出國報告（出國類別：研究）

環保大數據分析及區塊鏈技術研究

服務機關：行政院環保署
姓名職稱：魏文娟分析師

派赴國家：美國紐約
出國期間：**106** 年 **8** 月 **29** 日**~11** 月 **28** 日
報告日期：**107** 年 **2** 月 **26** 日

摘 要

本研究係行政院人事行政總處 106 年國外專題研究計畫，以「環保大數據分析及區塊鏈技術研究」為主題，規劃從實務與理論並進的研習著手，提升大數據分析、機器學習相關之資料科學及區塊鏈技術領域技術；正所謂科學技術需要「迎頭趕上」，筆者赴美實地研習美國哥倫比亞大學研究所層級之「大數據分析」課程，並對區塊鏈技術進行研究分析，說明其適用特性。運用國外研習時間進行專題研究，俾提供我國行政院環保署大數據分析相關運用參考。

本研究計畫經綜整美國哥倫比亞大學對資訊科學專業碩士學程研究生之教學課程及訓練資料深入研究，並藉於AWS雲端運算，利用PYSPARK及TensorFlow實作大數據分析應用於環境相關資料，藉由實務與理論相互映證，歸納研究大數據分析及人工智慧領域的挑戰與前景，分別就「英文能力提升」、「改變想法及做法」、「破壞性創新」及對想要在組織內啟動AI技術的建議做法等 4 個面向提出建議。

目　　錄

壹、源起與目的

壹、源起與目的

行政院環境保護署環境監測及資訊處於環境背景資料蒐集工作不餘遺力，因應電子化政府及資料開放政策，並建置有環境資源資料開放平臺、環境資源資料庫等，累積許多環境資源資料，面對大數據分析技術躍進，為了趕上新科技的腳步，一窺累積多年的環境資料庫究竟能帶給我們多少環境治理的金科玉律？因而再次引發了我追求技術的熱忱。

近年來資料科學、機器學習、深度學習等研究技術突飛猛進，主要基於下列三個因素：

一、GPU(graphics processing unit, NVIDIA)、TPU(著眼於ASIC專屬積體電路設計，旨於網路上執行各種神經網路模型之 CISC 指令，Google)等加速運算器的出現，其運算速度比電腦CPU快10~100倍，加上Google Cloud、Amazon AWS及Microsoft Azure等網路雲端運算服務，讓電腦計算能力大幅提升。

二、網際網路的蓬勃發展累積了大量資訊，不論在社群網路或是電子商務應用上，均提供機器學習與深度學習可加運用之訓練資料及測試資料。

三、各式演算法及程式開放源碼的建置(如Caffe、CNTK、Theano 、TensorFlow等)，讓數學、統計學所發展的各項模型演算程式迅速累積，透過程式開放平台（如github平台）提供給大眾使用。

隨著大數據(BigData)時代的來臨，誰能在龐雜的訊息中掌握關鍵分析技術，誰就能取得解決問題的先機。環保署規劃的前瞻計畫將全方位發展環境品質物聯網，建立更精準的環境品質服務平台，是以必須進一步提升資料的運用，職為因應未來物聯網（IOT）和雲端運算(Cloud Computing)之智能化(AI)與信息化(Information exchange)之需求，故規劃從實務與理論並進的研習著手，向國外取經，前往美國紐約哥倫比亞大學研習大數據分析課程，進行專題研究。

貳、過程

貳、過程

一、 研習進度規劃：

依行政院人事行政總處的規定之專案研究期限內，於 106 年 8 月 29 日出發前往美國紐約，進行 3 個月的研習，並於 11 月 28 日返國，詳細地研習進度規劃表如下：

研習進度	研習主題
8 月 29 日~ 9 月 8 日	大數據趨勢及應用 Introduction to Big Data trends and Applications
9 月 9 日~9 月 14 日	大數據平台 Big Data Platforms
9 月 15 日~9 月 21 日	大數據存儲和分析 Big Data Storage and Analytics
9 月 22 日~10 月 11 日	大數據分析機器學習演算法(3 週) Big Data Analytics ML Algorithms
10 月 12 日~10 月 18 日	機器推理 Machine Reasoning
10 月 19 日~10 月 25 日	影響環境因素大數據圖形關聯 Environmental factors linked Big Data Graph Computing
10 月 26 日~11 月 1 日	運用圖形運算分析影響環境因素 Graph Analytics on environmental factors
11 月 2 日~11 月 8 日	區塊鏈技術研究 Blockchain Technology for environmental protection application
11 月 9 日~11 月 16 日	影響環境因素大數據視覺化呈現 Environmental factors Visualization
11 月 17 日~11 月 28 日	環境保護因素區塊鏈建置評估 Environmental protection factors block chain construction evaluation

表 1 研習進度規劃表

二、 研究內容：

1、大數據趨勢及應用

“Big data is high-volume, high-velocity and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making.” —Gartner 2012

大數據依 Gartner 2012 定義為「高容量，高速度和多種類的信息資產，需要具有成本效益及創新形式之信息處理以制定具備洞察力之決策依據。」大數據發展在 2013 年風起雲湧，下圖為 Wikibo 率先在大數據分析(Big Data Analysis, BDA)領域的投資與市場方面的預測值，市場由 2013 年的銷售相關硬件，軟件和服務的供應商收入來衡量，達到了 186 億美元，與上一年相比增長 58%。

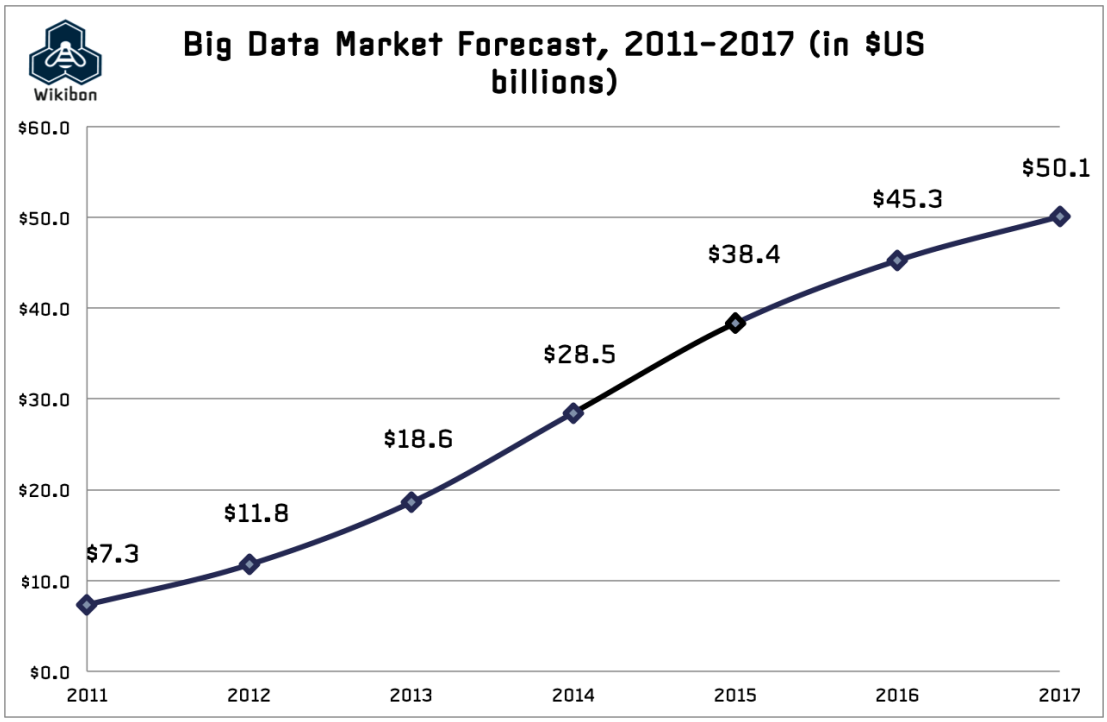


圖 1 BigData 2011~2017 市場預測 [1]

根據國際數據資訊(IDC)報導大數據領域投資及營收的實際數字 2016、 2017 年分別為 130、150 Billions，約達圖 1 預測的 3 倍左右，成長率分別為 11%、12.8%，並預估 2015~2020 年大數據分析領域將維持 2 位數以上之成長率。[2]

市場為何需要大數據分析技術？從圖 2 中列出包括創新技術、成本效率、數據成長、運算需求、分析需求、以及欲降低跨業門檻獲得成功等多項對大數據分析技術的需求。

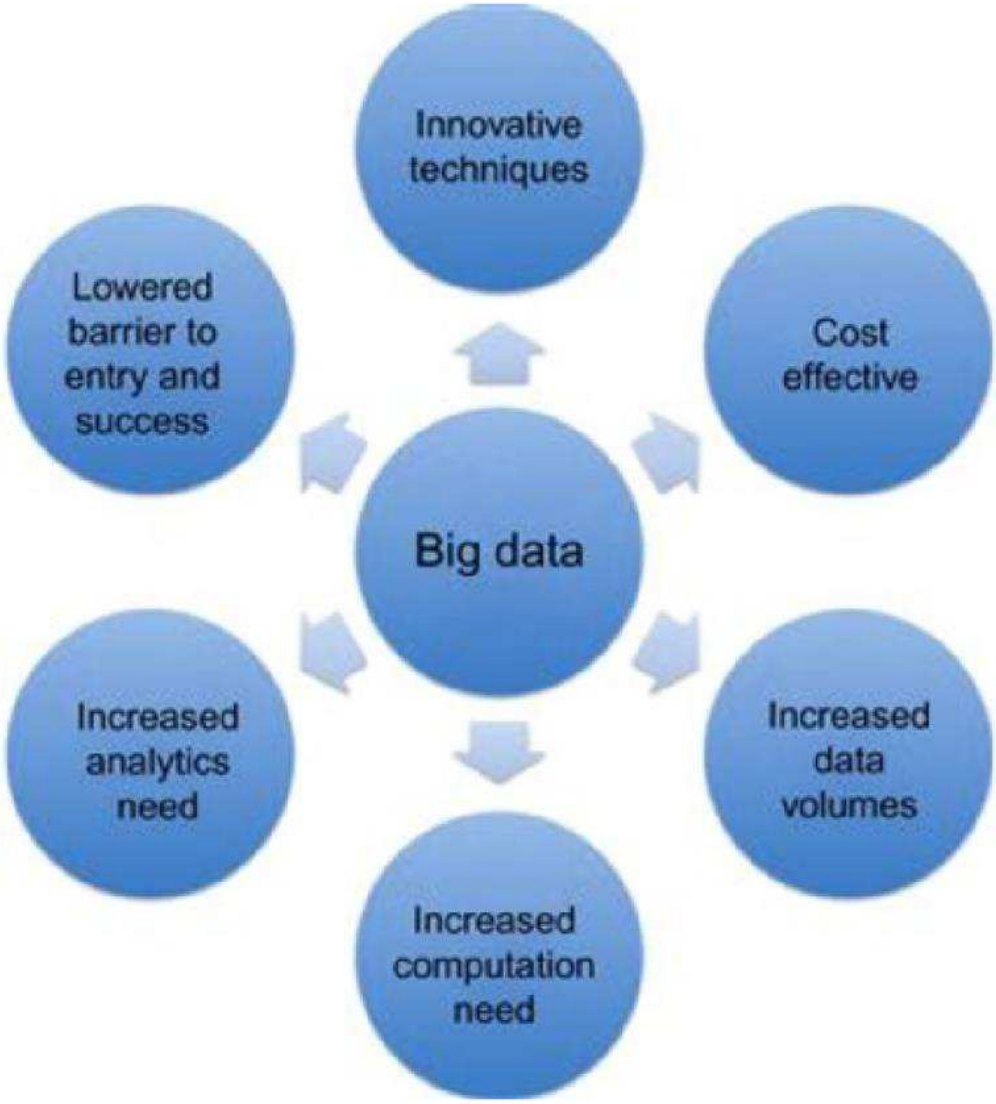


圖 2 What made Big Data needed? “Big Data Analytics”, David Loshin, 2013

大數據分析所需要的關鍵運算資源除了需要高速運算的 CPU processor 還有記憶體儲存空間以及網路頻寬速度，為了因應不斷成長的資料，系統擴充性(scalability)可依整體架構分為兩種，向上擴充(Scale UP)及向外擴充(Scale Out)其特徵為：

- 向上擴充(Scale UP))優點能充分利用資源，但缺點是需要考量個別系統架構差異進行系統演算法設計。

- 向外擴充(Scale Out)優點能獲得較多平行運算資源，缺點為分散儲存數據會造成 data access latency 變長。

配置原則為：

- 獨立的數據擴充時向上擴充並未比向外擴充產生明顯的優勢。
- 對有關聯的數據要儘量先向上擴充，才考慮向外擴充。

在 David Loshin 的 2013 年“Big Data Analytics”一書中列出傳統高性能應用功能特徵與新興大數據技術(當時主要是以 Hadoop 技術為主)之對比(表 2)，其分為 4 個面向：

(1)高性能應用典型作法是發展大量平行運算，開發者需有能力提高運算及進行系統優化與細部程式調整；大數據 Hadoop 則以其架構具可靠性、擴充性高並具有分散式及平行處理特性，是一個開源程式(免費)，其使用分散式檔案架構及分散式資料庫。

(2)高性能平台典型作法需採購昂貴的平行電腦、頻寬網路以及大量儲存設備；大數據 Hadoop 採用創新彈性可擴充的虛擬平台，充分將一般電腦硬體組成集群(clusters)資源或利用雲端設備加上開源的工作及技術。

(3)高性能資料管理典型作法受限於所使用的檔案系統或關聯性資料庫(其一般均以「列」為主的資料模式進行存取 Row Based data layout；大數據 Hadoop 有更多適切的替代方案如 NoSQL(Not Only SQL)，可針對業務特性，如採用快取記憶資料管理 In-Memory Data Management，或是以「行」為主的加速查詢(columnar layout to speed query response)，以及圖形資料庫(graph database)。

(4)高性能資源管理典型作法是採購高階產品，安裝並自行管理；大數據可利用 Hadoop 布署虛擬平台，讓中小型企業可利用雲端運算環境達成高速運算精進業務需求，不但務實有效且價格友善合理。

Aspect	Typical Scenario	Big Data
Application development	Applications that take advantage of massive parallelism developed by specialized developers skilled in high-performance computing, performance optimization, and code tuning	A simplified application execution model encompassing a distributed file system, application programming model, distributed database, and program scheduling is packaged within Hadoop, an open source framework for reliable, scalable, distributed, and parallel computing
Platform	Uses high-cost massively parallel processing (MPP) computers, utilizing high-bandwidth networks, and massive I/O devices	Innovative methods of creating scalable and yet elastic virtualized platforms take advantage of clusters of commodity hardware components (either cycle harvesting from local resources or through cloud-based utility computing services) coupled with open source tools and technology
Data management	Limited to file-based or relational database management systems (RDBMS) using standard row-oriented data layouts	Alternate models for data management (often referred to as NoSQL or "Not Only SQL") provide a variety of methods for managing information to best suit specific business process needs, such as in-memory data management (for rapid access), columnar layouts to speed query response, and graph databases (for social network analytics)
Resources	Requires large capital investment in purchasing high-end hardware to be installed and managed in-house	The ability to deploy systems like Hadoop on virtualized platforms allows small and medium businesses to utilize cloud-based environments that, from both a cost accounting and a practical perspective, are much friendlier to the bottom line

表 2 Contrasting Approaches in Adopting High-Performance Capabilities "Big Data Analytics", David Loshin, 2013

大數據分析平台及相關應用程式在近幾年有突飛猛進的發展，Hadoop技術在2013年獨領風騷的情形到了2014年已經被後起的Spark迎頭追上，2014~2017 年每年超過 400 位貢獻家投入 Spark 開發[3]。在 Machine Learning 的軟體開發情形亦是如火如荼，像 Theano 作為一個開源的深度學習框架，其由加拿大蒙特利爾大學於 2016 年 3 月第一次發行 0.8 版本，才短短的 18 個月即於 2017 年 9 月 28 日宣布更新 1.0 版本後不再更新維運，走入歷史[4]。目前超過 10 幾種 Deep Learning Software Framework[5]，其以開源程式為大宗，受到使用者青睞的則有 Google 的 TensorFlow、facebook 的 caffee2、Microsoft 的 CNTK。

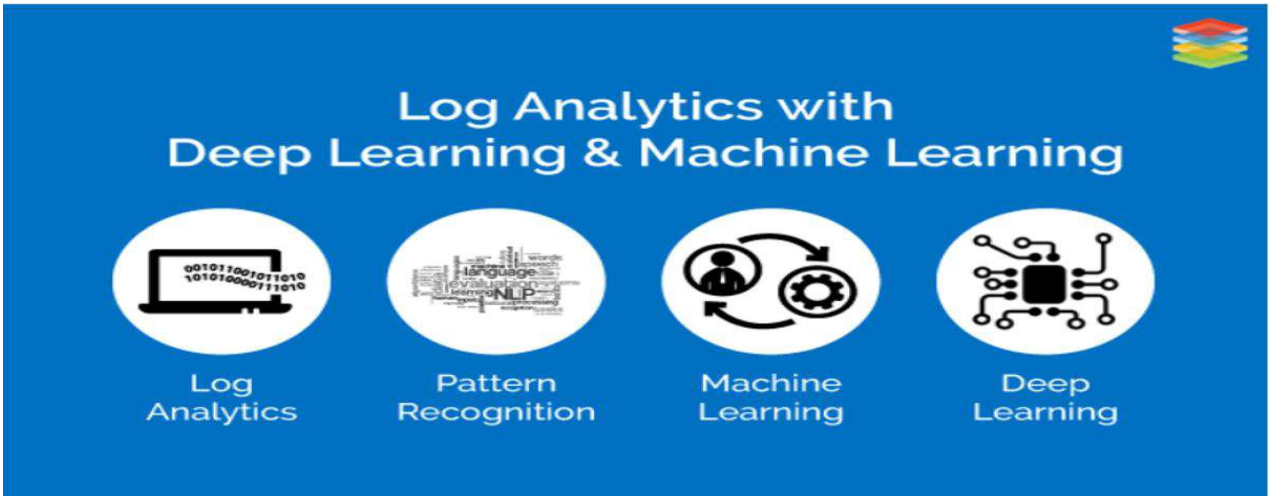
在大數據技術應用方面，IBM Deep Blue 於 1997 年 5 月二度挑戰卡斯巴羅夫比賽西洋棋，最終深藍電腦以 3.5－2.5 擊敗卡斯巴羅夫成為首個在標準比賽時限內擊敗西洋棋世界冠軍的電腦系統；2011 年 IBM IBM 開發的認知超級計算機 Watson 在美國電視益智節目" Jeopardy" 中勝出。2015 年 10 月，Google AlphaGo 擊敗樊麾，成為第一個無需讓子即可在 19 路棋盤上擊敗圍棋職業二段棋士的電腦圍棋程式；2016 年 3 月，透過自我對弈數以萬計盤進行練習強化後，AlphaGo 以 4:1 擊敗九段職業棋士李世乭，立下了里程碑；2017 年 5 月 23 至 27 日在

烏鎮圍棋峰會上，利用強化版 AlphaGo 和世界第一棋士柯潔比試獲勝；除了以上這些快炙人口的例子外，大數據應用主要受到大數據軟體開源性質，以及大數據供應商之業務模型，從其所具備之專業服務模式來幫助企業識別大數據應用案例，架構解決方案和維護性能等。

大數據分析相關技術包括：

- 平行運算 Massive Parallelism
- 大量數據儲存 Huge Data Volumes Storage
- 分散式資料管理 Data Distribution
- 高速網路 High-Speed Networks
- 高性能計算 High-Performance Computing
- 工作及時序管理 Task and Thread Management
- 資料探勘及分析 Data Mining and Analytics
- 資料擷取 Data Retrieval
- 機器學習 Machine Learning
- 資料視覺化 Data Visualization

大數據分析已成功地應用於廣泛的領域中，舉例如 Log Analysis、Pattern Recognition、機器學習、深度學習、Fintech、語音識別(自然語言處理)，Chatbot、電腦視覺，機器人仿人動作模擬，虛擬實境(VR、AR、MR、XR)等等。



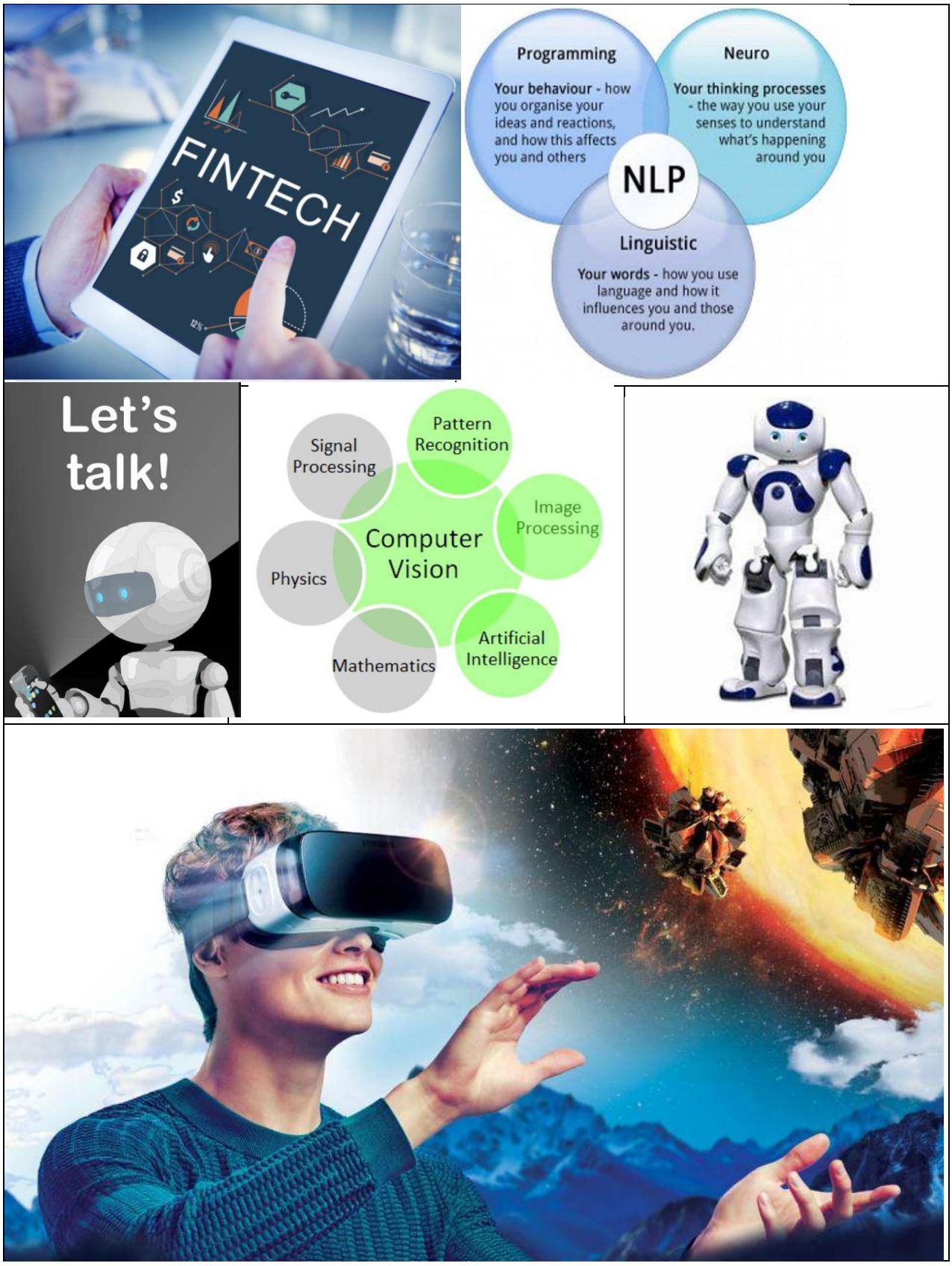


圖 3 大數據分析/機器學習之應用領域

2、大數據平台

本節介紹 2 種大數據平台：

(1) Hadoop是一款支援資料密集型分布式應用程式並以 Apache 2.0 許可協議發布的開源軟體框架，其根據 Google 公司 2003 年發表的 MapReduce 和 Google 檔案系統的論文實作而成，整個 Apache Hadoop「平台」包括 Hadoop 內核、MapReduce、Hadoop 分布式檔案系統（HDFS）以及一些相關項目，有 Apache Hive 和 Apache HBase 等等,所有的 Hadoop 模組都有一個基本假設，即硬體故障是常見情況，應該由框架自動處理，故其架構具有可靠性、擴充性，採分散式及平行處理特性，使用分散式檔案系統(Hadoop Distributed File System）及分散式資料庫。

