

## HW5

組員: 0853409 張舒婷 0853420 林若瑜 0853421 郭源芯

### 一、截圖與步驟描述

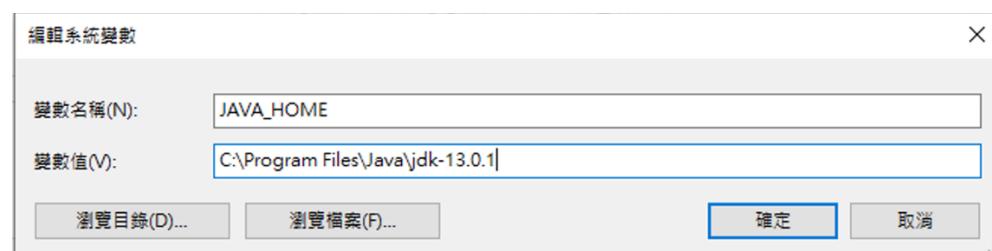
在 windows 環境底下安裝遇到了不少問題，因此也將問題和解決方法記錄起來

#### (一) 下載相關檔案



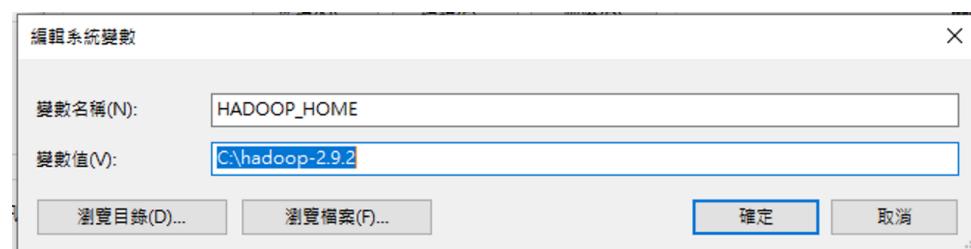
#### (二) 安裝 JDK 並設置環境變量

JAVA 安裝目錄在 C:\Program Files\Java，需要新建系統變量 JAVA\_HOME，值為 C:\Program Files\Java\jdk-13.0.1



