Lecture (講) 1 Introduction (介紹)

- 1.1 What Is Machine Learning (機器學習)?
- 1.2 Types of Machine Learning Systems (系統)
- 1.3 Main Challenges (主要挑戰)
- 1.4 Applications (應用)
- Aurelien Geron, Hands-On (實作) Machine Learning With Scikit-Learn and Tensorflow
- McKinsey (麥肯錫), Artificial intelligence: The next digital frontier, 2017 (Five case studies)
- 長榮大學資設院資管系許志華
- 上課內容 http://chhsu135.blogspot.com/
  - 標籤上課研究服務

1.1 What Is Machine Learning?

- Arthur Samuel coined (創造) the term "machine learning" (機器學習) in 1959
  - Machine Learning: Field of study (研究領域) that gives computers the ability to learn (學習能力) without being explicitly (明確地) programmed (程式設計).
- Herbert Simon: Learning is any process by which a system improves (改進) performance (性能) from experience (經驗).
  - Turing Award in 1975
  - Nobel Prize in Economics in 1978

Engineering-oriented (工程導向) one

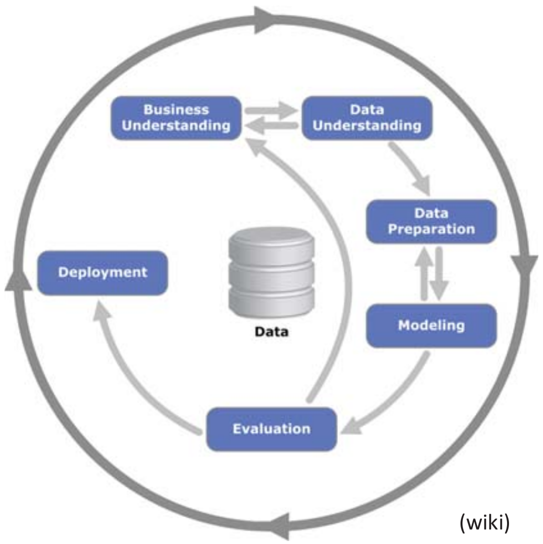
- Spam filter (垃圾郵件過濾器)
  - Classification (分類) task (工作) T: spam or ham (nonspam) for new emails. Binary (二元)
  - Experience (經驗) E is the *training data* (訓練資料): Given examples of spam emails (e.g., flagged (標示) by users)
  - Performance measure (性能測量) P: The ratio (比率) of correctly classified (正確分類) emails, *accuracy* (準確度)
    - 演算法設計
- *Tom M. Mitchell* (1997): Machine is learning if its performance on T, as measured (量測) by P, improves (改進) with experience E.

Data Mining (資料探勘)

- Jure Leskovec, Anand Rajaraman, Jeff Ullman
  - Mining of Massive Datasets (大規模數據挖掘)
  - http://www.mmds.org/
  - Discovery (發現) of useful, possibly unexpected (意外的), patterns (形態) in data

Cross Industry Standard Process (跨行業標準流程) for Data Mining (資料探勘)

- 了解商業問題
  - 領域知識
- 了解資料
- 準備資料
  - 資料前處理
- 建模
  - 數學，工具
- 評估
- 部署
  - 溝通
- A. Geron, Appendix (附錄) B, (6-page) checklist (清單)
- A. Ng, machine learning yearning (嚮往)



(wiki)

1.2 Types (類型) of Machine Learning Systems (系統)

- Supervised Learning (監督學習)
  - classification (分類)
  - Regression (迴歸)
- Unsupervised Learning (非監督學習)
- Reinforcement Learning (強化學習)
- 監督學習: The training data (訓練資料) you feed to the algorithm (演算法) includes the desired solutions (想要的答案), called *labels* (標籤)
  - improves (改進)

1.2.1 Supervised Learning (監督學習): Classification (分類)

- 5041: MNIST database (數據庫)
  - Modified (修正) National Institute of Standards and Technology (美國國家標準暨技術研究院)
  - American Census Bureau (人口普查局) employees (僱員) and high school students (高中生)
- Maine coon cat (緬因浣熊貓), Malayan tiger (馬來虎) (wiki)

https://www.kaggle.com/c/mnist-tutorial-machine-learning-challenge

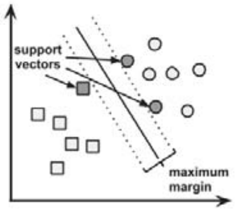
http://dogs-cats.wikia.com/wiki/File:Maine_coon_cat_2.jpg

資料標註/檢核人員

- 監督學習: labels (標籤) of the training data (訓練資料)
  - 若水國際：資料標註/檢核人員
    - 老子道德經，上善若水
    - 「善」字的拆解 (手、口、羊)
  - 有一家電商，為了讓消費者可以看到喜歡的衣服就直接拍照搜尋，必須讓機器學會辨認每一件衣服、褲子的特徵，須標註的點就非常細，有時一張照片要標出 250 到 300 個特徵點。商業周刊 1608 期

Binary Classification (二元分類)

- 資料科學在 Whoscall 產品體系中的角色 (SlideShare)
  – 圖：2 特徵 (feature) (較容易看)。
    - 找特徵需領域知識 (domain knowledge)
    - 打電話時間、對象、次數等等
    - Whoscall：實際上，找上五六十個，再縮小
  – 預測類別：正常，行銷或騷擾電話
    - Support vector machines (SVM ，支持向量機)

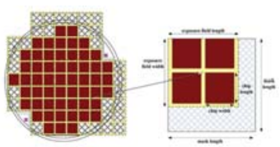


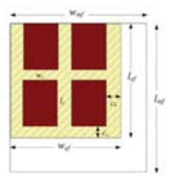
Multi-class Classification (多元分類)

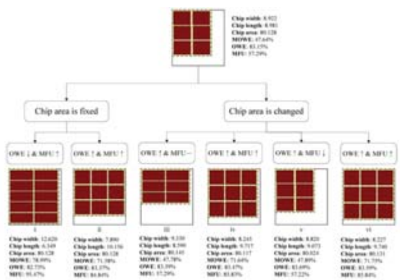
- handwritten character recognition (手寫字元辨識)
- speech recognition (語音辨識)
- image (影像)
  – 看 Obama 影片後產生新 …
- article (文件)
  – IBM Watson: DeepQA (問答) project competed on Jeopardy! (危險邊緣)
    - 10 分鐘就從 2000 萬份論文中，找出罕見白血病 (Inside, 2016)
  – 新聞：分類，自動產生 (假)
- 商品推薦 (recommendation)：喜歡某類，對某新產品的喜好度 (Netflix, Amazon, Appier)

Decision Tree (決策樹): Optimizing IC Feature Designs (特徵設計) to Enhance Overall Wafer Effectiveness (整體晶圓效能)

- Chen-Fu Chien and Chia-Yu Hsu, IEEE Tr. Semi. M., 2014
- 台積電：積體電路 (integrated circuit)
- $OWE = \frac{good\ die\ area}{total\ wafer\ area} \times 100\%$
- wafer exposure pattern (曝光模式)
- Mask-field-utilization (MFU,光罩場利用率)



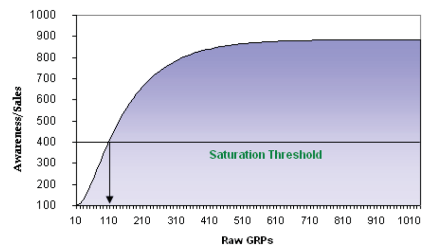




- $MOWE = OWE \times MFU$

1.2.2 Supervised Learning (監督學習): Regression (迴歸)

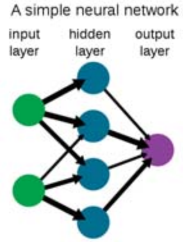
- Numeric prediction (數值預測)
  – x： Gross Ratings Point (總評級點) (或行銷費用、氣溫、 定價)、大訂單
    - 找特徵需領域知識 (domain knowledge)
  – y：銷售額、股價
  – Linear $y = 70 + 3\,x = f(x)$
  – Nonlinear (非線性)
  – 預測新資料 x 的 y 值

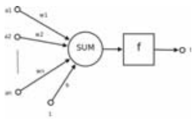


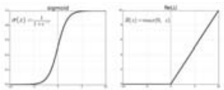
https://en.wikipedia.org/wiki/Advertising_adstock

Artificial Neural Networks (人工神經網路)

- Ref: Wiki, Bishop (輸入、隱藏、輸出層)
- 輸入: a_1 price, a_2 advertisement (廣告), …
- activation (激勵) function
- Many observations (觀察): Sales (銷售) s
- Training (訓練) or learning (學習)





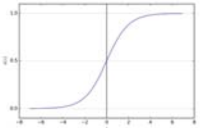


$$\min_{w_i,b} \sum_{i=1}^{n} \left(s(i) - \frac{1}{1 + e^{-a_1(i)w_1 - \cdots - a_n(i)w_n - b}} \right)^2$$

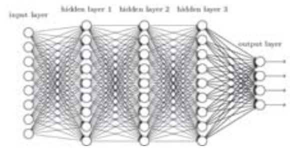
https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

Supervised Learning (監督學習): Regression (迴歸)

- Numeric prediction (預測)
  - Logistic Regression (邏輯斯迴歸): 0 ≤ 機率 ≤ 1
    - Customer churn (顧客變動)，借款拖欠風險，疾病檢測等等
    - 機率 > 0.5 ⇒ 類別 1 (分類)



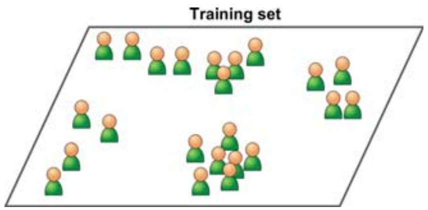
- Dual (雙重的) use: Neural Networks (神經網路), SVM (支持向量機)
  - Deep learning (深度學習)：隱藏層 ≥ 2

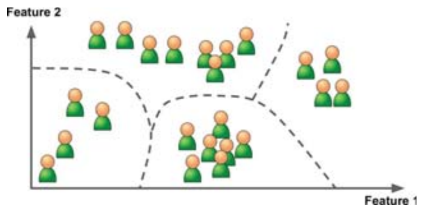


Nielsen

1.2.3 Unsupervised Learning (非監督學習)

- the training data (訓練資料) is unlabeled (未標誌的)
- The system tries to learn without a teacher.
- Example: Cluster analysis (集群分析) group the objects (i.e., the cases) into a smaller number of groups (also called *clusters*)
- 2 features (特徵) (較容易看)





Clustering (集群)

- 林洧楨，零售業營收年增率不到二%，它創五年新高 算出七種人氣店型　解密屈臣氏大數據戰法，商業周刊第 1562 期，2017/10
  - 七種店型：高價住宅、價值取向型住宅、目的消費型、遊客型、商業區型、車站商場型、學校型
  - 店面外觀沒變，但內部陳列卻是「看門道」的所在
  - 陸客 … 台製面膜等產品成立「台灣冠軍伴手禮」專區
  - 透過消費數據分析，可以分辨出受歡迎與不受歡迎產品，一週內先完成貨量調整與店型布局，降低庫存風險，然後在一個月內完成包含進撤櫃等所有細節調整

Unsupervised Learning (非監督學習)

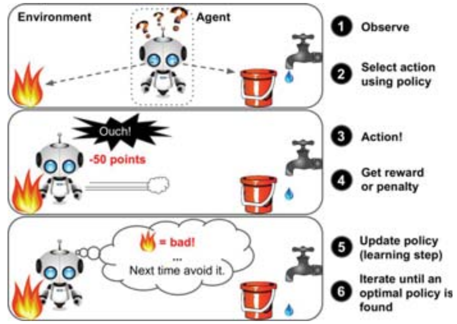
- Dimensionality reduction (降維)
  - 想可能的特徵
  - 再縮小：找出重要的。計算速度、較易理解等等
- Pattern detection (樣式辨認):
  - Association (關聯) Rules
    - 98% 的人買 AB 也買 C
    - 啤酒尿布是都市傳說

(http://chhsu135.blogspot.com/2016/01/blog-post.html)

https://www.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/42541/versions/3/screenshot.jpg

1.2.4 Reinforcement Learning (強化學習)

- The learning system, called an *agent* (代理程式), can observe the environment (觀察環境), select and perform actions (選擇並執行動作), and get *rewards* (報酬) in return (or *penalties* (懲罰) in the form of negative rewards).
  - 最大化目標



強化學習應用

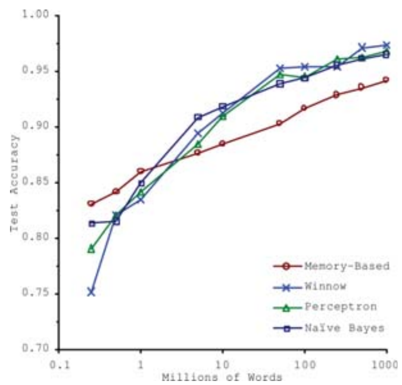
- Create auto-piloting (自動飛行) planes and auto-driving cars
- Chess engine: wiki
  - Environment (環境): State of the board
  - Reward (報酬): Win or lose at the end
  - Action (動作): Next move
- Google AlphaGo: RL + Deep learning (深度學習) + Monte Carlo Tree Search (蒙地卡羅樹搜尋)
- Bonsai & Siemens: Autotuning (自動調整) CNC 30x Faster, 2018

http://www.marketinginautomotive.com/wp-content/uploads/2013/06/Volvo-self-driving-car.jpg

https://cdn3.vox-cdn.com/uploads/chorus_asset/file/8109655/Pop.Up_copyright_Italdesign_2.jpg

1.3 Main Challenges (挑戰) of Machine Learning (機器學習)

- Insufficient Quantity (數量不足) of Training Data (訓練資料): Algorithms matter! (演算法很重要)
  - The Unreasonable Effectiveness (不合理的有效性) of Data
    - 橫軸：單位百萬個字
    - 縱軸：Test *accuracy* (準確度)

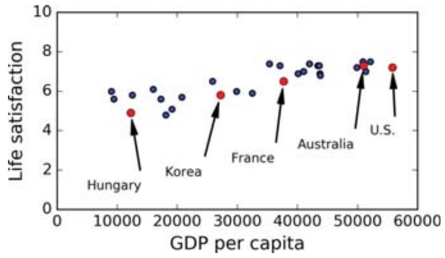


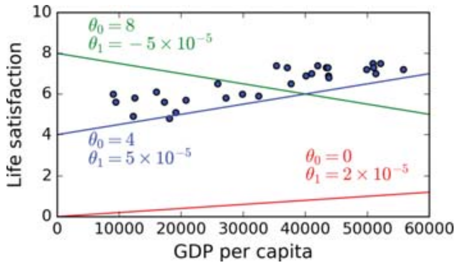
Example: Does money make people happier (幸福)?

- linear model (線性模型)

life_satisfaction (滿意) $= \theta_0 + \theta_1 \times$ GDP_ per_capita (人均國內生產總值)

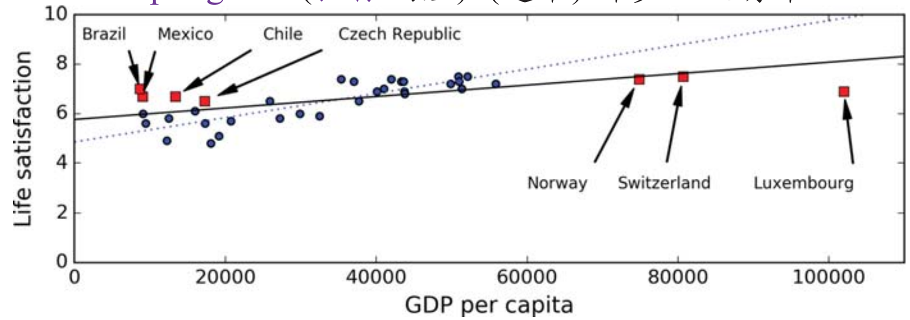
  - Greek letter (希臘字) θ theta ：歐洲文明的起源
  - 誤差平方最小：藍色





Nonrepresentative Training Data (不具有代表性的訓練資料)

- old model is represented by the dotted line (虛線)
- add the missing countries in red: get the solid line
  - 巴西、墨西哥：其他特徵
- Sampling bias (抽樣偏差): (選舉) 許多人沒有市話



Poor-quality data (質量差的數據)

- Outliers (異常值): 跑步時間負的，身高 260 cm (?)
  - Discard (丟棄) them or try to fix the errors manually (手動地)
- Missing (失踪) a few features (特徵): 5% of your customers did not specify (指定) their age
  - Ignore (忽視) this attribute (屬性), ignore these instances (例子), fill in (填寫) the missing values (e.g., with the median (中位數) age)

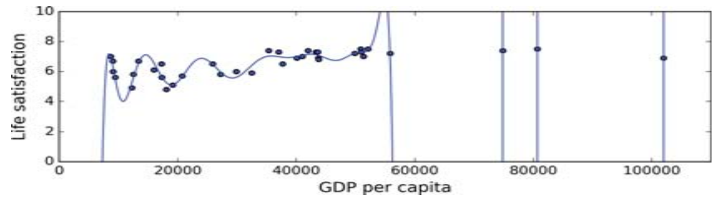
http://i.epochtimes.com/assets/uploads/2017/08/1708241220112378-600x400.jpg

Irrelevant (無關的) Features

- A critical (關鍵的) part of the success: Come up with a good set of features (特徵) to train on
- Feature engineering (特徵工程)
  - Feature selection (選擇)
  - Feature extraction (特徵萃取): dimensionality reduction algorithms (降維演算法) help
  - Creating new features by gathering (蒐集) new data

Overfitting (過度配適) the Training Data (訓練資料)

- Overgeneralizing (過度推論): Say you are visiting a foreign country and the taxi driver rips you off (敲竹槓). You might be tempted to (使很想要 (說或做)) say that *all* taxi drivers in that country are thieves.
- Overfitting: n 筆資料，高階多項式接近 n。
  - 訓練誤差小
  - 推論 (generalization)：5 萬 6 和 6 萬比，負的幸福



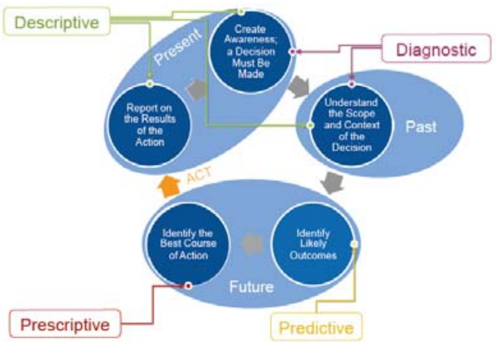
Underfitting (低度擬合) the Training Data (訓練資料)