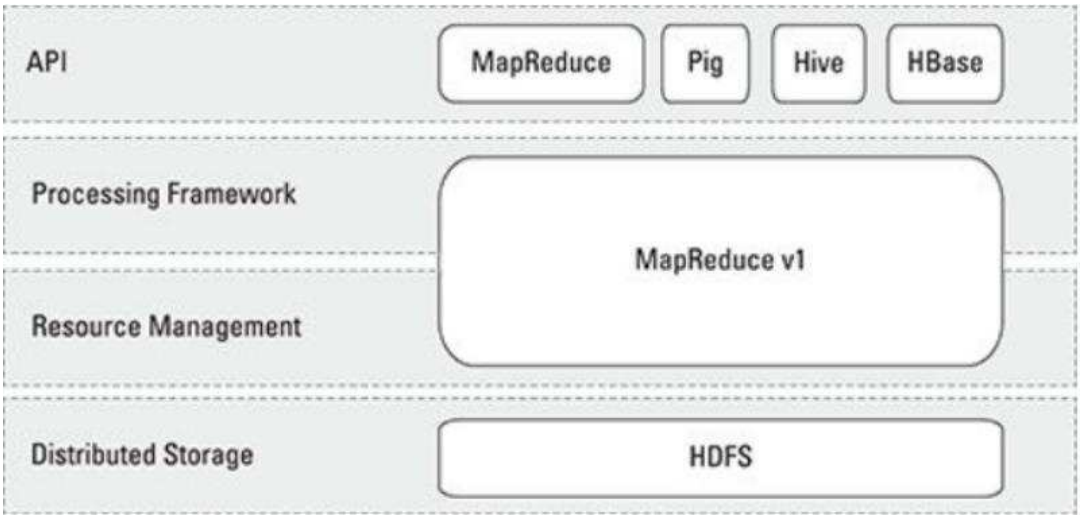


圖 4 Hadoop 1 架構示意圖

Apache Hadoop Software Library 採分散式處理框架，可運用簡單的編程模型處理跨電腦群集上的大量分散資料，上圖由底層往上分別為：

- Distributed Storage：分散式檔案系統(Hadoop Distributed File System）儲存分散式處理過程中所產生的資料、中間結果及最終結果。
- Resource Management：Hadoop 集群中的所有從節點(Slave Node)都具有 CPU 週期，RAM 和網絡頻寬。像 Hadoop 這樣的系統需

要能夠分配這些資源，以便多個應用程序與用戶可以預測和調節共享群集，這項工作由 JobTracker daemon 管控。

- Process Framework：Hadoop 1 以 MapReduce 方式執行所有應用程序，不斷進行分類與合併。
- Application Programming Interface(API)：提供程式設計者呼叫 Apache Hive 和 Apache HBase 等 API 與 MapReduce 介接，簡化設計。

HDFS 適合處理具備一次寫入和經常讀取之特性的應用，因此 Hadoop 平台上常見的應用包括日誌數據分析、風險建模分析、社會情緒分析、圖像分類、圖形分析、欺詐識別等。 (附錄 1 為安裝 Apache Hadoop 的程序筆記)。

(2) Spark 為一能處理 Hadoop 數據的快速和通用計算引擎。Spark 提供了一個簡單而富有表現力的編程模型，支持廣泛的應用程序，包括 ETL，機器學習，流處理和圖計算。

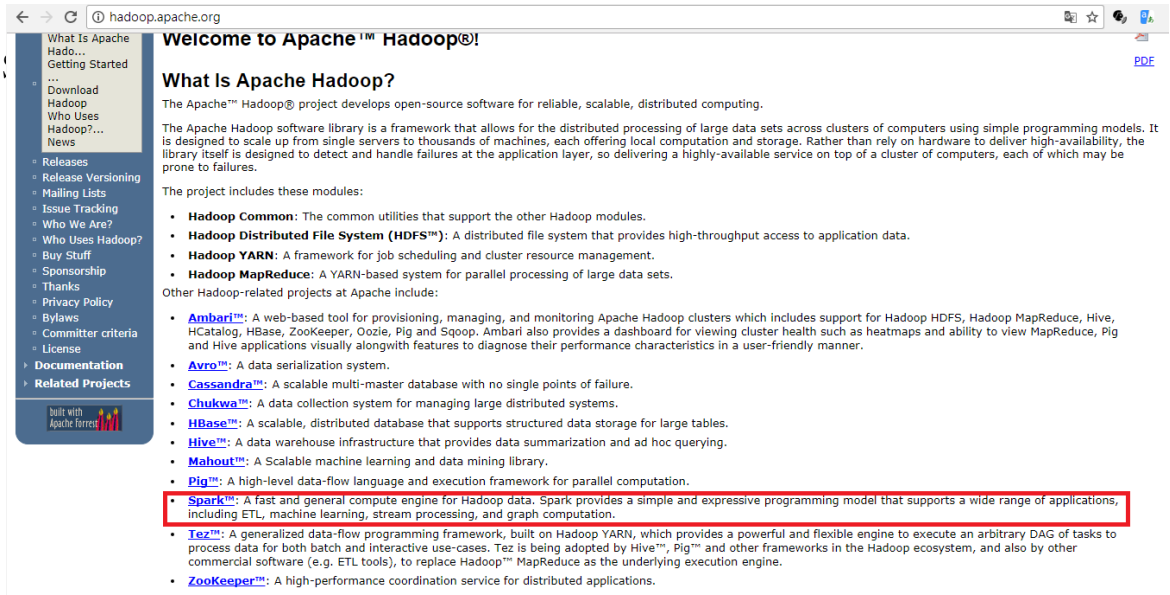


圖 5 Spark

相對於 Hadoop 的 MapReduce 會在執行完工作後將中介資料存放到磁碟中，Spark 使用了記憶體內運算技術，能在資料尚未寫入硬碟時即在記憶體內分析運算。Spark 在記憶體內執行

程式的運算速度能做到比 Hadoop MapReduce 的運算速度快上 100 倍，即便是執行程式於硬碟時，Spark 也能快上 10 倍速度。Spark 允許用戶將資料加載至叢集記憶體，並多次對其進行查詢，非常適合用於機器學習演算法。[6]

Spark Stack

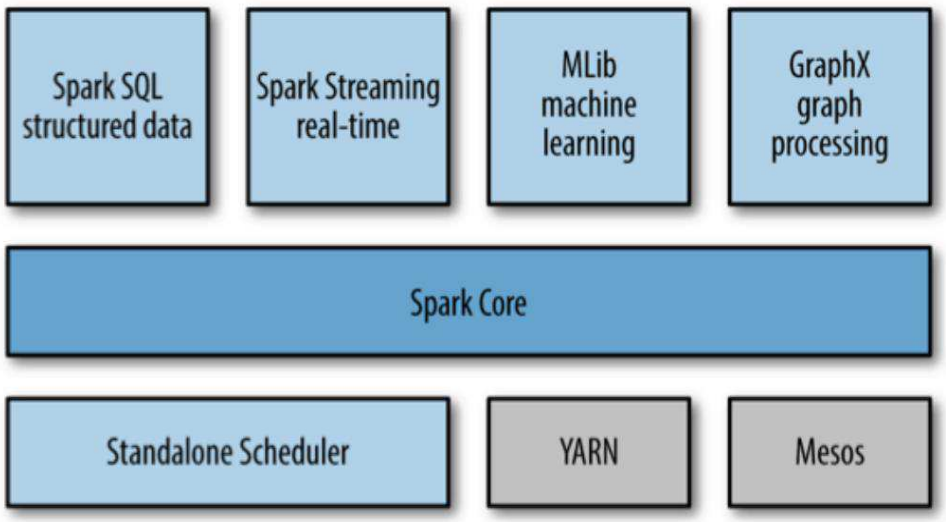


圖 6 Spark 架構圖[7]

即便是執行程式於硬碟時，Spark 也能快上 10 倍速度。Spark 允許用戶將資料加載至叢集記憶體，並多次對其進行查詢，非常適合用於機器學習演算法。

在實務上推薦學習者可利用 Virtual Machine (VM) 安裝 Ubuntu 後再安裝 java python scala spark 以 ipython notebook 進行實作，如遇上及大量的數據時，亦可於網路上開起雲端服務，附錄 2 於 AWS 上新開 1 台 ubuntu 及 install pyspark 實作流程。

3、大數據存儲和分析

下圖 7 為國際數據資訊(IDC)預測 2015~2020 年大數據相關服務收入市場分類及比例，可由此圖看出未來大數據技術的比重：

- 比重最高的為 Non-relational Analytic Data systems 佔 38.6%；NoSQL 指的是 Not only SQL，包括 graph data，此項比重之所以最高，誠如 Billy Lo, Head of Enterprise Architecture for Tangerine 所說的 "We expect 50 percent growth in data in the next five years and we need to be able to support that."[8]，傳統 BI Solution 因為速度太慢已不符需求，因此新的 DMSA (Data Management Solutions for Analytics)[9]系統將更為炙手可熱。另外 Gartner 在 IT market clock for database 分析報告中提到傳統 SQL DBMS 的 Lifecycle 即將到期，呼籲數據和分析領導者應該思考替換與之互補的 NoSQL 產品時機；Google 也於日前推出的影片介紹 Cloud Spanner[10] (Most databases today require making trade-offs between scale and consistency. With Cloud Spanner, you get the best of relational database structure and non-relational database scale and performance with external strong consistency across rows, regions, and continents)。
- 其次為 Cognitive software Platform(認知/人工智慧軟體平台，這些平台包括機器學習，推理，自然語言處理，語音或視覺辨識（物體辨識），人機交互，對話和敘述生成等技術，佔 23.3%，相關之 Data Science and Machine-Learning Platforms 相關產品推陳出新及競爭力演化請參閱圖 8 Gartner Magic quadrant。
- Content Analytics ，資料內容分為結構化(資料庫)和非結構化(文件，語音，圖像或視頻)，內容分析目標是獲得新見解(new insight)以幫助決策，佔 17.3%。
- Search Systems，搜索系統，佔 16.6%。
- IT Services，佔 14.6%。

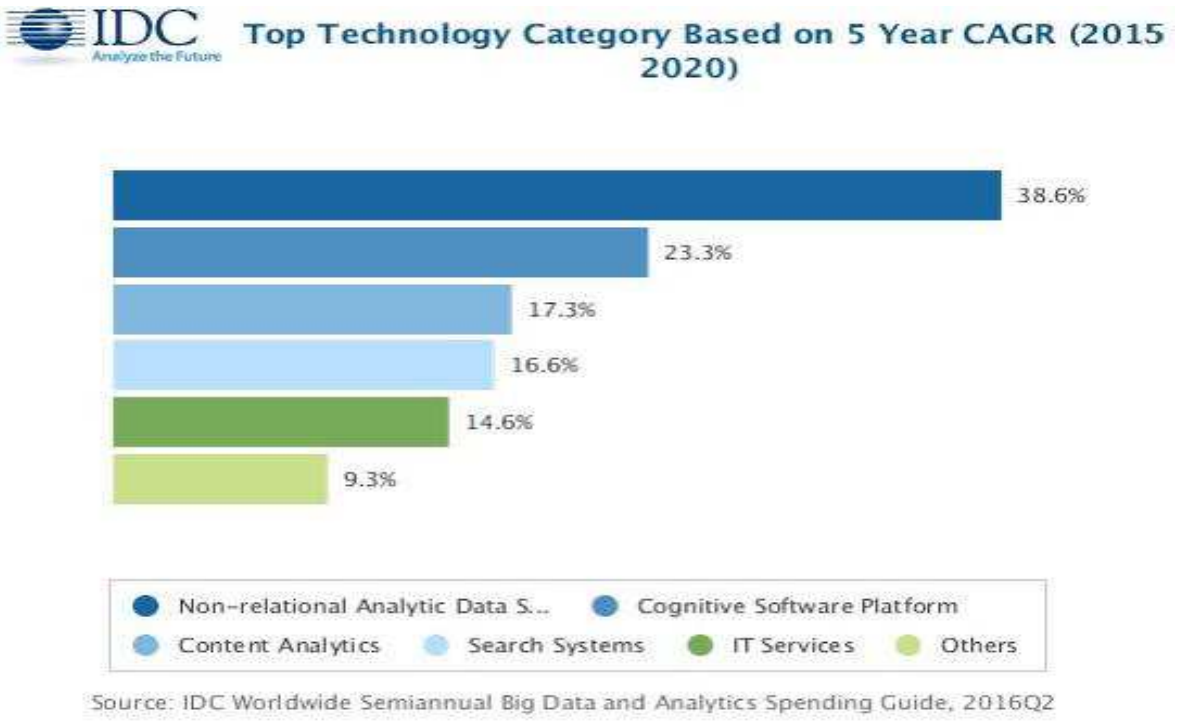


圖 7 2015~2020 年大數據相關服務收入市場分類比重

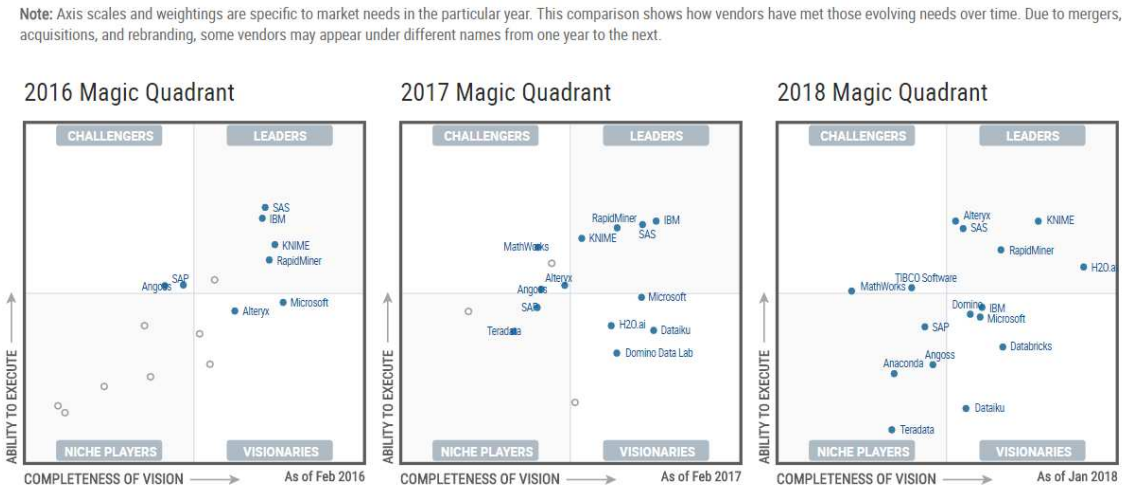


圖 8 Gartner’ s magic quadrant for 2016~2018 年 Data Science and Machine-Learning Platforms

4、大數據分析機器學習演算法

(1) 大數據分析的實作流程可用下圖來說明：

1) 首先是思考要做哪一方面的資料分析，然後蒐集相關的資料，資料的來源很多，Data Science Central 列出了 20 個網路公開的資料庫[11]，例如與健康有關有美國的 Healthdata.gov 及英國的 http://content.digital.nhs.uk/home 等。

2) 第 2 步是清理資料是要把資料整理成可以進行分析處理的狀態， 比如先統一格式，將資料整理成每個特徵欄位以行排列的格式(column features)，再處理 missing data、special data、及 outlie Data 等步驟。

3) 將資料分成 training set 及 test set，選擇大數據分析機器學習演算法對 training 資料進行分析。

4) 利用上述的模型再對 testing set 進行驗證。

5) 調整演算法的參數重複進行 training 及 testing，如過效果不錯，可嘗試布署，以真實資料試行預測，並比對結果。

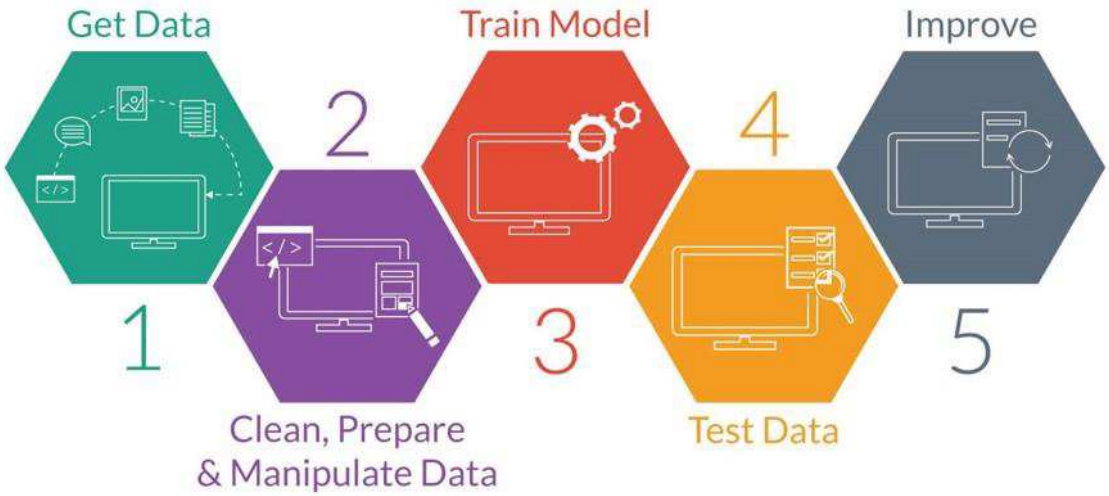


圖 9 大數據分析的實作流程

(2) 機器學習依學習時是否有標籤數據 label data 分為：

1) 監督式學習（Supervised Learning）：在訓練資料中有「標籤」資料，標籤就是答案或 output，譬如想由人的圖片分辨出性別，訓練資料中如果有已經標示出性別的圖片，以這些有「標籤」的圖片來訓練機器，讓機器學習出 general rule thatmapsinputs to outputs。

2) 非監督式學習（Unsupervised Learnin）：訓練資料中沒有「標籤」資料，讓機器依資料自己找出其結構中的特性，目的是找出資料中的 hidden patterns 做為特徵學習的方法。

(3) 因大數據分析所想要獲得的結果與問題類型有關，又可區分：

1) Classification 分類型問題：當所欲解決的問題是分類，例如分辨垃圾郵件或是正常郵件。為了要驗證預測值 predicted value與正確值 True Value，通常會利用 confusion matrix[12] 來 evaluate our model。

2) Regression 回歸問題：利用回歸方程的最小平方函數對一個或多個自變量和因變量之間關係進行建模的一種回歸分析。

3) clustering 分群：是將 inputs 分成群組，但因事前不知道資料特性適合分成幾組，需要機器自行分析特徵，學習分組程度，使用分群技巧有 3 步驟，an algorithm、a notion of both similarity of disimilarity、a stopping point。

Typical cluster models include:

- Connectivity models: for example, hierarchical clustering builds models based on distance connectivity.
- Centroid models: for example, the k-means algorithm represents each cluster by a single mean vector.
- Distribution models: clusters are modeled using statistical distributions, such as multivariate normal distributions used by

the expectation-maximization algorithm.

- Density models: for example, DBSCAN and OPTICS defines clusters as connected dense regions in the data space.
- Subspace models: in biclustering (also known as co-clustering or two-mode-clustering), clusters are modeled with both cluster members and relevant attributes.
- Group models: some algorithms do not provide a refined model for their results and just provide the grouping information.
- Graph-based models: a clique, that is, a subset of nodes in a graph such that every two nodes in the subset are connected by an edge can be considered as a prototypical form of cluster. Relaxations of the complete connectivity requirement (a fraction of the edges can be missing) are known as quasi-cliques, as in the HCS clustering algorithm.
- Neural models: the most well known unsupervised neural network is the self-organizing map and these models can usually be characterized as similar to one or more of the above models, and including subspace models when neural networks implement a form of Principal Component Analysis or Independent Component Analysis.

4) Tree based Methods：Tree is conceptually easy to understand 但卻有預測不準的問題，但是透過 aggregating many decision trees，使用 bagging、random forest，and boosting，the predictive performance of trees can be substantially improved.

