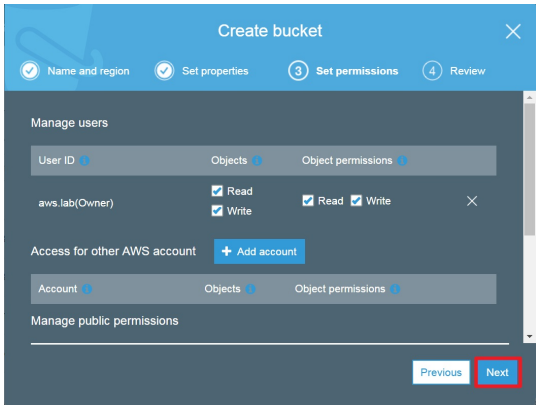
注意! S3 bucket名稱為全球唯一，重覆的名稱會無法建立，所以請勿照抄範例，請自己想一個適合的名稱

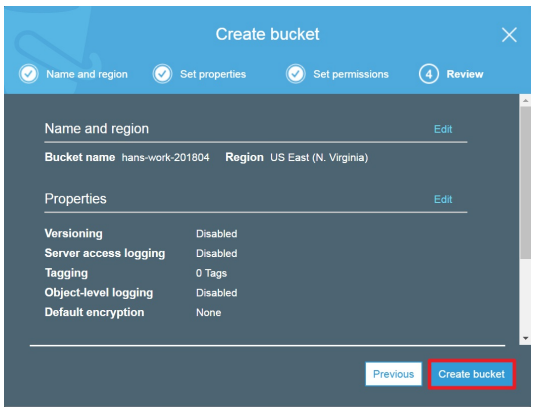
Region: `US East (N. Virginia)`

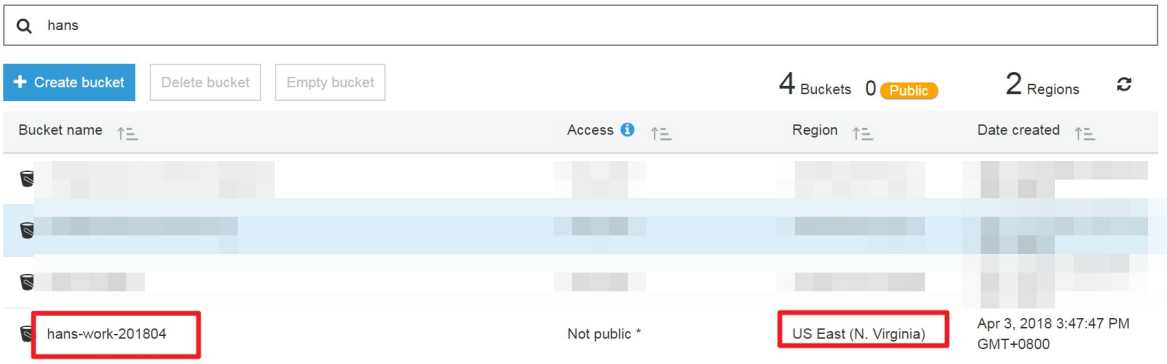


一直選Next直到建立好bucket







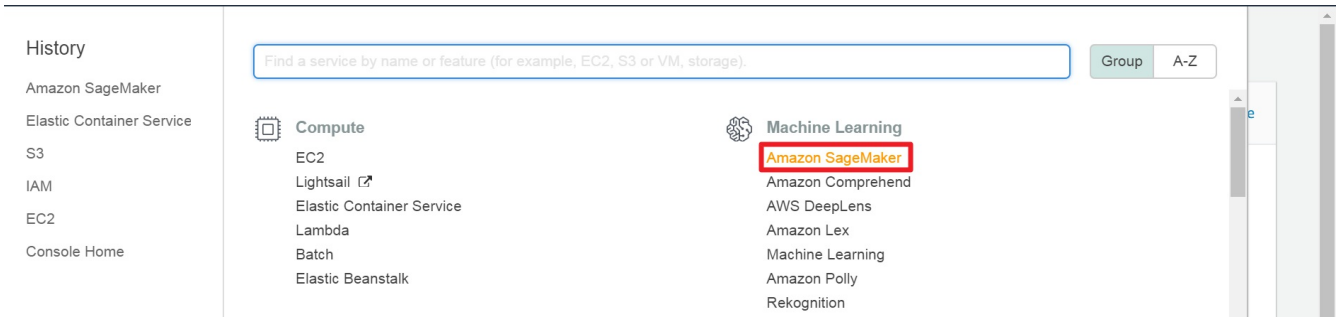


Module1 Introduction to Notebook Instance 15分鐘

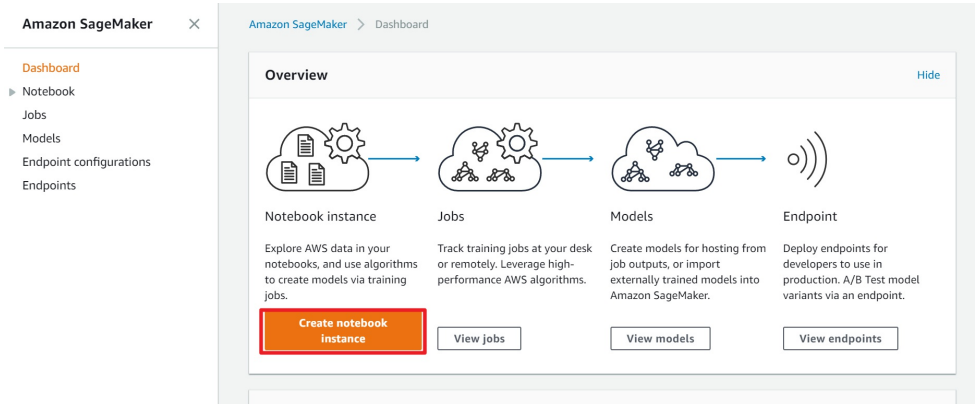
建立一個資料科學家探索資料用機器並探索資料

1. 建立Notebook Instance

   選擇服務SageMaker



   建立Notebook Instance

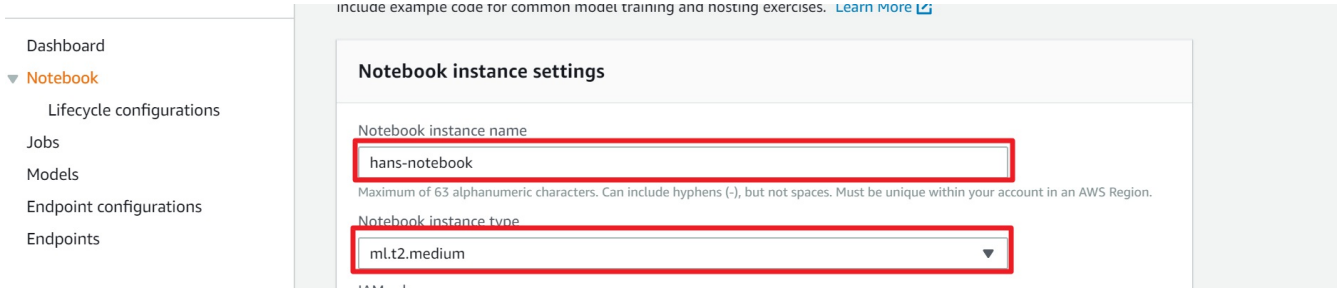


   設定Notebook Instance規格

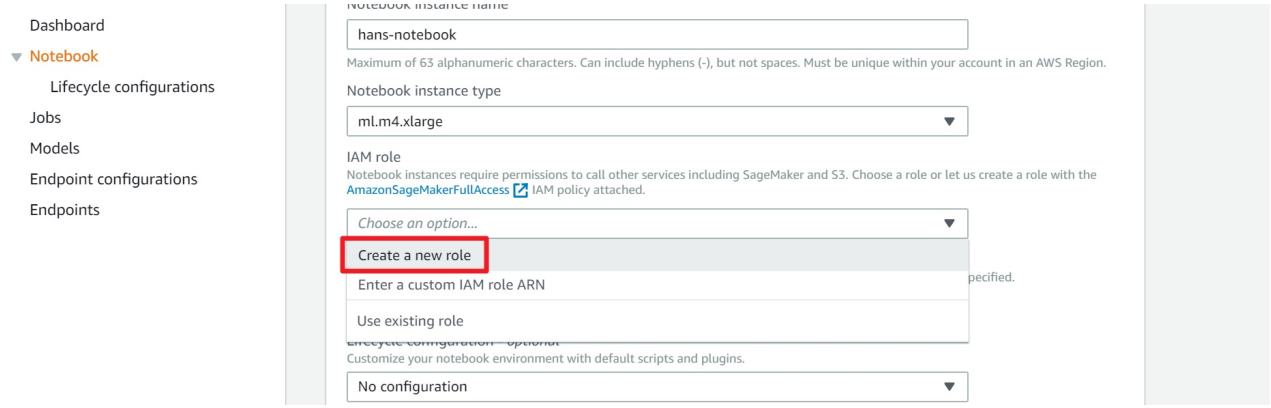
   Notebook instance name: 請填入想要的名稱

   Notebook instance type: 請選擇想要的型號

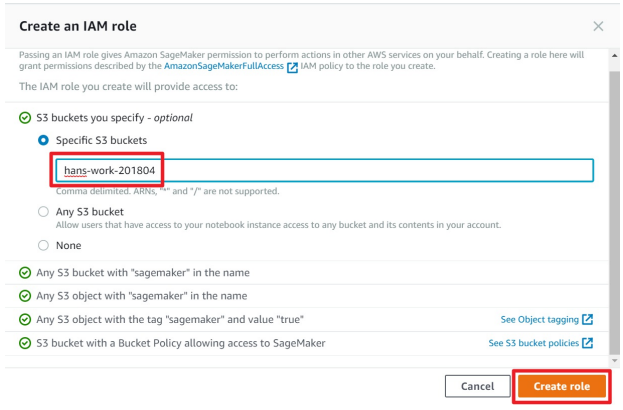
   建議instance type選擇`ml.t2.medium`或`ml.m4.xlarge`