- Your model is too simple (太簡單) to learn the underlying structure (在下面的結構) of the data
- The main options (主要選擇) to fix this problem are
  - Selecting a more powerful model (強大的模型), with more parameters (參數)
  - Feeding better features to the learning algorithm (學習演算法)
- 酒的價格 ≈ –0.4504 ＋ 0.6014 成長期平均溫度 – 0.003958 收成時雨量 ＋ 0.001043 冬季雨量
  - Orley Ashenfelter, 1990

https://i.ytimg.com/vi/05Jf8V3Q7i4/maxresdefault.jpg

1.4 Apply Relevant Data and Analytics to Decision Making (決策)

- Lisa Kart, Big Data Industry Insights 2015, Gartner (顧能)
- 描述 (顧客等待)，診斷，預測 (醫護、病痛等級)，指定 (服務時段與人數)
- M.Y. Sir, et al., Optimization of Multidisciplinary Staffing Improves Patient Experiences at the Mayo Clinic, *Interfaces*, 2017.



www.slideshare.net/denisreimer/big-data-industry-insights-2015

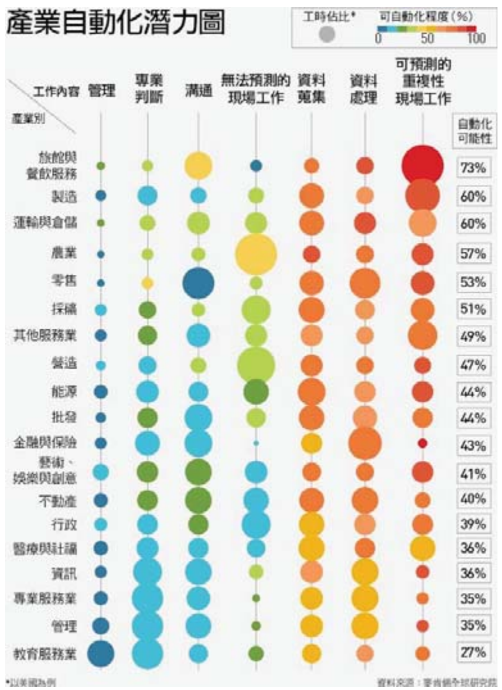
1.4.1 Artificial Intelligence (AI) for the Real World (現實世界的人工智慧)

- T.H. Davenport and R. Ronanki, Harvard Business Review (哈佛商業評論), 2018
  - Study of 152 projects (計畫)
  - Davenport and Patil: Data Scientist: The Sexiest Job (最性感的工作) of the 21st Century, HBR, 2012
- business need 1: automating business processes (自動化企業流程) (71/152)
  - updating customer files (更新客戶文件) with address changes or service additions (地址變更或服務添加)
  - “reading” legal and contractual documents (法律和合約文件) to extract provisions (提取條款) using natural language processing (自然語言處理)

29

McKinsey (麥肯錫)

- Harnessing automation (利用自動化) for a future that works, McKinsey Global Institute (全球研究所), January 2017. (當年最熱門的 MGI 文章)
- 天下雜誌：迎向人機合作的時代
- 教育服務業：流程 (持續) 改進
  - 顧客：雇主



30

例子

- 高盛利用自動交易程式，把原本在紐約總部的現股票交易櫃檯 600 名交易員變成只需要 2 人 (中時，2017)
- 儒鴻和資策會：訂單匯入自動化系統 原本一張訂單的處理時間為 48 小時，如今只要 2 小時就可以
- 自2011年以來，前10大銀行中已經裁掉 1 萬個以上的前台交易工作 (中時，2016)
  - 瑞士銀行資產交易所：2011, 2016
- 摩根大通 AI 軟體可替代律師年省 36 萬小時，科技新報，2017
- 4 小時審查 5 項保密協議，並確定 30 個法律問題，雷鋒網，2018
  - 人類律師平均準確率達 85%，而 AI 的準確率達 95%。AI 也在 26 秒內完成任務，人類律師平均需要 92 分鐘
- Google 自動廣告服務

31

business need 2: gaining insight (洞察力) through data analysis (57/152)

- predict what a particular customer (特定客戶) is likely to buy
- analyze warranty (保修) data to identify safety or quality problems in automobiles (汽車) and other manufactured products (製成品)
  - GE (通用電氣) 幫航空公司管理引擎，工業互聯網
- automate personalized targeting (個人化瞄準) of digital ads (數位廣告)
  - 平均一支行動廣告大概只有 0.5 秒的曝光時間 … 廣告投放延遲 1 秒鐘，威朋將損失 300 元，iThome，2014
- provide insurers with more-accurate and detailed actuarial modeling (精算建模)

32

business need 3: engaging with (與...接觸) customers and employees (24/152)

- intelligent agents that offer 24/7 customer service: password requests (密碼請求) to technical support (技術支援) questions—all in the customer’s natural language
- internal sites for answering employee questions on topics including IT, employee benefits, and HR policy;
- health treatment recommendation (治療建議) systems that help providers create customized care plans (客製化護理計劃) that take into account (考慮到) individual patients’ health status and previous treatments. (安智生醫 (Amwise) 的癌症精準醫療)

Examples

- 天下，637 期，「華生是我最好的總醫師」，台北醫學大學台北癌症中心執行副院長邱仲峯定位它的角色，「無形中， 把我們的時間、精力、診療水平拉升。 」
- 羅耀宗，掌握人工智慧應用的兩大「錢途」，哈佛商業評論，數位版文章，2018/8/24 (Michael Chui, Nicolaus Henke, and Mehdi Miremadi from McKinsey)
  – 將促銷活動個人化，單是實體零售商店的新增銷售（incremental sales）就會增加 1% 到 2 %
  – 在先進製造方面，營運活動往往產生最多的價值。這方面，人工智慧能夠協助根據需求背後的因果驅動因素，來做出預測，而不是根之前的結果來預測，因而改善預測準確度達 10% 到 20% 。這可能會使得存貨成本降低 5%，營收提高 2% 到 3%

four-step framework (四步架構) for integrating AI technologies (整合技術)

1) Understanding The Technologies: A main success factor (成功因素) is your people's willingness to learn.
   – 天下，637 期，台北醫學大學指派 33 歲醫生寫程式整合 Watson 和醫院資訊系統
2) Creating a Portfolio of Projects (項目組合)
   – in health care, knowledge tends to be siloed (孤立) within practices, departments, or academic medical centers.
   – 政府資料：連副閣揆張善政也要不到，聯合報，2015
3) Launching Pilots (啟動試驗)
4) Scaling Up (擴大): Requires collaboration (合作) between technology experts and owners of the business process being automated

1.4.2 Ethical Considerations (倫理的注意事項)

- Target
  – 搜集並研究其消費者的行為，在適當的時機寄折價券 (coupon)，以吸引顧客上門
  – 最戲劇化的故事 ：比家長更早發現其家中懷孕的高中女兒
  – 如何顧及顧客的隱私是一個重要的課題： 將顧客需要的折價券混合其他隨機的折價券
- 保險公司：生活習慣，某些疾病高風險群，拒保
- 個人資料保護法

書

- Cathy O'Neil, Weapons of Math Destruction (大數據的傲慢與偏見), 2016
  – 大學排名：員工捏造了幾乎每一方面的數據，包括SAT分數、錄取率、畢業率、新生續讀率、師生比率，以及校友捐款數據。拜假數據所賜，某大學的排名從第 50 位升至第 30 位
  – 研究者捏造的履歷表特別考慮種族因素 ... 研究者發現，白人姓名履歷表獲得雇主回應的次數比黑人姓名者多 50%
- Michael J. Sandel, Justice: What's the Right Thing to Do? (正義：一場思辨之旅), 2010

Lecture 2 Python

- 2.1 Python
- 2.2 開發環境
- 2.3 Data (資料)
- 2.4 Feature Engineering (特徵工程)
- 上課內容 http://chhsu135.blogspot.com/
  - 標籤上課研究服務
- 長榮大學資設院資管系許志華

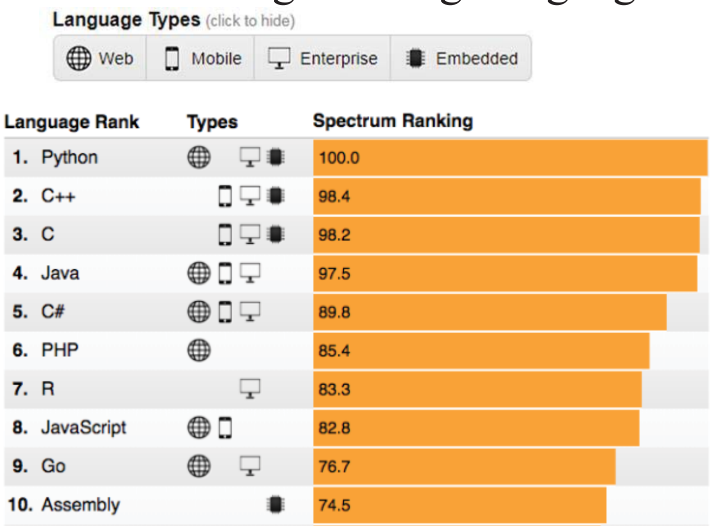
2.1 Python

- Guido van Rossum, 1991
- Wiki 英國發音 /ˈpaɪθən/ ，美國發音 /ˈpaɪθɑːn/
- Why Python?
  - Modern scripting language (命令稿語言): Object-oriented (物件導向)
  - Open source (開源)
  - Free
  - Popular: Next
  - Machine learning (機器學習): Scikit-Learn, TensorFlow
  - Web technology (網頁技術): Django, Flask
  - IoT (物聯網): (IBM) Node-RED

https://images2015.cnblogs.com/blog/1005077/201609/1005077-20160910235000035-2760349.png

https://images2015.cnblogs.com/blog/1033021/201610/1033021-20161018104511873-131390826.png

IEEE Spectrum: The 2018 Top Programming Languages



- 開發時間和計算時間的取捨
- 常使用的核心或底層部分使用c 或 c++
- TensorFlow implemented in C++

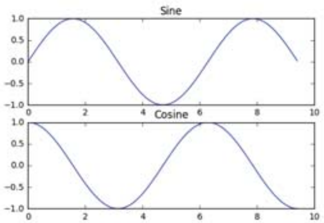
Version (版本)

- 2.6 以前
  `print ‘hello world’`
- 2.6 以後和第 3 版
  `print(‘hello world’)`
- What’s New In Python 3.0 (https://docs.python.org/3.0/whatsnew/3.0.html)
  - (按滑鼠右鍵) 選中譯
- https://pypi.python.org/pypi: 148,074 projects (2018/8/6)，157,457 projects (2018/11/6)。Sudoku (數獨)
- List of Python software (https://en.wikipedia.org/wiki/List_of_Python_software)

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

2.2 開發環境

- Anaconda: 附 Jupyter Notebook 和 Spider
- Integrated development environment (IDE, 整合開發環境)
- > 100 Python libraries (程式庫) and packages (套件): Ipython, NumPy, pandas, and Matplotlib, Scikit-Learn, and more
- version-consistent (版本一致): work with each other



安裝 Anaconda (1)

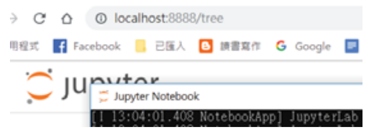
- 到 https://repo.continuum.io/archive/
- 到系統控制台 / 系統下，檢查您的系統 (或執行 dxdiag)
  - 如果是64 位元作業系統，下載 Anaconda3-5.2.0-Windows-x86_64.exe 後安裝。
  - 32 位元下載 Anaconda3-5.2.0-Windows-x86.exe 後安裝
  - 3-5.2.0 代表版本，可能改變
- 到控制台 / 系統 / 關於 / 系統資訊 / 進階系統設定 / 環境變數下，按 path 後，新增 Anaconda3和 Anaconda3\Scripts 所在的資料夾 (我的在 C:\Users\chhsu\Anaconda3)，然後按確定
  - 作業系統才知道到此處找 Anaconda，才可以使用之

安裝 Anaconda (2)

- https://conda.io/docs/user-guide/tasks/manage-pkgs.html
  - (安裝好之後) 清單：(cmd 下) conda list
  - Installing packages: conda install scipy=0.18.1
- (cmd 下) conda --version
- 在 cmd 下看 python 的版本 C:>python --v
- 虛擬環境及套件管理
  - http://yenlung-blog.logdown.com/posts/257347-anaconda-in-the-virtual-environment-and-package-management

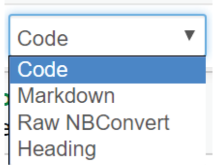
2.2.1 Jupyter Notebook (筆記本)

- 打開 (開始/程式) Anaconda3 內的 Jupyter Notebook
  - 前景：會出現瀏覽器，預設瀏覽器請選 google chrome
  - 背景：cmd
- The Jupyter Notebook: Create and share documents (創建和共享文檔) that contain live code, (LaTex, θ, θ) equations, visualizations (視覺化) and explanatory (解釋的) text.
- Kernel: IPython Notebook (ipynb 檔)
  - 使用 Notepad 打開：JSON



練習

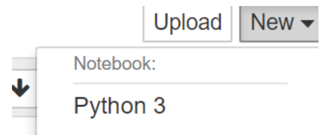
- 練習別人的檔案
  - Upload (上傳) lec02-1 Introduction to python.ipynb 到 workspace
  - 點選檔案



- 選單
  - Code
  - Markdown: Markdown is a popular markup language (標記式語言) that is a superset (超集合) of HTML.
- 再按 

(執行) : cell / run cell (欄位), select below

作業



- 修改部份上課內容指令，以觀察其結果的變化
- 寫新檔：右方 New 選單，選 Python 3
- File: Rename, download
  - File/ Download as/Notebook (.ipynb)

- 定義變數，下次需重新執行才能使用之
  - Cell /Run All, Run All Below
- 方法：聽過影片，跑過所有的指令。複製貼上需要的部份，然後修改相對應的部份
- + (insert cell below (在下方插入欄位))，也可以上下移動之

Change default directory (預設資料夾) of jupyter notebook

- 預設：Desktop (桌面)
- 在 cmd, 輸入:

`jupyter notebook --generate-config`

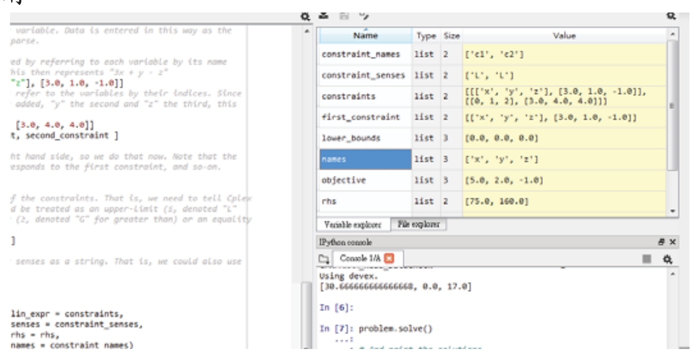
  - 在 Users/USERNAME/.jupyter, 產生 jupyter_notebook_config.py (configuration 配置)
- 找到下行並改成

`c.NotebookApp.notebook_dir =‘D:\\python‘`

- Windows 10: Jupyter notebook / 右鍵內容/捷徑
  - https://stackoverflow.com/questions/35254852/how-to-change-the-jupyter-start-up-folder
  - delete “%USERPROFILE%“，Start in `D:\\ python`

2.2.2 Spyder

- Integrated development environment (IDE, 整合開發環境)
- 在 Jupyter Notebook 將 ipynb 檔轉成 py 檔，再用 Spyder 打開
- 變數資訊
- 方便除錯
- debugger



2.2.3 TensorFlow and Keras

- Wiki: Comparison of deep learning software (深度學習軟體比較)
- TensorFlow by Google, Theano by University of Montreal, CNTK (Cognitive Toolkit) by Microsoft, PyTorch
- CPU (central processing unit ，中央處理器)
- GPU (graphics processing unit，圖形處理器)
  - Nvidia: GPU 擁有數千個核心，能有效處理平行運算
- Why gpu faster
  - https://medium.com/@andriylazorenko/tensorflow-performance-test-cpu-vs-gpu-79fcd39170c

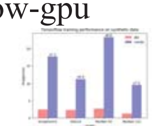
TensorFlow and Keras for CPU-only

- https://www.tensorflow.org/install/install_windows#installing_with_anaconda
  - 到Cmd，下指令 C:> pip3 install --upgrade tensorflow
- https://keras-cn.readthedocs.io/en/latest/for_beginners/keras_windows/
  - 到Cmd，下指令 C:> pip3 install keras -U --pre
- Google Colab: runs entirely in the cloud

Win10 安裝 TensorFlow-gpu & Keras

- 參考
  - https://medium.com/@WhoYoung99/2018%E6%9C%80%E6%96%B0win10%E5%AE%89%E8%A3%9Dtensorflow-gpu-keras-8b3f8652509a
  - https://www.quantinsti.com/blog/install-tensorflow-gpu
  - NVIDIA display driver: 搜尋 dxdiag

- Pip vs conda
  - https://www.anaconda.com/blog/developer-blog/tensorflow-in-anaconda/
  - (under cmd) conda install -c anaconda tensorflow-gpu



2.3 Data (資料)

- 林俊宏譯，大數據，天下文化，2013 ，第 112 頁
  - Data (資料): (拉丁文的意思) 既定的，講的是一件事實
  - 今日，資料指的是能夠紀錄、分析、重組的事務
- Data type:
  - Quantitative (定量的) or Numeric (數值的): Income (收入) of the buyer, Price (價格) of a product (產品)
  - Categorical data (類別資料): Gender (性別) (female (女性), male), Profession (職業) of the buyer

Data Preprocessing (資料預處理)

- 趨勢科技靠大數據打敗駭客：挾帶惡意程式「viagra」（威而鋼）的垃圾郵件，就有「v!agra」「vi@gra」等九百多種形式
  - 趨勢科技靠大數據打敗駭客，哈佛商業評論，2014
  - Regular Expressions (正規表示式)
- 林俊宏譯，大數據，天下文化，2013 ，第 98 頁
  - 維修孔有 38 種寫法： service box, SB, S, SBX, S/XB, SVBX, SERV BX, SERV/BOX, ….
  - Cynthia Rudin, et al. Machine Learning for the New York City Power Grid. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 34, No 2. February 2012.