5) Collaborative Filtering 協同過濾：此法應用廣泛，例如在電子商務中研究如何提供消費者更準確的個人化推薦服務，是提高轉換率和良好使用者經驗的重要關鍵。

6) Density estimation：finds thedistributionof inputs in some space 要找出輸入資料在不同空間中的分布。核密度估计（kernel density estimation）是在統計中用来估計未知的密度函数。

7) 降維問題 Dimensionality reduction：將 input mapping 到低維空間，主題建模問題 Topic modeling 與降維相關，是讓機器學習依文件內容建立不同主題模型。

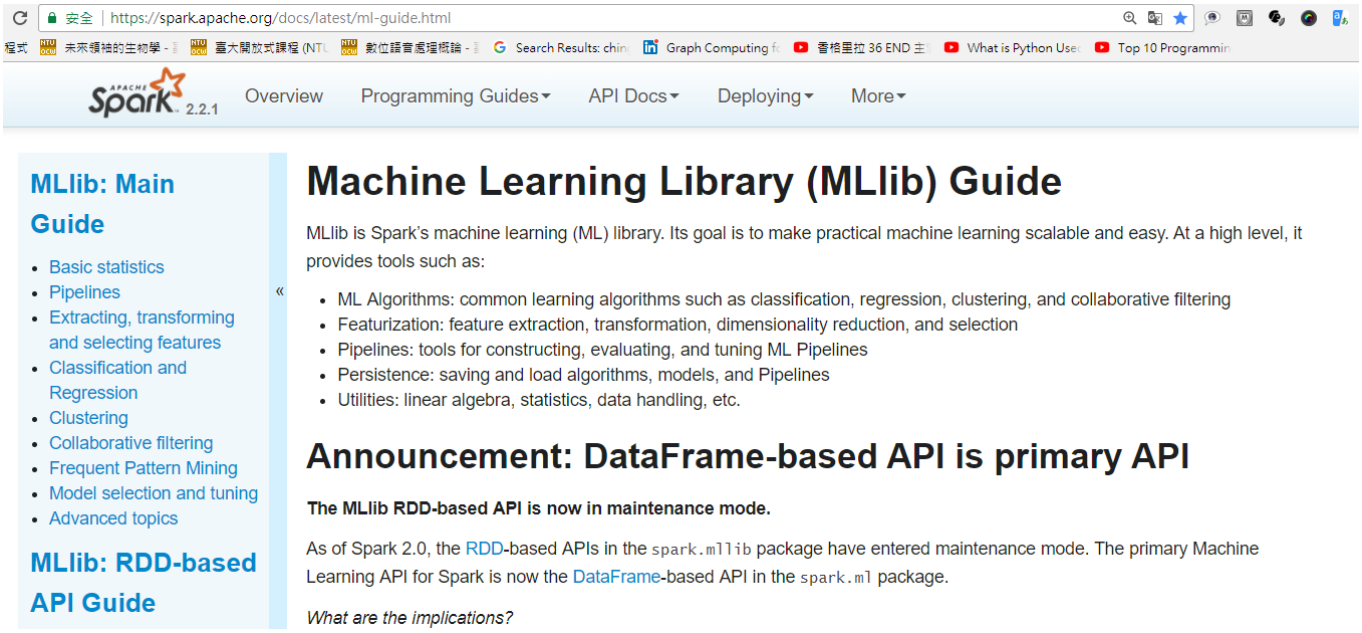
8) Reinforcement Learning： 強化學習是機器學習中的一個領域，強調如何基於環境而行動，以取得最大化的預期利益。其靈感來源於心理學中的行為主義理論，即有機體如何在環境給予的獎勵或懲罰的刺激下，逐步形成對刺激的預期，產生能獲得最大利益的習慣性行為。Reinforcement learning(RL) is an area ofmachine learninginspired bybehaviorist psychology, concerned with howsoftware agentsought to take actions in an environmentso as to maximize some notion of cumulative reward.

9) Deep Learning：Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised.

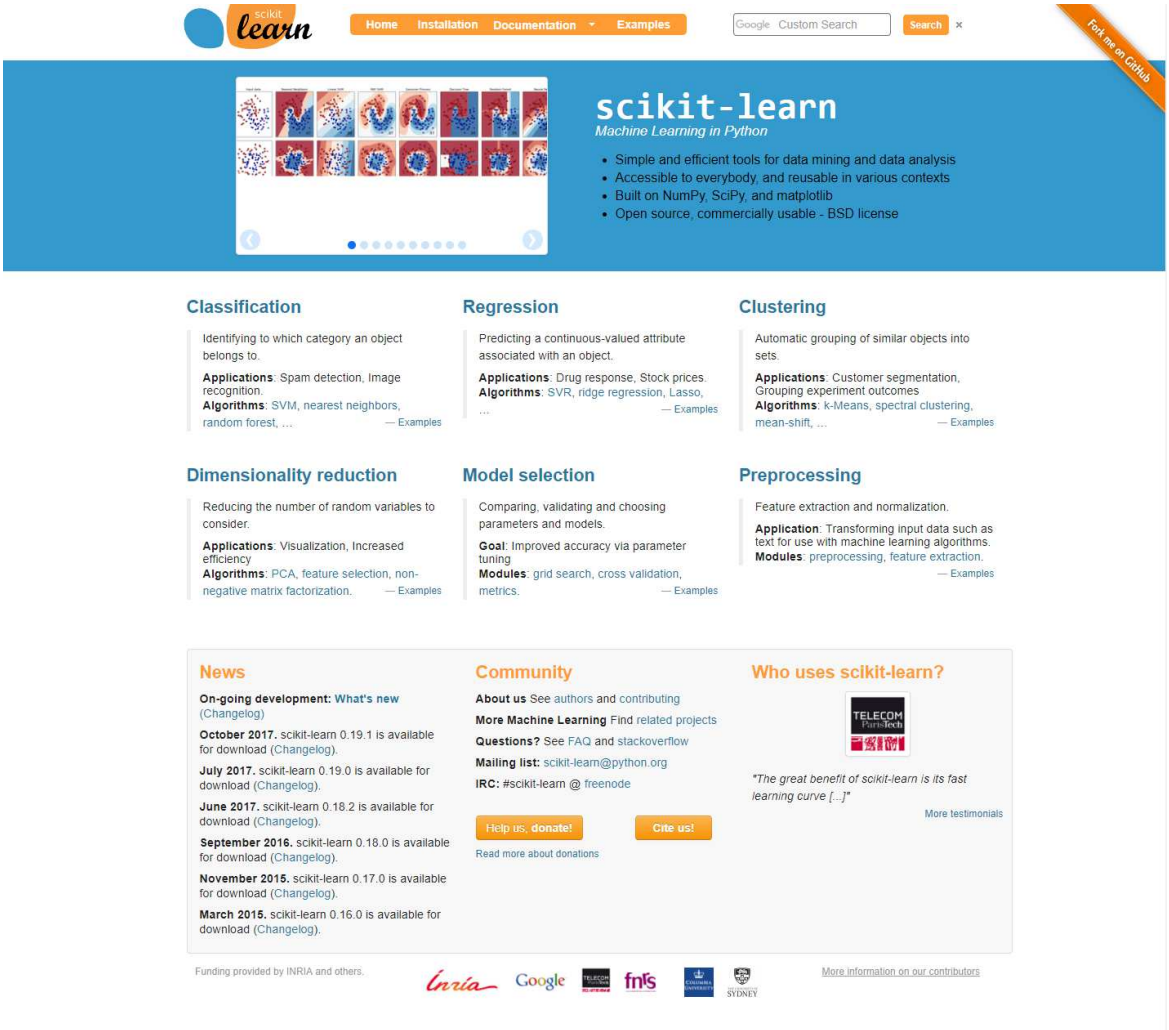
10) learning to learn learns its own inductive bias based on previous experience. Developmental learning, elaborated forrobot learning, generates its own sequences (also called curriculum) of learning situations to cumulatively acquire repertoires of novel skills through autonomous self-exploration and social interaction with human teachers and using guidance mechanisms such as active learning, maturation, motor synergies, and imitation.

以下列出 python、Spark、scikit-learn 的網路資源，以利持續學習。

- Python https://docs.python.org/3/tutorial/
- Spark 圖 10 https://spark.apache.org/docs/latest/ml-guide.html

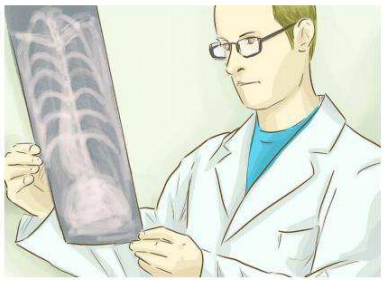


- Scikit-learn 圖 11 http://scikit-learn.org/stable/



Dealing with **uncertainty**:

You would like to determine how likely the patient is infected with inhalational anthrax given that the patient has a cough, a fever, and difficulty breathing

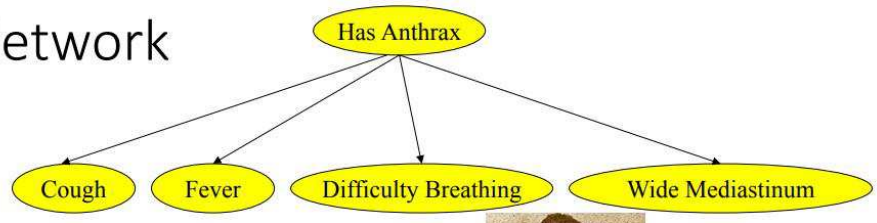


New evidence: X-ray image shows that the patient has a wide mediastinum.

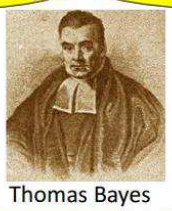
Belief update: your belief that the patient is infected with inhalational anthrax is now much **higher** now.

- In the previous slides, what you observed affected your belief that the patient is infected with anthrax
- This is called reasoning with uncertainty
- Wouldn't it be nice if we had some tools for reasoning with uncertainty? In fact, we do…

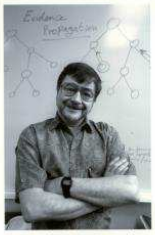
Bayesian Network



- Need a representation and reasoning system that is based on conditional independence
  - Compact yet expressive representation
  - Efficient reasoning procedures
- Bayesian Network is such a representation
  - Named after Thomas Bayes (ca. 1702 –1761)
  - Term coined in 1985 by Judea Pearl (1936 –), 2011 winner of the ACM Turing Award
  - Many applications, e.g., spam filtering, speech recognition, robotics, diagnostic systems and even syndromic surveillance



Thomas Bayes



Judea Pearl

6、影響環境因素大數據圖形關聯-Introduction to Graph Database

圖形資料庫(Graph Database)的前身是自 1999 年以來符合 W3C 標準的 RDF 格式(The Resource Description Format) 因其具有簡單及彈性的特徵並且和其同時期搭配的 SPARQL query language 在當時提供處理巨量資料的 infrastructure，因其與關聯式資料庫的架構大不相同，因此啟發了開原社群及資料庫業者(如 IBM、Oracle 及 Cray 等公司投入研究。

RDF 格式使用 3 parts statement called Triples(subject, predicate, object)可以當成是(ID, property Name, and property Value) ，這種簡單的架構可以讓 RDF 不依賴 schema[13]，茲列出範例如下：

Example: A company has nince of part p1234 in stock, then a simplified triple rpresenting this might be {p1234 inStock 9}.
Instance Identifier, Property Name, Property Value.
In a proper RDF version of this triple, the representation will be more formal. They require uniform resource identifiers (URIs).

```
@prefix fbd: <http://foobarco.net/data/>.
@prefix fbv: <http://foobarco.net/vocab/>.

fbd:p1234 fbv:inStock "9".
fbd:p1234 fbv:supplier "Joe's Part Company".
```

```
@prefix fbd:<http://foobarco.net/data/>.
@prefix fbv:<http://foobarco.net/vocab/>.
fbd:p1234 fbv:inStock "9".
fbd:p1234 fbv:name "Blue reverse flange".
fbd:p1234 fbv:supplier fbd:s9483.
fbd:s9483 fbv:name "Joe's Part Company".
fbd:s9483 fbv:homePage "http://www.joespartco.com".
fbd:s9483 fbv:contactName "Gina Smith".
fbd:s9483 fbv:contactEmail "gina.smith@joespartco.com".
```

The following SPAQRL query asks for all property names and values associated with the fbd:s9483 resource:

```
PREFIX fbd:<http://foobarco.net/data/>

SELECT ?property ?value
WHERE {fbd:s9483 ?property ?value.}
```

property	value
<http://foobarco.net/vocab/contactEmail>	"gina.smith@joespartco.com"
<http://foobarco.net/vocab/contactName>	"Gina Smith"
<http://foobarco.net/vocab/homePage>	"http://www.joespartco.com"
<http://foobarco.net/vocab/name>	"Joe's Part Company"

利用開源程式發展出的 Apache Jena，以 JAVA 寫成的 API 程式可針對 RDF 檔案產生對應之圖形。(Jena is a Java API which can be used to create and manipulate RDF graphs)。

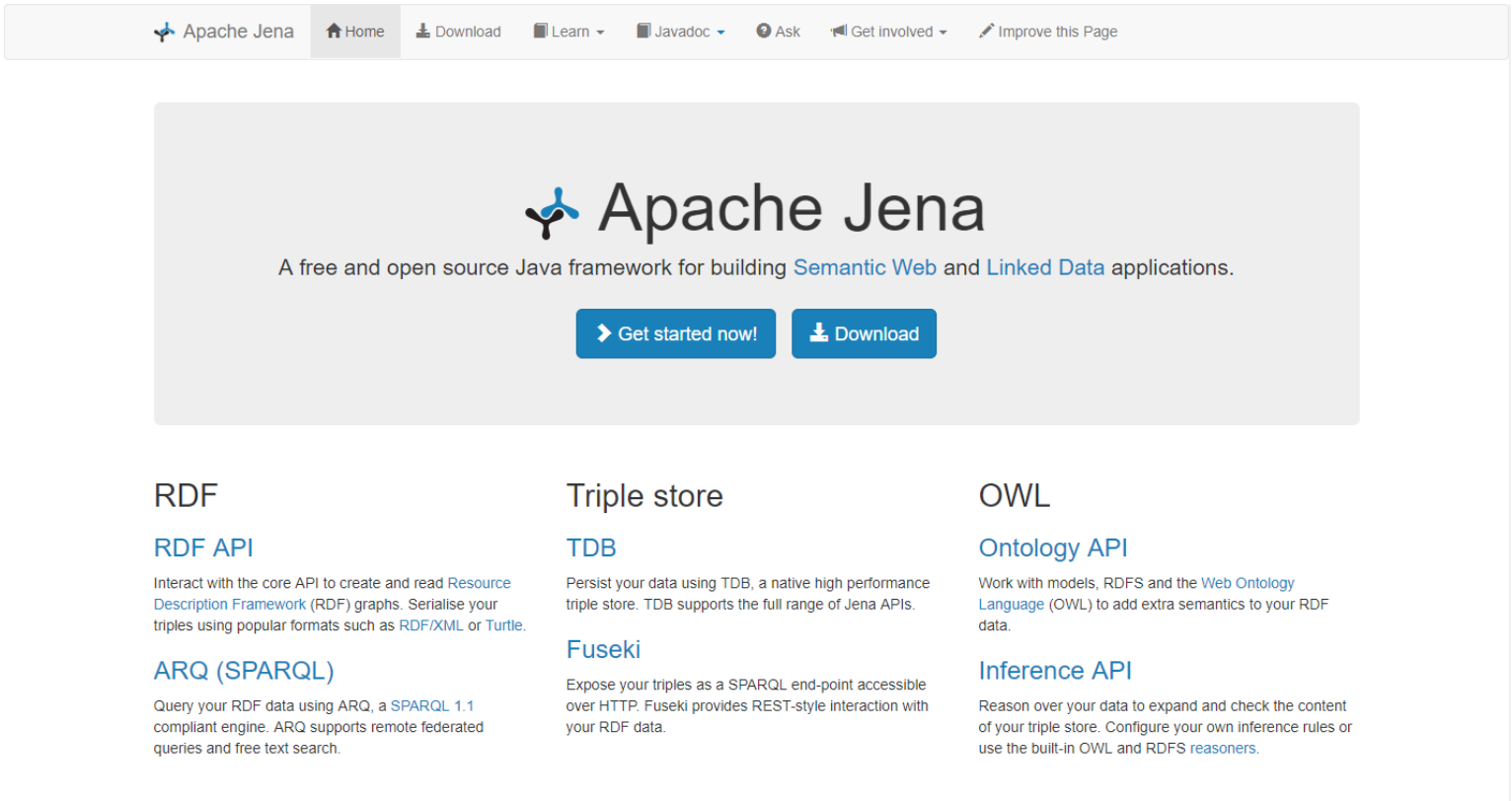


圖 12　Apache Jena (Apache License Version2.0)

參考資料 O'Reilly 的 Graph Databases 是以 Neo4J 當作介紹範例，其他開源軟體中 OrientDB是以 JAVA 開發的 NoSQL 資料庫，它以 multi-model database 著稱,支圖形資料庫援圖 、文件、key/value, 及物件模型(object models)。

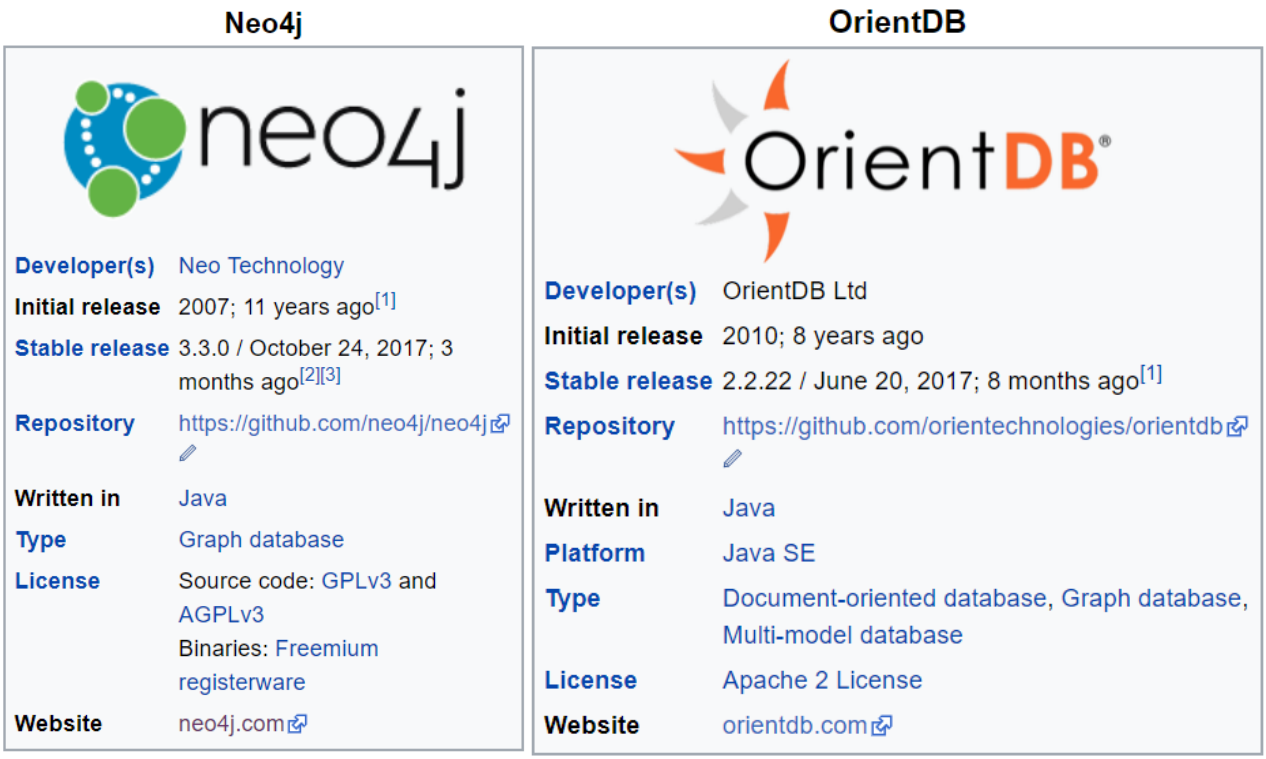
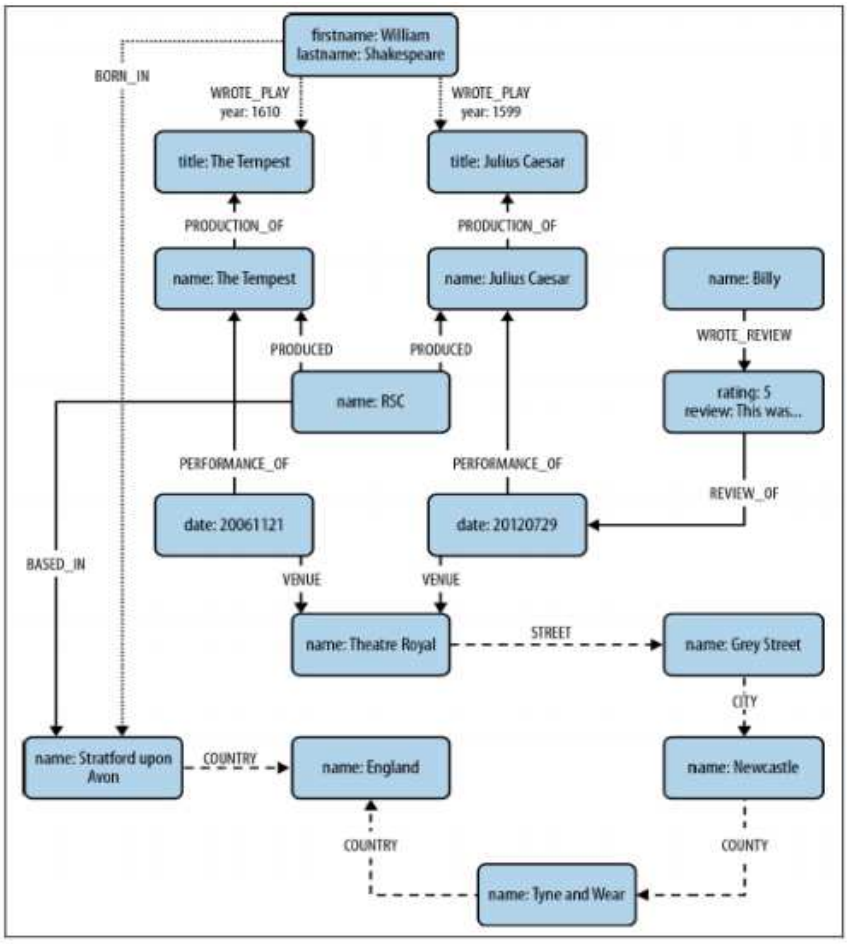


圖 13　Neo4j 與 OrientDB 特徵對照

接下來將利用 O'Reilly 的 Graph Databases 書中 Shakespeare 的範例來說明使用 Graph Databases 的特性，當思考類似的架構如果要用 SQL 架構來表達，需要建立多少個 Table 或是在查詢時需要如何下 SQL 指令時可以體會出 Graph Databases 的簡潔與效率。

```
CREATE (shakespeare { firstname: 'William', lastname: 'Shakespeare' }),
       (juliusCaesar { title: 'Julius Caesar' }),
       (shakespeare)-[:WROTE_PLAY { year: 1599 }]->(juliusCaesar),
       (theTempest { title: 'The Tempest' }),
       (shakespeare)-[:WROTE_PLAY { year: 1610}]->(theTempest),
       (rsc { name: 'RSC' }),
       (production1 { name: 'Julius Caesar' }),
       (rsc)-[:PRODUCED]->(production1),
       (production1)-[:PRODUCTION_OF]->(juliusCaesar),
       (performance1 { date: 20120729 }),
       (performance1)-[:PERFORMANCE_OF]->(production1),
       (production2 { name: 'The Tempest' }),
       (rsc)-[:PRODUCED]->(production2),
       (production2)-[:PRODUCTION_OF]->(theTempest),
       (performance2 { date: 20061121 }),
       (performance2)-[:PERFORMANCE_OF]->(production2),
       (performance3 { date: 20120730 }),
       (performance3)-[:PERFORMANCE_OF]->(production1),
       (billy { name: 'Billy' }),
       (review { rating: 5, review: 'This was awesome!' }),
       (billy)-[:WROTE_REVIEW]->(review),
       (review)-[:RATED]->(performance1),
       (theatreRoyal { name: 'Theatre Royal' }),
       (performance1)-[:VENUE]->(theatreRoyal),
       (performance2)-[:VENUE]->(theatreRoyal),
       (performance3)-[:VENUE]->(theatreRoyal),
       (greyStreet { name: 'Grey Street' }),
       (theatreRoyal)-[:STREET]->(greyStreet),
       (newcastle { name: 'Newcastle' }),
       (greyStreet)-[:CITY]->(newcastle),
       (tyneAndWear { name: 'Tyne and Wear' }),
       (newcastle)-[:COUNTY]->(tyneAndWear),
       (england { name: 'England' }),
       (tyneAndWear)-[:COUNTRY]->(england),
       (stratford { name: 'Stratford upon Avon' }),
       (stratford)-[:COUNTRY]->(england),
       (rsc)-[:BASED_IN]->(stratford),
       (shakespeare)-[:BORN_IN]->stratford
```