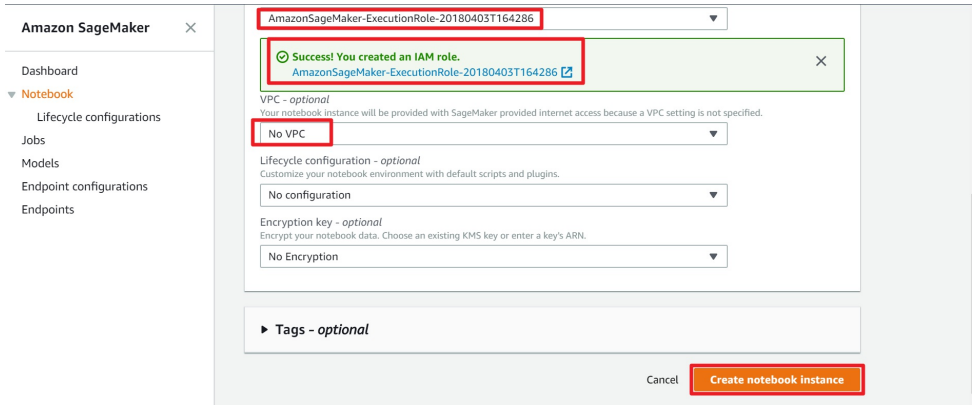
設定Notebook所使用的IAM Role權限



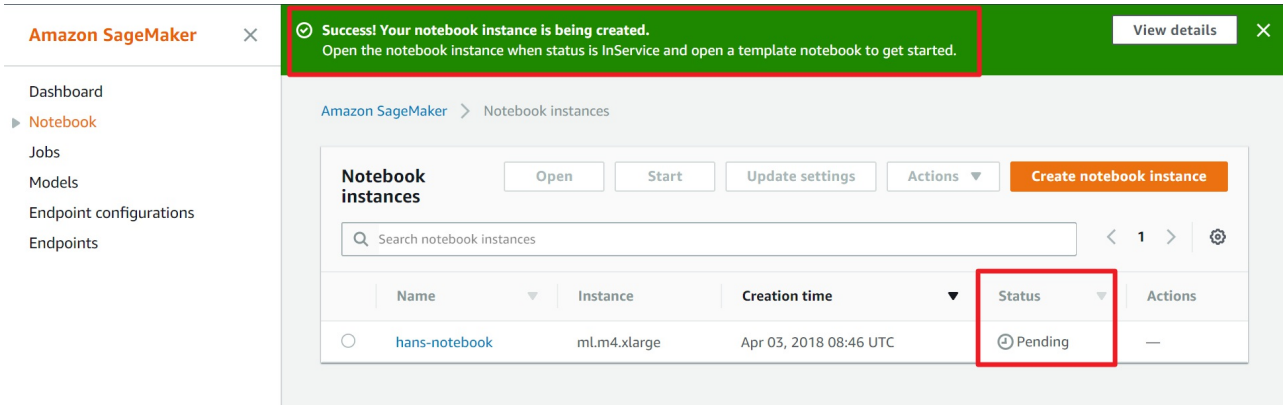
填入剛剛建立的S3 bucket名稱後建立Role



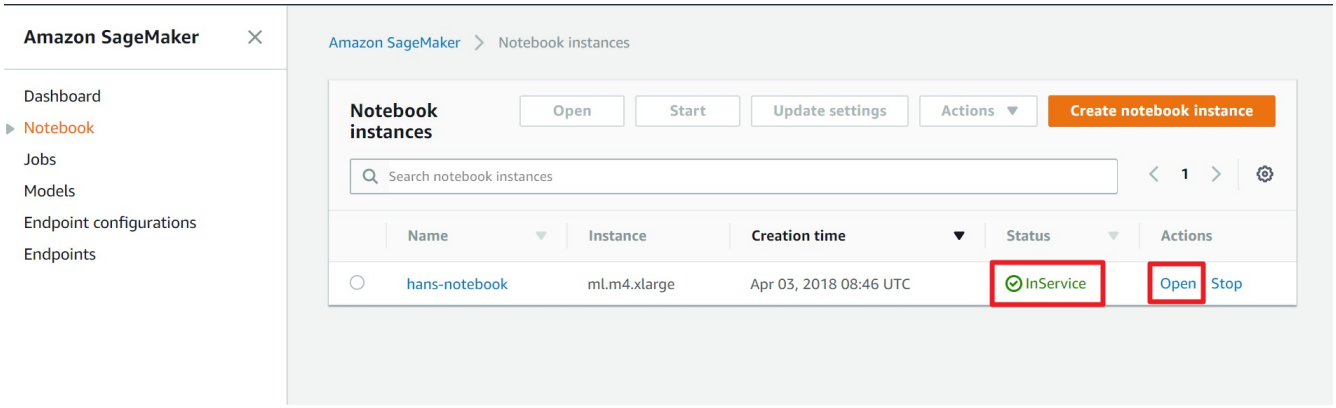
建立成功畫面並確認VPC - optional選擇No VPC後選擇Create notebook instance