Data Quality: Consistency (一致性)

- Unit (單位)
  - Material (物質): unit, batch (批), kg, etc.
  - Time: Second, minute, hour, etc.
- Balance equation (平衡方程式)
  - 物質守恆 (conservation)
    - 輸入 = 輸出 + 廢料
    - 存貨(t+1) = 存貨(t) + 輸入(t) – 輸出(t)
  - 時間守恆：到達 + 等待 + 處理 + 隨機時間 = 完成時間
  - 生產速度 * 生產時間 = 輸出

2.4 Feature Engineering (特徵工程)

- Feature engineering: The process of creating appropriate (適當) data features by applying business context
- 趙于婷，不喝酒、沒肝硬化也會肝癌　國衛院揪出「3大風險因子」，ETtoday 健康雲，2018年05月21日
  - 脂肪肝、糖尿病史和三酸甘油脂過高
- 王姿琳，數據釀的日本國宴酒，商業周刊，第1597期，2018-06-22
  - 數據分析職位：負責測量米的重量、洗米時間、水溫以及酒精度數、糖分度數等細節，隔天再依據前一天的數據判斷發酵的進度
  - 引進農業雲端系統「Akisai」，逐一記錄農田變化與施肥時間，果然成功提高兵庫縣山田錦產量一倍

Example from Facebook

- Xinran He, et al., Practical Lessons from Predicting Clicks on Ads (預測點擊廣告) at Facebook, ADKDD'14, August 2014
- 正確的特徵 (features) 和模型 (decision trees, logistic regression) 是關鍵。混和兩者的方法比個別方法增加了 3% 的準確度。至於特徵的選取，可以參考
  - Mia，超精準廣告背後，Facebook 怎麼對你「貼標籤」　華盛頓郵報整理的 98 項 Facebook 廣告指標，Inside，2016/9/13
  - Caitlin Dewey, 98 personal data points that Facebook uses to target ads to you, Washington Post, 2016/8/19

21

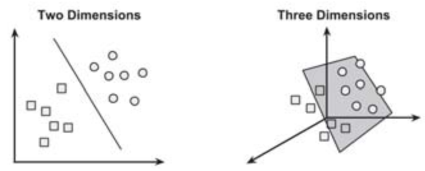
Some remarks

- Pedro Domingos, *A Few Useful Things to Know about Machine Learning*, Communications of the ACM, 2012.
  - 8. FEATURE ENGINEERING IS THE KEY: At the end of the day, some machine learning projects succeed (成功) and some fail. What makes the difference (區別)? Easily the most important factor (最重要因素) is the features used.
- Andrew Ng, *Machine Learning and AI via Brain simulations*, 2013.
  - Coming up with features is difficult, time-consuming (耗時), requires expert knowledge (專業知識). 'Applied machine learning' is basically feature engineering (特徵工程).

22

2.4.1 Whoscall

- 高義銘、郭建甫、鄭勝丰、宋政桓，資料科學在 Whoscall 產品體系中的角色，2014
- buzzorange：顯示來電者的身分，並警示該來電可能是行銷電話、騷擾電話，也能過濾掉拒接來電
  - 分類：正常，警示
  - 網路上即時搜尋，以及 500 萬用戶的回報
  - 若網路上搜不到、還沒有用戶回報，就無從判斷這通電話可能的身分、是不是惡意電話
  - 支持向量機 (SVM)



23

Call pattern (or feature)

- 正常：每天發話、接電話的頻率大概是 1 至 2 通，且通常有特定通話對象。
  - 行銷電話：每天發話的頻率在 10 通以上、發話相隔時間短、對象都不相同，且僅限於周一到周五電話行銷專員有上班的日子才有發話紀錄。
- 正常：每通電話平均的通話時數約在 1 分 12 秒
  - 詐騙電話 (Fraud Number) 的平均通話時數 30 秒不到，行銷電話 (Marketing Numbers) 的平均通話時數 36 秒不到，顯然是被接起之後立刻就被掛斷。

24

Machine learning (機器學習)

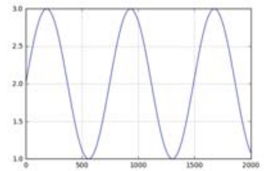
- Whoscall (86 - 91 頁) 開始在做資料分析時，萃取了五六十個特徵，但最後發現只要十幾二十個就可以判斷。
- 找出 Call Pattern 之後，whoscall 在一通電話之間判斷其是否為惡意電話的準確度高達 93%，在兩通電話後，判斷的準確率則提升至 96%

Lecture 3 Regression (迴歸)

- 3.1 Simple linear regression (簡單線性迴歸)
- 3.2 Multivariate Regression (多變量迴歸)
- 3.3 predicting medical expenses (預測醫藥費用)
- 3.4 Get the Data
- Bertsimas, et al., The Analytics Edge
- Bonaccorso, Machine Learning Algorithms
- Geron, Hands-On Machine Learning With Scikit-Learn and Tensorflow
- 長榮大學資設院資管系許志華

3.1 Simple linear regression (簡單線性迴歸)

- Bordeaux Wine (法國波爾多葡萄酒)
- Correlation (相關): price and AGST 0.659563 (最高)
  - File (檔案): wine-data.csv
- 酒的價格 (price) = –3.41776 + 0.63509 成長期平均溫度 + 誤差
  - AGST: Average Growing Season Temperature
- 非線性：水位 = 2 + sin (c 分鐘)



解釋

- 預測價格 = –3.41776 + 0.63509 成長期平均溫度
- Year 1952: AGST 17.1167 (度 C), real price (實際價格) 7.495
  - predicted (預測) price = – 3.41776 + 0.63509 (17.1167) ≈ 7.45296
  - Square error (方差): $(7.45296 – 7.495)^2 \approx 0.00177$
  - Supervised learning (監督式學習): real price 已知
- 平均溫度↑(↓) 1 度 C ⇒ 價格↑(↓) 0.63509
- 如果均溫↑(↓) 30 度⇒ 價格↑(↓) 0.63509×30？

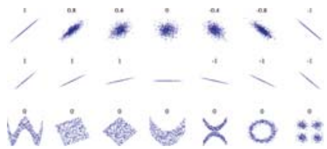
3.1.1 Correlation and Causation (相關和因果)

- 林俊宏譯，大數據 (Big Data)，第 4章相關性 (正是如此) 不再拘泥於因果 (causality) 關係 (為何如此)
- More firemen's presence (消防員出現) during a fire instance signifies (表示) that the fire is big but the fire is not caused (造成) by firemen.
- global temperature (全球溫度), Pirate (海盜)

https://www.forbes.com/sites/erikaandersen/2012/03/23/true-fact-the-lack-of-pirates-is-causing-global-warming/#5fe501e43a67

Correlations (相關) (正是如此)

- **Correlation** between two variables: Indicates how closely their relationship follows a straight line (直線)
- price and AGST 0.659563: Fairly strong positive association (正關聯), temperature ↑ $\Rightarrow$ price ↑
- 沃爾瑪 (Wal-Mart) 和天睿 (Teradata) 合作：颱風來襲前，手電筒和 Pop-Tarts 熱銷
- The Signal and the Noise (精準預測): ch 6 冰淇淋和森林火災 (相關，都發生在暑期，沒有因果)



https://www.amazon.com/Pack-Ultimate-Tarts-Variety-Flavors/dp/B017N01NXM

Causation (因果) (為何如此) (1/3)

- 想知道因果關係，必須做實驗
  - 1885 年 Louis Pasteur 發明狂犬病疫苗，『拯救』了 Joseph Meister：遭到患有狂犬病的狗咬傷，只有 1/7 的人會染病
    - 因果關係：(1) 被狗咬傷是否會染病 (2) 疫苗是否能對抗狂犬病毒
  - 控制組和對照組：仔細控制可能的原因，以證明事實確實如此。藥物實驗
  - Randomized controlled experiments (隨機對照實驗)

Causation (因果) (為何如此) (2/3)

- 尹俞歡，訂房網 Booking.com 母集團，股價破兩千美元秘密 荷蘭直擊　亞馬遜後最成功網路公司，商業周刊，1600 期，2018.07.11
  - 這裡 …，有一千八百名工程師在此工作…對網站進行永無止境的改版測試。大家一天至少會做一千次以上的 A/B 測試。
  - A/B 測試：針對某項網站上的功能，提供兩種版本給消費者使用，若數據顯示，某個選項最能留下消費者、或是減少後續客服次數，就會被正式採用
    - 刷卡時，卡片種類選單該用文字或圖像呈現？結果證明，以圖像呈現信用卡，消費者不易選錯，成單率也較高
  - 15% 抽成率，98%的毛利率，三十億美元買關鍵字廣告
  - https://conversionsciences.com/blog/correlation-causation-impact-ab-testing/ (選中譯)

Causation (因果) (為何如此) (3/3)

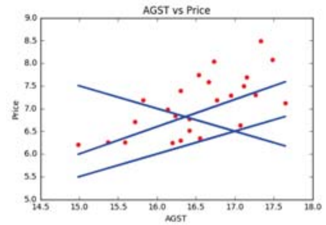
- 紐約市
  - 每年因地下管道失火，紐約市有幾百個檢修人員出入孔悶燒或爆飛。
  - 預測人孔事故最重要指標：電纜的年份，過去是否曾發生事故
  - Cynthia Rudin, et al. Machine Learning for the New York City Power Grid (電網). IEEE Transactions on Pattern Analysis (圖形分析) and Machine Intelligence (機器智慧), Vol. 34, No 2. February 2012. (Columbia University and Consolidated Edison Company)

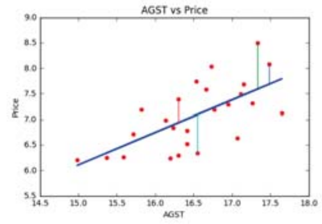
More

- E. Almquist and G. Wyner, Boost (促進) Your Marketing (行銷) ROI with Experimental Design (實驗設計), Harvard Business Review (哈佛商業評論), 2001
  – ROI (Return on Investment，投資報酬率)
- T.H. Davenport, How to Design Smart Business Experiments (商業實驗), Harvard Business Review, 2009
- E.T. Anderson and D. Simester, A Step-By-Step (逐步) Guide to Smart Business Experiments, Harvard Business Review, 2011.

3.1.2 Objective (目標)

- 預測價格 (price) $\hat{y}_i = \theta_0 + \theta_1$ 成長期平均溫度
  – Greek θ theta: Unknowns (未知)
- Minimize (最小化) the mean squared error (均方差，MSE): $\frac{1}{m}\sum(y_i - \hat{y}_i)^2$ (m 筆資料)
  – Predicted (預測) y value y-hat, actual y value y_i
- Optimal solution (最佳解) $\theta_0 \approx -3.41776$, $\theta_1 \approx 0.63509$





R-Squared for Goodness of Fit (適合度)

- actual y_i,預測 $\hat{y}_i$, (所有的) mean $\bar{y}$
- SST (sum of square total, 總離均差平方和) = $\sum_i (y_i - \bar{y})^2$
- SSR (sum of square residual, 殘差平方和) = $\sum_i (\hat{y}_i - \bar{y})^2$
- R-Squared = SSR / SST ≈ 0.43502 (Wine price by AGST)
  – total proportion (總比例) of variance (變異數) in the dependent variable (相依變數) explained by the independent variable (獨立變項).
  – It is a value between 0 and 1
  – 越大越好

3.2 Multivariate Regression (多變量迴歸)

- Bordeaux Wine (法國波爾多葡萄酒)
- March 1990, Orley Ashenfelter, a Princeton economics professor (經濟學教授), claims he can predict wine quality (品質) without tasting (品嚐) the wine
  酒的價格 ≈ –0.4504 + 0.6014 成長期平均溫度 – 0.003958 收成時雨量 + 0.001043 冬季雨量
  – 特徵 (feature): Domain knowledge (領域知識)
  – 下雨搶收
- 世界知名的品酒專家 Robert Parker 說：『Ashenfelter is an absolute total sham (騙子).』
- 在拍賣市場中，結果驗證了 Ashenfelter 是對的

Regression (迴歸)

- Linear (線性) $\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$
  酒的價格 ≈ –0.4504 + 0.6014 成長期平均溫度
  – 0.003958 收成時雨量 + 0.001043 冬季雨量
- Model parameters (模型參數): the bias term (偏項) θ_0, the feature weights (特徵權重) $\theta_1, \theta_2, \ldots, \theta_n$ in the hypothesis function (假設函數) h

$$\theta^T \equiv [\theta_0 \quad \theta_1 \quad \ldots \quad \theta_n], x \equiv \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}, \hat{y} = h_\theta(x) = \theta^T x$$

- Nonlinear: $x_1 x_2, 1/(1 + {x_2}^2), \sin x_3, e^{x_4}, \log x_5$

輸入與輸出變數的類別與範圍

- 酒的價格 ≈ –0.4504 + 0.6014 成長期平均溫度
  – 0.003958 收成時雨量 + 0.001043 冬季雨量
- `Python FileName.describe()`
  - 酒的價格 ∈ [6.2049, 8.4937]
  - 成長期平均溫度 ∈ [14.9833, 17.6500]
  - 收成時雨量 ∈ [38, 292]
  - 冬季雨量 ∈ [376, 830]

解釋

- 酒的價格 ≈ –0.4504 + 0.6014 成長期平均溫度 (AGST) (17.1167) – 0.003958 收成時雨量 (160) + 0.001043 冬季雨量 (600) ≈ 9.836

Year	Price	AGST	HarvestRain	WinterRain
1952	7.495	17.1167	160	600

  - Real price (真實的價格): 7.495
- 均溫和冬雨固定不變，收成時雨量 160 mm，↑(↓) 100 mm (變成 260 (60) mm)，酒的價格↓(↑) 0.3958
- 收成雨固定不變，AGST↑(↓) 1 度 C，冬季雨量↑(↓) 100 mm ⇒ 價格↑(↓) 0.6014 + 0.1043 = 0.7057

3.3 predicting medical expenses (預測醫藥費用)

- In order for (為了) an insurance company (保險公司) to make money, it needs to collect more in yearly premiums (保險費) than it spends on medical care (醫療保健) to its beneficiaries (受益人).
- As a result, insurers invest a great deal of time and money to develop models that accurately forecast (準確預測) medical expenses.
- Ref: Brett Lantz, Machine Learning with R, Packt Publishing, 2013.

成因

- Medical expenses (醫藥費用) are difficult to estimate (很難估計) because the most costly conditions (昂貴的條件) are rare (罕見) and seemingly random (看似隨機).
  - lung cancer (肺癌) is more likely among smokers (吸煙者) than non-smokers
  - heart disease (心臟病) may be more likely among the obese (肥胖)

Business objective (目標)

- Goal: Use patient data (患者數據) to estimate (估計) the average medical care expenses (平均醫療費用) for such population segments (人口區段)
- Business objective (目標): These estimates could be used to create actuarial tables (保險精算表) which set the price of yearly premiums (年保費) higher or lower depending on (根據) the expected treatment (治療) costs.

Collecting data (收集資料)

- These data were created for this book using demographic (人口統計學的) statistics from the U.S. Census Bureau (美國人口普查局)
  - a simulated dataset (模擬數據集)
- The insurance.csv file includes 1338 examples of beneficiaries (受益人) currently enrolled (參加) in the insurance (保險) plan
  - Features (特徵) *x* of the patient
  - Charges (收費) *y*: the total medical expenses (總醫療費用) charged to the plan for the calendar year (公曆年)

Frame the Problem (框問題): System design (系統設計)

- supervised, unsupervised, or reinforcement learning?
- Each row: instance (例子)
  - 第 0 筆 (列), instance (例子): 19, female (女), …
  - a typical *multivariate* (多變量) regression (迴歸) task: you are asked to predict (預測) a value (charges)
  - output label (標示): charges (收費)
  - features (特徵) to make a prediction of charges (收費)
  - NaN: not a number

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.9	0	yes	southwest	16884.9240
1	18	NaN	33.77	1	no	southeast	1725.5523

Features (特徵) *x*

- age of the primary beneficiary (主要受益人)
- sex: policy holder's gender, either male or female.
- bmi: body mass index (身體質量指數) = weight (in kilograms) / height (in meters) squared
- children: An integer, number of children / dependents (受撫養者) covered by the insurance plan
- smoker: yes or no, the insured regularly smokes tobacco.
- region: Beneficiary's (受益人) place of residence (住所) in the U.S., northeast (東北), southeast, southwest (西南), or northwest.