上面指令為 Neo4j Create 範例左(圖 14)為 create 指令建立的圖(graph)

上頁 Shakespeare 的範例 Neo4j 利用 create 指令可建構出圖，當思考類似的架構如果要用 SQLDB 架構來表達，我們可能需要對作者、劇本、劇場、演出等實體建立 Table。

接下來的查詢範例是查 1608 年之後 Shakespeare 在 Newcastle Theatre Royall 演過的是哪個戲劇?

```
START theater=node:venue(name='Theatre Royal'),
      newcastle=node:city(name='Newcastle'),
      bard=node:author(lastname='Shakespeare')
MATCH (newcastle)<-[:STREET|CITY*1..2]-(theater)
      <-[:VENUE]-()-[:PERFORMANCE_OF]->()-[:PRODUCTION_OF]->
      (play)<-[w:WROTE_PLAY]-(bard)
WHERE w.year > 1608
RETURN DISTINCT play.title AS play
```

```
+---------------+
| play          |
+---------------+
| "The Tempest" |
+---------------+
1 row
```

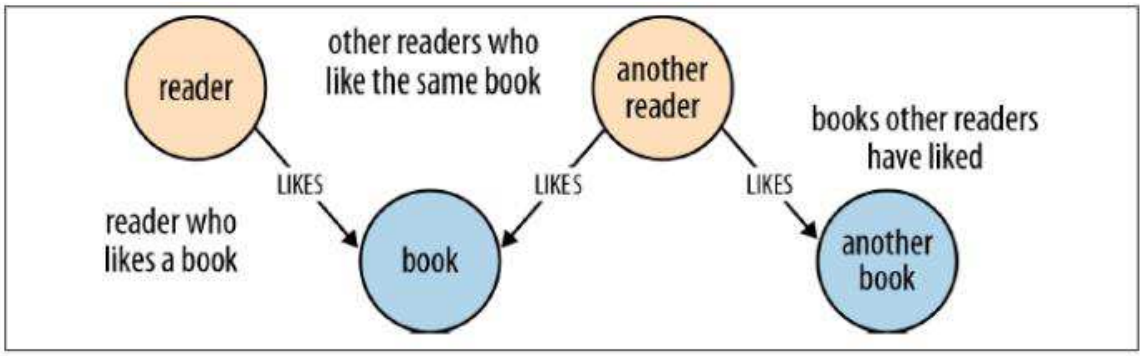
那麼 Shakespeare 在 Newcastle Theatre Royall 演過的是哪幾個戲劇? 請依照演出次數排序

```
START theater=node:venue(name='Theatre Royal'),
      newcastle=node:city(name='Newcastle'),
      bard=node:author(lastname='Shakespeare')
MATCH (newcastle)<-[:STREET|CITY*1..2]-(theater)
      <-[:VENUE]-()-[p:PERFORMANCE_OF]->()-[:PRODUCTION_OF]->
      (play)<-[:WROTE_PLAY]-(bard)
RETURN   play.title AS play, count(p) AS performance_count
ORDER BY performance_count DESC
```

```
+-------------------------------------+
| play            | performance_count |
+-------------------------------------+
| "Julius Caesar" | 2                 |
| "The Tempest"   | 1                 |
+-------------------------------------+
2 rows
```

另一個第 4 章的範例也可以看出 Graph Database 也適合用在推薦系統。

圖 15　Graph Database 以 user reference 推薦書籍應用



Because this data model directly encodes the question presented by the user story, it lends itself to being queried in a way that similarly reflects the structure of the question we want to ask of the data:

```
START reader=node:users(name={readerName})
      book=node:books(isbn={bookISBN})
MATCH reader-[:LIKES]->book<-[:LIKES]-other_readers-[:LIKES]->books
RETURN books.title
```

7、運用圖形運算分析影響環境因-Graph Database Analysis

這一節所使用的範例是林教授於 2008~2009 年間於 IBM 公司內部建立社群網路 SmallBlue / Atlas 所收集到其顧問群同事們使用社群網路，對於在公司內部所建立的社群群組，與在公司外部建立的鏈結，對於本身的工作及公司的利潤產生有什麼正、負面的影響，作為範例說明[14]，參考資料有系統展示的影片連結。

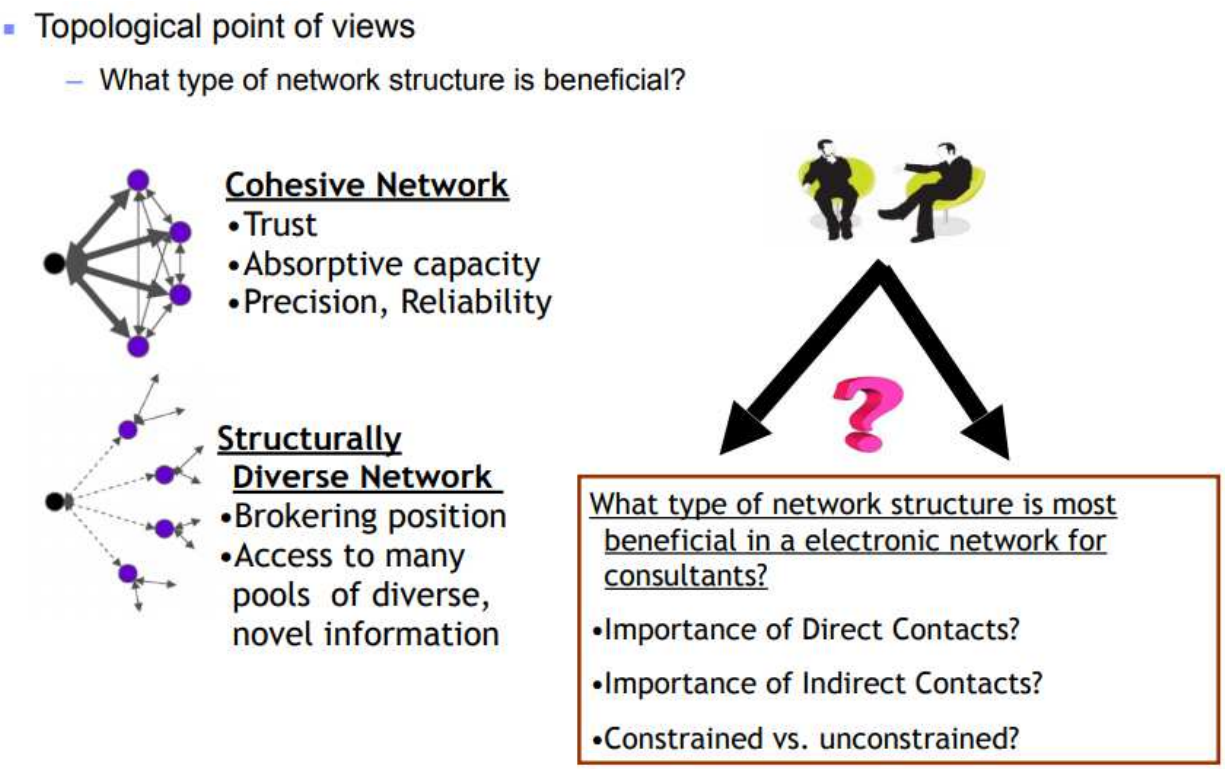


圖 16 Economic Issues-Network Topology and Worker Productivity

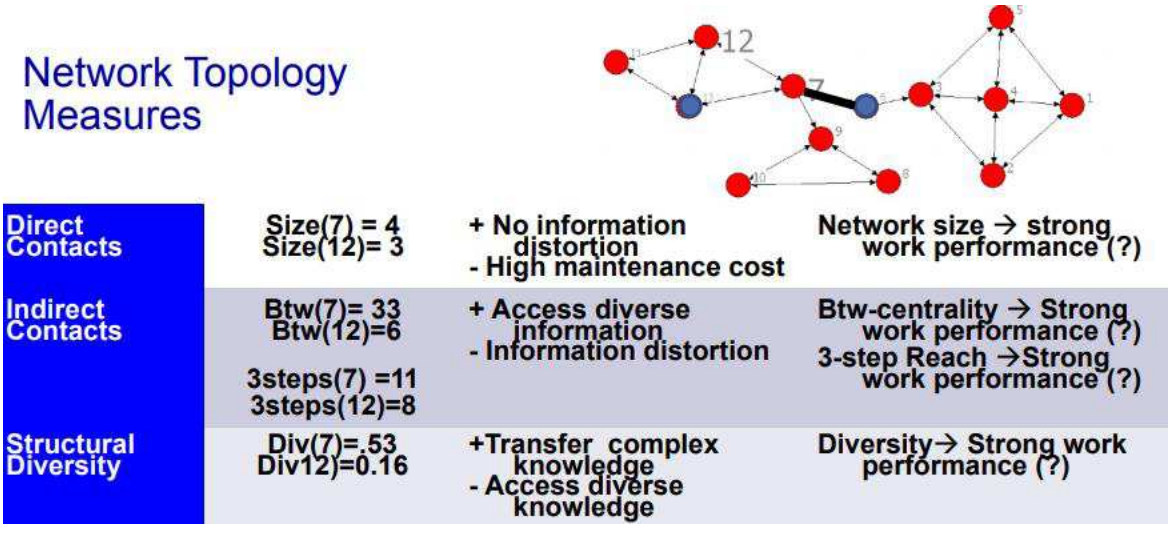
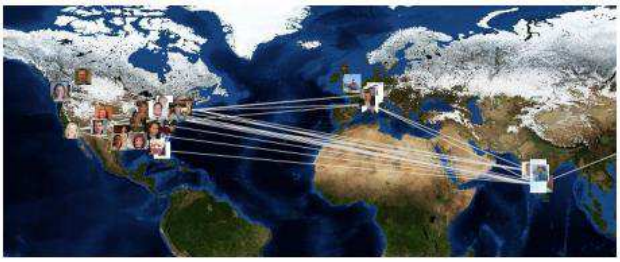


圖 17 Network Topology Measures

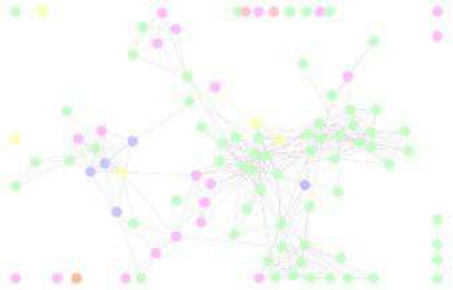
在為期 2 年的時間分析蒐集 2038 位 IBM 顧問使用社群網路 SmallBlue 的情形發現：

- 開始使用使用 SmallBlue 之後 social Capital grew，相當於每位替公司增加 584 美元之，每年約 7010 美元利潤。
- 經過 Joint analysis 發現於網路中增加認識 1 位顧問朋友，每年能為 IBM 公司帶進 948 美元利潤。(此項分析刊登於 Businessweek Top Story, April 8 2009)

圖 18 Enterprise becomes more successful utilizing Social Network Analysis



Example: Healthcare experts in the world



Connections between different divisions

Example: Healthcare experts in the U.S.

Key social bridges

圖 19　Betweenness 相當於 Bridges

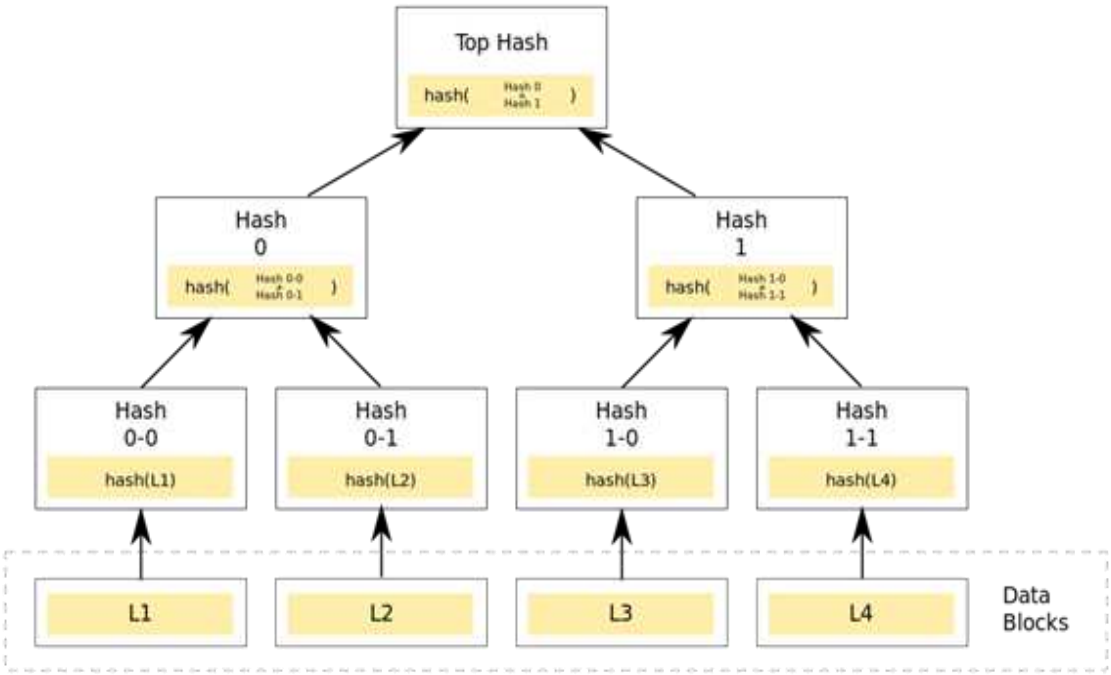
Observations from Personal Social Networks vs. Revenue

- Structural Diverse networks with abundance of structural holes are associated with higher performance.
  - *Having diverse friends helps.*
- Betweenness is negatively correlated.
  - *Being a bridge between a lot of people is not helpful.*
- Network reach are highly corrected.
  - *The number of people reachable in 3 steps is positively correlated with higher performance.*
- Having too many strong links — the same set of people one communicates frequently is negatively correlated with performance.
  - *Perhaps frequent communication to the same person may imply redundant information exchange.*
    - Future textual analysis can be done to confirm this.

8、區塊鏈技術研究

區塊鏈(原名 block chain 兩字分開，2016 年後改 blockchain 兩字合併)技術源自 Markle tree(hash tree)、Timestamps、cryptography。

（1） Markle tree(hash tree) 2 元樹範例如下圖所示，其中每個葉節點用數據塊 hash 值作標籤，並且每個非葉節點用其子節點的標籤的加密 hash 值作標籤。hash 樹能對大型數據結構之內容進行高效安全的驗證。



An example of a binary hash tree. Hashes 0-0 and 0-1 are the hash values of data blocks L1 and L2, respectively, and hash 0 is the hash of the concatenation of hashes 0-0 and 0-1.

圖 20 a binary hash tree[15]

圖中 L1~L4 是資料塊，Hsah0-0 是 L1 的 Hash 值(經過 hash function 計算後的結果)，Hsah0-1 是 L2 的 Hash 值，Hash 0 則是 Hsah0-0 與 Hsah0-1 concatenate 之後，在經過 hash function 計算後所

得的 hash 值，Top Hash 則是 Hash 0+ Hash 1 的 Hash 值。

（2） Timestamps

時間戳的應用是為了想要留下一個時間序的證據，應用時將運算出來的雜湊值用 TSA private key 來加密，作法是將資料與時間戳進行 hash 運算產生的 hash 值，送到時間戳授權機構(TSA)以其 private key 來加密，驗證時用 TSA public key 還原值與原 hash (Data+TimeStamp)運算值比對，如兩者相同表其內容及時間均為真，未受篡改。

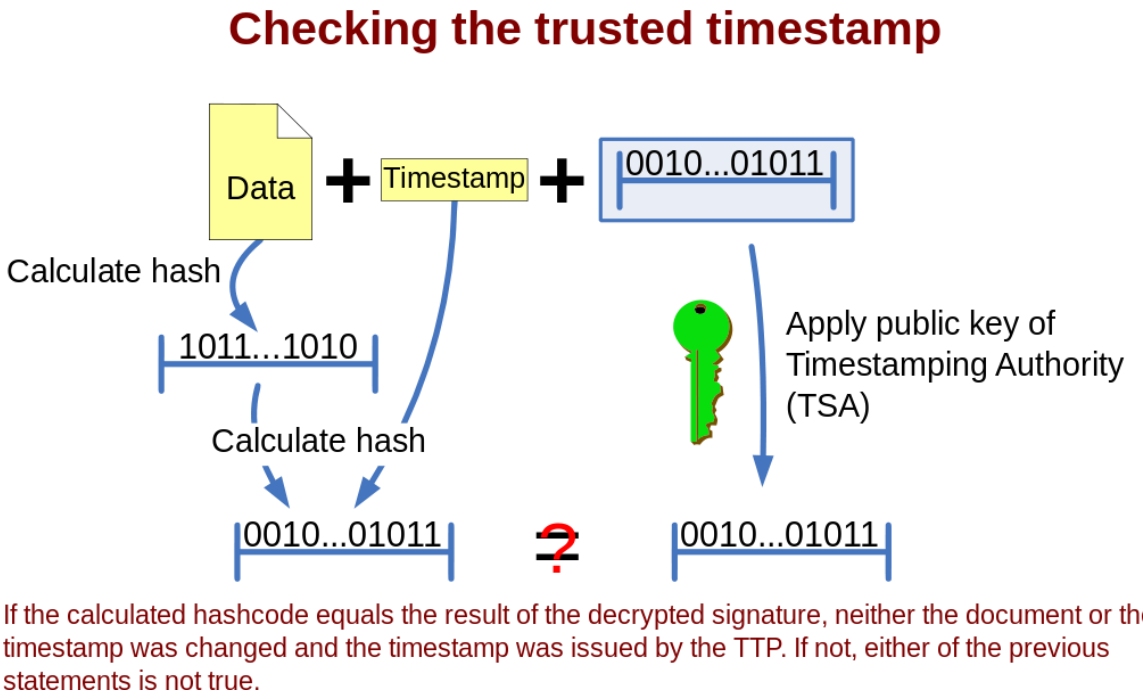


圖 21　Timestamps 一般性運用

在分散式 bitcoin 區塊鏈上運用時間戳，作為該交易確切產生時間的安全證明。

分散區塊鏈加時間戳亦被應用於其他領域，例如儀表板相機，以確保視頻文件在錄製時的完整性[16]，又如為證明在社交媒體平台上共享的創意內容和想法的優先順序等。

（3） Cryptography

Blockchain security methods include the use of public-key cryptography. A *public key* (a long, random-looking string of numbers) is an address on the blockchain. Value tokens sent across the network are recorded as belonging to that address. A *private key* is like a password that gives its owner access to their digital assets or the means to otherwise interact with the various capabilities that blockchains now support. Data stored on the blockchain is generally considered incorruptible.