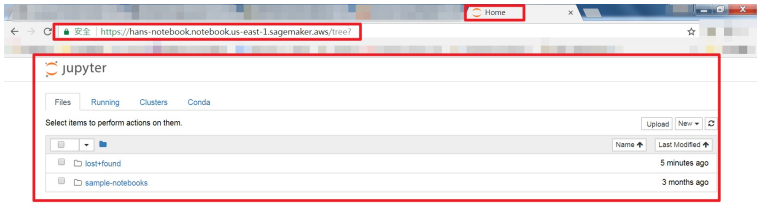
會出現建立中的畫面



建立完成看到InService後點選Open

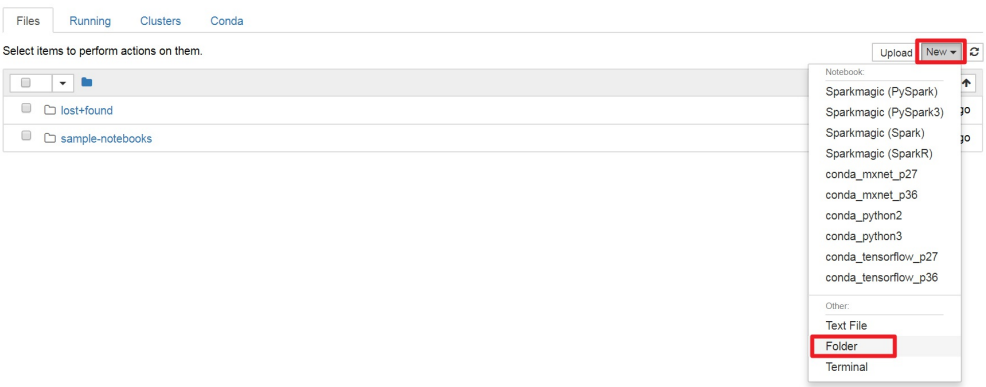


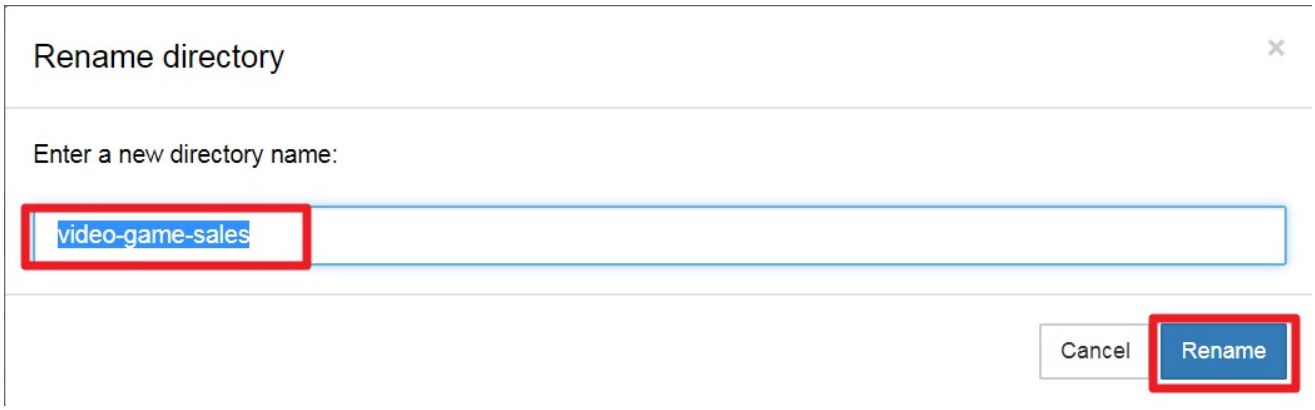
瀏覽器會跳出新的視窗，恭喜你！成功建立第一個資料科學家所使用的環境



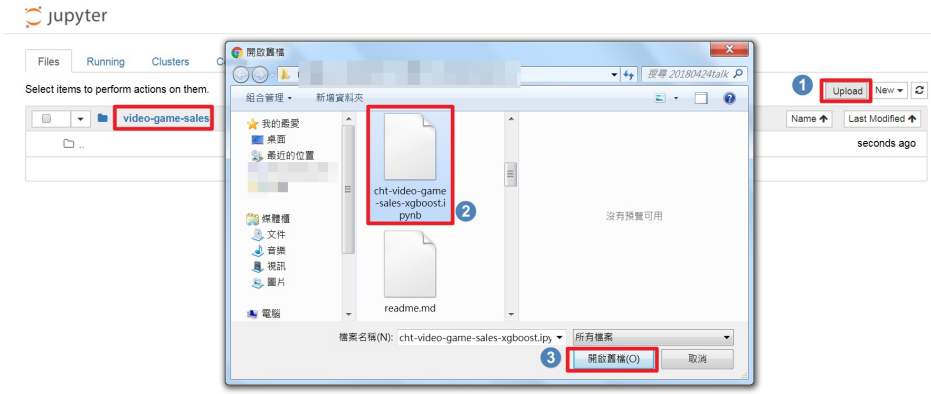
2. 探索資料

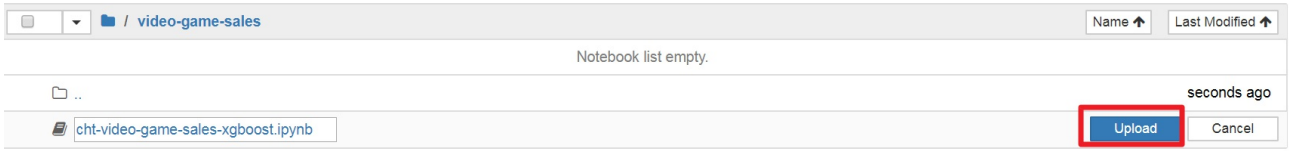
選擇New建立新的資料夾並改名為`video-game-sales`





點選進入新建的資料夾並上傳檔案cht-video-game-sales-xgboost.ipynb





進入Notebook開始開發

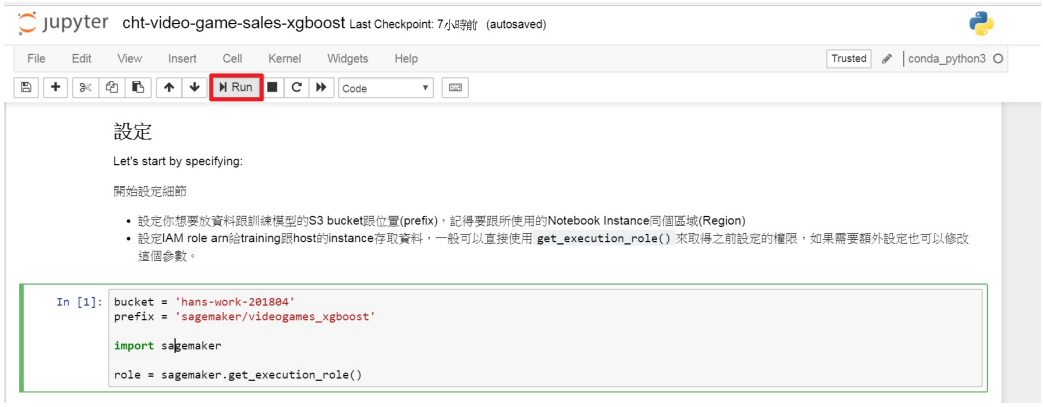
簡易介紹Jupyter Notebook使用方式，Jupyter Notebook是一個資料科學家常使用的開發工具，舉例來說，剛剛上傳的cht-video-game-sales-xgboost.ipynb為一個Notebook，一個Notebook又由一個個Cell組成程式碼來達到與資料分析跟互動。 紅框框起來的部分是編輯程式碼及執行的地方，稱為Cell， 透過滑鼠左建點擊選取想要的Cell 使用Shift + Enter來執行該

個Cell內的程式碼

- 設定IAM role arn給training跟host的instance存取資料，一般可以直接使用 `get_execution_role()` 來取得之前設定的權限，如果需要額外設定也可以修改這個參數。

```
In [1]: bucket = 'hans-work-201804'
        prefix = 'sagemaker/videogames_xgboost'

        import sagemaker

        role = sagemaker.get_execution_role()
```

或選Run也可以執行該Cell的程式碼



執行中的Cell會顯示左側框框(In[ ])會顯示*號

```
In [1]: bucket = 'hans-work-201804'
        prefix = 'sagemaker/videogames_xgboost'

        import sagemaker

        role = sagemaker.get_execution_role()
```

接下來載入所需要的Python套件

```
In [*]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        from IPython.display import Image
        from IPython.display import display
        from sklearn.datasets import dump_svmlight_file
        from time import gmtime, strftime
        import sys
        import math
        import json
        import boto3
```

修改S3 Bucket成自己剛剛建立的Bucket名稱

開始設定細節

- 設定你想要放資料跟訓練模型的S3 bucket跟位置(prefix)，記得要跟所使用的Notebook Instance同個區域(Region)
- 設定IAM role arn給training跟host的instance存取資料，一般可以直接使用 `get_execution_role()` 來取得之前設定的權限，如果需要額外設定也可以修改這個參數。

```
In [ ]: bucket = '<your-s3-bucket-name>'
        prefix = 'sagemaker/videogames_xgboost'

        import sagemaker

        role = sagemaker.get_execution_role()
```

接下來可以照著Notebook上的指示開發

請注意!!! 本程式碼腳本會上下互相依賴，請務必照順序執行Cell，請勿隨意重新整理網頁，如果重新整理網頁之前執行的內容會消失

Module2 Introduction to ML Training Service 約15分鐘

建立一個資料科學家運算跟分析機器

1. 建立ML Training Service Instance

   設定要訓練的工作，例如要用什麼機器，在哪個Region

訓練模型

資料已經準備完畢，接下來是使用XGBoost model來訓練。XGBoost有很多超參數可以調整來優化模型，以下列出比較會用到的參數。

- max_depth：樹的深度，太小會不足以反應資料的特性，太多會導致過度擬合
- eta：學習的步驟大小，用來避免過度擬合
- eval_metric：用來評估模型的指標
- scale_pos_weight：如果遇到不平衡的分佈可以用這個參數來作調整

直接開始訓練吧!