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.9	0	yes	southwest	16884.9240

3.4 Get the Data (資料取得)

- lec03-1 insurance.ipynb
- data/lec03-insurance.csv
  - Directory (目錄) data 下 lec03-insurance.csv 檔
  - csv: Comma-Separated Values (逗點分隔值)
  - Use Notepad++ to open it

lec03-insurance.csv

```
1 age,sex,bmi,children,smoker,region,charges
2 19,female,27.9,0,yes,southwest,16884.924
3 18,male,33.77,1,no,southeast,1725.5523
```


3.4.1 Pandas

- Developed (開發) by Wes McKinney in 2008 while at AQR Capital Management (資本管理) for financial (財務) data
  - MIT with an B.S. in Mathematics (數學) in 2007
  - convinced (說服) management to open source (開源) the library when left the company
- Data Structures (資料結構)
  - One-dimensional (維): Series
  - Two-dimensional: DataFrame (資料格式)

https://twitter.com/wesmckinn

DataFrame (資料格式)

- `import pandas as pd # 輸入`
- `insurance = pd.read_csv("data/lec03-insurance.csv")`
  `# 資料夾 data 下 lec03-insurance.csv`
- `insurance.head(2)# head(): default (預設) 5 個`

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.9	0	yes	southwest	16884.9240
1	18	NaN	33.77	1	no	southeast	1725.5523

- one row (列) per beneficiary (受益人): instance (例子)
  - 6 attributes (屬性)
  - 1 target (目標)

insurance.info()
information (資訊)

- 1338 筆資料
- float64 (64 位元浮點數)
- integer (整數)
- object (物件)
- Missing data (遺漏資料)
  - 1337 of sex, bmi
- insurance 隨問題變化，常見錯誤
- info 是 Python 函數

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
age         1338 non-null int64
sex         1337 non-null object
bmi         1337 non-null float64
children    1338 non-null int64
smoker      1338 non-null object
region      1338 non-null object
charges     1338 non-null float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.2+ KB
```

- insurance.shape
- (1338, 7)

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.9	0	yes	southwest	16884.9240
1	18	NaN	33.77	1	no	southeast	1725.5523

Access the elements (存取元素) (1)

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.9	0	yes	southwest	16884.9240
1	18	NaN	33.77	1	no	southeast	1725.5523

- Column (行): insurance.age, insurance['age'], insurance.iloc[:,0]
  - insurance.iloc[:,0]:方便自動化
  - Python 從 0 開始
  - iloc: integer-location based indexing (基於整數位置的索引)
  - insurance 和 age 隨問題變化，常見錯誤

```
0    19
1    18
2    28
3    33
```


Access the elements (存取元素) (2)

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.9	0	yes	southwest	16884.9240
1	18	NaN	33.77	1	no	southeast	1725.5523

```
age                18
sex               NaN
bmi             33.77
children            1
smoker             no
region      southeast
charges       1725.55
Name: 1, dtype: object
```

- Row (列): insurance.iloc[1]
- Element (元素)

insurance.iloc[0, 5], insurance['region'][0]:
('southwest', 'southwest')

- 部分

insurance.iloc[0, 5:] :

```
region      southwest
charges       16884.9
Name: 0, dtype: object
```

insurance.iloc[1, 1:3] : 第 1 到 2 項，不包括第 3 項

```
sex      NaN
bmi    33.77
Name: 1, dtype: object
```


insurance.describe()(描述)(1)

- count (計數), mean (平均值), std (standard deviation，標準差), min (minimum，最小值), max (maximum，最大值)

	age	bmi	children	charges
count	1338.000000	1337.000000	1338.000000	1338.000000
mean	39.207025	30.667004	1.094918	13270.422265
std	14.049960	6.099040	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.315000	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.700000	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

- Age: Excluding those above 64 years, since they are generally covered by the government
- An ideal BMI ∈ [18.5 24.9]

`insurance.describe()`(描述)(1)

- count (計數), mean (平均值), std (standard deviation，標準差), min (minimum，最小值), max (maximum，最大值)

	age	bmi	children	charges
count	1338.000000	1337.000000	1338.000000	1338.000000
mean	39.207025	30.667004	1.094918	13270.422265
std	14.049960	6.099040	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.315000	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.700000	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

- (excluding those above 64 years, since they are generally covered by the government)
- `insurance.age.mean(), insurance.describe().iloc[1, 0]`
- `(39.20702541106129, 39.20702541106129)`
- `insurance.describe().iloc[4:7, 0]`

```
25%    27.0
50%    39.0
75%    51.0
Name: age, dtype: float64
```


3.4.2 Nominal data (名目資料)

- Categorical (類別) data
- `insurance.region.describe()`

```
count          1338
unique            4
top       southeast
freq            364
Name: region, dtype: object
```

- Descriptive statistics (描述性統計)
  - Mode (眾數): the most common value, southeast
  - Proportion (比例): `364/1338`, `insurance.region.describe()[3] / insurance.region.describe()[0]`
  - `(0.2720478325859491 7, 0.27204783258594917)`

`insurance.describe(include = 'all')`

- Google describe pandas
  https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.describe.html
- NaN not a number

	age	sex	bmi	children	smoker	region	charges
count	1338.000000	1337	1337.000000	1338.000000	1338	1338	1338.000000
unique	NaN	2	NaN	NaN	2	4	NaN
top	NaN	male	NaN	NaN	no	southeast	NaN
freq	NaN	675	NaN	NaN	1064	364	NaN
mean	39.207025	NaN	30.667004	1.094918	NaN	NaN	13270.422265
std	14.049960	NaN	6.099040	1.205493	NaN	NaN	12110.011237
min	18.000000	NaN	15.960000	0.000000	NaN	NaN	1121.873900
25%	27.000000	NaN	26.315000	0.000000	NaN	NaN	4740.287150
50%	39.000000	NaN	30.400000	1.000000	NaN	NaN	9382.033000
75%	51.000000	NaN	34.700000	2.000000	NaN	NaN	16639.912515
max	64.000000	NaN	53.130000	5.000000	NaN	NaN	63770.428010

`value_counts()`

- `c = insurance.region.astype('category').values`

```
[southwest, southeast, southeast, northwest,
northwest, ..., northwest, northeast, southeast,
southwest, northwest]
Length: 1338
Categories (4, object): [northeast, northwest, southeast,
southwest]
```

- `c.value_counts(), insurance.region.value_counts()`
  - Count (計數)
  - `insurance, region`: 隨問題變
  - `value_counts`: python

```
southeast    364
southwest    325
northwest    325
northeast    324
Name: region, dtype: int64
```

3.4.3 NumPy

- scientific computing (科學計算): array (陣列), linear algebra (線性代數), and more
- https://docs.scipy.org/doc/numpy-1.14.0/genindex.html
  - 驚人的函式庫
- `import numpy as np`
- `A = np.array([2, 2, 4, 80])`
- `np.mean(A), np.median(A) #平均、中位數`
- `(22.0, 3.0)` # (2 + 4) / 2 = 3
- {2, 2, 4, 8}: mean $\frac{2+2+4+8}{4} = 6$, median $\frac{2+4}{2}$, and mode 2
- {2, 2, 4, 80}: mean 22, median 3, and mode (眾數) 2

NumPy Functions

- `from scipy import stats #` scipy also include pandas and NumPy
- `A = np.array([2, 2, 2, 4, 80])`
- `stats.mode(A)`
- `# ModeResult(mode=array([2]), count=array([3]))`
- `stats.mode(A)[0][0], stats.mode(A)[1][0]) # 2, 3`

- `B = np.array([2, 2, 4, 8])`
- Dispersion measures (分散度測量)
  - Variance (變異數)
    $\frac{1}{4}[(2-4)^2+(2-4)^2+(4-4)^2+(8-4)^2] = 6$
  - Standard deviation (sd): sqrt(var) (square root (平方根))
- `np.var(B), np.std(B), [np.min(B), np.max(B)]`
- `(6.00, 2.45, [2, 8])`

2-dim array (1)

- Dimension (維)
- `import numpy as np`

```
b = np.array(([4, 5, 1], [6, 2, 7]))
b
array([[4, 5, 1],
       [6, 2, 7]])

 b.shape, b.shape[0], b.shape[1] # 形狀
((2, 3), 2, 3)
# 2 row (列) 3 column (行)
```


2-dim array (2)

```
array([[4, 5, 1],
       [6, 2, 7]])
```

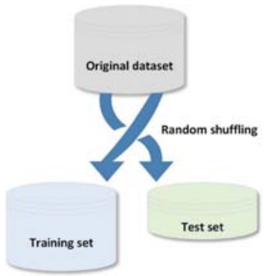
- Column (行): `b[:, 2]`, `array([1, 7])`
- Row (列): `b[1]: array([6, 2, 7])`
- 元素 `b[0, 2]:1`
- 部分 `b[:, 0:2]`

```
array([[4, 5],
       [6, 2]])
```

- `np.zeros((2,3))`
- `[[0. 0. 0.]`
  `[0. 0. 0.]]`
- `np.eye(3,3) # identity matrix (單位矩陣)`
- `[[1. 0. 0.]`
  `[0. 1. 0.]`
  `[0. 0. 1.]]`

3.4.4 Create a Test Set (測試集)

- split the data into training and test sets (訓練集和測試集)
  - 訓練集：用於訓練模型 (training the model)
  - 測試集：用於測試其性能 (test its performances)
- pick some instances (例子) randomly (隨機地)
  - typically 10 - 30% of the dataset
  - Random shuffling (隨機洗牌)



Random shuffling (隨機洗牌)

- `import numpy as np`
- `np.random.permutation(5)# 排列，執行兩次`
- `array([2, 1, 4, 0, 3])`
- `array([1, 3, 0, 2, 4])`

- `np.random.seed(42)# random.seed 隨機種子`
- `shuffled_indices = np.random.permutation(5)`
- `print(shuffled_indices, shuffled_indices[:3], shuffled_indices[3:])`
- `[1 4 2 0 3] [1 4 2] [0 3]`
- `[1 4 2 0 3] [1 4 2] [0 3]`

split_train_test
自行定義函數 (function)

```
def split_train_test(data, test_ratio):
    np.random.seed(42) # 以保持相同輸出
    shuffled_indices = np.random.permutation(len(data))
    test_set_size = int(len(data) * test_ratio)
    test_indices = shuffled_indices[:test_set_size]
    train_indices = shuffled_indices[test_set_size:]
    return data.iloc[train_indices], data.iloc[test_indices] #回傳
train_set, test_set = split_train_test(insurance, 0.2)
print(len(train_set), "train +", len(test_set),
  "test", ' = total ',  len(train_set) + len(test_set))
1071 train + 267 test  = total  1338
```

train_set.head(2)

	age	sex	bmi	children	smoker	region	charges
846	51	female	34.20	1	no	southwest	9872.7010
560	46	female	19.95	2	no	northwest	9193.8385

train_test_split()

- `from sklearn.model_selection import train_test_split #` scikit-learn: machine learning (機器學習) in Python

- `X_train, X_test, Y_train, Y_test = train_test_split(insurance.iloc[:, 0:6], insurance.iloc[:, 6], test_size=0.2, random_state = 42)`

X_train.head(2)

	age	sex	bmi	children	smoker	region
560	46	female	19.95	2	no	northwest
1285	47	female	24.32	0	no	northeast

```
Y_train.head(2)

560      9193.8385
1285     8534.6718
Name: charges, dtype: float64
```

Lecture 4 Regression (迴歸)

- 4.1 Discover (發現) and Visualize (視覺化) the Data to Gain Insights (洞察)
- 4.2 Prepare the Data for Machine Learning Algorithms (機器學習演算法)
- 4.3 Select and Train a Model
- 4.4 Zara
- Bonaccorso, Machine Learning Algorithms
- Geron, Hands-On Machine Learning With Scikit-Learn and Tensorflow
- 長榮大學資設院資管系許志華

4.1 Discover (發現) and Visualize (視覺化) the Data to Gain Insights (洞察)

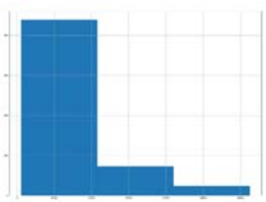
- lec04 insurance.ipynb
- put the test set aside and only explore the training set
  - `insurance = train_set.copy()`

	age	sex	bmi	children	smoker	region	charges
846	51	female	34.20	1	no	southwest	9872.7010
560	46	female	19.95	2	no	northwest	9193.8385

- 4.1.1 Single variable (單變量)
- 4.1.2 Looking for Correlations (尋找相關性)

4.1.1 Single variable (單變量)

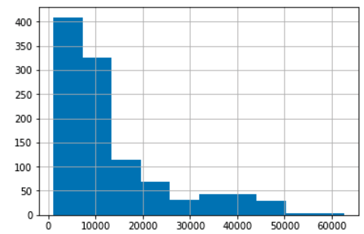
- `import matplotlib.pyplot as plt`
  - matplotlib: Python plotting library
  - Matlab: matrix laboratory (矩陣實驗室)
- `insurance.charges.hist(bins = 3, figsize = (20,15))`
  - DataFrame insurance 檔案
  - charges 變數
  - histograms (直方圖)
  - bins (箱)
  - `figsize: figure size`



- `plt.savefig("data//charges")`
  - 在 data 下，存檔 charges
- `plt.show()`

Histograms (直方圖)

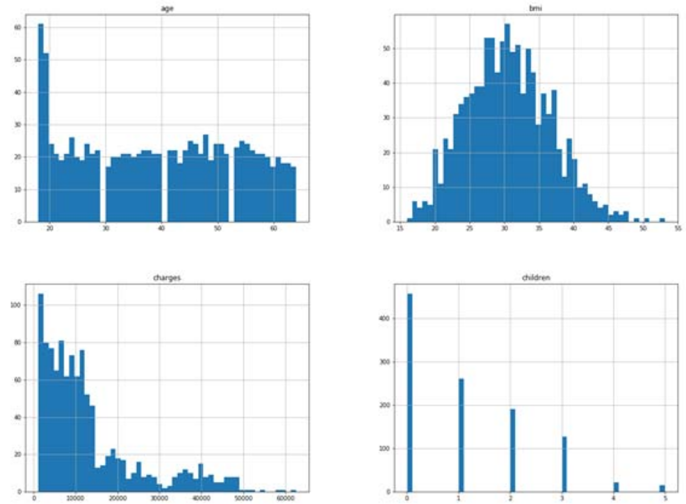
- `insurance.charges.hist(bins = 10)`
- horizontal axis (橫軸): a given value range
  - `insurance.charges.min(),insurance.charges.max()`
  - `(1121.8739, 62592.87309)`
  - `insurance.charges.describe()`
- vertical axis (垂直軸)
  - number of instances (例子)



5

Histograms (直方圖)

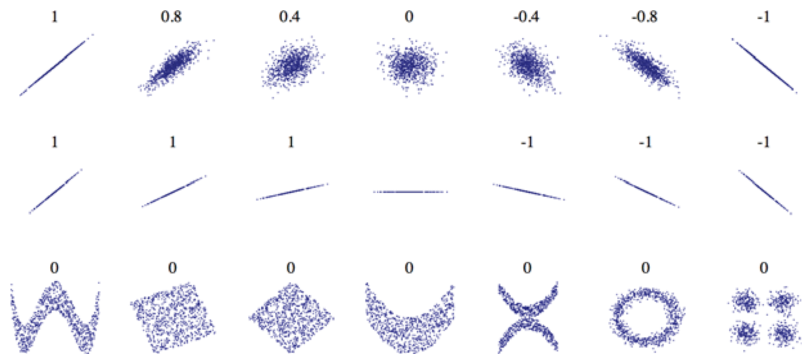
- `insurance.hist(bins=50, figsize=(20,15))`



6

4.1.2 Looking for Correlations (尋找相關性)

- wiki



7

Correlations (相關性) (1)

- `corr_matrix = insurance.corr()`
- # standard correlation coefficient (標準相關係數) (also called Pearson's r) between every pair of attributes
- cor(x, y) = cor(y, x)
- 跟自己是 1
- None of the correlations (相關) are considered strong
- The maximum one: age and charges with 0.281
  - As age increases, so does charges.

	age	bmi	children	charges
age	1.000000	0.119908	0.060911	0.281396
bmi	0.119908	1.000000	-0.005760	0.198274
children	0.060911	-0.005760	1.000000	0.071906
charges	0.281396	0.198274	0.071906	1.000000

8

Correlations (相關性) (2)

	age	bmi	children	charges
age	1.000000	0.119908	0.060911	0.281396
bmi	0.119908	1.000000	-0.005760	0.198274
children	0.060911	-0.005760	1.000000	0.071906
charges	0.281396	0.198274	0.071906	1.000000

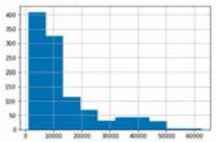
- `corr_matrix[“charges”].sort_values() # ascending (上升) default (預設)`
- `corr_matrix["charges"].sort_values(ascending=False)`

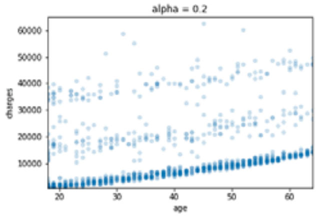
```
children     0.071906
bmi          0.198274
age          0.281396
charges      1.000000
Name: charges, dtype: float64
```

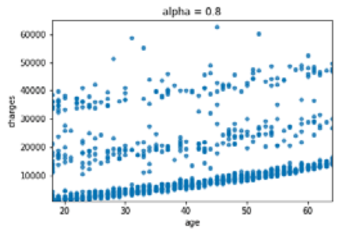
```
charges      1.000000
age          0.281396
bmi          0.198274
children     0.071906
Name: charges, dtype: float64
```

correlation scatterplot (散佈圖)

- `insurance.plot(kind="scatter", x="age", y="charges", alpha=0.2)`
  - alpha: 0.0 transparent (透明)through 1.0 opaque (不透明)
- `plt.axis([18, 64, 1000, 65000])`
- `plt.show()`
- Correlations (相關性): 0.281
- Several straight lines



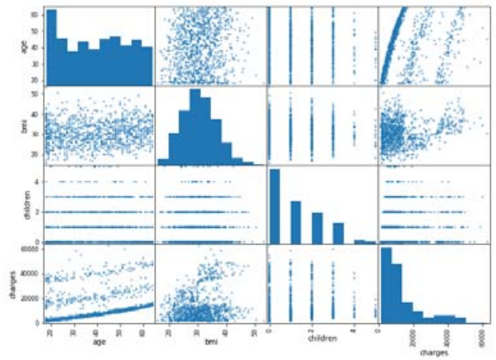




Pandas' `scatter_matrix` function

- `attributes = ["age", "bmi", "children", "charges"]`
- `scatter_matrix(insurance[attributes])`
- `plt.show()`
- bmi and charges:
  2 distinct groups
  of points

```
charges     1.000000
age         0.281396
bmi         0.198274
children    0.071906
Name: charges, dtype: float64
```



4.2 Prepare the Data for Machine Learning Algorithms (機器學習演算法)

	age	sex	bmi	children	smoker	region	charges
846	51	female	34.20	1	no	southwest	9872.70100
560	46	female	19.95	2	no	northwest	9193.83850

- `insurance = train_set.drop("charges", axis=1)`
- `insurance_labels = train_set["charges"].copy()`

	age	sex	bmi	children	smoker	region
846	51	female	34.20	1	no	southwest
560	46	female	19.95	2	no	northwest