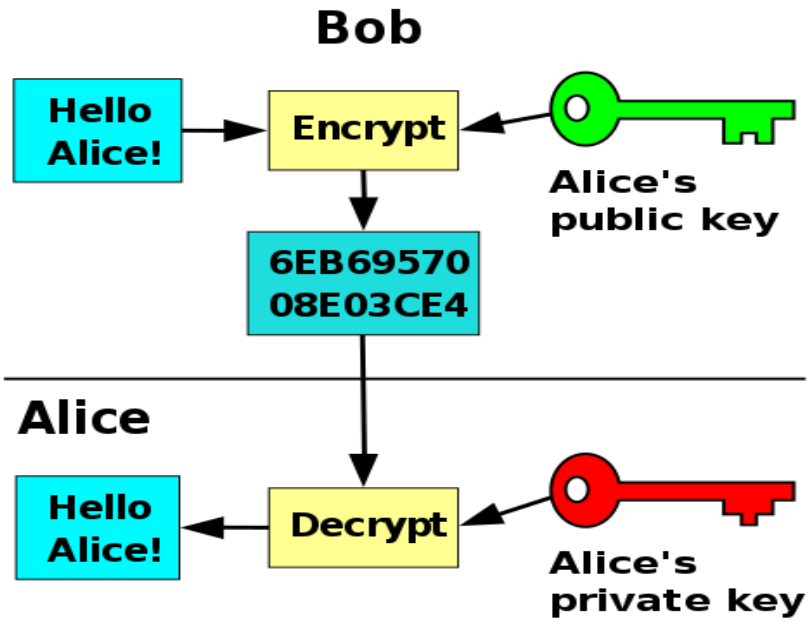


圖 22 public-key cryptography 架構

以 blockchain 技術基礎為 2008 年的論文 Bitcoin: A Peer-to Peer Electronic Cash System, 文章作者以 Satoshi Nakamoto (中本聰)為名，其以區塊鏈技術設計的電子錢幣 Bit Coin，交易架構如下圖，其所用到技術包括 hash function、Markle tree、timestamps、cryptography 等。

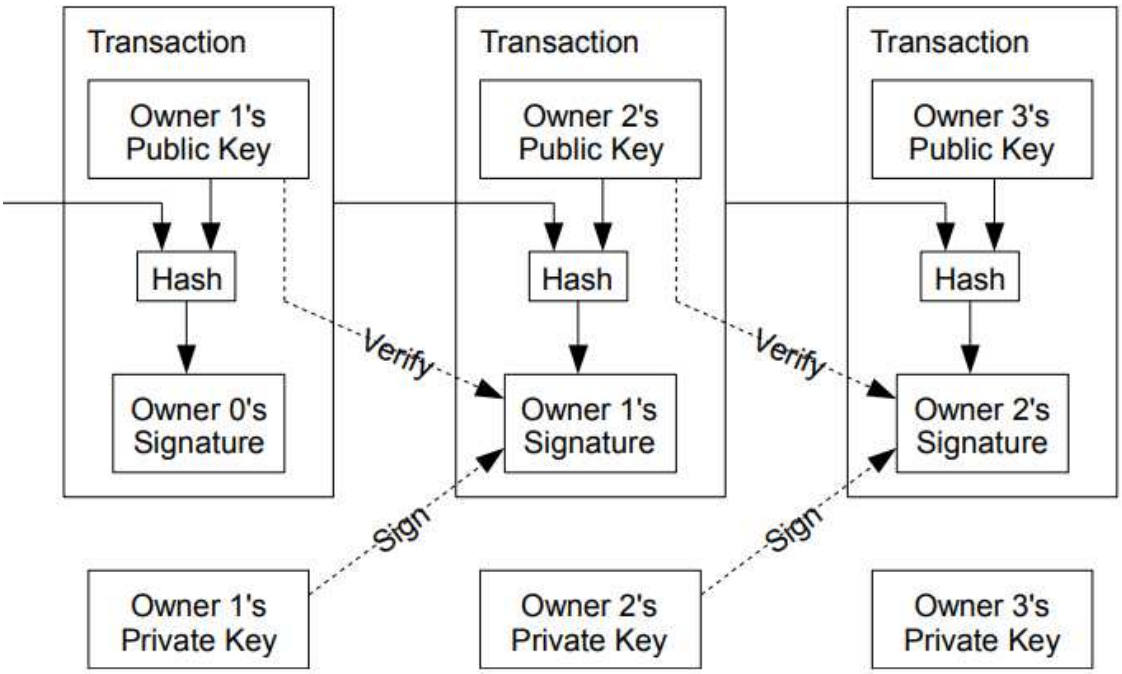


圖 23 A diagram of a bitcoin transfer

比特幣是運用密碼協議在網路上可進行點對點支付的網絡(The bitcoin network is a peer-to-peer payment network that operates on a cryptographic protocol. Users send and receive bitcoins, the units of currency, by broadcasting digitally signed messages to the network using bitcoin cryptocurrency wallet software. Transactions are recorded into a distributed, replicated public database known as the blockchain, with consensus achieved by a proof-of-work system called *mining)*

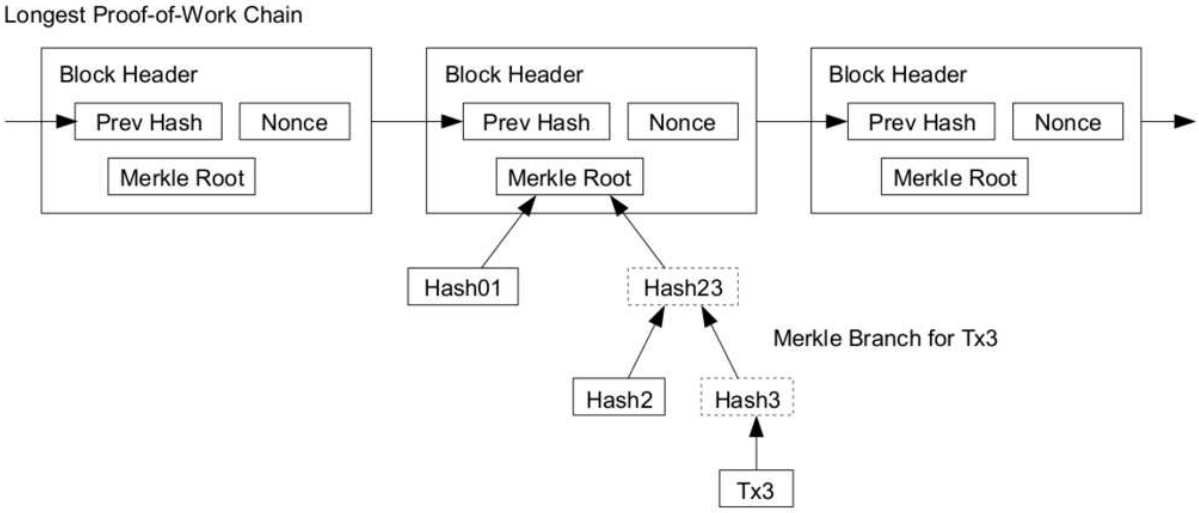


圖 24 Longest Proof-of-Work Chain

用戶以比特幣錢包軟件使用加密貨幣，將交易資訊以數字簽名加密後放上網絡廣播。分散於網路中的 bitcoin servers 比特幣伺服器利用強大運算找處謎題解答 proof of work 又稱挖掘，達成交易成立共識，使交易被記錄到分散式的公共資料庫(稱為區塊鏈)。

Economist 於 2015 年 The great chain of being sure about things 中的圖說明較為清楚，其將交易資訊 Any length of data (Transaction A、Transaction B…) 透過 hash function 產生固定長度雜湊值(Hash value #A、Hash value #B…)組成 Markle tree，將此 Markle tree 的 Root Top hash 放入 block 中，前頭加上前一個 block 的雜湊值，後面加上 timestamps，組成一個密碼謎題(cryptography puzzle)，讓 bitcoin servers 比特幣伺服器利用強大運算找處謎題解答。

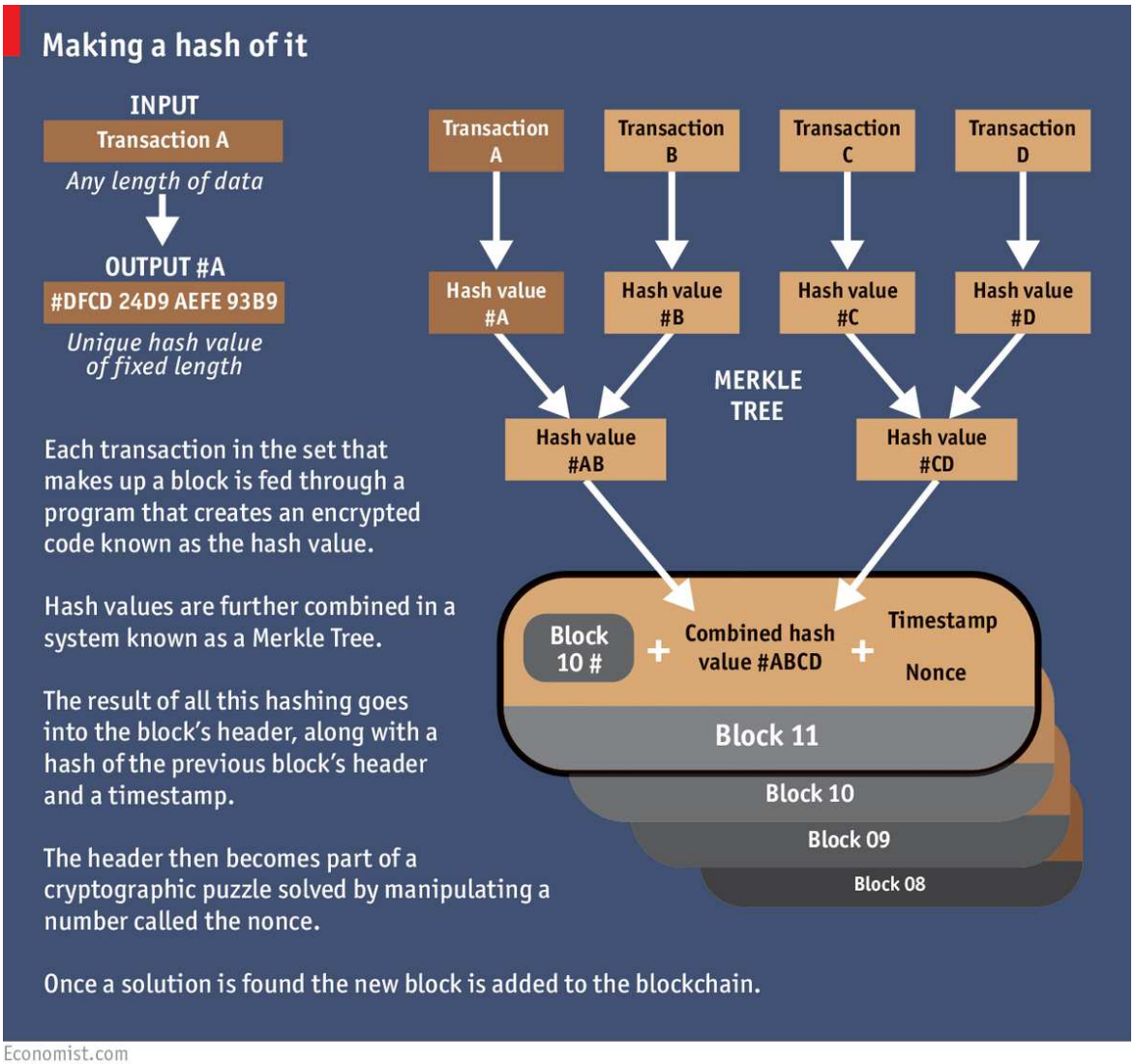
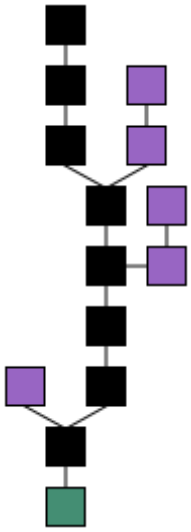


圖 25 Economist 2015 年 The great chain of being sure about things[17]



鏈結如何形成?左圖中綠色是最早起始塊，區塊鏈的作法是取鏈結最長為主，因此由起始塊（綠色）到當前塊的最長系列塊（黑色）組成主鏈。孤兒塊（紫色）存在於主鏈之外，最終都會被捨去。

Blockchain formation. The main chain (black) consists of the longest series of blocks from the genesis block (green) to the current block. Orphan blocks (purple) exist outside of the main chain.

圖 26

區塊鏈的特色為：

（1） 去中心化 (decentralized)

（2） 共同維護公開帳本 (public ledger)

（3） 防止抹滅或竄改 (tamper resistant)

（4） 具備時戳 (timestamps)

（5） 自動解決交易衝突 (conflict resolution)

圖 27 區塊鏈的四大構成要素

9、影響環境因素大數據視覺化呈現-Visualization

視覺化呈現在大數據分析領域佔有極為重要的地位，因為人是視覺的動物，對人類而言視覺震撼力是無可比擬的，如果對巨量數據有新的發現卻不能以有效的方式將資料的特性表達出來，那就無法由資料分析中說出一個動人的故事，要能讓數據說話，除了分析技術外，靠的就是視覺化的技術，林教授建議可以朝精通下面這些軟體著手 SVG、D3.js、Bootstrap、HTML、 Javascript、 and CSS 等：

(1) SVG (Scalable Vector Graphics) is an XML-based markup language for describing two-dimensional vector graphics. SVG is essentially to graphics what HTML is to text.

for circles, rectangles, and simple and complex curves. A simple SVG document consists of nothing more than the <svg> root element and several basic shapes that build a graphic together. In addition there is the <g> element, which is used to group several basic shapes together.

SVG Supports gradients, rotations, filter effects, animations, interactivity with JavaScript, and so on. But all these extra features of the language rely on this relatively small set of elements to define the graphic area.

(2) D3.js 圖 28 Data-Driven Documents D3 網站效果驚人。

(3) Bootstrap 以基本圖案為主的繪圖工具。

(4) 學習以上各種繪圖語法及軟體工具，當然也不要忘了最基本的 HTML(5), Javascript, 以及 CSS。

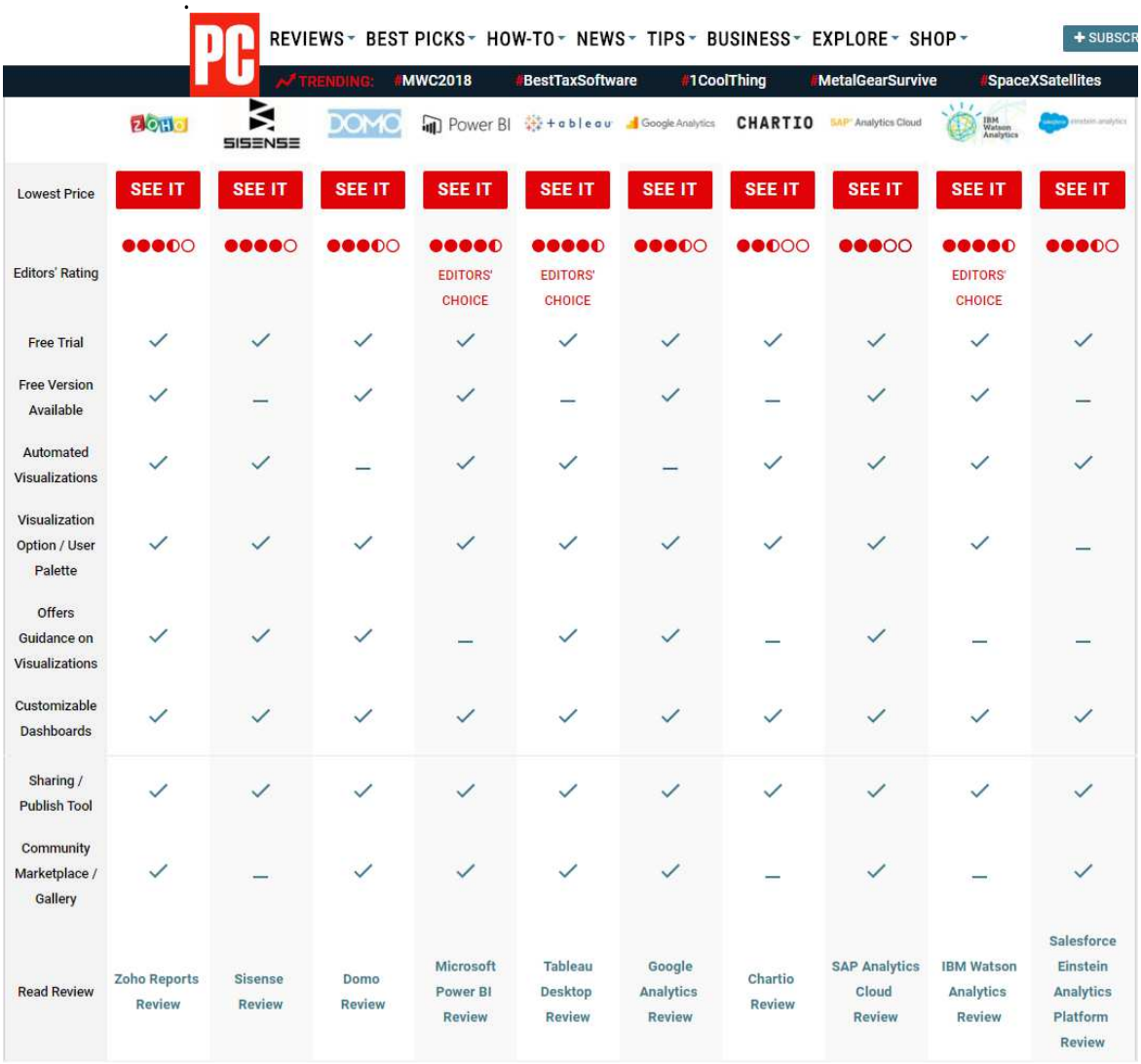


圖 29 PC Magazine Survey 2018 年最夯的視覺化軟體工具

10、 環境保護因素區塊鏈建置評估

回到原始命題：研究區塊鏈技術應用於本署空氣品質監測資料之收集、傳輸、處理、儲存、及彙整應用等各階段驗證之可行性。

評估結論：區塊鏈是為確保嵌入的資料不受竄改，並非驗證該資料是否正確，當空氣品質監測資料超出正常監測範圍，需進一步查察原因，發現污染源或機器故障異常進行資料勘誤，例如美國環保署的監測資料其校對及勘誤時間甚至長達 1 年以上；或要驗證資料的正確性，可用大數據分析進行空氣品質預測，再進而比對準確度，對於 Outlier 如係設備故障，或可利用設備維護參數進行分析研究和機器學習以習得故障之前的徵兆，如同人賴生病時會有徵兆一般，以預測來預防故障，進行維修替換，另如係資料傳輸安全的疑慮，則可於傳輸時加密因應。

區塊鏈技術的價值是在於 Powering the Internet of Value，可將有價的資料，如地產資料、智慧財資料等、區塊鏈可以協助政府機構將現有紀錄數位化，並在安全的基礎設施中對其進行管理。

當正確資料確立後，可考慮以區塊鏈技術架構公開、透明、去中心化的分散性資料運算及保存平台，政府機構的 IT 部門可以創建規則和算法，例如允許區塊鏈中的數據 在滿足預定條件時自動與第三方共享。

從長遠來看，區塊鏈技術甚至可以讓個人和組織直接掌握政府對他們所持有的所有信息。 這樣的透明度反過來可以使機構更容易實現網絡化公共服務的創建。

參、心得與建議

參、心得與建議

本次有幸參加行政院人事行政總處106年國外專題研究計畫，特別是能學習到目前當紅的大數據分析及機器學習相關主題，整個計畫從提案、英語考試選拔、專題研究面試，到出國學習整個過程，除了看到主辦單位的用心，經歷了一場自我成長學習的旅程，更看到整個大數據分析及機器學習領域的擴大應用場景，特別是與人類未來發展息息相關的人工智慧(AI)，在不同學校、企業及國家萌芽、成長的情形，茲提出下列數點心得建議，供各界參考：

一、 英文能吸收很重要：

英文這點在計畫選拔雖然已經強調過了，但是重要的原因可藉由嚴長壽於逢甲通識課程影片[18;1:15:00~]闡釋後更為明顯突出，他提出在台灣即使成績最好的學生進入最好的學校，學到最棒的知識都不一定能有競爭力，因為世界出現了更新的，打破疆域的學習方式，就是可汗學院 khanacademy 之外的 coursera、udacity、edX 等網路教學，世界一流頂尖大學如哈佛、MIT、史丹佛等的老師於 2013 年已開始將課程開在網路學院上，成為免費學習資源，所以北京教育政策當局就對大陸和最優秀的學生說不要以為你上北大、清華就很了不起，現在印度、孟加拉最遠偏區的學生，上百萬上千萬的學生在上哈佛、MIT、史丹佛的課程了，唯一的原因是因為英語是他們第一語言，要獲得世界頂尖的教學課程，沒有任何門檻，唯一的門檻就是英文，競爭已經沒有國界了，而且新的觀點一直不斷的加上來，知識每天在改變，而大數據要告訴我們的是世界上的知識，包括研究報告有 54%是以英文發表的，只有 4%是以中文發表的，如果不懂英文就把自己侷限在 4%的知識裡，科技已經把整個學習模式完全打破了，這是一個教育革命時代的來臨，我們必須用全新的觀點去看教育的未來。

因此能用英文吸收變成重中之重，以下提供增進英語的免費應

用軟體及網路聽課的撇步來協助學習：

- 利用 google 外掛 Speakit 讓文章用聽的，利用 NLP 語音辨識及合成的技術突破來訓練英語聽力，將各種網路資源直接變成有聲的資源，彌補在台灣學習英文只重視閱讀的偏差，並可增進閱讀各式各樣題材，讓以往艱澀難懂的英文論文可用聽的方式吸收，輕輕鬆鬆起來論文毫不費力，相信掌握的聽力，可以讓原本是 GEPT 中級的程度者輕鬆進階到中高級，讓中高級程度者再次突破進階到高級進階。
- 手機 APP 安裝 Google Translation，利用語音辨識查詢能讓原本聽不懂的音更有魅力。
- 在 Youtube 或網路學院上課，善用速率調整工具。

圖 30 世界上以英語為第一語言的國家[19]

二、 改變想法、作法：

在大數據分析課堂上，林教授語重心長的歸納十幾年發展經驗“Design a smart machine is more important than design a smart rule by yourself. It is human being’s nature to create rules for solving a problem. Don’t think you are the one to solve the problem, instead to design a system and let the system automatically solving problem. Then it becomes possible to scale to handle all kinds of situation.”

後來在另一場 Taiwan R User Group/MLDM Monday 所舉辦的 Data science：from data driven to deep learning，講者吳沛燊臺大醫院復健科醫師及均一教育平台資料科學家產闡釋得更清楚，其所分享的 Traditional Programming 與 Machine Learning 差異 (圖 31)，圖中上半部是傳統的程式設計，輸入端是資料及可執行的機器碼，中間是運算過程，產出是運算或 query 的結果，此處所要執行的步驟都已經是制定好的，然而下半部是機器學習的示意圖，輸入端是資料與結果，以 supervise learning 為例，用資料特徵項目 featrues 為 Data，label 為 Output，中間是機器學習模式演算法(例如 SVM、 logistic regression、Random Forest 等)，讓電腦產出程式 program (例如 how to adjust parameters to obtain maximum probability 或產出 logistic function 的 weight 等等)，一樣是用電腦做事情但是用途很不一樣，讓電腦自己產生 program 可以有很大的擴充性，例如說類神經網路做 1 層或是多層，串聯或併聯，讓整個模型越強大，以記憶越多的東西，學會更複雜的 Pattern。

所以機器學習是一種 paradigm shift (典範轉移)， paradigm 不是被動為人模仿的 example（例子、子範），而是主動的母範，範式轉移或思角轉向，在科學範疇裡，一種在基本理論上從根本假設的改變。將帶給各種應用巨大的轉變。

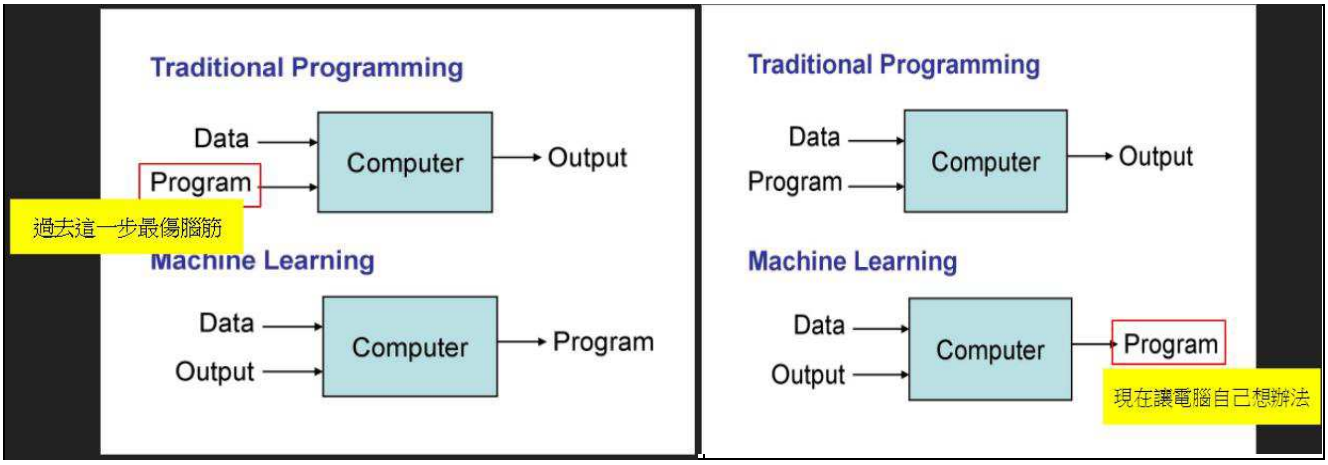


圖 31 Traditional Programming 與 Machine Learning 差異[20]