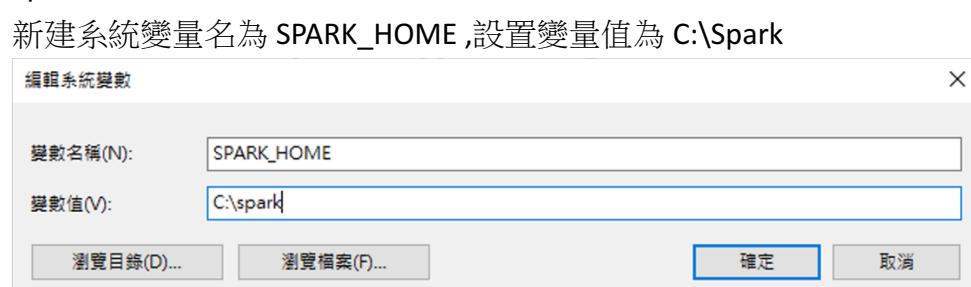
解壓 hadoop-2.9.2.tar.gz 到 C 盤根目錄

新建系統變量名 HADOOP\_HOME， 變量值為 C:\hadoop-2.9.2



解壓 spark-3.0.0-bin-hadoop3.2.tgz 到 C 盤根目錄，spark-3.0.0 目錄重命名為 Spark。

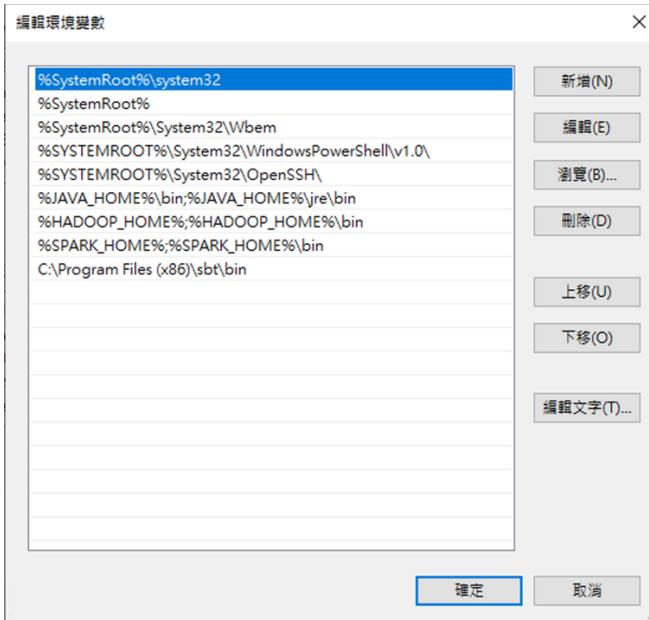
新建系統變量名為 SPARK\_HOME ,設置變量值為 C:\Spark



在 PATH 中加入

%JAVA\_HOME%\bin;%JAVA\_HOME%\jre\bin

%HADOOP\_HOME%;%HADOOP\_HOME%\bin  
 %SPARK\_HOME%;%SPARK\_HOME%\bin



配置 hadoop 文件（編輯 C:\hadoop-2.9.2\etc\hadoop 下的四個檔案）：

Name	Date modified	Type	Size
core-site.xml	3/30/2018 5:31 AM	XML Document	1 KB
hadoop-env.sh	3/30/2018 5:31 AM	Windows Command File	4 KB
hadoop-env.sh	3/30/2018 5:52 AM	SH File	16 KB
hadoop-metrics2.properties	3/30/2018 5:31 AM	PROPERTIES File	4 KB
hadoop-policy	3/30/2018 5:31 AM	XML Document	11 KB
hadoop-user-functions.sh.example	3/30/2018 5:31 AM	EXAMPLE File	4 KB
hdfs-site.xml	3/30/2018 5:33 AM	XML Document	1 KB
httpfs-env.sh	3/30/2018 5:33 AM	SH File	2 KB
httpfs-log4j.properties	3/30/2018 5:33 AM	PROPERTIES File	2 KB
httpfs-signature.secret	3/30/2018 5:33 AM	SECRET File	1 KB
httpfs-site.xml	3/30/2018 5:33 AM	XML Document	1 KB
kms-acls	3/30/2018 5:31 AM	XML Document	4 KB
kms-env.sh	3/30/2018 5:31 AM	SH File	2 KB
kms-log4j.properties	3/30/2018 5:31 AM	PROPERTIES File	2 KB
kms-site.xml	3/30/2018 5:31 AM	XML Document	1 KB
log4j.properties	3/30/2018 5:31 AM	PROPERTIES File	14 KB
mapred-env.sh	3/30/2018 5:44 AM	Windows Command File	1 KB
mapred-env.sh	3/30/2018 5:44 AM	SH File	2 KB
mapred-queues.xml.template	3/30/2018 5:44 AM	TEMPLATE File	5 KB
mapred-site.xml	3/30/2018 5:44 AM	XML Document	1 KB
ssl-client.xml.example	3/30/2018 5:31 AM	EXAMPLE File	3 KB
ssl-server.xml.example	3/30/2018 5:31 AM	EXAMPLE File	3 KB
user_ec_policies.xml.template	3/30/2018 5:33 AM	TEMPLATE File	3 KB
workers	3/30/2018 5:31 AM	File	1 KB
yarn-env.sh	3/30/2018 5:43 AM	Windows Command File	3 KB
yarn-env.sh	3/30/2018 5:43 AM	SH File	6 KB
yarnservice-log4j.properties	3/30/2018 5:43 AM	PROPERTIES File	3 KB
yarn-site.xml	3/30/2018 5:43 AM	XML Document	1 KB

### 1. 編輯 C:\hadoop-2.9.2\etc\hadoop 下的 core-site.xml

```
<configuration>
<property>
    <name>hadoop.tmp.dir</name>
    <value>/C:/hadoop/workplace/tmp</value>
</property>
<property>
    <name>dfs.name.dir</name>
    <value>/C:/hadoop/workplace/name</value>
</property>
<property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:9000</value>
</property>
</configuration>
```