```
846    9872.7010
560    9193.8385
Name: charges, dtype: float64
```


Difficulties (困難)

- 1071 entries (項目)
- Missing data: sex and bmi 1070
- Categorical (類別) attributes (屬性): sex, smoker, region
- input numerical attributes (數字屬性) very different scales (尺度)
  - age: 18 to 64
  - bmi: 15.96 to 53.13
  - children: 0 to 5

Data preprocessing (資料預處理)

- 4.2.1 Dealing with missing Data (處理遺漏資料):
  - 遺漏資料：欄位沒有資料，例如使用者沒有填寫、在製品加工失敗等等
  - 4 techniques for data imputation (插補技術)
  - 選 5 筆資料，更容易觀看與理解 (NaN, not a number)
- 4.2.2 Managing categorical data (處理類別資料)
- 4.2.3 Data scaling and normalization (資料縮放與正規化)
- `insurance5 = pd.read_excel ('data/lec03-insurance-5.xlsx')`

	age	sex	bmi	children	smoker	region	charges
0	19	female	NaN	0.0	yes	southwest	16884.92400
1	18	NaN	33.77	1.0	no	NaN	1725.55230
2	37	male	29.83	2.0	no	northeast	6406.41070
3	60	female	NaN	NaN	no	northwest	28923.13692
4	25	NaN	26.22	0.0	NaN	northeast	2721.32080

4.2.1 Dealing with missing Data: (1) Delete (刪除)

- More suitable (適當) and effective (有效) when the number of missing value rows count (列數) is insignificant (微不足道) (say < 5%) compare to (相比於) the overall record count.
- pandas dropna() function: not available (無法使用)
- `insurance5.dropna()`
  - 不是 `insurance5 = insurance5.dropna()`，所以 `insurance5` 沒有變，以下可以重複使用

	age	sex	bmi	children	smoker	region	charges
0	19	female	NaN	0.0	yes	southwest	16884.92400
1	18	NaN	33.77	1.0	no	NaN	1725.55230
2	37	male	29.83	2.0	no	northeast	6406.41070
3	60	female	NaN	NaN	no	northwest	28923.13692
4	25	NaN	26.22	0.0	NaN	northeast	2721.32080

	age	sex	bmi	children	smoker	region	charges
2	37	male	29.83	2.0	no	northeast	6406.4107

(2) Replace with summary (替換為摘要) (1/2)

- Probably the most commonly used (最常用的)
- For quantitative variables (量變數), mean/average or mode or median (中位數) value of the respective column (各欄目)
  - `insurance5.bmi.mean()`
  - (33.77+29.83+26.22)/3 ≈ 29.3
- For categorical (類別) variables, the mode (眾數) (most frequent) summation technique works better.
  - `insurance5.sex.mode()`
  - `0    female`
  - `dtype: object`
  - `insurance5.sex.mode()[0]`
  - `'female'`

	age	sex	bmi	children	smoker	region	charges
0	19	female	NaN	0.0	yes	southwest	16884.92400
1	18	NaN	33.77	1.0	no	NaN	1725.55230
2	37	male	29.83	2.0	no	northeast	6406.41070
3	60	female	NaN	NaN	no	northwest	28923.13692
4	25	NaN	26.22	0.0	NaN	northeast	2721.32080

(2) Replace with summary (替換為摘要) (2/2)

- `insurance5.fillna(insurance5.median()).fillna (insurance5.mode().iloc[0])`

	age	sex	bmi	children	smoker	region	charges
0	19	female	NaN	0.0	yes	southwest	16884.92400
1	18	NaN	33.77	1.0	no	NaN	1725.55230
2	37	male	29.83	2.0	no	northeast	6406.41070
3	60	female	NaN	NaN	no	northwest	28923.13692
4	25	NaN	26.22	0.0	NaN	northeast	2721.32080

	age	sex	bmi	children	smoker	region	charges
0	19	female	29.83	0.0	yes	southwest	16884.92400
1	18	female	33.77	1.0	no	northeast	1725.55230
2	37	male	29.83	2.0	no	northeast	6406.41070
3	60	female	29.83	0.5	no	northwest	28923.13692
4	25	female	26.22	0.0	no	northeast	2721.32080

(3) Random replace (隨機替換)

- a randomly picked value from the respective column (各欄目)
- Appropriate (適當) where the missing values row count is insignificant (微不足道)
- `import random`
- `random.randrange(insurance.bmi.min().round(), insurance.bmi.max().round()) # 3 次`
- `17, 24, 19`
- `region_cat =insurance.region.astype('category').values.categories`
- `np.random.choice(region_cat) # 3 次`
- `('southwest', 'northwest', 'southwest')`

(4) Using predictive model (預測模型)

- This is an advanced technique (先進的技術).
- Train a regression (迴歸) model for continuous variables (連續變項) with the available (可用) data and use the model to predict the missing values.
- N. DeHoratius, A.J. Mersereau, and L. Schrage, Retail Inventory (零售庫存) Management When Records Are Inaccurate (不準確), *MSOM*, 2008. (Best Paper)
  - Bayesian (貝氏) inventory
- D. Bertsimas, C. Pawlowski, and Y.D. Zhuo, From Predictive Methods to Missing Data Imputation (插補): An Optimization (最佳化) Approach, JMLR, 2018.
  - Integer programming difficult, first-order conditon

4.2.2 Managing categorical data (類別資料)

- cannot immediately (立即) be processed by any algorithm (演算法)
- `insurance.region.astype('category').values.categories`
- `Index(['northeast', 'northwest', 'southeast', 'southwest'], dtype='object')`
  - 東北，西北，東南，西南
- `insurance.region[0] = 'southwest'`
  - LabelEncoder (標籤編碼): 3 # Python 從 0 開始
  - OneHotEncoder (獨熱編碼，一位有效編碼): [0, 0, 0, 1]

LabelEncoder (標籤編碼)

- `from sklearn.preprocessing import LabelEncoder`
  - scikit-learn (sklearn): Machine Learning in Python
  - sklearn.preprocessing package (前處理套件)
- `encoder = LabelEncoder()`
- `insurance_cat = insurance["region"]`
- `insurance_cat_encoded = encoder.fit_transform(insurance_cat)`
- `print(encoder.classes_)`
- `['northeast' 'northwest' 'southeast' 'southwest']`
- `print(insurance_cat_encoded)`
- `[3 2 2 ... 2 3 1]`

OneHotEncoder (獨熱編碼)

- `from sklearn.preprocessing import LabelBinarizer # 標籤二元化`
- `encoder = LabelBinarizer()`
- `insurance_cat_1hot = encoder.fit_transform(insurance_cat)`
- `insurance_cat_1hot`
-
```
array([[0, 0, 0, 1],
       [0, 0, 1, 0],
       [0, 0, 1, 0],
       ...,
       [0, 0, 1, 0],
       [0, 0, 0, 1],
       [0, 1, 0, 0]])
```
- `# 標籤編碼 [3 2 2 ... 2 3 1]`

LabelEncoder vs. OneHotEncoder

- `Index(['northeast', 'northwest', 'southeast', 'southwest'], dtype='object')`
  - 東北，西北，東南，西南
- OneHotEncoder (獨熱編碼)
  - `'northeast'`= [1, 0, 0, 0]
  - `'northwest'`= [0, 1, 0, 0]
  - `'southwest'`= [0, 0, 0, 1]
  - A classifier (分類器) based on the distance (距離): $\sqrt{(0-1)^2+(1-0)^2}=\sqrt{2}$
- LabelEncoder (標籤編碼)
  - `'northeast'`= 0
  - `'northwest'`= 1
  - `'southwest'`= 3
  - A classifier based on the distance: 1, 3, 2

4.2.3 Data scaling and normalization (資料縮放與標準化)

- Cannot compare with (比較) different scales (不同的尺度)
- 正規化：以行 (column) 特徵為依據
- min-max normalization (極值標準化)

$$X_{new} = \frac{X - min(X)}{max(X) - min(X)} \in [0,1]$$

- `from sklearn.preprocessing import MinMaxScaler`
- `ss = MinMaxScaler() # scaler 純量`
- `scaled_data = ss.fit_transform(x)`
- X = (1, 2, 6)
  - min(X) = 1
  - max(X) = 6
  - Xnew = (1–1, 2 – 1, 6 – 1) / (6 – 1) = (0, 1/5, 1)

z-score standardization (Z-分數標準化)

$$X_{new} = \frac{X-\mu}{\sigma} = \frac{X - Mean(X)}{StdDev(X)}$$

- Prob(Xnew >= 3) = Prob(Xnew <= –3) = 0.1%
- X = (1, 2, 6), mean = 3
  - std = $\sqrt{\frac{1}{3}\left((1-3)^2+(2-3)^2+(6-3)^2\right)} \approx 2.16$
- $X_{new} = \frac{X-\mu}{\sigma}$: (–0.93, –0.46, 1.39)
  - X < μ ⇒ 負的， X > μ ⇒ 正的
- `from sklearn.preprocessing import StandardScaler`
- `ss = StandardScaler()`
- `scaled_data = ss.fit_transform(x)`

4.2.4 Size of Data Frame

	age	sex	bmi	children	smoker	region
0	51	female	34.20	1	no	southwest

- 數值資料 3 個
- Text and Categorical Attributes: sex 2, smoker 2, region 4 種
  - 2 classes: `sex_ohe = encoder.fit_transform(insurance["sex"])`
  - `encoder.classes_`
  - `sex_ohe = np.hstack((1 - sex_ohe, sex_ohe))`

```
array([[0],
       [0],
       [0],
       ...,
       [1],
       [0],
       [1]])
```

```
array(['female', 'male'], dtype='<U6')
```

	female	male
0	1	0
1	1	0

- 共 3 + 2 + 2 + 4 = 11 種

concat (concatenate 連接)

	age	sex	bmi	children	smoker	region
0	51	female	34.20	1	no	southwest
1	46	female	19.95	2	no	northwest
2	47	female	24.32	0	no	northeast
3	52	female	24.86	0	no	southeast
4	39	female	34.32	5	no	southeast

- `insurance_df = pd.concat([insurance_num, sex_df, smoker_df, region_df], axis=1)`

	age	bmi	children	female	male	no	yes	northeast	northwest	southeast	southwest
0	51	34.20	1	1	0	1	0	0	0	0	1
1	46	19.95	2	1	0	1	0	0	1	0	0
2	47	24.32	0	1	0	1	0	1	0	0	0
3	52	24.86	0	1	0	1	0	0	0	1	0
4	39	34.32	5	1	0	1	0	0	0	1	0

Better approach

- `cat_df = pd.concat([insurance["sex"], insurance["smoker"], insurance["region"]], axis=1)`
- `cat_df.head()`

	sex	smoker	region
0	female	no	southwest
1	female	no	northwest
2	female	no	northeast

- `pd.get_dummies(cat_df)`

	sex_female	sex_male	smoker_no	smoker_yes	region_northeast	region_northwest	region_southeast	region_southwest
0	1	0	1	0	0	0	0	1
1	1	0	1	0	0	1	0	0

Correlation (相關性)

- `all_data = pd.concat([insurance_df, insurance_labels], axis=1)`
- `all_data.corr()["charges"].sort_values(ascending=False)`
- Smoker yes: 0.780075 (>> 0.281396 for age)

```
charges      1.000000
yes          0.780075
age          0.281396
bmi          0.198050
children     0.071906
southeast    0.067639
male         0.057049
northeast   -0.000472
northwest   -0.033618
southwest   -0.035414
female      -0.057049
no          -0.780075
Name: charges, dtype: float64
```

4.3 Select and Train a Regression (迴歸) Model

- framed the problem (框架問題)
- got the data and explored (探索) it
- sampled a training set and a test set (訓練集和測試集取樣)
- cleaned up and prepared your data
- now ready to select and train (選擇和訓練) a Machine Learning model (機器學習模型)

Select a Performance Measure (績效衡量)

- Mean Absolute Error (MAE，平均絕對誤差): l_1 norm (範數)

$$MAE(X,h) = \frac{1}{m}\sum_{i=1}^{m}\left|h(x^{(i)}) - y^{(i)}\right|$$

- Root Mean Square Error (RMSE，均方根誤差): l_2 norm

$$RMSE(X,h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2}$$

- Error (–5, 1, 0.5)
  - l_1 norm 5 + 1 + 0.5
  - l_2 norm 25 + 1 + 0.25 (專注於大的數值，忽略小的)
  - RMSE more sensitive (敏感) to outliers (異常值) than MAE

4.3.1 Training and Evaluating (訓練與評估) on the Training Set

- `from sklearn.linear_model import LinearRegression`
- `#` 正規化：subtract the mean and divide by the l2-norm
- `lr = LinearRegression(normalize=True)`
- `# Train the model` (訓練模型)
- `lr.fit(insurance_df, insurance_labels)`
  - `insurance_df`: *X*
  - `insurance_labels` (保險標籤): *y*
- `print( "Score {:.4f}".format(lr.score(insurance_df, insurance_labels)) )`
- `Score 0.7420 #` coefficient of determination (決定係數)，解釋能力

Linear Regression Equation (線性迴歸方程式)

- `print('y = %.3f '% lr.intercept_)`
- `for i, c in enumerate(lr.coef_):`
  `print('%.3f '% c, insurance_df.columns.values[i])`
- Intercept 截距， coefficient 係數， enumerate 列舉

- children ∈{0,1,…,5}: 每增加 1 個，增加 426.727 元
- female 比 male 多 11.864
- smoke 吸煙多花 23654.323
- northeast 比 southwest 多

(468.154 – (–353.502) =) 821.656

```
y = -624.669
256.606   age
338.995   bmi
426.727   children
5.932   female
-5.932   male
-11827.161   no
11827.161   yes
468.154   northeast
91.163   northwest
-197.367   southeast
-353.502   southwest
```

Prediction and Error (預測和誤差)

- `some_data = insurance_df.iloc[:4] # 4 個例子`
- `some_labels = insurance_labels.iloc[:4]`
- `print("Predictions:\t", lr.predict(some_data)) #預測的 y`
- `Predictions: [12307.89076923  7065.56550747 8327.11919265  9127.68678932]`
- `print("Labels:\t\t", list(some_labels)) # 真正的 y`
- `Labels: [9872.701, 9193.8385, 8534.6718, 27117.99378]`
- `lr.predict(insurance_df).min(), lr.predict(insurance_df).max() # 預測的範圍`
- `(-1960.3279647819486, 40804.71968748159)`
- `insurance_labels.min(), insurance_labels.max() #真正的範圍`
- `(1121.8739, 62592.87309)`
- `from sklearn.metrics import mean_squared_error`
- `print(np.sqrt( mean_squared_error(insurance_labels, lr.predict(insurance_df) ))) # 均方根誤差`, sqrt: square root
- `6099.432725507942`

The 4th Sample

- `insurance_df.iloc[3]`
- `insurance_labels.iloc[3]`
- 真正 `27117.99378`
- 預測 `9127.68678932` (是真正的 1/3)
- `insurance_all[(insurance_all.age == 52) & (insurance_all.female == 1) & (insurance_all.children == 0) ]`

```
y = -624.669
256.606  age
338.995  bmi
426.727  children
5.932  female
-5.932  male
-11827.161  no
11827.161  yes
468.154  northeast
91.163  northwest
-197.367  southeast
-353.502  southwest
```

	age	bmi	children	female	male	no	yes	northeast	northwest	southeast	southwest	charges
3	52	24.86	0	1	0	1	0	0	0	1	0	27117.99378
633	52	23.18	0	1	0	1	0	1	0	0	0	10197.77220
670	52	31.20	0	1	0	1	0	0	0	0	1	9625.92000
789	52	37.40	0	1	0	1	0	0	0	0	1	9634.53800

4.3.2 Multicollinearity (多重共線性)

- (wiki) Collinearity is a linear association (線性關聯) between two explanatory variables (解釋變量)
  - perfectly collinear: $X_{2i} = \lambda_0 + \lambda_1 X_{1i}$ for all observations (觀察) i. (係數) female + male = 1
  - `insurance_df.corr()`

	age	bmi	children	female	male
age	1.000000	0.119711	0.060911	0.009243	-0.009243
bmi	0.119711	1.000000	-0.005710	-0.014016	0.014016
children	0.060911	-0.005710	1.000000	-0.017156	0.017156
female	0.009243	-0.014016	-0.017156	1.000000	-1.000000
male	-0.009243	0.014016	0.017156	-1.000000	1.000000

female	male
1	0
1	0

northeast	northwest	southeast	southwest
0	0	1	0
1	0	0	0
0	0	0	1

- Multicollinearity
  - Northeast + northwest + southeast + southwest = 1

drop one of the problematic variables (丟掉一個有問題的變數)

- James, et al., An Introduction to Statistical Learning
- `cat_df.head(2)`
- `cat_df.index`
- `RangeIndex(start=0, stop=1071, step=1)`
- `new_df = pd.DataFrame(index = cat_df.index)`
- `for i in cat_df:`
  `new_df = new_df.join(pd.get_dummies(cat_df[i]).iloc[:, 1:])`
- `# 取第 1 行之後`

	sex	smoker	region
0	female	no	southwest
1	female	no	northwest

	sex_female	sex_male	smoker_no	smoker_yes	region_northeast	region_northwest	region_southeast	region_southwest
0	1	0	1	0	0	0	0	1
1	1	0	1	0	0	1	0	0

	male	yes	northwest	southeast	southwest
0	0	0	0	0	1
1	0	0	1	0	0

Linear Regression Equation (線性迴歸方程式)

- `lr2 = LinearRegression(normalize=True)`#左，除 l2 norm
- `from sklearn.preprocessing import StandardScaler`
- `ss = StandardScaler()` #右邊，Z-分數標準化，除標準差
- `scaled_data2 = ss.fit_transform(insurance_new_df)`
- `lr5 = LinearRegression(normalize= False)`
- `lr5.fit(scaled_data2, insurance_labels)`

```
y = -11977.744
256.606   age
338.995   bmi
426.727   children
-11.864   male
23654.323   yes
-376.992  northwest
-665.521  southeast
-821.656  southwest
```

```
y = 13342.847
3609.246   age
2046.531   bmi
518.410   children
-5.931   male
9556.463   yes
-160.783   northwest
-293.446   southeast
-354.561   southwest
```


statsmodels.api: Statistics in Python

- `import statsmodels.api as sm`
- `X2 = sm.add_constant(scaled_data2)`#截距
- `est = sm.OLS(insurance_labels, X2).fit()`
- `# Ordinary least squares` 最小平方法
- `print(est.summary())`
- statistically nonsignificant (統計不顯著)

```
y = 13342.847
3609.246   age
2046.531   bmi
518.410   children
-5.931   male
9556.463   yes
-160.783   northwest
-293.446   southeast
-354.561   southwest
```

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.334e+04	187.166	71.289	0.000	1.3e+04	1.37e+04
x1	3609.2458	189.357	19.060	0.000	3237.689	3980.803
x2	2046.5312	195.861	10.449	0.000	1662.212	2430.850
x3	518.4104	187.694	2.762	0.006	150.117	886.704
x4	-5.9306	187.761	-0.032	0.975	-374.355	362.494
x5	9556.4627	188.271	50.759	0.000	9187.036	9925.889
x6	-160.7833	228.746	-0.703	0.482	-609.628	288.062
x7	-293.4463	237.771	-1.234	0.217	-760.000	173.107
x8	-354.5610	230.532	-1.538	0.124	-806.911	97.789

4.3.3 Backward selection (向後選擇)

- G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning* (統計學習) : *With Applications in R*, Springer, 2013.
- Backward remove the variable (刪除變數) with the largest (最大) p-value — that is, the variable selection that is the least statistically significant (最小統計顯著性).
- The new (n − 1)-variable model is fit, and the variable with the largest p-value is removed.
- This procedure continues until a stopping rule (停止規則) is reached.
- For instance, we may stop when all remaining (其餘) variables have a p-value below some threshold (閾值).
  - commonly set to (通常設置為) 0.05, 0.01, 0.005, or 0.001

Drop nonsignificant variables (不顯著變數)

- `insurance_back = insurance_df.drop(['female', 'male', 'no', 'northeast', 'northwest', 'southeast', 'southwest'], axis=1)`
- `insurance_back.head()`

	age	bmi	children	yes
0	51	34.20	1	0
1	46	19.95	2	0
2	47	24.32	0	0
3	52	24.86	0	0
4	39	34.32	5	0