In [59]:
```
job_name = 'videogames-xgboost-' + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
print("Training job", job_name)

containers = {
                'us-west-2': '433757028032.dkr.ecr.us-west-2.amazonaws.com/xgboost:latest',
                'us-east-1': '811284229777.dkr.ecr.us-east-1.amazonaws.com/xgboost:latest',
                'us-east-2': '825641698319.dkr.ecr.us-east-2.amazonaws.com/xgboost:latest',
                'eu-west-1': '685385470294.dkr.ecr.eu-west-1.amazonaws.com/xgboost:latest'
              }

create_training_params = \
{
    "RoleArn": role,
    "TrainingJobName": job_name,
```

In [17]:
```
job_name = 'videogames-xgboost-' + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
print("Training job", job_name)

containers = {
                'us-west-2': '433757028032.dkr.ecr.us-west-2.amazonaws.com/xgboost:latest',
                'us-east-1': '811284229777.dkr.ecr.us-east-1.amazonaws.com/xgboost:latest',
                'us-east-2': '825641698319.dkr.ecr.us-east-2.amazonaws.com/xgboost:latest',
                'eu-west-1': '685385470294.dkr.ecr.eu-west-1.amazonaws.com/xgboost:latest'
              }

create_training_params = \
{
    "RoleArn": role,
    "TrainingJobName": job_name,
    "AlgorithmSpecification": {
        "TrainingImage": containers[boto3.Session().region_name],
        "TrainingInputMode": "File"
    },
    "ResourceConfig": {
        "InstanceCount": 1,
        "InstanceType": "ml.m4.xlarge",
        "VolumeSizeInGB": 10
    },
```

   XGBoost演算法要輸出模型的位置以及超參數等設定

```
    ],
    "OutputDataConfig": {
        "S3OutputPath": "s3://{}/{}/xgboost-video-games/output".format(bucket, prefix)
    },
    "HyperParameters": {
        "max_depth":"3",
        "eta":"0.1",
        "eval_metric":"auc",
        "scale_pos_weight":"2.0",
        "subsample":"0.5",
        "objective":"binary:logistic",
        "num_round":"100"
    },
    "StoppingCondition": {
        "MaxRuntimeInSeconds": 60 * 60
    }
}

Training job videogames-xgboost-2018-04-18-07-02-00
```

   開始訓練，這邊準備的範例約需要訓練6-10分鐘

```
Training job videogames-xgboost-2018-04-09-07-27-01
```

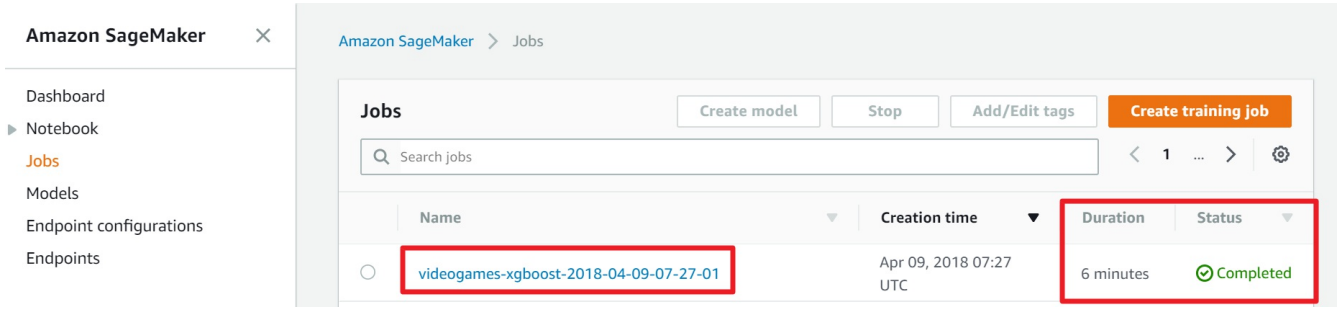
```
In [*]: %%time

sm = boto3.client('sagemaker')
sm.create_training_job(**create_training_params)

status = sm.describe_training_job(TrainingJobName=job_name)['TrainingJobStatus']
print(status)

try:
    sm.get_waiter('training_job_completed_or_stopped').wait(TrainingJobName=job_name)
finally:
    status = sm.describe_training_job(TrainingJobName=job_name)['TrainingJobStatus']
    print("Training job ended with status: " + status)
    if status == 'Failed':
        message = sm.describe_training_job(TrainingJobName=job_name)['FailureReason']
        print('Training failed with the following error: {}'.format(message))
        raise Exception('Training job failed')
```

```
InProgress
```

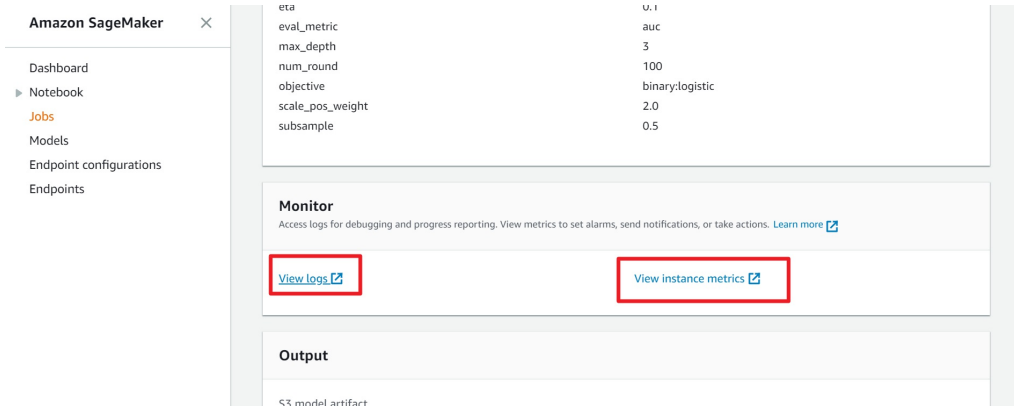
2. 監控訓練狀態

可以回到AWS的Console看到UI介面上的設定

訓練的狀態



CloudWatch會紀錄訓練的log



Module3 Introduction to ML Hosting Service 約15分鐘

建立一個機器學習提供預測或分群服務使用

1. 建立ML Hosting Service Instance

   指定模型

部署模型

現在將訓練好的模型部署到serverless的結點作服務

1. 指定container
2. 指定使用model
3. 建立服務model

In [61]:
```
create_model_response = sm.create_model(
    ModelName=job_name,
    ExecutionRoleArn=role,
    PrimaryContainer={
        'Image': containers[boto3.Session().region_name],
        'ModelDataUrl': sm.describe_training_job(TrainingJobName=job_name)['ModelArtifacts']['S3ModelArtifacts']})

print(create_model_response['ModelArn'])
```

```
arn:aws:sagemaker:us-east-1:326623638833:model/videogames-xgboost-2018-04-09-07-27-01
```