2. 編輯 C:\hadoop-2.9.2\etc\hadoop 下的 mapred-site.xml

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
  <property>
    <name>mapred.job.tracker</name>
    <value>hdfs://localhost:9001</value>
  </property>
</configuration>
```

3. 編輯 C:\hadoop-2.9.2\etc\hadoop 下的 hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.data.dir</name>
    <value>C:/hadoop/workplace/data</value>
  </property>
</configuration>
```

4. 編輯 C:\hadoop-2.9.2\etc\hadoop 下的 yarn-site.xml

```
<configuration>
  <!-- Site specific YARN configuration properties -->
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
  <property>
    <name>yarn.resourcemanager.address</name>
    <value>140.113.72.196:8032</value>
  </property>
  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>140.113.72.196:8030</value>
  </property>
  <property>
    <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>140.113.72.196:8035</value>
  </property>
  <property>
    <name>yarn.resourcemanager.admin.address</name>
    <value>140.113.72.196:8033</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.address</name>
    <value>140.113.72.196:8088</value>
  </property>
</configuration>
```

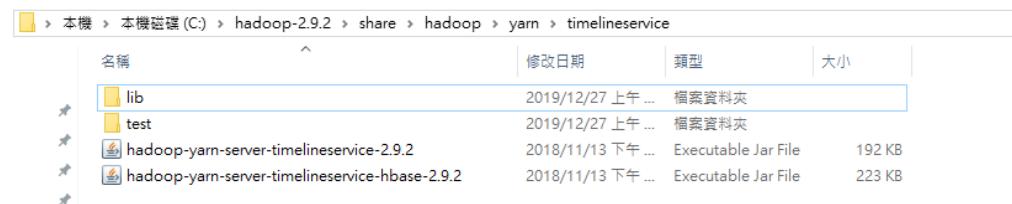
C:\hadoop-2.9.2\sbin 裡的 start-all.cmd 前面加 cd path 導至 bin

```
20
21 echo This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
22 cd C:\hadoop-2.9.2\bin
```

若沒有加上導至 bin 的程式碼，再後面的步驟就會一直出現以下截圖的錯誤

```
C:\hadoop-2.9.2\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
系統找不到檔案 hadoop。
系統找不到檔案 hadoop。
starting yarn daemons
系統找不到檔案 yarn。
系統找不到檔案 yarn。
```

複製 C:\hadoop-2.9.2\share\hadoop\yarn\timelineservice\hadoop-yarn-server-timelineservice-2.9.2 到上一層 C:\hadoop-2.9.2\share\hadoop\yarn

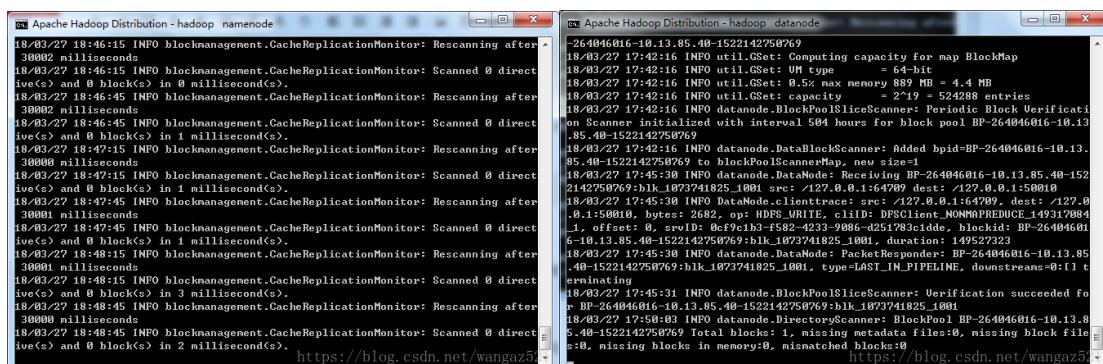


若沒有上面步驟，會出現 resource manager 在執行一小段時間後就關掉的狀況。這裡透過 export.hadoop.yarn.log 才觀察到的問題。

以上設定完成後先至 bin 底下下 hdfs namenode -format 指令

```
C:\hadoop-2.9.2\bin>hdfs namenode -format
19/12/27 10:24:48 INFO namenode.NameNode: STARTUP_MSG:
*****STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = DESKTOP-I6NJKCD/140.113.72.196
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 2.9.2
STARTUP_MSG: classpath = C:\hadoop-2.9.2\etc\hadoop;C:\hadoop-2.9.2\share\hadoop\common\lib\apacheds-i18n-2.0.0-M15.jar;C:\hadoop-2.9.2\share\hadoop\security\lib\*
```