```
y = 13342.847
3609.246   age
2046.531   bmi
518.410   children
-5.931   male
9556.463   yes
-160.783   northwest
-293.446   southeast
-354.561   southwest
```

	coef	std err	t	P>|t|
const	1.334e+04	187.166	71.289	0.000
x1	3609.2458	189.357	19.060	0.000
x2	2046.5312	195.861	10.449	0.000
x3	518.4104	187.694	2.762	0.006
x4	-5.9306	187.761	-0.032	0.975
x5	9556.4627	188.271	50.759	0.000
x6	-160.7833	228.746	-0.703	0.482
x7	-293.4463	237.771	-1.234	0.217
x8	-354.5610	230.532	-1.538	0.124

Linear Regression Equation (線性迴歸方程式)

- Score 0.7414 # sklearn.linear_model
- statsmodels.api

```
y = 13342.847
3610.375   age
1988.111   bmi
520.803   children
9557.737   yes
```

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.334e+04	187.051	71.332	0.000	1.3e+04	1.37e+04
x1	3610.3747	189.038	19.099	0.000	3239.445	3981.304
x2	1988.1110	188.423	10.551	0.000	1618.389	2357.833
x3	520.8026	187.444	2.778	0.006	153.002	888.603
x4	9557.7368	187.337	51.019	0.000	9190.145	9925.329

- Optimal solution (最佳解) $\hat{\theta} = (X^T X)^{-1} X^T y$ (in lecture 5)
- `array([13342.84660963, 3610.37465382, 1988.11102153, 520.80264097, 9557.73679047])`

4.3.4 Improving model performance (提高模型性能)

- Model specification (規格): Adding nonlinear (非線性) relationships (關係)

$$y = \alpha + \beta_1 \text{age} + \beta_2 \text{age}^2$$

```
insurance_back['age2'] = insurance_back ['age'] ** 2
```

- Transformation (轉換): converting a numeric variable to a binary indicator: If bmi > 30, then 1. Otherwise 0
  - If the variable is not statistically significant (統計顯著), exclude (排除) it in the future.
- Adding interaction effects (交互作用): bmi30 * smoker 代表 3 項 # 0 或 1

```
insurance_back['bmi30_smoker']
=(insurance_back['bmi']>30) * insurance_back['yes']
```


Nonlinear Regression Equation (非線性迴歸方程式)

- Score 0.8609 (> `0.7414 slide 41)`
- `rmse 4479.113003368343 (< 6107.166690520324)`
- `lr3.predict(insurance_back).min(), lr3.predict(insurance_back).max()# 預測`
- `(2480.331494519739, 50050.25808433389)`
- `insurance_labels.min(), insurance_labels.max() # 真正`
- `(1121.8739, 62592.87309)`

```
y = 13342.847
-15.327   age
263.812   bmi
737.059   children
5415.566   yes
3731.687   age2
6061.605   bmi30_smoker
```


Backward selection (向後選擇)

- Drop 2 statistically nonsignificant (不顯著) variables age and bmi

	coef	std err	t	P>|t|
const	1.334e+04	137.316	97.169	0.000
x1	-15.3266	945.675	-0.016	0.987
x2	263.8121	149.632	1.763	0.078
x3	737.0592	143.929	5.121	0.000
x4	5415.5659	195.207	27.743	0.000
x5	3731.6873	945.093	3.948	0.000
x6	6061.6046	202.920	29.872	0.000

```
y = 13342.847
-15.327   age
263.812   bmi
737.059   children
5415.566   yes
3731.687   age2
6061.605   bmi30_smoker
```

- New R^2 Score 0.8605 (< 0.8609)

	children	yes	age2	bmi30_smoker
0	1	0	2601	0
1	2	0	2116	0

```
y = 13342.847
735.777   bmi
5324.123   children
3747.216   yes
6196.459   age2
```

4.3.5 Test set (測試集合)

	children	yes	age2	bmi30_smoker
0	2	0	2025	0
1	0	0	1296	0

- 經過同樣的轉換，太麻煩
  - Transformation Pipeline (轉換工作流) in lecture 5
- R^2 Score 0.883 (> 0.8609 slide 43 訓練集合)
- `rmse 4278 (< 4479 訓練集合)`

- D. Bertsimas and A. King, An Algorithmic Approach (演算法方法) to Linear Regression, *Operations Research*, 2016.
  - mixed integer quadratic optimization (MIQO,混合整數二次最佳化)
  - explicitly addresses (明確地處理) various competing objectives (各種競爭目標) and demonstrate the effectiveness (證明有效性) of our approach on both real and synthetic data sets.

4.4 Markdown pricing (降價定價) of Zara

- (wiki) 西班牙零售服裝製作與銷售公司
- Felipe Caro and Jérémie Gallien, Clearance Pricing Optimization (清倉定價最佳化) for a Fast-Fashion Retailer (快時尚零售商), *Operations Research*, 2012.
  - 上課網頁
- 在一季中，Zara 的主要競爭者提供約 2000 到 4000 件商品，但是 Zara 卻提供平均約 11,000 件的商品。
- 原先 Zara 使用人工決定換季大拍賣的折價價格，作者使用數學模型來處理此問題
  - 決策：次數，時間點，幅度
- 營收管理：定價 (行銷)，庫存 (作業管理)
  - 航空、旅館 …

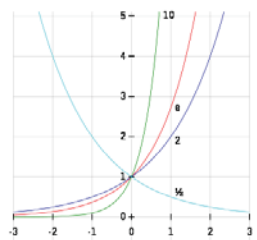
4.4.1 第一階段 (1)

- 根據歷史資料，得到需求曲線 λ
- 考慮等五項因素：Purchase quantity (商品的採購量), Age of an article, Previous period demand, Broken assortment effect (破碎分類效應), Price discount (價格折扣)
- When inventory (庫存) is low, the remaining items (剩下的物品) are usually those that are less attractive (吸引力) to customers
  - Threshold (閾值) f for the entire country
- price tag (價格標籤) p_r^T
- 利用 (指數) 線性迴歸得到其參數 β

$$\lambda_r^w = F(C_r, A_r^w, \lambda_r^{w-1}, I_r^w, p_r^w)$$
$$= \exp\left(\beta_{0r} + \beta_1 \ln(C_r) + \beta_2 A_r^w + \beta_3 \ln(\lambda_r^{w-1}) + \beta_4^w \ln(\min\{1, \frac{I_r^w}{f}\}) + \beta_5^w \ln\left(\frac{p_r^w}{p_r^T}\right)\right)$$

第一階段 (2)

- 需求曲線 λ = $\exp\left(\beta_{0r} + \beta_1 \ln(C_r) + \beta_2 A_r^w + \beta_3 \ln(\lambda_r^{w-1}) + \beta_4^w \ln(\min\{1, \frac{I_r^w}{f}\}) + \beta_5^w \ln\left(\frac{p_r^w}{p_r^T}\right)\right)$
- $\lambda(w) = \beta_0 ({C_r}^{\beta_1})(e^{\beta_2 A})(\lambda(w-1)^{\beta_3})\left(\min\left\{1, \frac{I_r}{f}\right\}^{\beta_4}\right)\left(\left(\frac{p^w}{p^T}\right)^{\beta_5}\right)$
  - w: week, $\lambda(w-1)$ 前一週的需求
- Age of an article: A (≥ 0) ↑ ⇒ λ ↓
  - 使用時間變短，變舊
  - β_2 正或負？淺藍負的，其餘正
  - 負
  - 如果 $\beta_2 = -0.1$



A	0	1	2	3	4	5
exp(beta2*A)	1	0.90484	0.81873	0.74082	0.67032	0.60653

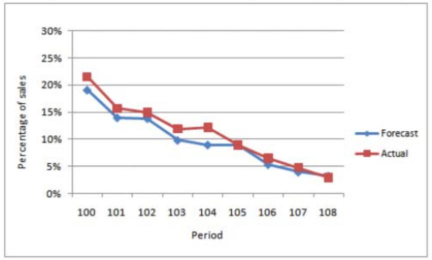
第一階段 (3)

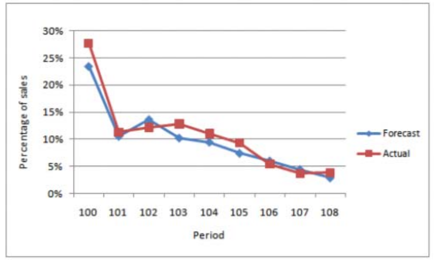
- 需求曲線 $\lambda(w) = \beta_0(C_r{}^{\beta_1})(e^{\beta_2 A})(\lambda(w-1)^{\beta_3})\left(\min\left\{1,\frac{I_r}{f}\right\}^{\beta_4}\right)\left(\left(\frac{p^w}{p^T}\right)^{\beta_5}\right)$
  - w: week, $\lambda(w-1)$ 前一週的需求
- Broken assortment effect (破碎分類效應)
  - I_r inventory (庫存) of article r available in the entire country
  - If $I_r < f$ (閾值), then $\min\left\{1,\frac{I_r}{f}\right\} = \frac{I_r}{f} < 1 \Rightarrow \beta_4 > 1$
  - If $I_r > f$, then $\min\left\{1,\frac{I_r}{f}\right\} = 1 \Rightarrow \lambda(w)$ 不變
- current price p^w, price tag (價格標籤) $p_r{}^T$, so $\frac{p^w}{p_r{}^T} < 1$
  - $p^w \downarrow \Rightarrow$ Discount (折扣) $\uparrow \Rightarrow \lambda \uparrow \Rightarrow \beta_5$ 正或負？
  - β_5 負，$\frac{p^w}{p_r{}^T}$ =0.2

beta	-2	-1	0	1	2
(pw/pr)^beta	25	5	1	0.2	0.04

第一階段 (4)

- mean absolute deviation (MAD, 平均絕對偏差): Weekly aggregate forecast (每週匯總預測) versus actual sales for Belgium (比利時) (left) and Ireland (愛爾蘭) (right)





4.4.2 第二階段

- 決定價格，以最大化營收
  - 價格會影響需求，營收則是需求和售價的乘積
- 限制
  - 需求曲線
  - 庫存 (t+1) = 庫存 (t) 減 需求(t)
- 最佳解
  - 因為多階段和不確定，使用動態規劃 (dynamic programming)
  - 因為太複雜，所以將變數取期望值，變成非線性規劃 (nonlinear programming)
  - 最後再線性化 (linearization) 變成混合整數規劃 (mixed integer programming)，以方便 Cplex求解

結果

- 作者於 2009 年年初上線測試，在比利時和愛爾蘭的測試結果顯示增加了 7 千 3 百萬美元的收入
  - 此計畫幾乎不增加任何的成本，因此可將之視為利潤的增加
  - 原本的總收入是 12 億 5 千 9 百萬美元，所以增加了 5.8%。
  - 在營收管理中，有一派的說法是競爭已經呈現在原本的需求曲線中，不需要再度考慮之。
  - Zara 的產品具有特殊性，較具有價格的競爭力。
- 由於此實驗的成功，Zara 將在全球推行此計畫。
- 後續許多相關的計畫
  - http://chhsu135.blogspot.com/search?q=zara

1

Lecture 5 Training models (訓練模型)

- 5.1 Linear regression (線性迴歸)
- 5.2 Gradient Descent (梯度下降法)
- 5.3 Polynomial (多項式) regression
- 5.4 Learning curves (學習曲線)
- 5.5 Regularization (正規化): Ridge, Lasso, and ElasticNet
- 5.6 Multi Asset Trend Following Strategy (多資產趨勢跟踪策略) by J.P.Morgan (摩根大通)
- Bonacccorso, Machine Learning Algorithms
- Geron, Hands-On Machine Learning With Scikit-Learn and Tensorflow
- 長榮大學資設院資管系許志華

2

5.1 Linear regression (線性迴歸)

- 酒的價格 (y) = –0.4504 + 0.6014 成長期平均溫度 (AGST) – 0.003958 收成時雨量 + 0.001043 冬季雨量 + ε
- 預測價格 ($\hat{y}$)=$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 = [\theta_0 \quad \theta_1 \quad \theta_2 \quad \theta_3]\begin{bmatrix}1\\x_1\\x_2\\x_3\end{bmatrix}$

$$\equiv \theta^T x = [1 \quad x_1 \quad x_2 \quad x_3]\begin{bmatrix}\theta_0\\\theta_1\\\theta_2\\\theta_3\end{bmatrix} = x^T\theta$$

- independent variable (獨立變數) x_i: 均溫、收雨、冬雨。Greek θ theta, ε epsilon. Model parameters (參數) θ_i
- n the number of features (特徵), x_i the ith feature value.

3

Vector form (向量形式)

- 酒的價格 ($\hat{y}$) = –0.4504 + 0.6014 成長期平均溫度 (AGST) – 0.003958 收成時雨量 + 0.001043 冬季雨量 $= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 = \theta^T x$
- Predicted (預測) value $\hat{y}$
- 總共 m 筆資料

$$\hat{y} \equiv \begin{bmatrix}\hat{y}_1\\\vdots\\\hat{y}_m\end{bmatrix} = \begin{bmatrix}1 & x_{11} & x_{21} & x_{31}\\\vdots & & \vdots & \\1 & x_{1m} & x_{2m} & x_{3m}\end{bmatrix}\begin{bmatrix}\theta_0\\\theta_1\\\theta_2\\\theta_3\end{bmatrix} \equiv X\theta$$

  - $X = [1 \quad \text{dataFrame}]$
  - $X \in R^{m\times(n+1)}$, 特徵 n = 3 now
  - Matrix multiplication (矩陣乘法)

4

5.1.1 Objective (目標): Mean Square Error (MSE，均方誤差)

$$MSE(\theta) = \min \frac{1}{m}\sum\left(y_i - \hat{y}_i\right)^2$$ (m 筆資料)

- $y = \begin{bmatrix}y_1\\\vdots\\y_m\end{bmatrix}, y^T = [y_1 \quad \cdots \quad y_m], \hat{y} = \begin{bmatrix}\hat{y}_1\\\vdots\\\hat{y}_m\end{bmatrix} = X\theta,$

$$\Rightarrow \sum(y_i - \hat{y}_i)^2 = [y_1 - \hat{y}_1 \quad \cdots \quad y_m - \hat{y}_m]\begin{bmatrix}y_1 - \hat{y}_1\\\vdots\\y_m - \hat{y}_m\end{bmatrix}$$

$$= (y - X\theta)^T(y - X\theta) = (y^T - \theta^T X^T)(y - X\theta)$$

  - Transpose (轉置), Unknowns (未知) θ
- $\min_\theta \frac{1}{m}(y - X\theta)^T(y - X\theta)$

Optimal Solution (最佳解): Coefficient (係數)

- $\min_{\theta} \frac{1}{m} (y - X\theta)^T (y - X\theta)$
- Optimal solution $\hat{\theta} = (X^T X)^{-1} X^T y$ (by calculus)
  - normal equation (正規方程式)
  - 在某些條件下，$X^T X$ 的反矩陣存在
  - Transpose (轉置): $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}$
  - Inverse matrix (反矩陣): Assume B $\in R^{2\times2}$, $BB^{-1} = B^{-1}B = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
    - Example: $\begin{bmatrix} 1 & 3 \\ 1 & 4 \end{bmatrix} \begin{bmatrix} 4 & -3 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

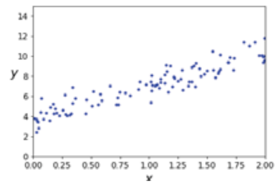
Dimension (維度) of the matrices

- Optimal solution (最佳解) $\hat{\theta} = (X^T X)^{-1} X^T y$
- y$\in R^{m\times1}$
- Transpose (轉置): $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}$
- $X \in R^{m\times(n+1)} \Rightarrow X^T \in R^{(n+1)\times m}$
- $X^T X \in R^{(n+1)\times(n+1)} \Rightarrow (X^T X)^{-1} \in R^{(n+1)\times(n+1)}$
- $X^T y \in R^{(n+1)\times1}$
- $\hat{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} = (X^T X)^{-1} X^T y \in R^{(n+1)\times1}$

Computational Complexity (計算複雜性)

- $\hat{\theta} = (X^T X)^{-1} X^T y$
- If $X^T X$ $n \times n$ matrix, inverse (反矩陣) $O(n^{2.4})$ to $O(n^3)$ (depending on the implementation (實現))
  - A function t(n) ∈ O(g(n)) if exist positive constant c and some nonnegative (非負) integer n_0 such that $t(n) \le cg(n)$ for all n $\ge n_0$. (上界 $g(n)$)
  - Double the number of features (特徵增加一倍): $\frac{(2n)^{2.4}}{n^{2.4}} \approx 5.28, \frac{(2n)^3}{n^3} = 8$
- Predictions by $\theta^T x$: linear

5.1.2 Python



- Python: lec05 training_linear_models
- `import numpy as np`
- `m = 100`
- `np.random.seed(68) # 以便產生一致的結果`
- `X = 2 * np.random.rand(m, 1) # [0, 2)`
- `# random samples from a uniform distribution (均勻分佈) over [0, 1), size (100, 1)`
- `np.random.seed(76)`
- `y = 4 + 3 * X + np.random.randn(m, 1)`
- `# random.randn: a sample (or samples) from the standard normal distribution (標準常態分配)`
- `# [4, 10) + 亂數`

Optimal Solution (最佳解)

- Slide 3: $\begin{bmatrix} 1 & x_{11} \\ \vdots & \vdots \\ 1 & x_{1m} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} \equiv X\theta, m = 100$
- `X_b = np.c_[np.ones((100, 1)), X]`
- $\hat{\theta} = (X^T X)^{-1} X^T y$
- `theta_best = np.linalg.inv(X_b.T.dot(X_b)) . dot(X_b.T).dot(y)`
  - Linear algebra, inverse (反矩陣), transpose (轉置)
  - `np.dot(a,b) = a.dot(b)`

```
array([[3.74570384],
       [3.31633466]])
```

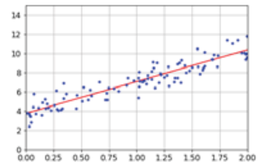

Draw the line

```
array([[3.74570384],
       [3.31633466]])
```

- `# new test instances`
- `X_new = np.array([[0], [2]])`
- `# add x0 = 1 to each instance`
- `X_new_b = np.c_[np.ones((2, 1)), X_new]`
- `print('X_new_b is\n ', X_new_b)`
- `y_predict = X_new_b.dot(theta_best)`
- `print('y_predict is\n ', y_predict)`