三、 破壞性創新 Disruptive innovation

亦被稱作破壞性科技、突破性創新，是指將產品或服務透過科技性的創新，並以低價特色針對特殊目標消費族群，突破現有市場所能預期的消費改變。許多人對人工智慧科技的疑慮來自於擔心人工智慧將取代人類的工作，讓許多藍領及白領階級失業，然而Tim O'Reilly解釋人類不會沒有工作Why we'll never run out of jobs[21]，他給了3個理由[22]，第一個原因：Nick Hanauer 指出「科技是用來解決人類提出的問題，世界上只要還有問題存在，就需要人類繼續工作去解決它」。例如Facebook創辦人Mark Zuckerberg與其夫人捐出30億美金用來解決人類疾病的問題[23]，二戰之後義大利的照片就像現在的敘利亞，人類不斷的重建受到戰爭、天災人禍摧殘的地區(颱風地震和核災氣爆等等)，還有很多很多問題，像氣候變遷、能源缺乏、垃圾處理、永續生態等等，這些都是需要人工智慧與人類智慧一起攜手合作解決的問題。第二個原因：在2004年由Clayton Christensen所提出的「當有吸引力產品因模組及商品化而的利潤消失時，則通常在其他相鄰的領域即將有誘人利潤之加值產品出現機會」(when attractive profits disappear at one stage in the value chain because a product becomes modular and commoditized , the opportunity to earn attractive profits with proprietary products will usually emerge at an adjacent stage.)這裡產品加值通常是靠設計與創造力，例如PC利潤消失時以Window軟體為主的Microsoft因為加值的OS贏過

IBM，但是也別擔心競爭不過Microsoft，靠internet的開放以及開源軟體的群體智慧創造使得Google、FB、Apple等公司崛起，因此現在應該致力於思考商品化(commoditized)之後加值的產品創新(Value Added Product innovation)的模式，當AI技術將某些工作(job)商品化之後，仍然會有其他相鄰的領域出現利潤之加值產品(something else become valuable)那麼，在下一階段究竟是甚麼會帶來可觀的利潤？至少可以確定的是必須靠設計與創造力進行產品加值，就像Apple公司捲土重來之時，賈伯斯宣傳的是會買蘋果產品的人是因認同蘋果產品所代表的身分地位，而非僅為了實用功能而已(a product is bought of what it means instead of what it does)。第三個原因：經濟的轉變需要時間及努力(Economic transformation takes time and efforts)。

適應未來的快速競爭，破壞性創新是擴大和開發新市場，提供新的功能的有力方法，藉由新科技及創造力去迎接挑戰，而不是消極的擔心AI會不會搶走自己手中的工作，善於利用各種網路與開源資源，改變想法跟上時代進步的腳步，「迎頭趕上」是適應未來的的不二法門。

四、 對想要在組織內啟動 AI 技術的建議做法：

1. 人力調配：在一場訪問ANDREW NG的webniar中，受訪者提出對有意於組織內啟動AI技術的機構與企業，可以在組織內新增 Chief AI Officer 職位、AI任務小組，以橫向 Matrix Across 的方式協助各部門導入AI，另外基層AI人力培訓則可獎勵員工以不算昂貴的價錢，利用網路課程進行AI基礎能力培訓[24]，於我國的公務機關，建議訂出AI數位學習補助或獎勵，利用網路數位課程如coursera、udacity、edX或是udemy、deeplearning.ai等網路學院，其於設計上都有認證制度，可以做為獎勵的目標值，設計「公務人力AI學習制度」以快速累積AI能力，利用網路學習與獎勵創造「繼續學習的環境」，讓員工完成短期、中期與長期學習目標，不要自外於AI的發展，獎勵與科技接軌，以累積機器學習典範轉移的適應能力，引領員工開創未來工作適應力，將對AI的排斥與恐懼轉為知識與能力，讓員工看得到將來AI發展的前景，並能迎頭趕上、與時併進。

2. 資源調配：正所謂舊的不去，新的不來，但在科技的領域要擁抱新的技術，並不簡單，拿文書處理的Microsoft office為例，其版權費用相當高，雖然政府已經逐步推廣Open Office欲降低成本，但因投入 Open Office 的資源有限(即使是免費軟體，並不表示不費力就會使用)，成效有限，除此之外，像DBMS、及Cloud Computing 等服務，都不斷推陳出新，以DBMS為例，下圖為 Gartner 將50多種新興分析平台 (Business intelligence Analysis Platforms)分類之情形，主要因資料種類(包括image、video、audio、text、graph等等)多元化，需要針對此資料多元化進行處理(包括AI/BI 分析技術所需的各種流程、運算、視覺化等功能之需求)。讓傳統以SQL為主的資料管理捉襟見肘，不敷所需。國內的傳統廠商R&D的投入較少，新創服務多靠政府計劃帶動，政府成為領頭羊的作用，然而若於資源調配上卻趨於保守，導致對新技術無法確切掌握，要追趕AI技術反而成為緣木求魚。

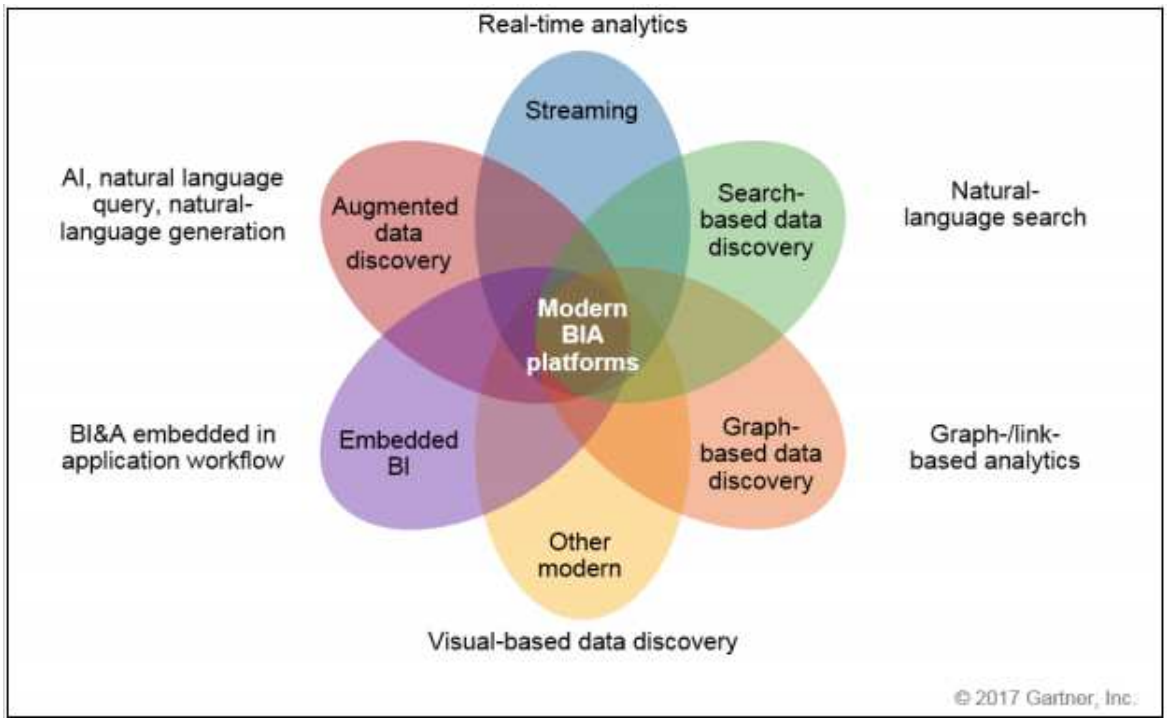


圖 32 Gartner 2017 Business intelligence Analysis Platforms分類圖

3. 政府用人的困境：考選部曾103年第 11 屆第 282 次會議考選部重要業務報告提出政府資訊人才考試晉用現況與面對未來之挑戰[25]，在面對的挑戰部分尚未提及AI的需求，然而技術人員在整個公務體系中無暢通之升遷管

道，導致國家不重視務實的技術領域(例如教改中教育部的技職教育被犧牲，恐因整個國家的公務體系中已無務實的技術高層人員能把關的原因)，考選部的報告對資訊技術的副處長因職系限制無法轉調至其他單位，又因未設置跨越科長及副處長間之第十職等高級分析師，致科長晉陞至副處長升遷管道出現斷層，且各機關修改組織法增設高級分析師職缺困難重重等(現實的狀況是除中央少數的單位可能有資訊職系的副處長外，多數的機關資訊職系的最高職缺就是科長，不轉換職系就是沒有缺)。面對AI新科技技術衝擊與挑戰，政府應本考用合一制度，增設導入AI所需的職務職系(資料科學)與人才，考試的科目搭配技術領域的日新月異而快速變化調整，建議參考業界取才的線上實作測試 (如 kaggle 競賽方式)。如能從改善用人制度方面著手，網羅或培訓AI專業的人才，以合理的升遷管道來協助拓展政府各部門數據運用發展，以培養並快速累積此領域的人才，落實於制度中，以創造環境及條件讓政府組織內能充分利用AI以增進政府組織內之洞察力及領導力。

肆、參考資料

1. http://wikibon.org/wiki/v/Big_Data_Vendor_Revenue_and_Market_Forecast_2013-2017
2. https://www.idc.com/getdoc.jsp?containerId=prUS42371417
3. https://www.openhub.net/p/apache-spark
4. https://buzzorange.com/techorange/2017/10/02/theano-say-goodbye/
5. 16 https://en.wikipedia.org/wiki/Comparison_of_deep_learning_software
6. https://zh.wikipedia.org/wiki/Apache_Spark
7. http://shop.oreilly.com/product/0636920028512.do
8. https://customers.microsoft.com/en-US/story/a-growing-banks-fruitful-new-approach-to-data-helps-it
9. https://blogs.technet.microsoft.com/dataplatforminsider/2017/03/07/gartner-names-microsoft-a-leader-in-the-magic-quadrant-for-data-management-solutions-for-analytics-dmsa/
10. https://www.youtube.com/watch?v=bh0AEVwSJzA&list=TLGG3MVSS87ieHUyMzAyMjAxOA

    learning Spark by Holden Karau, Andy Konwinski, Patrick Wendell, and Matei Zaharia (O'Reilly). Copyright 2015 Databricks, 978-1-449-35862-4."
11. https://www.datasciencecentral.com/profiles/blogs/top-20-open-data-sources
12. https://en.wikipedia.org/wiki/Confusion_matrix
13. http://online.liebertpub.com/doi/full/10.1089/big.2012.0004
14. http://smallblue.research.ibm.com/projects/snvalue/
15. https://en.wikipedia.org/wiki/Merkle_tree
16. https://en.wikipedia.org/wiki/Trusted_timestamping
17. https://www.economist.com/news/briefing/21677228-technology-behind-bitcoin-lets-people-who-do-not-know-or-trust-each-other-build-dependable
18. https://www.youtube.com/watch?v=XzdZIICk5cc&t=4612s
19. https://zh.wikipedia.org/wiki/%E8%8B%B1%E8%AF%AD%E5%9B%BD%E5%A

E%B6%E5%92%8C%E5%9C%B0%E5%8C%BA%E5%88%97%E8%A1%A8

20. https://www.facebook.com/Tw.R.User/videos/948393558647146/
21. https://www.youtube.com/watch?v=K7ZFxPnL0Ww
22. https://www.youtube.com/watch?v=xRmQTWpkaVU&t=153s
23. http://www.yoursnews.in/zuckerberg-said-can-we-cure-all-diseases-in-our-childrens-lifetime/
24. https://hbr.org/webinar/2017/07/deep-learnings-next-frontier
25. http://wwwc.moex.gov.tw/main/content/wHandMenuFile.ashx?menu_id=2619&strType=

伍、附錄

1. 安裝Apache Hadoop的程序筆記
2. 於AWS上新開1台 ubuntu 及install pyspark 實作流程。

Hadoop Installation Tutorial for mac and linux

Step 1: Install Prerequisites

- Install Homebrew for Easy Installation of Hadoop

go to https://brew.sh/ for simple one-line installation guide.

Step 2: Install Java

- Download and Install Java from

https://www.java.com/en/download/mac_download.jsp

- Check your java version by executing

```
java -version
```

Step 3: Use Homebrew to install Hadoop

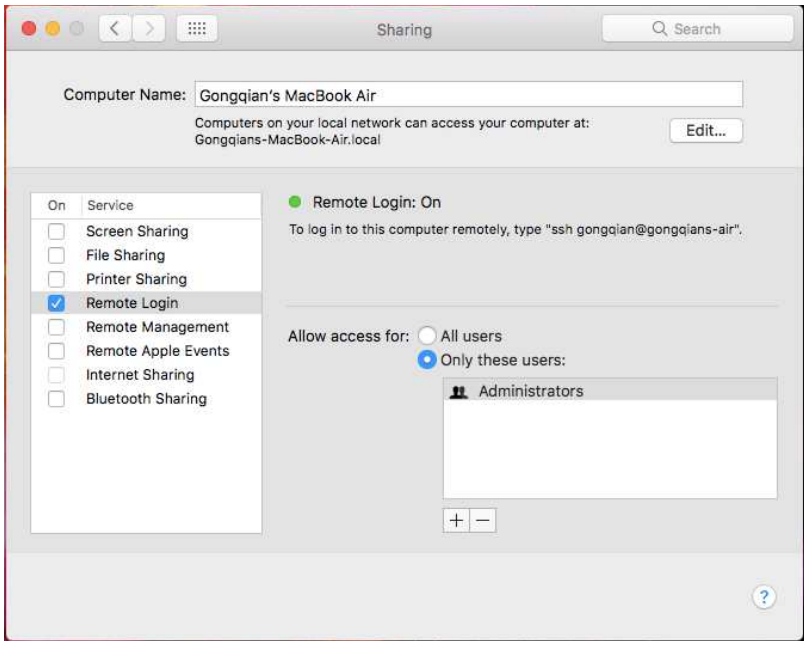
- Install Hadoop

```
brew install Hadoop
```

- Check Hadoop version

```
hadoop version
```

- enable "remote sharing" : System Preference > Sharing > Remote Login



Step 4: Configure Hadoop

Please learn how to use vim before you configuring Hadoop.

Be careful of unexpected spaces when you copy and paste command lines.

- a) Edit hadoop-env.sh file

```
sudo vim /usr/local/Cellar/hadoop/2.8.1/libexec/etc/hadoop/hadoop-env.sh
```

Delete the line started with "# export HADOOP_OPTS=Djava…"

Copy and paste this line in the same position

```
export HADOOP_OPTS="$HADOOP_OPTS -Djava.net.preferIPv4Stack=true -
Djava.security.krb5.realm= -Djava.security.krb5.kdc="
```

- b) Edit hdfs-site.xml

```
sudo vim /usr/local/Cellar/hadoop/2.8.1/libexec/etc/hadoop/hdfs-site.xml
```

Between the <configuration></configuration> tags insert:

```
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
```

- c)Edit core-site.xml

```
sudo vim /usr/local/Cellar/hadoop/2.8.1/libexec/etc/hadoop/core-site.xml
```

Between the <configuration></configuration> tags insert:

```
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
```

- d) Edit mapred-site.xml

```
cd /usr/local/Cellar/hadoop/2.8.1/libexec/etc/hadoop
mv mapred-site.xml.template mapped-site.xml
sudo vim /usr/local/Cellar/hadoop/2.8.1/libexec/etc/hadoop/mapred-site.xml
```

Between the <configuration></configuration> tags insert:

```
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
```

- e) Edit yarn-site.xml

```
sudo vim /usr/local/Cellar/hadoop/2.8.1/libexec/etc/hadoop/yarn-site.xml
```

Between the <configuration></configuration> tags insert:

```
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
```

- f) Add easy launch words to your "profile"

```
sudo vim ~/.profile
```

In a blank section copy and paste the two following lines:

```
alias hstart="/usr/local/Cellar/hadoop/2.8.1/sbin/start-dfs.sh;/usr/local/Cellar/
hadoop/2.8.1/sbin/start-yarn.sh"
alias hstop="/usr/local/Cellar/hadoop/2.8.1/sbin/stop-yarn.sh;/usr/local/Cellar/
hadoop/2.8.1/sbin/stop-dfs.sh"
```

```
source ~/.profile
```

Step 5: Configure SSH Key for localhost

- Generate ssh key (skip password setting)

```
ssh-keygen -t rsa
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

- Test you can access localhost by typing:

```
ssh localhost
```

output should be like "Last login: Sun Sep 16 23:51:46 2017 from ::1"

Step 6: Setup local variables

```
sudo vim ~/.bashrc
```

Copy and paste the following, check and adjust jdk version before copy and paste:

```
# Set Hadoop-related environment variables
export HADOOP_HOME=/usr/local/Cellar/hadoop/2.8.1
# Set JAVA_HOME (we will also configure JAVA_HOME directly for Hadoop later on) export
JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home
# Some convenient aliases and functions for running Hadoop-related commands unalias fs
&> /dev/null
alias fs="hadoop fs"
unalias hls &> /dev/null
alias hls="fs -ls"
#  If you have LZO compression enabled in your Hadoop cluster and
#  compress job outputs with LZOP (not covered in this tutorial):
#  Conveniently inspect an LZOP compressed file from the command
# line; run via:
#
#  $ lzohead /hdfs/path/to/lzop/compressed/file.lzo
#
#  Requires installed 'lzop' command.
#
lzohead () {
hadoop fs -cat $1 | lzop -dc | head -1000 | less
}
# Add Hadoop bin/ directory to PATH
```

Then type:

```
source ~/.bashrc

export PATH=$PATH:$HADOOP_HOME/bin

export JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk(your jdk
version might be different)/Contents/Home
```

Step 7: Format HDFS and make new directory

```
cd /usr/local/Cellar/hadoop/2.8.1/

./sbin/start-dfs.sh

./bin/hdfs namenode -format

./bin/hdfs dfs -mkdir /user
```

```
./bin/hdfs dfs -mkdir /user/(your computer's name)

./sbin/start-yarn.sh

./sbin/stop-yarn.sh

./sbin/stop-dfs.sh
```

Step 8: Start Hadoop (runs start-dfs.sh and start-yarn.sh)

- To start: `hstart`
- Check what nodes are running by typing: `jps`

```
Gongqians-MacBook-Air:~ gongqian$ jps
4577 Jps
3106 DataNode
3922 NodeManager
3012 NameNode
3222 SecondaryNameNode
3821 ResourceManager
Gongqians-MacBook-Air:~ gongqian$
```

- Then stop: `hstop`

Step 9: Try file system shell commands in HDFS

- Check Hadoop website

hadoop.apache.org/docs/r2.8.1/hadoop-project-dist/hadoop-common/FileSystemShell.html

---

step 10:Download Hadoop Examples jar and test some example

```
cd /usr/local/Cellar/hadoop/2.8.1/
wget https://www.dropbox.com/s/cyuah7lc31g0x3h/hadoop-mapreduce-examples-2.6.0.jar
```

- Example 1: Approximate the value of Pi

Execute: `./bin/hadoop jar hadoop-mapreduce-examples-2.6.0.jar pi 10 100`

- Example 2: Solve a Sudoku Puzzle

Create the following file:

```
sudo vim  puzzle1.dta
```

Copy and paste following text into puzzle1.dta:

```
8 5 ? 3 9 ? ? ? ?
? ? 2 ? ? ? ? ? ?
? ? 6 ? 1 ? ? ? 2
? ? 4 ? ? 3 ? 5 9
? ? 8 9 ? 1 4 ? ?
3 2 ? 4 ? ? 8 ? ?
9 ? ? ? 8 ? 5 ? ?
? ? ? ? ? ? 2 ? ?
? ? ? ? 4 5 ? 7 8
```

Execute: `./bin/hadoop jar hadoop-mapreduce-examples-2.6.0.jar sudoku puzzle1.dta`

- Example 3: Word counter

Create file romeo_juliet.txt

check http://shakespeare.mit.edu/romeo_juliet/full.html ,and copy full script into romeo_juliet.txt

Add romeo_juliet.txt to HDFS directory then execute:

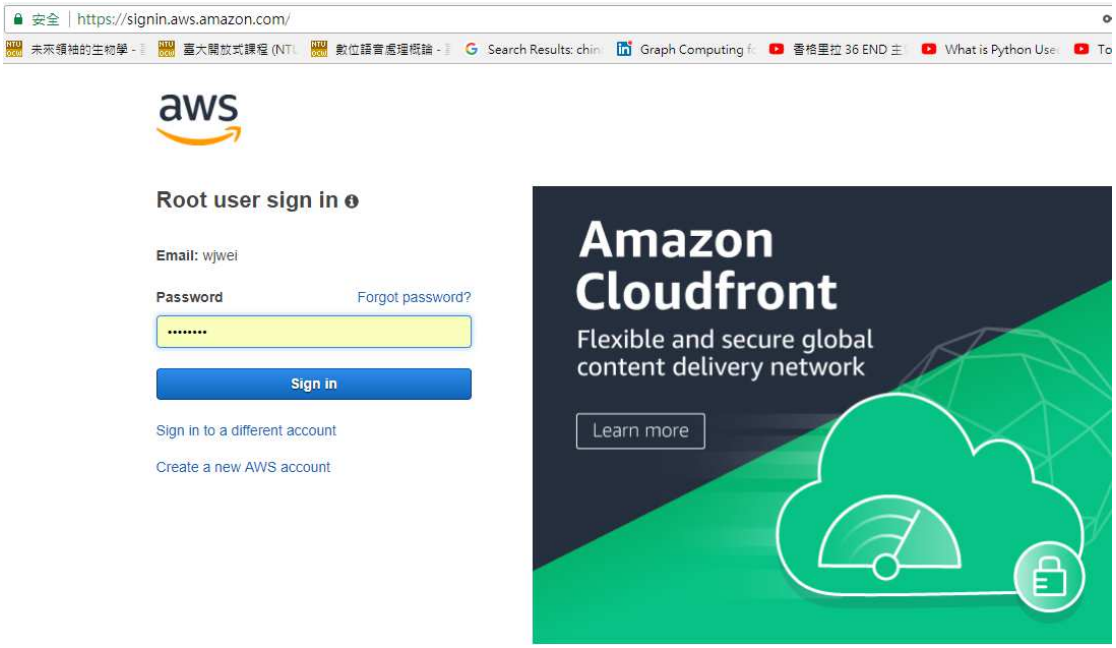
Execute: `./bin/hadoop jar hadoop-mapreduce-examples-2.6.0.jar wordcount romeo_juliet.txt output.txt`

Step 11: Installation Success

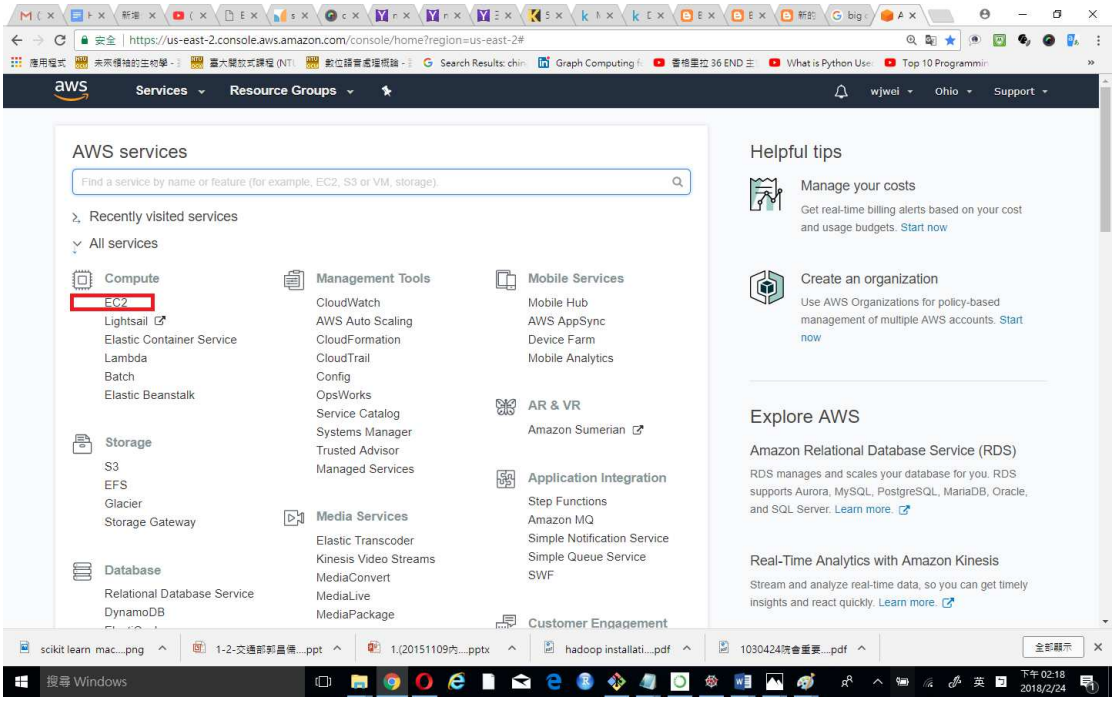
Now you can try 5 simple operations on your three datasets

附錄 2 於 AWS 上新開 1 台 ubuntu 及 install pyspark 實作流程

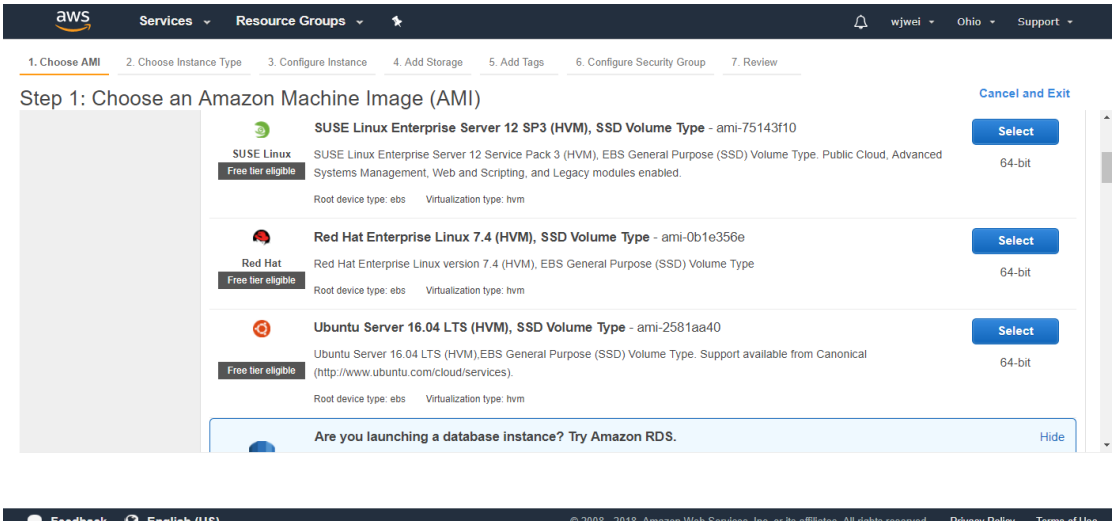
- 使用者須先由 https://aws.amazon.com/tw/ 註冊帳號
- Login



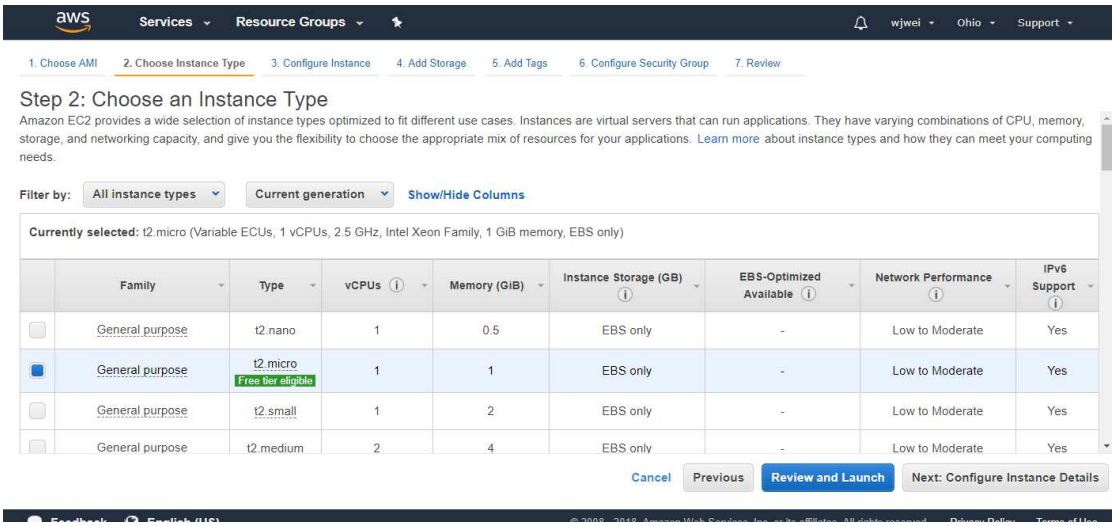
- 先選 ec2



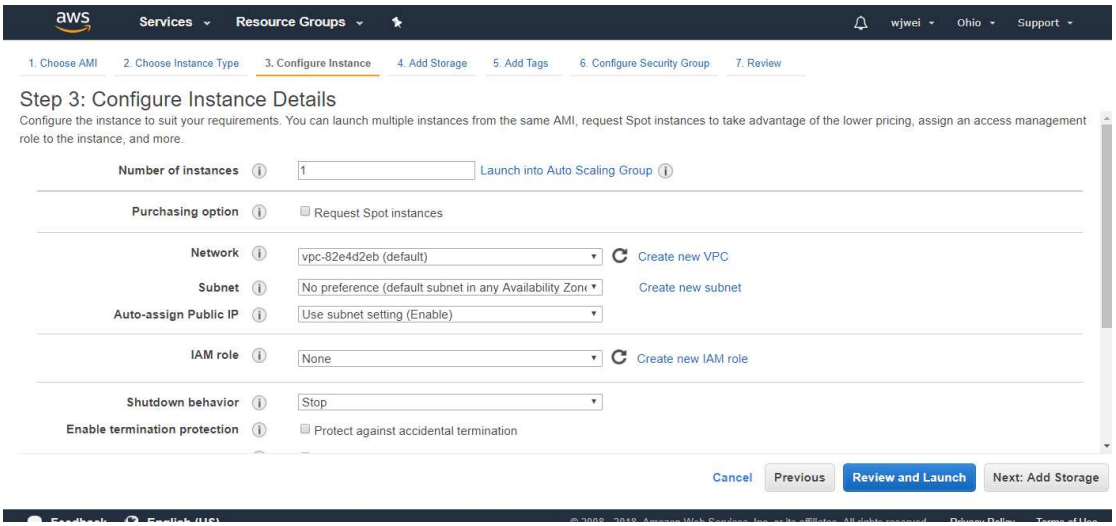
- Step1: Choose an Amazon Machine Image 選擇 ubuntu 系統



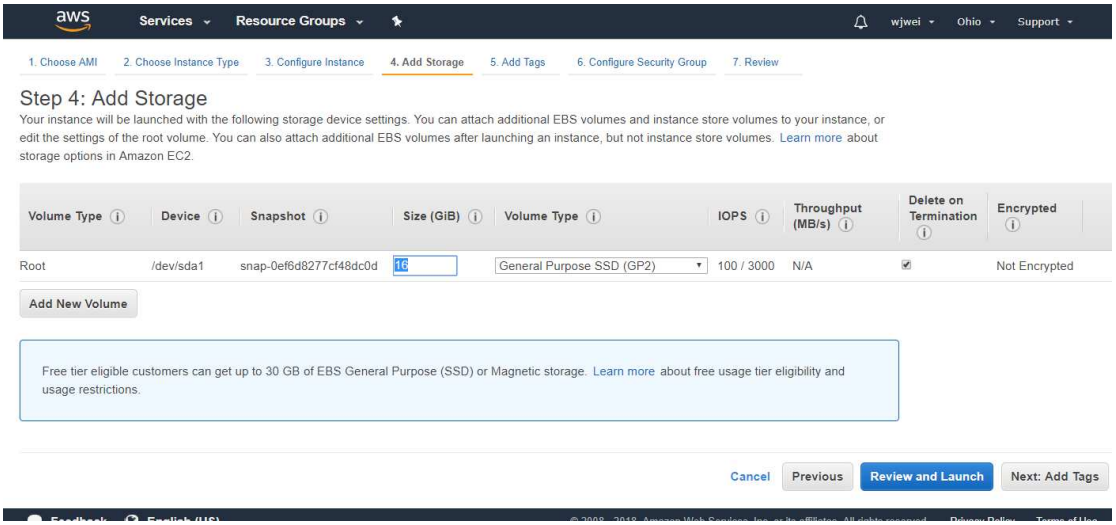
- Step2:Choose an Instance Type



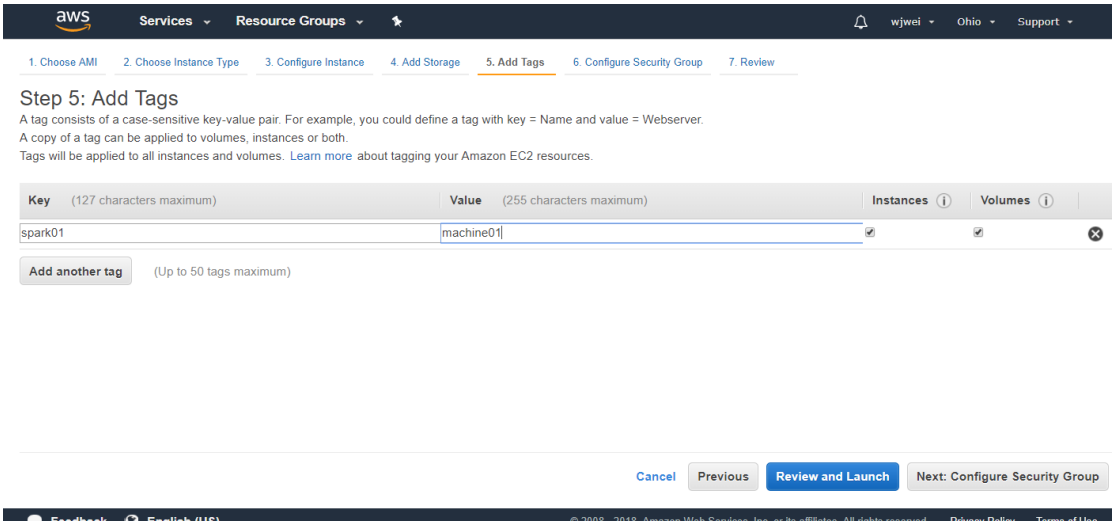
- Step3



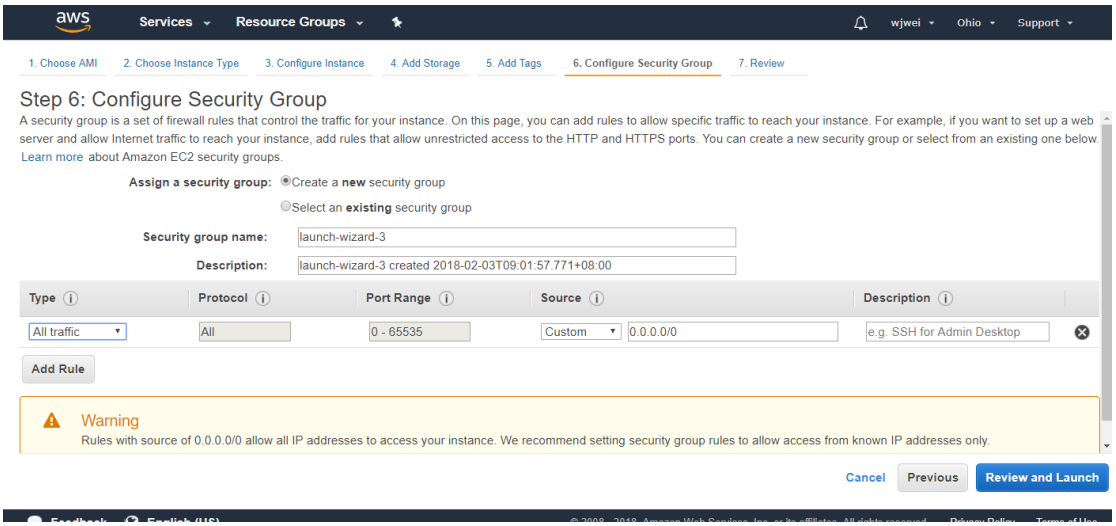
- Step4



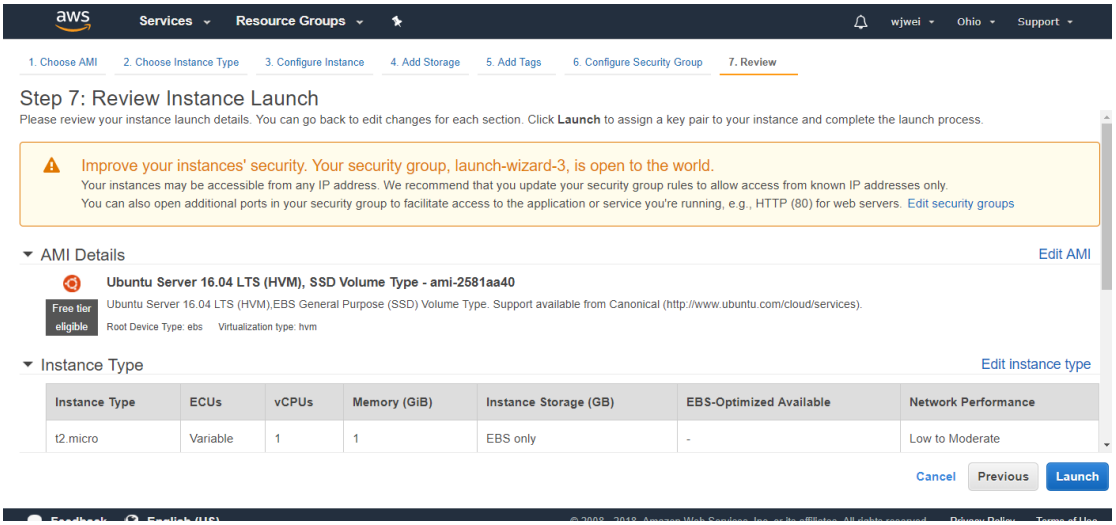
- Step5



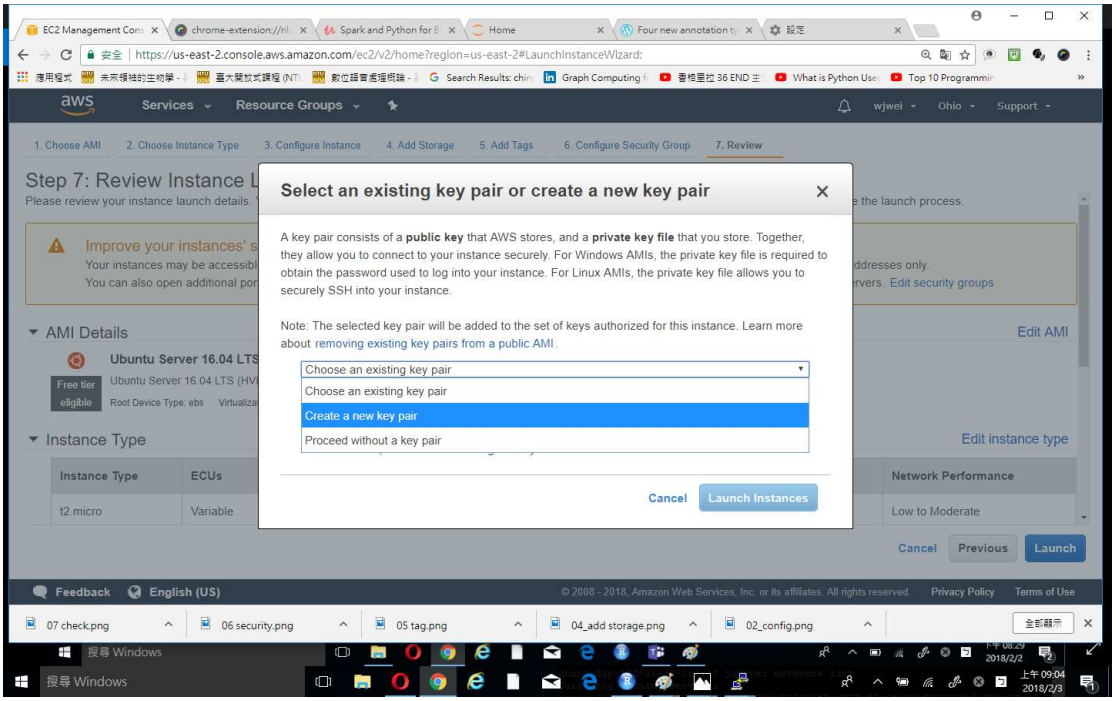
- Step6



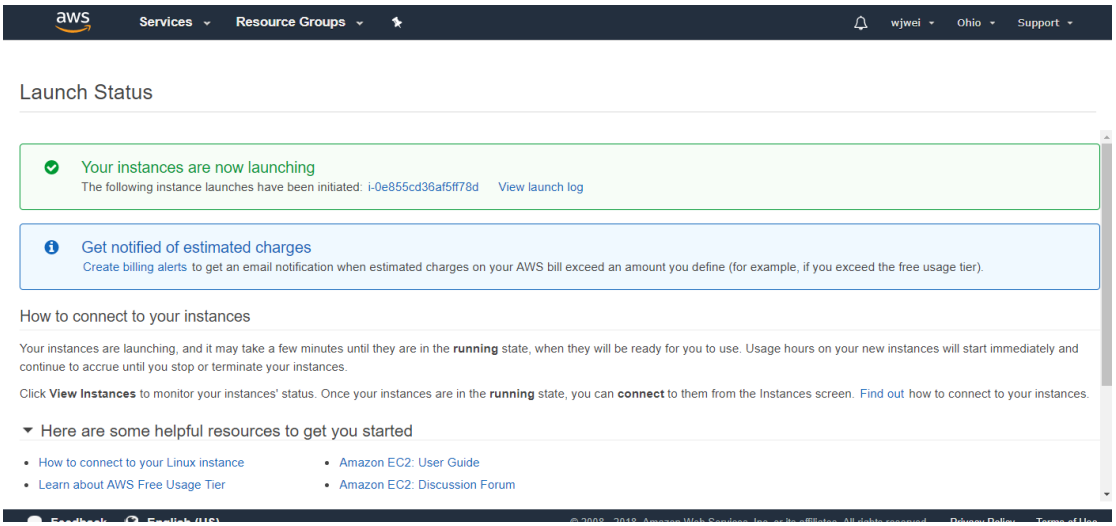
- Step7

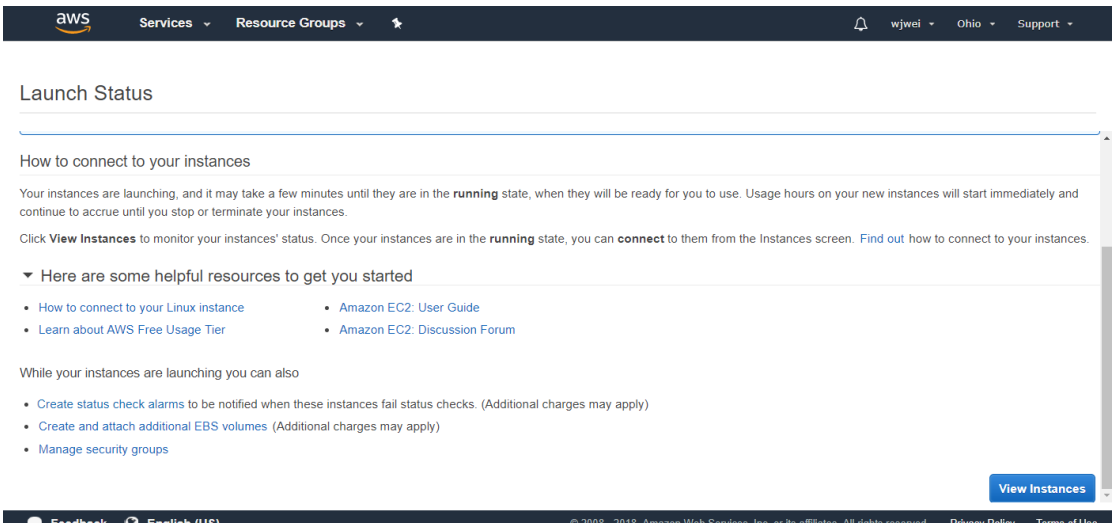


- Step 8

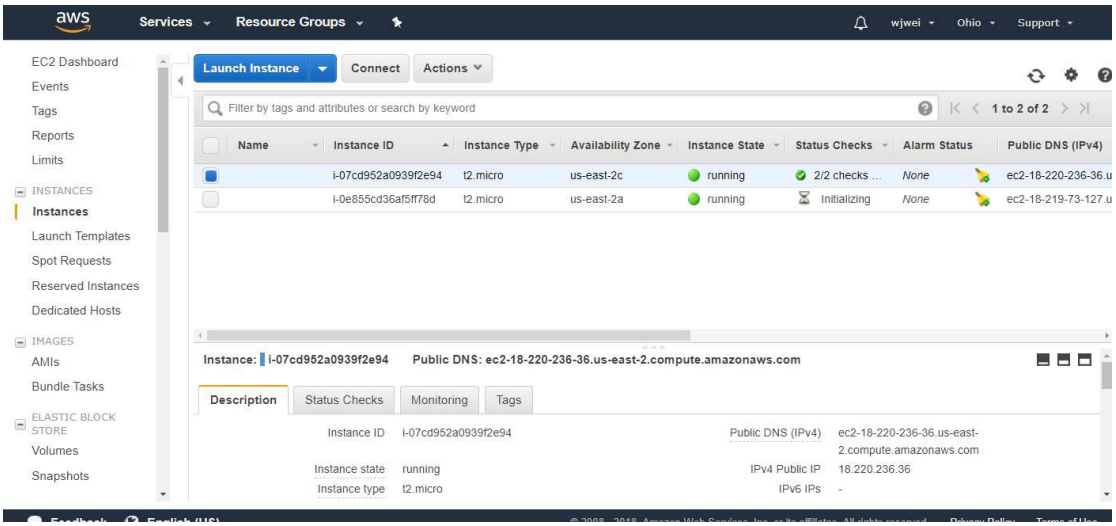


此處會產生一組 PEM 檔，後續使用者端連線前需要下載 puttygen 用來產生.ppk 檔，做為每次連線認證碼，並下載 putty.exe 以執行 local 端遠端登入連線。(下載 puttygen 及 putty.exe 畫面紀錄 ec2 unbuntu 開設完成之後)





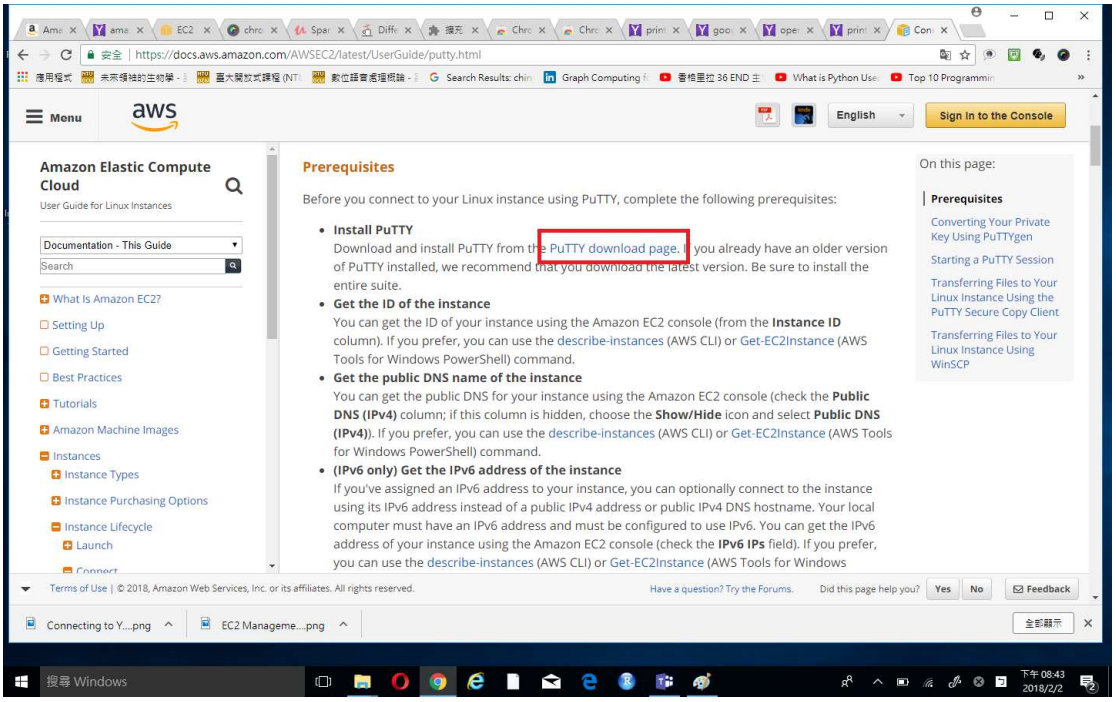
完成於 AWS 上新開 1 台 ubuntu

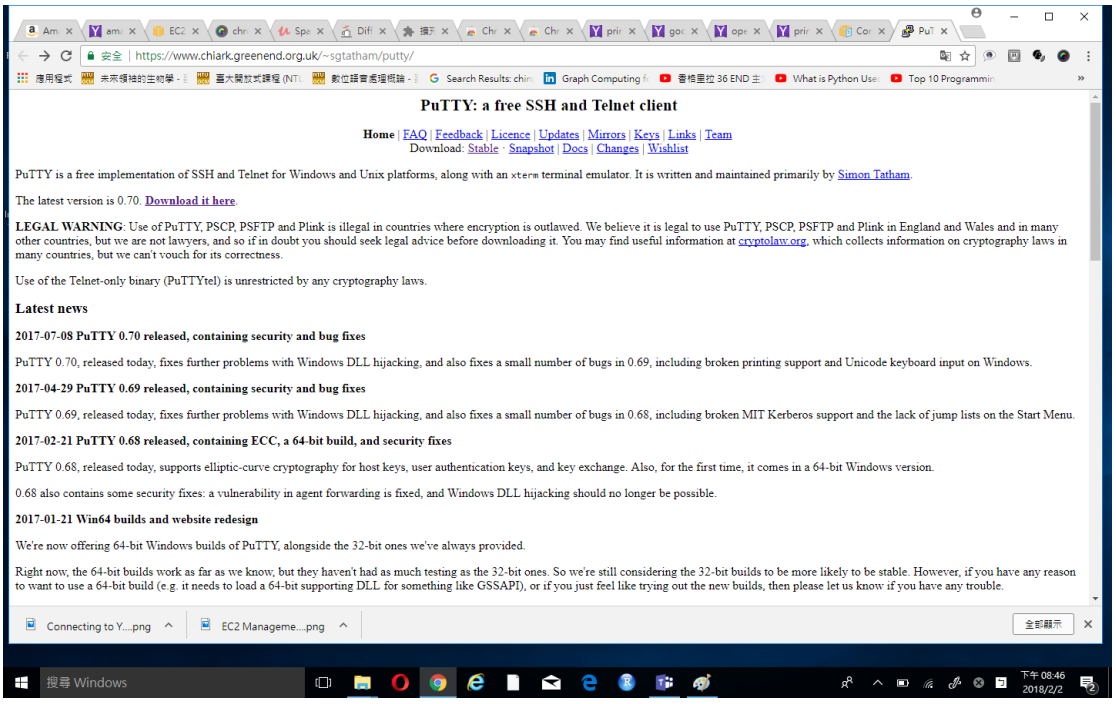


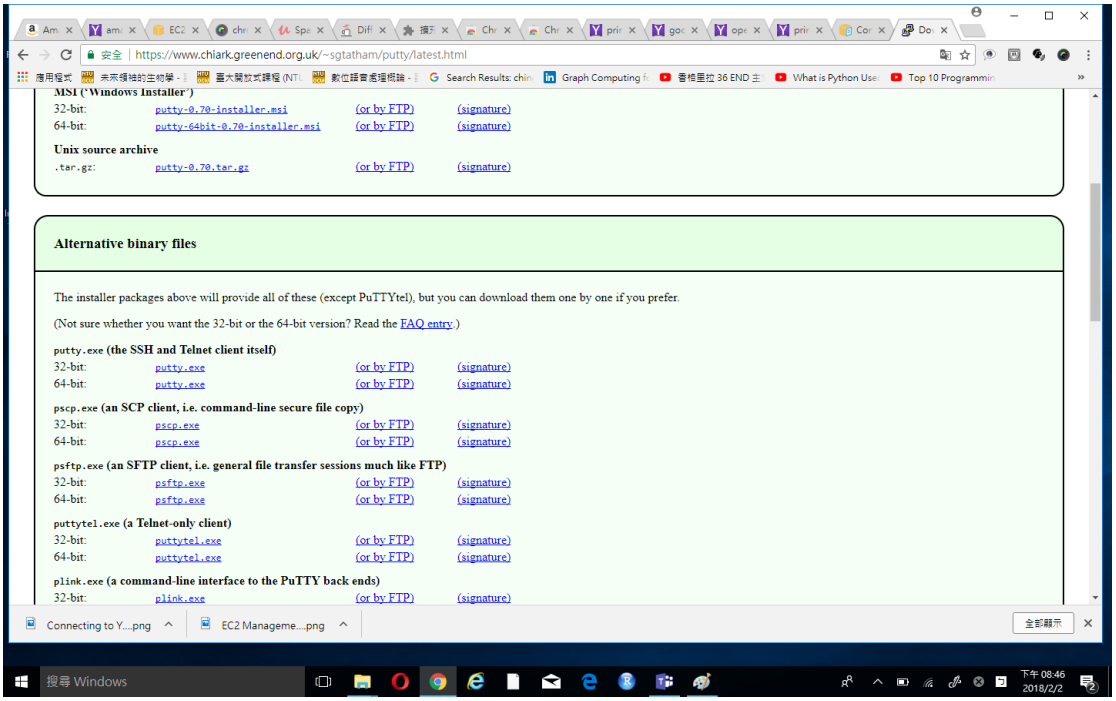
於瀏覽器中搜尋 ssh windows ec2

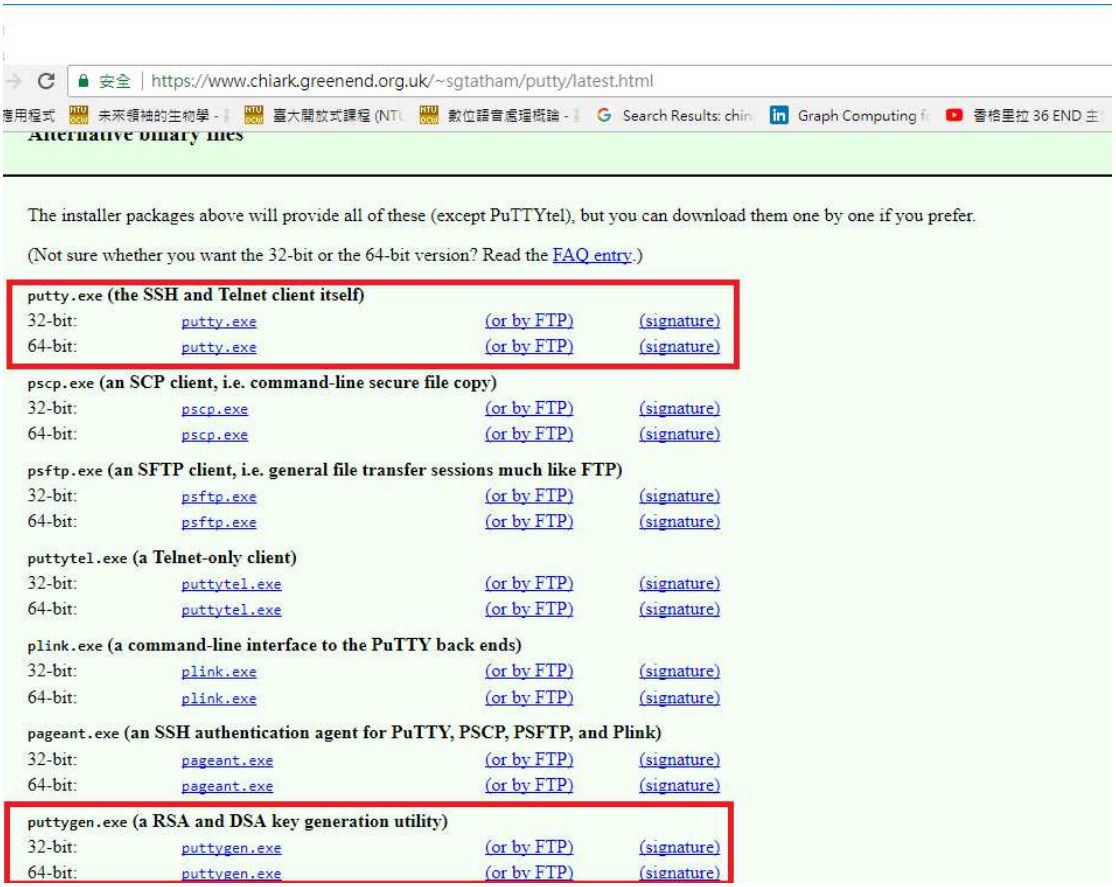
Connecting to Your Windows Instance - Amazon Elastic Compute Cloud
https://docs.aws.amazon.com/AWSEC2/.../WindowsGuide/connecting_to_windows_in...
Amazon **EC2** instances created from most **Windows** Amazon Machine Images (AMIs) enable you to connect using Remote Desktop. Remote Desktop uses the Remote Desktop Protocol (RDP) and enables you to connect to and use your instance in the same way you use a computer sitting in front of you. It is available on ...