在切換到 sbin 目錄執行 start-all.cmd，會啟動四個 process，並跳出以下四個視窗：



```

Apache Hadoop Distribution - yarn resourcemanager
三月 27, 2018 5:49:49 下午 com.sun.jersey.spi.container.GuiceComponentProviderFactory register
信息: Registering org.apache.hadoop.yarn.server.resourcemanager.webapp.RMWebServices as a root resource class
三月 27, 2018 5:49:49 下午 com.sun.jersey.spi.container.GuiceComponentProviderFactory register
信息: Registering org.apache.hadoop.yarn.webapp.GenericExceptionHandler as a provider class
三月 27, 2018 5:49:49 下午 com.sun.jersey.server.impl.application.WebApplicationImpl _initiate
信息: Initiating Jersey application, version 'Jersey: 1.9.0/02/2011 11:17 AM'
三月 27, 2018 5:49:49 下午 com.sun.jersey.spi.container.GuiceComponentProviderFactory getComponentProvider
信息: Binding org.apache.hadoop.yarn.server.resourcemanager.webapp.JAXBCContextResolver to GuiceManagedComponentProvider with the scope "Singleton"
三月 27, 2018 5:49:50 下午 com.sun.jersey.spi.container.GuiceComponentProviderFactory getComponentProvider
信息: Binding org.apache.hadoop.yarn.webapp.GenericExceptionHandler to GuiceManagedComponentProvider with the scope "Singleton"
三月 27, 2018 5:49:51 下午 com.sun.jersey.spi.container.GuiceComponentProviderFactory getComponentProvider
信息: Binding org.apache.hadoop.yarn.server.resourcemanager.webapp.RMWebServices to GuiceManagedComponentProvider with the scope "Singleton"
18/03/27 17:51:07 INFO logs: Aliases are enabled
https://blog.csdn.net/wangaz5

Apache Hadoop Distribution - yarn nodemanager
18/03/27 17:42:12 INFO http.HttpServer2: Jetty bound to port 8042
18/03/27 17:42:12 INFO northbay.log: jetty-6.1.26
18/03/27 17:42:12 INFO northbay.log: Extract jar file:/D:/hadoop/hadoop-2.5.2/share/hadoop/yarn/hadoop-yarn-common-2.5.2.jar!/webapps/node to C:/Users/WANGAN/Downloads\LocalTemp\Jetty_0_0_0_8042_node_19tJBxwebapp
18/03/27 17:42:12 INFO northbay.log: Started HttpServer2$SelectChannelConnectorWithSafeStartup@0.0.0.0:8042
18/03/27 17:42:12 INFO webapp.WebMpp: Web app /node started at 8042
18/03/27 17:42:14 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8081
18/03/27 17:42:14 INFO nodeanager.NodeStatusUpdaterImpl: Sending out 0 NM container statuses: []
18/03/27 17:42:14 INFO nodeanager.NodeStatusUpdaterImpl: Registering with RM using containerID: []
18/03/27 17:42:15 INFO security.NMContainerTokenSecretManager: Rolling master-key for nn-tokens, got key with id :71452497
18/03/27 17:42:15 INFO security.NMContainerTokenSecretManagerInNM: Rolling master-key for nn-tokens, got key with id :125595692
18/03/27 17:42:15 INFO nodeanager.NodeStatusUpdaterImpl: Registered with ResourceManager as JRMIE1ADCF7E8F.JDR.R.360bmyhD.local:64187 with total resource of <memory:8192, vCores:8>
18/03/27 17:42:15 INFO nodeanager.NodeStatusUpdaterImpl: Notifying ContainerManager to unblock new container requests
https://blog.csdn.net/wangaz5

```

在 cmd 到 java\jdk-version\bin 底下下 jps 確認正在執行的 process

```
C:\Java\jdk-13.0.1\bin>jps
10868 ResourceManager
16132 NameNode
7268
11656 NodeManager
9432 Jps
```