   設定EC2規格

建立服務

1. 決定EC2 instance的機型
2. 決定EC2的數量
3. 決定model名稱

設定好後建立endpoint

In [62]:
```
xgboost_endpoint_config = 'videogames-xgboost-endpoint-config-' + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
print(xgboost_endpoint_config)
create_endpoint_config_response = sm.create_endpoint_config(
    EndpointConfigName=xgboost_endpoint_config,
    ProductionVariants=[{
        'InstanceType': 'ml.t2.medium',
        'InitialInstanceCount': 1,
        'ModelName': job_name,
        'VariantName': 'AllTraffic'}])

print("Endpoint Config Arn: " + create_endpoint_config_response['EndpointConfigArn'])
```

   建立服務的Endpoint約需要8-10分鐘

In [*]:
```
%%time

xgboost_endpoint = 'EXAMPLE-videogames-xgb-endpoint-' + strftime("%Y%m%d%H%M", gmtime())
print(xgboost_endpoint)
create_endpoint_response = sm.create_endpoint(
    EndpointName=xgboost_endpoint,
    EndpointConfigName=xgboost_endpoint_config)
print(create_endpoint_response['EndpointArn'])

resp = sm.describe_endpoint(EndpointName=xgboost_endpoint)
status = resp['EndpointStatus']
print("Status: " + status)

try:
    sm.get_waiter('endpoint_in_service').wait(EndpointName=xgboost_endpoint)
finally:
    resp = sm.describe_endpoint(EndpointName=xgboost_endpoint)
    status = resp['EndpointStatus']
    print("Arn: " + resp['EndpointArn'])
    print("Status: " + status)

    if status != 'InService':
        message = sm.describe_endpoint(EndpointName=xgboost_endpoint)['FailureReason']
        print('Endpoint creation failed with the following error: {}'.format(message))
        raise Exception('Endpoint creation did not succeed')
```

```
EXAMPLE-videogames-xgb-endpoint-201804090742
```

2. 開始評估模型

```
In [64]: runtime = boto3.client('runtime.sagemaker')
```

```
In [65]: def do_predict(data, endpoint_name, content_type):
             payload = '\n'.join(data)
             response = runtime.invoke_endpoint(EndpointName=endpoint_name,
                                                ContentType=content_type,
                                                Body=payload)
             result = response['Body'].read()
             result = result.decode("utf-8")
             result = result.split(',')
             preds = [float((num)) for num in result]
             preds = [round(num) for num in preds]
             return preds

         def batch_predict(data, batch_size, endpoint_name, content_type):
             items = len(data)
             arrs = []

             for offset in range(0, items, batch_size):
                 if offset+batch_size < items:
                     results = do_predict(data[offset:(offset+batch_size)], endpoint_name, content_type)
                     arrs.extend(results)
                 else:
                     arrs.extend(do_predict(data[offset:items], endpoint_name, content_type))
                 sys.stdout.write('.')
             return(arrs)
```

建立混淆矩陣 (Confusion Matrix)

```
In [66]: %%time
         import json

         with open('test.libsvm', 'r') as f:
             payload = f.read().strip()

         labels = [int(line.split(' ')[0]) for line in payload.split('\n')]
         test_data = [line for line in payload.split('\n')]
         preds = batch_predict(test_data, 100, xgboost_endpoint, 'text/x-libsvm')

         print ('\nerror rate=%f' % ( sum(1 for i in range(len(preds)) if preds[i]!=labels[i]) /float(len(preds))))
```

```
.........
error rate=0.134496
CPU times: user 24 ms, sys: 8 ms, total: 32 ms
Wall time: 1.04 s
```

```
In [ ]:
```

```
In [67]: pd.crosstab(index=np.array(labels), columns=np.array(preds))
```

Out[67]:

col_0	0	1
row_0		
0	622	49
1	59	73

計算預測的準確率

```
In [67]: pd.crosstab(index=np.array(labels), columns=np.array(preds))
```

Out[67]:

col_0	0	1
row_0		
0	622	49
1	59	73

欄 0 1 代表預測狀況 列 0 1 代表實際狀況

從矩陣可以分析

622 + 49 + 59 + 73 = 803

判斷正確的狀況為 622 + 73 / 803 = 0.8655 判斷錯誤的狀況為 49 + 59 / 803 = 0.1345

從矩陣可以分析看到整體正確率為86.55% 從矩陣可以分析看到整體誤判率為13.45%

Clean Up

清除你的環境避免額外的收費

1. 清除ML Hosting Service (清除Endpoints)