```
X_new_b is
 [[1. 0.]
 [1. 2.]]
y_predict is
 [[ 3.74570384]
 [10.37837316]]
```

- `plt.plot(X_new, y_predict, "r-") # red`
- `plt.plot(X, y, "b.") # blue`
- `plt.axis([0, 2, 0, 15])`
- `plt.grid()`



5.1.3 正規方程式推導

- Why? Useful for the understanding (理解) of machine and deep learning
- $\min_{\theta} f(\theta) \equiv \min_{\theta} \frac{1}{m}(y - X\theta)^T (y - X\theta) = \frac{1}{m}(y^T - \theta^T X^T)(y - X\theta) = \frac{1}{m}(y^T y - \theta^T X^T y - y^T X\theta + \theta^T X^T X\theta)$. In general, AB ≠ BA
- $(AB)^T = B^T A^T, (ABC)^T = C^T (AB)^T = C^T B^T A^T$
- Scaler $g(\theta) \equiv y^T X\theta = (y^T X\theta)^T = \theta^T X^T y, (y^T)^T = y, g(\theta) = (y^T X)\theta \equiv c^T\theta = c_1\theta_1 + \cdots + c_n\theta_n$
- Gradient (梯度) $\frac{\partial g(\theta)}{\partial\theta} \equiv \begin{bmatrix} \frac{\partial g(\theta)}{\partial\theta_1} \\ \vdots \\ \frac{\partial g(\theta)}{\partial\theta_n} \end{bmatrix} = \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = c = X^T y$

推導

- $f(\theta) \equiv \frac{1}{m}(y^T y - \theta^T X^T y - y^T X\theta + \theta^T X^T X\theta)$
- $h(\theta) \equiv \theta^T X^T X\theta \equiv d^T\theta = \theta^T d \ \ ((ABC)^T = C^T B^T A^T)$
  - $d \equiv X^T X\theta \Rightarrow d^T = (X^T X\theta)^T = \theta^T X^T X$
  - Product rule (乘積規則)

$$\frac{\partial h(\theta)}{\partial\theta} = \frac{\partial d^T\theta}{\partial\theta} + \frac{\partial\theta^T d}{\partial\theta} = 2d$$

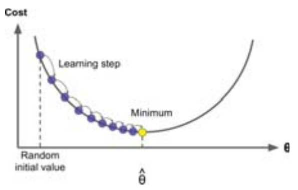
- Gradient (梯度) $\frac{\partial f(\theta)}{\partial\theta} = \nabla_\theta f(\theta) = \frac{1}{m}(0 - 2X^T y + 2X^T X\theta)$

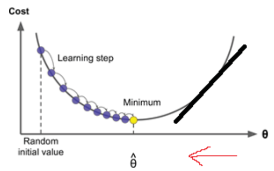
$$\frac{\partial f(\theta)}{\partial\theta} = 0 \Longrightarrow \theta = (X^T X)^{-1} X^T y$$

- Second-order partial derivative (偏微) $\frac{\partial^2 f(\theta)}{\partial\theta^2} = \frac{2}{m} X^T X$

5.2 Gradient Descent (梯度下降法)

- $f(\theta) = \theta^2 - 6\theta + 11$
- tangent line (切線) $f'(\theta) = 2\theta - 6$
- Critical point (臨界點) $f'(\theta) = 0 \Rightarrow \hat{\theta} = 3$
- $f''(\theta) = 2 > 0$ for all $\theta \Rightarrow$ 廣域極小 (global minimum)
- In general, $f'(\theta) = 0$ difficult to solve, e.g, neural net
- $\theta^{(next)} = \theta - \eta f'(\theta)$, Learning rate (學習率) (eta) $\eta = 0.1$
  - a very generic optimization algorithm (最佳化演算法)

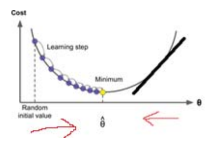




Numerical Example (數值例子)

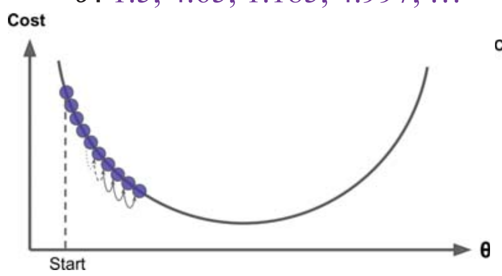
- $f'(\theta) = 2\theta - 6, \theta^{(next)} = \theta - \eta f'(\theta), \eta = 0.1$
- $-f'(\theta) > 0$ for $\theta < 3$, 代表往右走
  - (從左邊) $\theta^0 = 0, f'(0) = -6, -f'(0) = 6$
  - $\theta^1 = 0 - (0.1)(-6) = 0.6$，從 0 到 0.6，往右走
  - $\theta^2 = 0.6 - (0.1)(-4.8) = 1.08$，從 0.6 到 1.08
- $-f'(\theta) < 0$ for $\theta > 3$ 代表往左走
  - (右) $\theta^0 = 4, f'(0) = 2,\ -f'(0) = -2$ 代表往左走
  - $\theta^1 = 4 - (0.1)(2) = 3.8$，從 4 到 3.8

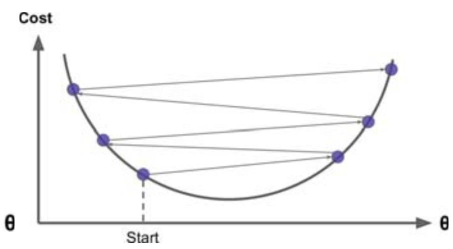
https://cp.cw1.tw/files/md5/30/e6/30e698c34a68081167e8ea47ae29712f-62723.jpg



Learning rate hyperparameter (超參數)

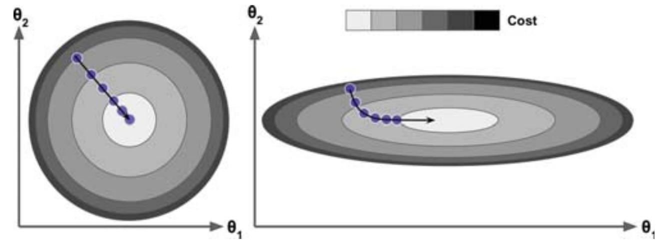
- (左)Too small: Taking a long time to converge. (右) too large
- $f'(\theta) = 2\theta - 6, \theta^{(next\ step)} = \theta - \eta f'(\theta), \eta = 1.05$
- $\theta^0 = 1.5, f'(1.5) = -3, \theta^1 = 1.5 - (1.05)(-3) = 4.65$
- $\theta^1 = 4.65, \theta^2 = 4.65 - (1.05)(3.3) = 1.185, \theta^3 = 4.997$
- θ: 1.5, 4.65, 1.185, 4.997, … 目標 3





Feature Scaling (特徵縮放)

- $\theta^{(next\ step)} = \theta - \eta f'(\theta)$,
- $f(\theta_1, \theta_2) = {\theta_1}^2 + {\theta_2}^2, \frac{\partial f}{\partial \theta_1} = 2\theta_1, \frac{\partial f}{\partial \theta_2} = 2\theta_2$ same speed
- $f(\theta_1, \theta_2) = {\theta_1}^2 + \frac{{\theta_2}^2}{9}$ (橢圓), $\frac{\partial f}{\partial \theta_1} = 2\theta_1, \frac{\partial f}{\partial \theta_2} = 2\theta_2/9$
  - normalization (正規化): $\theta_{2n} = \theta_2/3, {\theta_{2n}}^2 = \frac{{\theta_2}^2}{9}$



5.2.1 Batch Gradient Descent (BGD, 批量梯度下降)

- $\nabla_\theta MSE(\theta) = \begin{bmatrix} \frac{\partial}{\partial\theta_0}MSE(\theta) \\ \frac{\partial}{\partial\theta_1}MSE(\theta) \\ \vdots \\ \frac{\partial}{\partial\theta_n}MSE(\theta) \end{bmatrix} = \frac{2}{m}X^T(X\theta - y)$
- $\theta^{(next\ step)} = \theta - \eta\nabla_\theta MSE(\theta) = \theta - \frac{2\eta}{m}X^T(X\theta - y)$
  - η eta
- involves calculations (計算) over the full training set **X**, at each Gradient Descent step
  - terribly slow (慢) on very large training sets

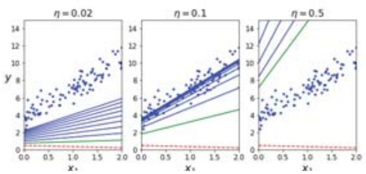
Python

- $\theta^{(next\ step)} = \theta - \frac{2\eta}{m}X^T(X\theta - y)$

```
eta = 0.1 # learning rate
n_iterations = 1000 # 執行次數
m = 100 # 樣本數
theta = np.random.randn(2,1)
# random initialization (隨機初始化)
for iteration in range(n_iterations):
    gradients = 2/m * X_b.T.dot(X_b.dot(theta) - y)
    theta = theta - eta * gradients
theta_best # by normal equation
theta # by Batch Gradient Descent (批量梯度下降)
array([[3.74570384],
       [3.31633466]])
```


Learning rate (學習率)

```
def plot_gradient_descent(theta, eta, theta_path=None):
    m = len(X_b)
    plt.plot(X, y, "b.") # 藍點
    n_iterations = 1000
    for iteration in range(n_iterations):
        if iteration < 10: # Show the first 10 steps
            y_predict = X_new_b.dot(theta)
            style = "r--" if iteration == 0 else
                 ("g" if iteration == 1 else "b-")
plt.subplot(131); plot_gradient_descent(theta, eta=0.02)
plt.subplot(132); plot_gradient_descent(theta, eta=0.1)
plt.subplot(133); plot_gradient_descent(theta, eta=0.5)
```



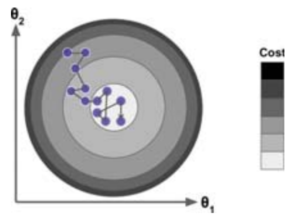
5.2.2 Stochastic Gradient Descent (隨機梯度下降)

- Batch $\nabla_\theta MSE(\theta) = \frac{2}{m}\sum_{i=1}^{m}\left(\theta^T x^{(i)} - y^{(i)}\right)x^{(i)}$ (向量)
- Stochastic: pick random i, $2\left(\theta^T x^{(i)} - y^{(i)}\right)x^{(i)}$ (向量)
  - picks a random instance (例子) in the training set (訓練集合) at every step and computes the gradients based only on (只根據) that single instance
  - much faster
  - very little data to manipulate (運用) at every iteration (迭代), possible to train on huge training sets

21

Behavior (行為)

- Stochastic Gradient (隨機梯度) much less regular (有規律的) than Batch (批量) Gradient:
  - the cost function will bounce up and down (上下), decreasing only on average.
- Over time it will end up (最終) very close to the minimum, but once it gets there it will continue to bounce around, never settling down (平靜下來) to optimal



22

Learning schedule (排程)

- determines the learning rate η (eta) at each iteration
  - If η is reduced too quickly, get stuck in (陷進去) a local minimum (區域極小)
  - If η is reduced too slowly, jump around (跳來跳去) the minimum for a long time and end up with (以結束) a suboptimal solution (次佳解)
  - *Hyperparameters* (超參數) t0 = 5, t1 = 50, $\eta(t) = \frac{5}{50+t}$, $\eta(0) = 1/10$, $\eta(100) = 5/150$, $\eta(t) \to 0$ as $t \to \infty$

http://4.bp.blogspot.com/-oOneMevpAOk/U8VffRGTXZI/AAAAAAAAAPA/c5Hu21GKJ9U/s1600/1521782_441053755995069_1226722732_n.jpg

23

Condition for convergence (收斂)

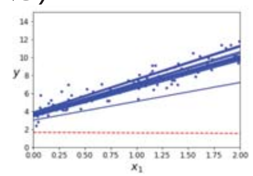
- $\sum_{t=1}^{\infty} \eta(t) = \infty$, $\sum_{t=1}^{\infty} \eta(t)^2 < \infty$, and more
  - Previous slide: $\eta(t) = \frac{5}{50+t}$ OK (by calculus series test)
- H. Robbins and S. Monro. A Stochastic Approximation (隨機逼近) Method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.
- L. Bottou, F. E. Curtis, J. Nocedal, Optimization Methods for Large-Scale Machine Learning, arXiv:1606.04838v3
- J.C. Spall, *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*, John Wiley & Sons, 2003.

24

5.2.3 Python (1)

```
• def learning_schedule(t):
•     return t0 / (t + t1)
• np.random.seed(139)
• theta = np.random.randn(2,1)
  – # random initialization (隨機初始化)
  – 斜率 -0.05880282，往下傾斜
• n_epochs = 50  # 學習次數
• for epoch in range(n_epochs):
•   for i in range(m): # 所有資料
•     if epoch == 0 and i < 20: # 前面 20 次
•        y_predict = X_new_b.dot(theta)
•        style = "b-" if i > 0 else "r--" #開始紅色
•        plt.plot(X_new, y_predict, style)
```

```
[[ 1.64772631]
 [-0.05880282]]
```



Python (2)

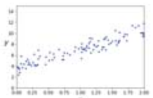
```
for i in range(m): # 所有資料 (previous slide)
    random_index = np.random.randint(m)
    # pick a random index
    xi = X_b[random_index:random_index+1]
    # pick a random one
    yi = y[random_index:random_index+1]
    gradients = 2 * xi.T.dot(xi.dot(theta) - yi)
    eta = learning_schedule(epoch * m + i)
    theta = theta - eta * gradients
    theta_path_sgd.append(theta)
numpy.linalg.norm(theta_best - theta ) / numpy.linalg.norm(theta_best)
0.0067064113837883976
```


SGD with Scikit-Learn (1)

```
theta_best        array([[3.74570384],
                         [3.31633466]])
from sklearn.linear_model import SGDRegressor
sgd_reg = SGDRegressor(max_iter = 50, penalty=None, eta0 = 0.1, random_state=42)
sgd_reg.fit(X, y.ravel())
theta_sgd = np.array(([sgd_reg.intercept_[0]], [sgd_reg.coef_[0]]))
              array([[3.74054316],
                     [3.28946307]])

numpy.linalg.norm(theta_best - theta_sgd ) / numpy.linalg.norm(theta_best)
0.0054694295318836305
```


SGD with Scikit-Learn (2)

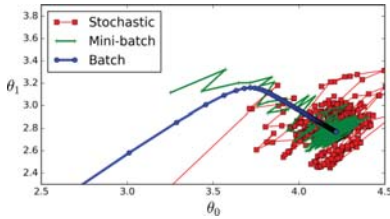
- `theta_best` `array([[3.74570384], [3.31633466]])`
- `sgd_reg = SGDRegressor(max_iter = 50, learning_rate = 'optimal')`
  - eta = 1.0 / (alpha * (t + t0)) where t0 is chosen by a heuristic proposed by Leon Bottou
  - alpha: Defaults (預設選項) to 0.0001



- `sgd_reg.intercept_, sgd_reg.coef_`
- `([-6.14453061e+11], [-1.82557376e+12]) # 發散`
- `max_iter = 5000`
- `3.5575439, 0.9535904`
- `sgd_reg1 = SGDRegressor(max_iter = 50, alpha = 0.1, learning_rate = 'optimal')`
- `(array([3.56415024]), array([0.84713983]))`
- `max_iter = 500: 3.56424868, 0.8440939`

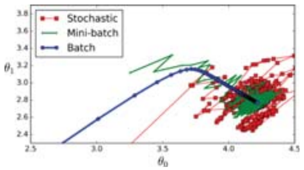
5.2.4 Mini-batch (小批量) Gradient Descent (梯度下降)

- Batch $\frac{2}{m}\sum_{i=1}^{m}\left(\theta^T x^{(i)} - y^{(i)}\right) x^{(i)}$ (向量)
- Stochastic: Pick random i, $2\left(\theta^T x^{(i)} - y^{(i)}\right)x^{(i)}$ (向量)
- Mini-batch: small random set S (<m) of instances $\frac{2}{S}\sum_{i\in S}\left(\theta^T x^{(i)} - y^{(i)}\right)x^{(i)}$ (向量)
- Use a lot in deep learning




```
t0, t1 = 10, 1000
def learning_schedule(t):
    return t0 / (t + t1)
```

Python