開啟網頁確認 localhost:8088 跟 localhost:50070 有出現以下兩個畫面

Cluster Metrics											
	Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores	
About	0	0	0	0	0	0 B	8 GB	0 B	0	8	
Nodes											
Node Labels											
Applications											
NEW											
NEW_SAVING											
SUBMITTED											
ACCEPTED											
RUNNING											
FINISHED											
FAILED											
KILLED											
Scheduler											
Tools											

All Applications

Cluster Nodes Metrics											
	Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes					
1	0	0	0	0	0	0					
Scheduler Metrics											
	Scheduler Type	Scheduling Resource Type	Minimum Allocation			Maximum Allocation	Maximum Cluster				
	Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>			<memory:8192, vCores:4>	0				
Show 20 ▾ entries											
ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores
											Allocated Memory MB
											Reserved CPU Vcores
											Reserved Memory MB
											% of Queue
											% of Cluster
											Pro

No data available in table

Showing 0 to 0 of 0 entries

Overview 'localhost:9000' (active)											
Started:	Fri Dec 27 11:51:09 +0800 2019										
Version:	2.9.2, r626afbeae31ca687bc2f8471dc841b66ed2c6704										
Compiled:	Tue Nov 13 20:42:00 +0800 2018 by ajisaka from branch-2.9.2										
Cluster ID:	CID-ffa63578-c4cb-43d1-895a-d571e593728f										
Block Pool ID:	BP-968912654-140.113.72.196-1577418654922										

## Summary

Security is off.  
Safemode is off.  
1 files and directories, 0 blocks = 1 total filesystem object(s).  
Heap Memory used 76.4 MB of 117 MB Heap Memory. Max Heap Memory is 1000 MB.  
Non Heap Memory used 36.73 MB of 40.38 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	0 B
DFS Used:	0 B (100%)
Non DFS Used:	0 B

連接 hdfs，創建目錄並丟檔案到上面，並測試檔案有辦法下載

```
C:\hadoop-2.9.2\bin>hadoop fs -mkdir /sample
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/C:/hadoop-2.9.2/share/hadoop/common/lib/hadoop-auth-2.9.2.jar) to method sun.security.krb5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
mkdir: '/sample': File exists

C:\hadoop-2.9.2\bin>hadoop fs -ls /
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/C:/hadoop-2.9.2/share/hadoop/common/lib/hadoop-auth-2.9.2.jar) to method sun.security.krb5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Found 1 items
drwxr-xr-x - fish supergroup 0 2019-12-27 14:22 /sample
```

### (三) spark 連接 worker 和 master(140.113.72.196)

```

C:\spark>cd bin
C:\spark\bin>spark-class org.apache.spark.deploy.worker.Worker spark://140.113.72.196:7077
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
19/12/25 19:40:51 INFO Worker: Started daemon with process name: 14692@67-0456744-H1
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/C:/spark/jars/spark-unsafe_2.12-3.0.0-preview.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
19/12/25 19:40:51 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
19/12/25 19:40:52 INFO SecurityManager: Changing view acls to: User
19/12/25 19:40:52 INFO SecurityManager: Changing modify acls to: User
19/12/25 19:40:52 INFO SecurityManager: Changing view acls groups to:
19/12/25 19:40:52 INFO SecurityManager: Changing modify acls groups to:
19/12/25 19:40:52 INFO SecurityManager: SecurityManager: authentication disabled; users with view permissions: Set(User); groups with view permissions: Set()
19/12/25 19:40:52 INFO SecurityManager: SecurityManager: authentication disabled; users with modify permissions: Set(User); groups with modify permissions: Set()
19/12/25 19:40:53 INFO Utils: Successfully started service 'sparkWorker' on port 54872.
19/12/25 19:40:53 INFO Worker: Starting Spark worker 140.113.72.218:54872 with 8 cores, 6.7 GiB RAM
19/12/25 19:40:53 INFO Worker: Running Spark version 3.0.0-preview
19/12/25 19:40:53 INFO Worker: Spark home: C:/spark
19/12/25 19:40:53 INFO ResourceUtils: -----
19/12/25 19:40:53 INFO ResourceUtils: Resources for spark.worker:
19/12/25 19:40:53 INFO ResourceUtils: -----
19/12/25 19:40:54 INFO Utils: Successfully started service 'WorkerUI' on port 8081.
19/12/25 19:40:54 INFO WorkerWebUI: Bound WorkerWebUI to 0.0.0.0, and started at http://67-0456744-H1:8081
19/12/25 19:40:54 INFO Worker: Connecting to master 140.113.72.196:7077...
19/12/25 19:40:54 INFO TransportClientFactory: Successfully created connection to /140.113.72.196:7077 after 94 ms (0 ms spent in bootstraps)