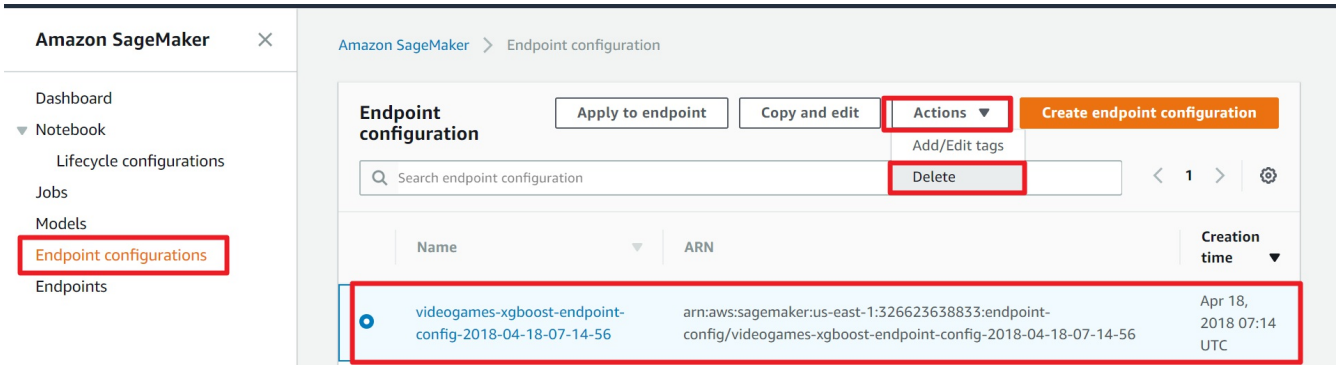
額外評估

可以透過其他方式來增加準確率, 例如調整超參數跟作特徵工程。

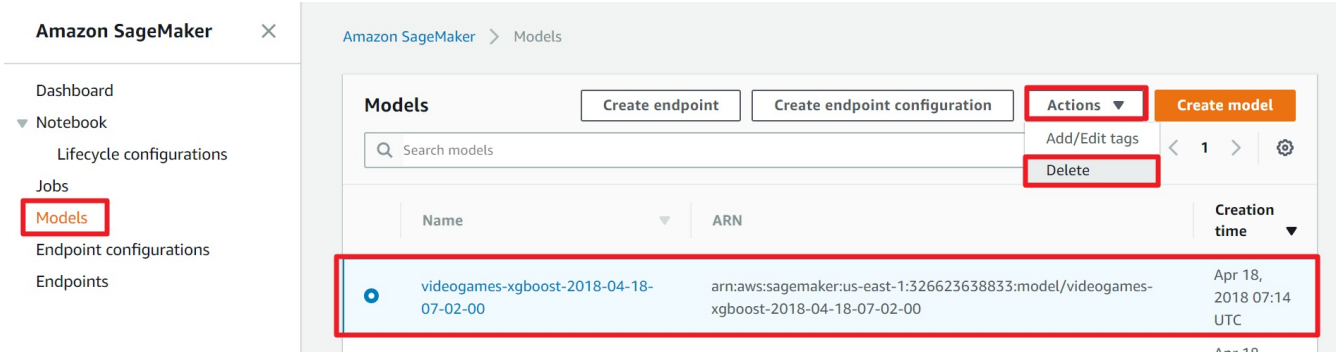
```
In [38]: sm.delete_endpoint(EndpointName=xgboost_endpoint)
```

```
Out[38]: {'ResponseMetadata': {'HTTPHeaders': {'connection': 'keep-alive',
            'content-length': '0',
            'content-type': 'application/x-amz-json-1.1',
            'date': 'Tue, 17 Apr 2018 09:19:06 GMT',
            'x-amzn-requestid': '2d292c20-f37e-4fe5-9b47-81c5d3f06227'},
           'HTTPStatusCode': 200,
           'RequestId': '2d292c20-f37e-4fe5-9b47-81c5d3f06227',
           'RetryAttempts': 0}}
```