選上述連結可前往下載 puttygen 及 putty.exe 之網頁



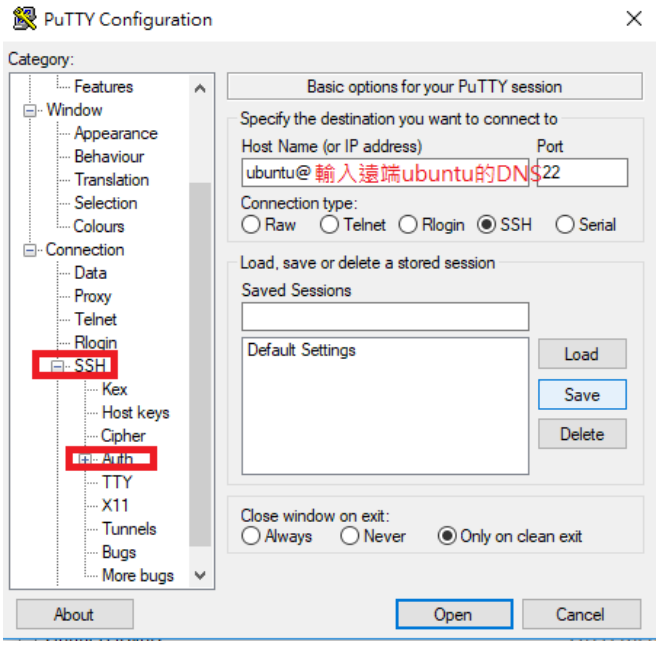


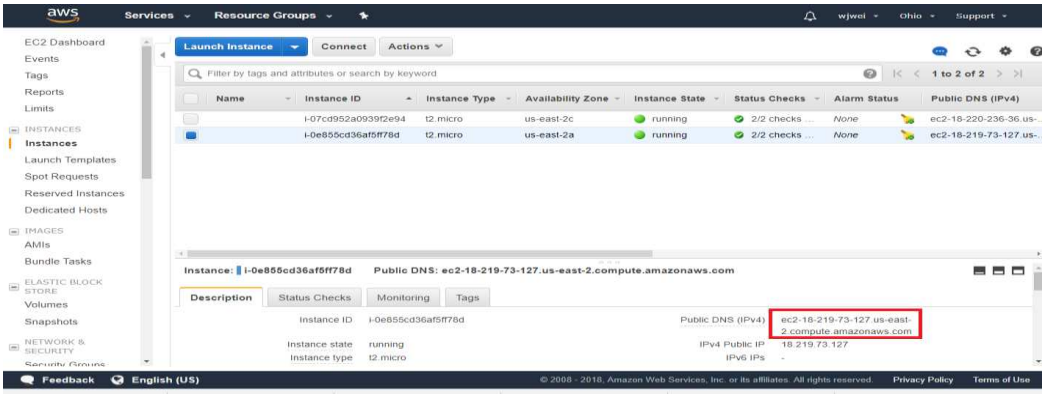


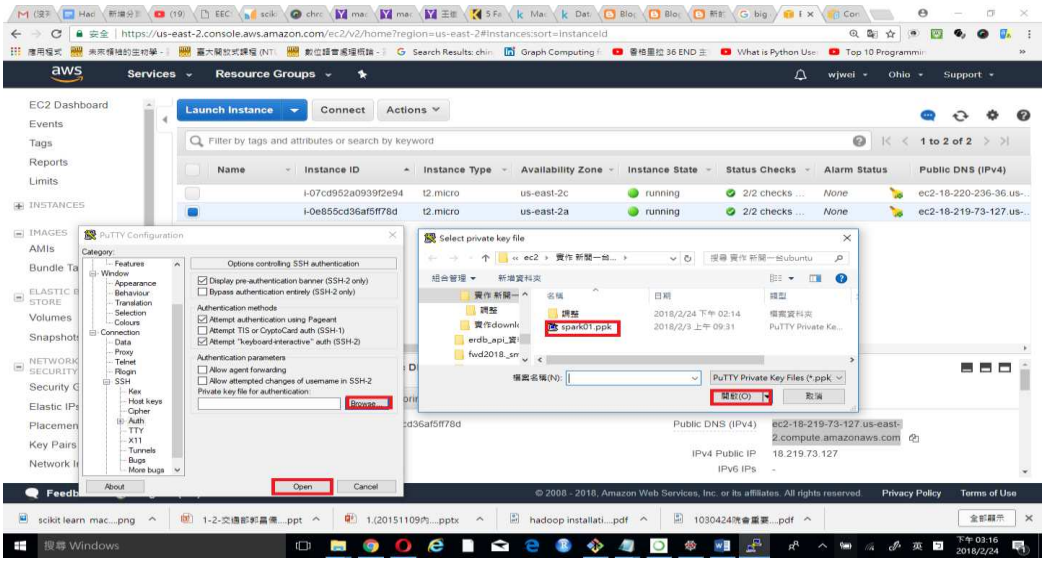


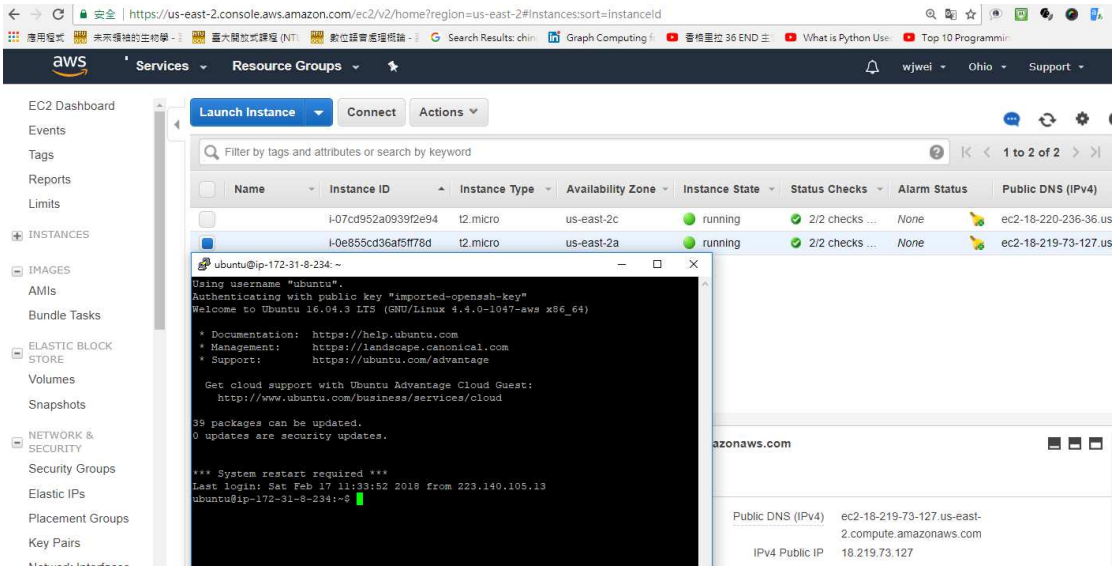
執行 puttygen.exe 產生 .ppk 檔，詳細步驟請閱讀網站說明。

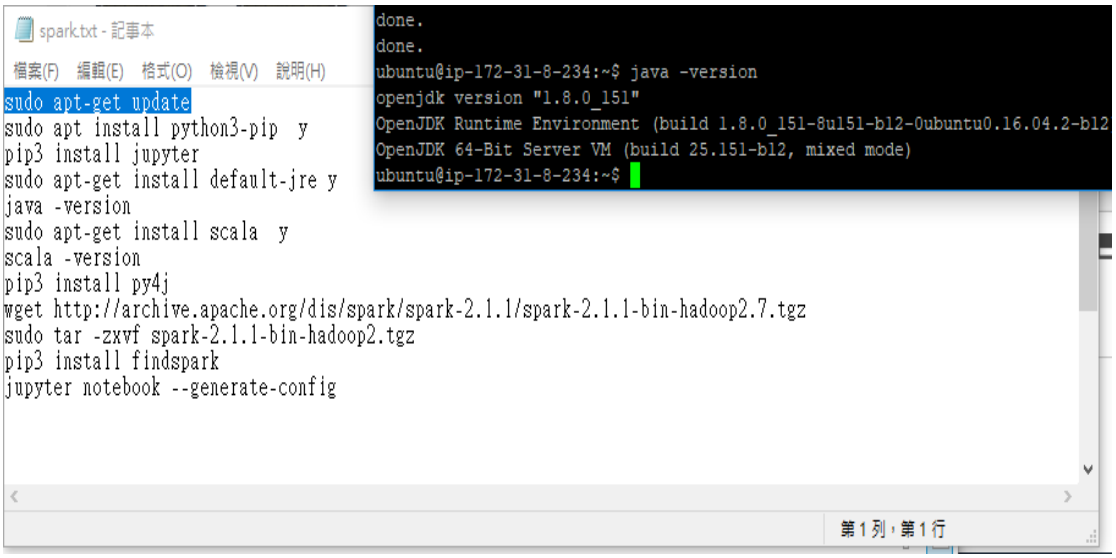
遠端連線執行 putty.exe








Services
Resource Groups
wjwei
Ohio
Support
EC2 Dashboard
Events
Tags
Reports
Limits
INSTANCES
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Launch Instance
Connect
Actions
Filter by tags and attributes or search by keyword
1 to 2 of 2
Name
Instance ID
Instance Type
Availability Zone
Instance State
Status Checks
Alarm Status
Public DNS (IPv4)
i-07cd952a0939f2e94
t2.micro
us-east-2c
running
2/2 checks ...
None
ec2-18-220-236-36.us
i-0e855cd36af5ff78d
t2.micro
us-east-2a
running
2/2 checks ...
None
ec2-18-219-73-127.us
ubuntu@ip-172-31-8-234: ~
Using username "ubuntu".
Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-1047-aws x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud
39 packages can be updated.
0 updates are security updates.
*** System restart required ***
Last login: Sat Feb 17 11:33:52 2018 from 223.140.105.13
ubuntu@ip-172-31-8-234:~$
azonaws.com
Public DNS (IPv4)
ec2-18-219-73-127.us-east-2.compute.amazonaws.com
IPv4 Public IP
18.219.73.127


spark.txt - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)
sudo apt-get update
sudo apt install python3-pip y
pip3 install jupyter
sudo apt-get install default-jre y
java -version
sudo apt-get install scala y
scala -version
pip3 install py4j
wget http://archive.apache.org/dis/spark/spark-2.1.1/spark-2.1.1-bin-hadoop2.7.tgz
sudo tar -zxvf spark-2.1.1-bin-hadoop2.7.tgz
pip3 install findspark
jupyter notebook --generate-config
done.
done.
ubuntu@ip-172-31-8-234:~$ java -version
openjdk version "1.8.0_151"
OpenJDK Runtime Environment (build 1.8.0_151-8u151-b12-0ubuntu0.16.04.2-b12
OpenJDK 64-Bit Server VM (build 25.151-b12, mixed mode)
ubuntu@ip-172-31-8-234:~$
第 1 列，第 1 行