```

(四) 將 worker 連接 master(140.113.72.196)後完成連接的畫面(localhost:8080)

 **Spark Master at spark://140.113.72.196:7077**

**URL:** spark://140.113.72.196:7077  
**Alive Workers:** 4  
**Cores in use:** 32 Total, 32 Used  
**Memory in use:** 34.9 GiB Total, 4.0 GiB Used  
**Resources in use:**  
**Applications:** 1 Running, 0 Completed  
**Drivers:** 0 Running, 0 Completed  
**Status:** ALIVE

#### ▼ Workers (4)

Worker Id	Address	State	Cores	Memory	Resources
worker-20191225193720-140.113.72.201-53620	140.113.72.201:53620	ALIVE	8 (8 Used)	14.9 GiB (1024.0 MiB Used)	
worker-20191225193745-140.113.72.202-49859	140.113.72.202:49859	ALIVE	8 (8 Used)	6.7 GiB (1024.0 MiB Used)	
worker-20191225194053-140.113.72.218-54872	140.113.72.218:54872	ALIVE	8 (8 Used)	6.7 GiB (1024.0 MiB Used)	
worker-20191225195020-140.113.72.211-49191	140.113.72.211:49191	ALIVE	8 (8 Used)	6.7 GiB (1024.0 MiB Used)	

## 二、作業目標對應之結果

### (一) 將資料讀取進來(可用 pandas 套件)、資料前處理、把空值以 0 替代

```
# read files
global Path
sc = SparkContext()
if sc.master[0:5]=='local': #代表由本機執行，直接讀取本機檔案
    Path="file:///"
else: #在cluster執行，讀取hdfs檔案
    Path="hdfs://master:9000/sample/"

df = pd.read_csv(Path+'character-deaths.csv')

# fill blanks
df = df.fillna(0)
```

### (二) Death Year , Book of Death , Death Chapter 三者取一個，將有數值的轉成

1、將 Allegiances 轉成 dummy 特徵(底下有幾種分類就會變成幾個特徵，值是 0 或 1，本來的資料集就會再增加約 20 種特徵)

```
# turn num into 1
bookofdeath = np.zeros(df.shape[0])
for i in range(df.shape[0]):
    if (df['Book of Death'][i] != 0):
        bookofdeath[i] = 1
df.insert(2,'book of death',bookofdeath)

# drop Death Year, Death Chapter
df = df.drop("Book of Death", axis=1)
df = df.drop("Death Year", axis=1)
df = df.drop("Death Chapter", axis=1)

# 將Allegiances轉成dummy
dummies_1 = pd.get_dummies(df, columns=['Allegiances'], drop_first=True)
df = dummies_1
```

### (三) 亂數拆成訓練集(75%)與測試集(25%)把處理好的資料存進 train.csv 、

test.csv

```
# feature target
target = df.loc[:, ['book of death']]
df_feature = df.drop("book of death", axis=1)
features = df_feature

# 亂數拆成訓練集(75%)與測試集(25%)
#x_train, x_test, y_train, y_test = train_test_split(features, target, test_size=0.25)
#train=pd.concat([x_train,y_train],axis=0)
msk = np.random.rand(len(df)) < 0.75
train = df[msk]
test = df[~msk]

#sp_train = sparkSession.createDataFrame(train)
#sp_train.write_csv('hdfs://cluster/user/hdfs/train.csv')
#sp_test = sparkSession.createDataFrame(test)
#sp_test.write_csv('hdfs://cluster/user/hdfs/test.csv')

train.to_csv('train.csv', index=None, header=True)
test.to_csv('test.csv', index=None, header=True)
```