2. 清除Endpoint Configuration

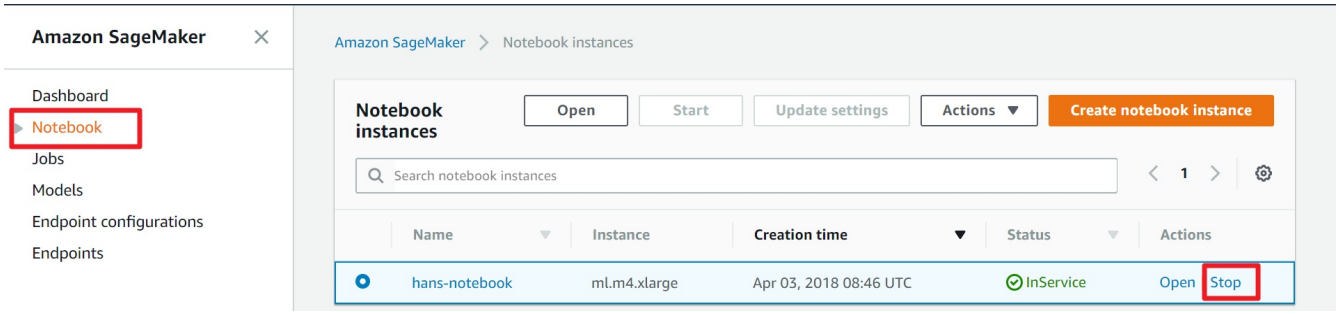


3. 清除Models

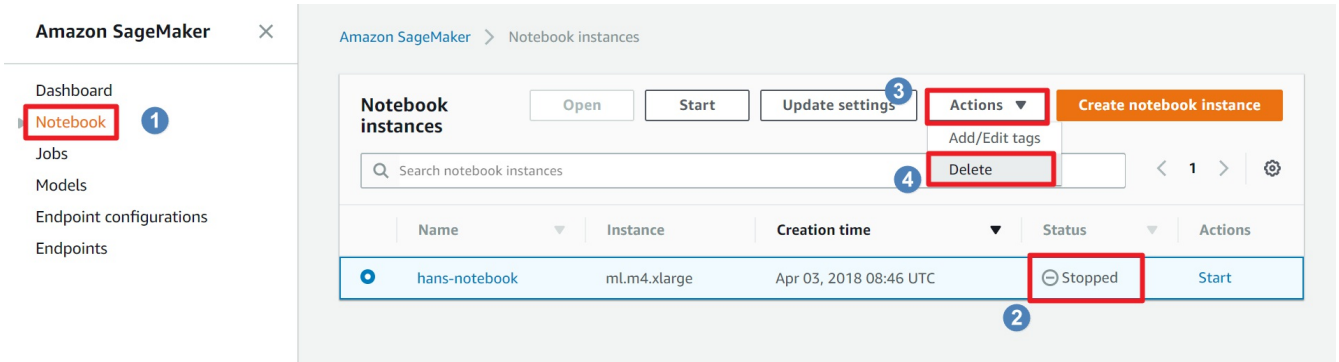


4. 清除Notebook

先停止

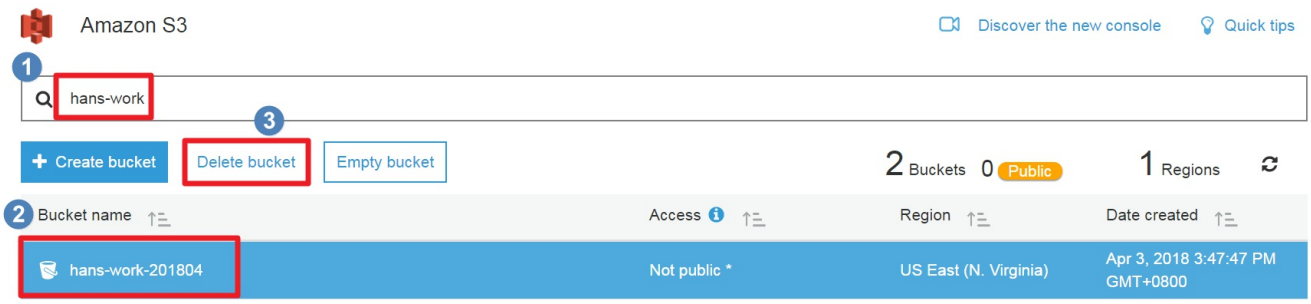


確認停止後刪除

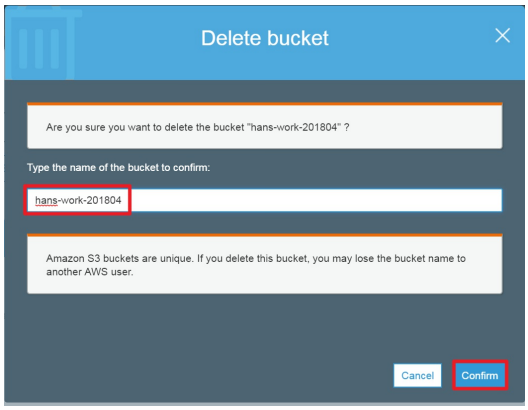


5. 清除S3

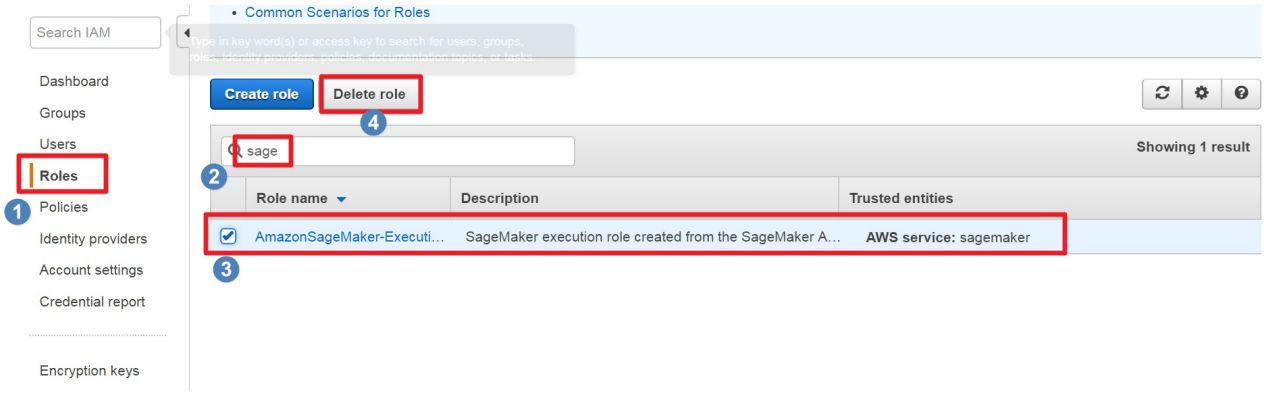
搜尋剛剛建立的S3 Bucket並選擇刪除

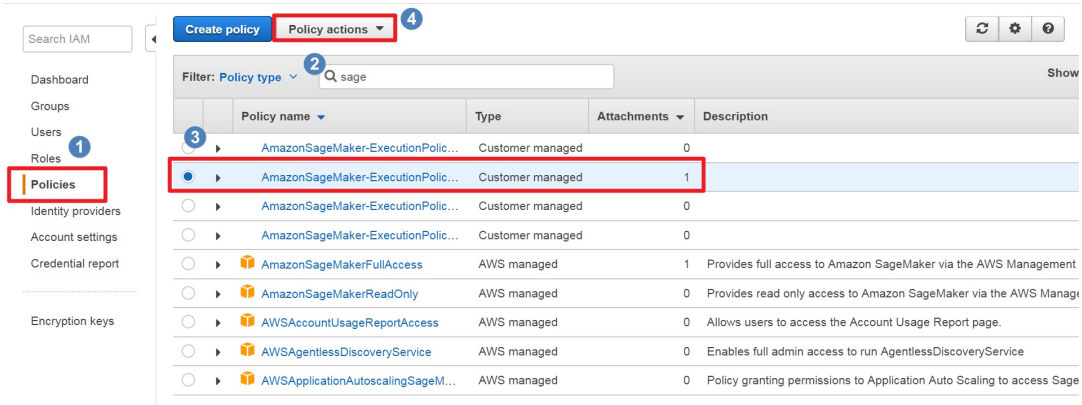


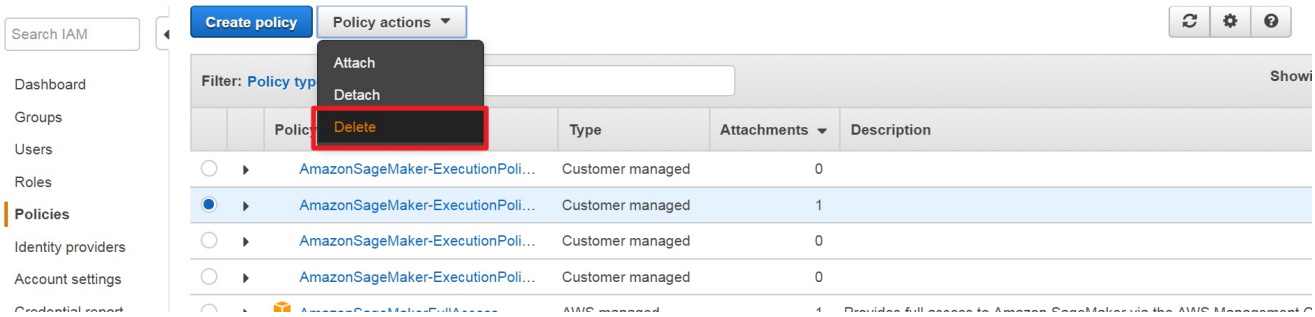
填入想要刪除的S3 Buckcet名稱後按確認



6. 清除IAM






Search IAM
Dashboard
Groups
Users
Roles
Policies
Identity providers
Account settings
Create policy
Policy actions
Attach
Detach
Delete
Filter: Policy typ
Showi
Type
Attachments
Description
AmazonSageMaker-ExecutionPoli...
Customer managed
0
AmazonSageMaker-ExecutionPoli...
Customer managed
1
AmazonSageMaker-ExecutionPoli...
Customer managed
0
AmazonSageMaker-ExecutionPoli...
Customer managed
0