### (四) 讀資料並把資料轉成 RDD 檔

```
train_rawDataWithHeader=sc.textFile(Path+'train.csv')
test_rawDataWithHeader=sc.textFile(Path+'test.csv')
#train_data=prepare_data(train)
header = train_rawDataWithHeader.first()
rawData = train_rawDataWithHeader.filter(lambda x: x != header)
rData = rawData.map(lambda x: x.replace("\n", ""))
train_data = rData.map(lambda x: x.split(","))
#test_data=prepare_data(test)
header = test_rawDataWithHeader.first()
rawData = test_rawDataWithHeader.filter(lambda x: x != header)
rData = rawData.map(lambda x: x.replace("\n", ""))
test_data = rData.map(lambda x: x.split(","))

lines=train_data+test_data
categoriesMap = lines.map(lambda fields: fields[0]).distinct().zipWithIndex().collectAsMap()
train_RDD=train_data.map(lambda r:LabeledPoint(extract_label(r),extract_features(r,categoriesMap,len(r)-1)))
test_RDD=test_data.map(lambda r: (extract_features(r,categoriesMap,len(r)-1),r[-1]))
```

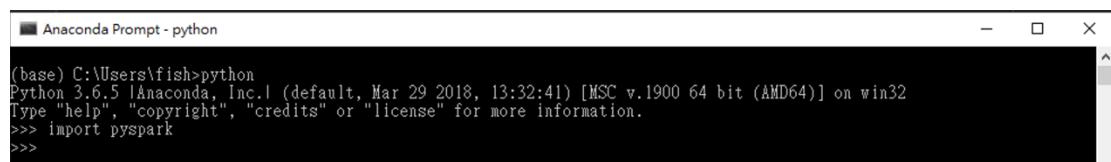
## (五) 建立決策樹模型、產出預測結果、計算 accuracy, recall, precision

```
model=DecisionTree.trainClassifier(train_RDD,numClasses=2,categoricalFeaturesInfo={},impurity='entropy',maxDepth=14,maxBins=9)

count = 0
num = 0
positive = 0
negative = 0
truePositive = 0
trueNegative = 0
falsePositive = 0
falseNegative = 0
for data in test_RDD.take(test_data.count()):
    num+=1
    preds = int(model.predict(data[0]))
    #print(str(preds)+':'+str(data[1]))
    if(preds == int(data[1])):count=count+1
    if int(data[1]) == 0: negative += 1
    if int(data[1]) == 1: positive += 1
    if (int(preds) == 1 and int(data[1]) == 1) : truePositive +=1
    if (int(preds) == 0 and int(data[1]) == 1) : falseNegative +=1
    if (int(preds) == 0 and int(data[1]) == 0) : trueNegative +=1
    if (int(preds) == 1 and int(data[1]) == 0) : falsePositive +=1
acc = count/num
precision = float(truePositive) / (float(truePositive) + float(falsePositive))
recall = float(truePositive) / (float(truePositive) + float(falseNegative))
print('Accuracy:' + str(acc))
print('Precision:' + str(precision))
print('Recall:' + str(recall))
```

## (六) 執行程式

需要先導入 pyspark



```
Anaconda Prompt - python
(base) C:\Users\fish>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import pyspark
>>>
```

使用 pycharm 執行以上程式

執行期間打開 localhost:8080 可以看到目前有 Spark shell 的應用正在運行（如下圖所示）：

### ▼ Running Applications (1)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20191225211405-0000 (kill)	pyspark-shell	32	1024.0 MiB		2019/12/25 21:14:05	fish	RUNNING	5 s

## (七) 實際產出結果

```
Accuracy:0.9591836734693877
Precision:0.5
Recall:0.8
```

討論：

我們直接將 spark 裝在 windows 系統上，因此跟書上步驟不同，過程有遇到不同的問題，不過最後還是可以順利預測結果。

參考資料：

<https://kknews.cc/zh-tw/code/nrxbg52.html>

<http://www.lawrencelin.info/2017/11/spark-on-yarn-in-windows/>

<https://www.cnblogs.com/chevin/p/9090683.html>

<https://zhuanlan.zhihu.com/p/29362852>

<https://blog.csdn.net/wangaz521/article/details/79717177>

<https://www.codementor.io/@jadianes/spark-python-mllib-decision-trees-du107qr0j>