```
• n_iterations = 50
• minibatch_size = 20 # mini-batch S
• t = 0
• for epoch in range(n_iterations):
•     shuffled_indices = np.random.permutation(m) # 排列
•     # m: 訓練實例, 100
•     X_b_shuffled = X_b[shuffled_indices]
•     y_shuffled = y[shuffled_indices]
•     for i in range(0, m, minibatch_size): # i 位置
•     # 執行 100 / 20 = 5 次 , i: 0,20, 40, 60, 80
•         t += 1
•         xi = X_b_shuffled[i:i+minibatch_size]
•         yi = y_shuffled[i:i+minibatch_size]
•         gradients = 2 * xi.T.dot(xi.dot(theta_mb) - yi)
•         eta = learning_schedule(t)
•         theta_mb = theta_mb - eta * gradients
•         theta_path_mgd.append(theta_mb)
```

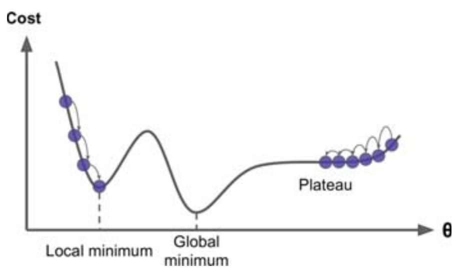

5.2.5 Comparison (比較) of algorithms

- m is the number of training instances (例子)
- n is the number of features (特徵)
- (*) learning rate, number of iterations
- (**) learning rates, (***) batch size

Algorithm	Large m	Large n	Hyperparameters	Scaling (縮放) required	Scikit-learn
Normal equation	Fast	Slow	0	No	LinearRegression
Batch	Slow	Fast	2 (*)	Yes	n/a
Stochastic	Fast	Fast	≥ 2 (**)	Yes	SGDRegressor
Mini-batch	Fast	Fast	≥ 2 (***)	Yes	n/a

5.2.6 Local (區域) and global (廣域) minimum (極小)

- If the random initialization (初始化) starts on the left, then it will converge to a *local minimum*. (> *global minimum*)
- If it starts on the right, then it will take a very long time to cross the plateau (台地), and if you stop too early you will never reach the global minimum.



Convex optimization (凸優化) and global minimum

- Objective: Minimize a convex (凸) function (函數)
  Constraints (限制式): Convex set (凸集合)
- Definition: A function f is convex if
  $f(\theta x_1 + (1-\theta)x_2) \leq \theta f(x_1) + (1-\theta)f(x_2)$
  – for all $x_1, x_2, 0 \leq \theta \leq 1$
  – 曲線上的點 ≤ 端點間所構成線段

- Theorem: A local optimal solution (區域最佳解) will also be a global (廣域) optimal solution.
- S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004. (applications, algorithms)

Symmetric (對稱) matrix

- Definition (定義): Matrix $A \in R^{m\times n}$ is symmetric if $A = A^T$
- $A = \begin{bmatrix}1\\2\end{bmatrix}, A^T = [1 \quad 2]$ Not symmetric
- $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}^T = \begin{bmatrix}1 & 3\\2 & 4\end{bmatrix}$ Not symmetric
- $\begin{bmatrix}1 & 2\\2 & 4\end{bmatrix}^T = \begin{bmatrix}1 & 2\\2 & 4\end{bmatrix}, a_{12} = a_{21}$
- Conclusion (結論): square and $a_{ij} = a_{ji}$

Convex functions (凸函數) (1)

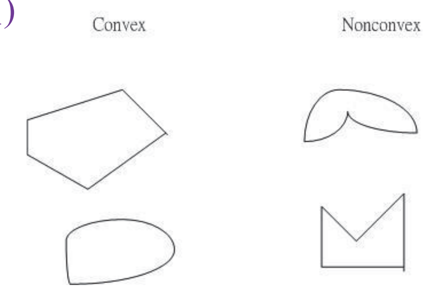
- 二次微分 (對稱) 矩陣 $\nabla^2 f(x) \equiv$ A，求 A 的特徵值 (eigenvalues) | A – λ I | = 0
- 定理： A 的特徵值 ≥ 0 (稱為半正定) ⇔ $f(x)$ convex
  - $f(x) = x^2, f'(x) = 2x, f''(x) = 2$
  - $f(x) = x_1^2 + x_2^2$, $\nabla^2 f(x) = \begin{bmatrix}2 & 0\\0 & 2\end{bmatrix}$, 特徵值 2,2 (python)
  - Positive semidefinite (半正定): $z^T A z \geq 0, \forall z$
- Ex: $z^T \begin{bmatrix}1 & 2\\4 & 6\end{bmatrix} z = z^T \begin{bmatrix}1 & 3\\3 & 6\end{bmatrix} z$

Convex functions (凸函數) (2): Least square

- $\nabla_\theta{}^2 MSE(\theta) = \frac{2}{m} X^T X$, dimension (維度) (n+1) × (n+1)
  - 對稱 $(X^T X)^T = X^T (X^T)^T = X^T X$
  - Positive semidefinite (半正定) $z^T (\frac{2}{m} X^T X) z = \frac{2}{m} z^T X^T X z = \frac{2}{m} (Xz)^T (Xz) \geq 0$
  - In general, AB ≠ BA. But 2 / m scalar OK
- Sufficient condition (充分條件) to be invertible (可逆): $X \in R^{m\times(n+1)}$
  - (Full) Rank(X) = min(m, n+1)
  - Rank: Number of linear independent (線性獨立的) vectors

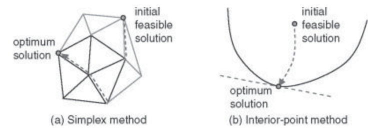
Constraints (限制式): Convex Set (凸集合)

- Definition: A set S is convex if $\theta x + (1-\theta) y \in S$, for all $x, y \in S, 0 \leq \theta \leq 1$ (Greek theta)
- Non-convex: [1,2] ∪ [4,5].
  - Why difficult? Need to check every region
- 2 dimensions (維度)



Interior-point algorithms (內點演算法)

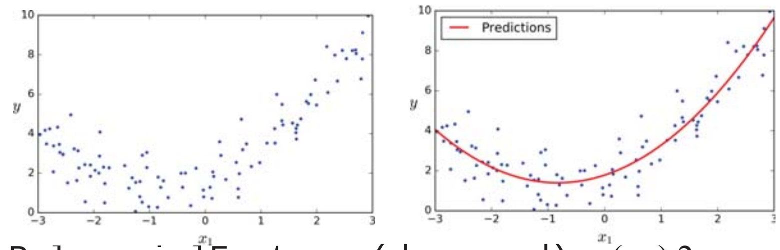
- 1947, George Dantzig, Simplex algorithm (單形法)
  - Worst case time complexity (複雜度) : Exponential (指數)
- 1984, Narendra Karmarkar, linear programming problems in polynomial time (多項式時間)
  - Many different algorithms: (Soviet) Dikin in 1967
- Yurii Nesterov and Arkadi Nemirovski, 1994: for convex (凸) nonlinear programming (非線性規劃) problems
  - self-concordance (自我和諧) functions $|f'''(x)| \le 2f''(x)^{3/2}$
  - Polynomial-time complexity (複雜度) for nonlinear programming problem!



https://www.sciencedirect.com/topics/computer-science/interior-point

5.3 Polynomial regression (多項式迴歸)

- $m = 100$
- X = 6 * np.random.rand(m, 1) – 3
- Quadratic function $y = 0.5x^2 + x + 2$ + Gaussian noise
- the model estimates $\hat{y} = 0.56x^2 + 0.93x + 1.78$



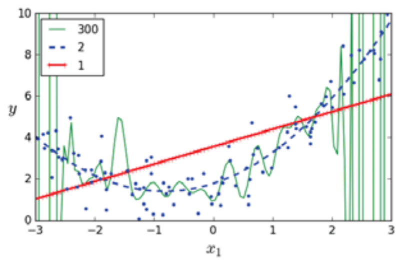
- `PolynomialFeatures(degree=d)`: (n=) 2 features a and b, degree = 3, total $(n + d)! / d!\, n!$ features (特徵) = 10, 常數 , $a, b, a^2, b^2, ab, a^3, a^2b, ab^2, b^3$

PolynomialFeatures (多項式特徵)

```
• from sklearn.preprocessing import
  PolynomialFeatures
• poly_features = PolynomialFeatures(degree=2,
  include_bias=False) # no bias column
• X_poly = poly_features.fit_transform(X)
• print(X.shape, X[0], X[0]**2)
• (100, 1) [-0.75275929] [0.56664654]
• print(X_poly.shape, X_poly[0])
• (100, 2)  [-0.75275929 0.56664654]
• lin_reg = LinearRegression()
• lin_reg.fit(X_poly, y)
• lin_reg.intercept_, lin_reg.coef_
• (array([1.78134581]), array([[0.93366893,
  0.56456263]])) # estimates ŷ = 0.56x² + 0.93x + 1.78
```

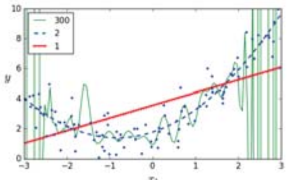

5.4 Learning curves (學習曲線)

- perform high-degree Polynomial Regression: overfit (過度配適)
- linear model is underfitting (低度擬合)
- The model that will generalize best: Quadratic model (二次模型)
- Problem: In general you won't know what function generated the data.



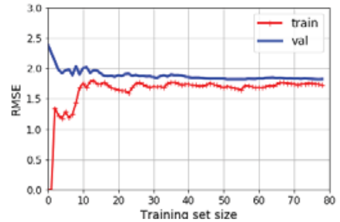
5.4.1 Python Transformation Pipeline (轉換工作流)

```
• from sklearn.preprocessing import StandardScaler
• from sklearn.pipeline import Pipeline
• for style, width, degree in (("g-", 1, 300), ("b--", 2,
  2), ("r-+", 2, 1)):
•     polybig_features = PolynomialFeatures(degree=degree)
•     std_scaler = StandardScaler()
•     lin_reg = LinearRegression()
•     polynomial_regression = Pipeline([
•             ("poly_features", polybig_features),
•             ("std_scaler", std_scaler),
•             ("lin_reg", lin_reg), ])
•     polynomial_regression.fit(X, y)
•     y_newbig = polynomial_regression.predict(X_new)
•     plt.plot(X_new, y_newbig, style, label=str(degree),
  linewidth=width)
```



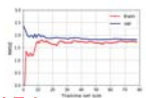
5.4.2 generalization performance of a model (模型推論性能)

- learning curves (學習曲線): simply train the model several times on different sized subsets (子集合) of the training set
- Linear Regression model (a straight line): one or two instances in the training set, the model can fit them perfectly, which is why the curve starts at zero.
  - Underfit: add more training examples will not help.
  - need to use a more complex model or better features



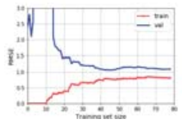
Python: Linear regression

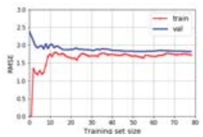
```
• def plot_learning_curves(model, X, y):
•     X_train, X_val, y_train, y_val = train_test_split
              (X, y, test_size=0.2, random_state=10)
•     train_errors, val_errors = [], []
•     for m in range(1, len(X_train)): # 取前段
•         model.fit(X_train[:m], y_train[:m])
•         y_train_predict =model.predict(X_train[:m])
•         y_val_predict = model.predict(X_val)
•         train_errors.append(mean_squared_error(
              y_train_predict, y_train[:m]))
•         val_errors.append(mean_squared_error(
              y_val_predict, y_val))
• print(train_errors_sqrt[75:])
• print(val_errors_sqrt[75:])
• [1.74667291 1.74445756 1.73538016 1.72683815]
• [1.8223893  1.81918743 1.81688302 1.81961098]
```



learning curves of a 10th-degree polynomial model (上圖)

- (上) The error on the training data is much lower than with the Linear Regression model (下).
- overfitting model
  - The model performs significantly better on the training data than on the validation data.
  - If a much larger training set, the two curves would continue to get closer.
- `print(train_errors_sqrt[76:])`
- `print(val_errors_sqrt[76:])`
- `[0.81384053 0.80899334 0.80769541]`
- `[1.08293063 1.08236688 1.09150266]`





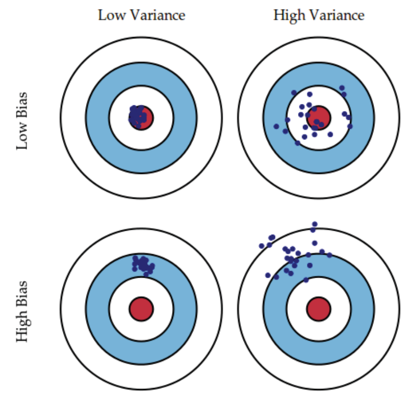
Comparison

- train_errors_sqrt[78], val_errors_sqrt[78]
- Linear: 1.72683815, 1.81961098]
- Quadratic: 0.8377079, 1.05919642
- 10^{th}: 0.80769541, 1.09150266

- abs(train_errors_sqrt[78] - val_errors_sqrt[78])/ train_errors_sqrt[78]
- Linear: 0.05372410103790786
- Quadratic: 0.2643982669004327
- 10^{th}: 0.35137906066527425

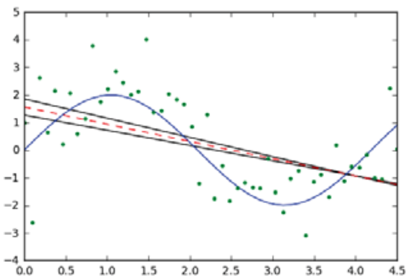
5.4.3 The Bias/Variance Tradeoff (偏誤及變異數之折衷)

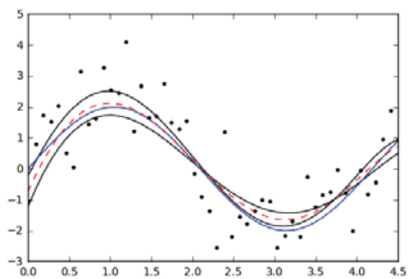
- http://scott.fortmann-roe.com/docs/BiasVariance.html
- the target (目標) (0,0)
- (1,1),(−1, −1). Average $\hat{h}$ =(0,0). $\left[Bias\left(\hat{h}(x_0)\right)\right]^2 = 0$, $Var\left(\hat{h}(x_0)\right) = \frac{1}{2}(2+2) = 2$
- (−4, −3),(0, 5). Average $\hat{h}$ =(−2,1). $\left[Bias\left(\hat{h}(x_0)\right)\right]^2 =$ $4+1, Var\left(\hat{h}(x_0)\right) = \frac{1}{2}[(4+$



Example of Expected MSE (1)

- https://gist.github.com/fagonzalezo/6819785
- https://scikit-learn.org/stable/modules/learning_curve.html
- (藍色) $y = 2\sin(1.5x)$ + standard normal distribution
- Fit a linear function and polynomial (degree 5) twice (黑色), average (紅色)

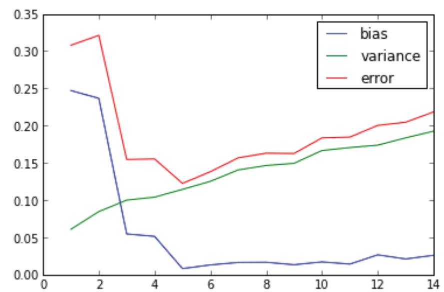




Example of Expected MSE (2)

$$E\left(y_0 - \hat{h}(x_0)\right)^2 = \left[Bias\left(\hat{h}(x_0)\right)\right]^2 + Var(\hat{h}(x_0)) + Var(\in)$$

- $y = 2\sin(1.5x)$ + standard normal distribution
- Pick 20 points, fit 100 models, then average
- 橫軸 多次多項式 (max = 15)



Expected mean square error (MSE)

$$E\left(y_0 - \hat{h}(x_0)\right)^2 = \left[Bias\left(\hat{h}(x_0)\right)\right]^2 + Var(\hat{h}(x_0)) + Var(\epsilon)$$

- Bias (偏誤): wrong assumptions, assume data is linear or quadratic
  - A high-bias model: most likely to underfit the training data.
- Variance (變異數): the model's excessive sensitivity to small variations
  - A model with many degrees of freedom (such as a high-degree polynomial (高階多項式) model) is likely to have high variance, overfit the training data.
- *Irreducible error* (無法降低錯誤): noisiness of the data itself.
  - way to reduce: Clean up the data (e.g., fix the data sources, such as broken sensors, or detect and remove outliers)

5.5 Regularization (正規化)

- Reduce overfitting (過度配適):
  - Regularize the model (i.e., to constrain (限制) it): the fewer degrees of freedom (自由度) it has, the harder it will be for it to overfit the data.
  - Regularize a polynomial model: Reduce the number of polynomial degrees (次數)
- For a linear model, regularization is typically achieved by constraining the weights (權重) of the model.
- Ridge Regression, Lasso Regression, and Elastic Net

5.5.1 Ridge Regression (脊迴歸)

- also called *Tikhonov regularization* (正規化)
  - Russian, 1906 - 1993
- $\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$
  - x_i: Feature, might be function of the other feature
- $J(\theta) = MSE(\theta) + \frac{\alpha}{2}\sum_{i=1}^{n} {\theta_i}^2$
  - If (hyperparameter 超參數) $\alpha = 0$, then Linear Regression
  - If α very large, then all weights very close to zero and the result is a flat line going through the data's mean.
- The regularization term (項目) should only be added to the cost function during training (訓練).
- Evaluate the model's performance using the unregularized performance measure.

Closed-form solution (閉合形式解答) (1)

- $J(\theta) = MSE(\theta) \implies (X^T X)\hat{\theta} = X^T y$
- $J(\theta) = MSE(\theta) + \frac{\alpha}{2}\sum_{i=1}^{n} {\theta_i}^2$
- $\frac{\partial}{\partial \theta_j}\frac{\alpha}{2}\sum_{i=1}^{n} {\theta_i}^2 = \alpha\theta_j, \frac{\partial}{\partial {\theta_j}^2} = \alpha, \frac{\partial}{\partial\theta_i\partial\theta_j} = 0, \forall i \neq j$
- The optimal solution (最佳解): $(X^T X + \alpha I)\hat{\theta} = X^T y$
  - I: identity (單位) matrix. If $n = 2$, $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
  - Slide 3: $X = \begin{bmatrix} 1 & x_{11} & x_{21} & x_{31} \\ \vdots & & \vdots & \\ 1 & x_{1m} & x_{2m} & x_{3m} \end{bmatrix} = [1 \quad \text{dataFrame}]$

53

Closed-form solution (閉合形式解答) (2)

- The optimal solution (最佳解): $(X^T X + \alpha I)\hat{\theta} = X^T y$
  - Def: positive-definite (正定) matrix: $z^T Az > 0, \forall z \neq \mathbf{0}$
  - Theorem: If A positive-definite, then A^{-1}exists.
  - $z^T(X^T X + \alpha I)z = z^T X^T Xz + \alpha z^T z \geq 0 + \alpha z^T z >$ $0 \Rightarrow$ Inverse of $(X^T X + \alpha I)$ exists
    - $z^T z$ >0: Ex $[-1 \quad 0]\begin{bmatrix}-1\\0\end{bmatrix} = 1 + 0$
  - Efficient implementation: Cholesky factorization (分解) for positive-definite $A \equiv X^T X + \alpha I = LL^T$
  - https://en.wikipedia.org/wiki/Cholesky_decomposition

54

5.5.2 Python: zip

- https://docs.python.org/3.3/library/functions.html#zip
- `x = [1, 2, 3]`
- `y = [4, 5, 6]`
- `zipped = zip(x, y)`
- `list(zipped)`
- `[(1, 4), (2, 5), (3, 6)]`

```
for alpha, style in zip(alphas, ("b-", "g--", "r:")):
  model = model_class(alpha, **model_kargs)
          if alpha > 0 else LinearRegression()
```

- alphas=(0, 10, 100): parameters for regularization (正規化參數)
- `**model_kargs`: accept an arbitrary number of arguments (任意數量的引數), e.g., tol=1, random_state=42

55

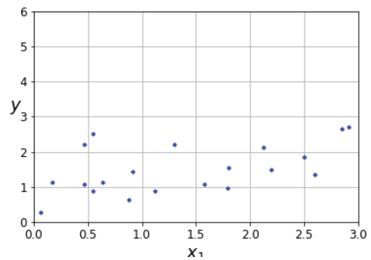
Python

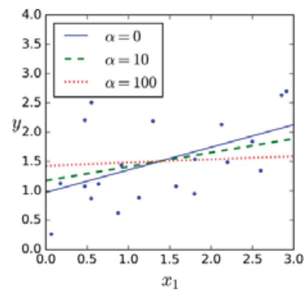
```
def plot_model(model_class, polynomial, alphas,
**model_kargs): # model_class: Ridge, Lasso
   for alpha, style in zip(alphas, ("b-", "g--", "r:")):
       model = model_class(alpha, **model_kargs)
               if alpha > 0 else LinearRegression()
       if polynomial:
               model = Pipeline([("poly_features",
                      PolynomialFeatures(degree=10,
                      include_bias=False)),
                      ("std_scaler", StandardScaler()),
                       ("regul_reg", model),])
        model.fit(X, y)
        y_new_regul = model.predict(X_new)
        lw = 2 if alpha > 0 else 1
        plt.plot(X_new, y_new_regul, style, linewidth=lw,
                label=r"$\alpha = {}$".format(alpha))
```

56

5.5.3 Example of Ridge Regression (脊迴歸)

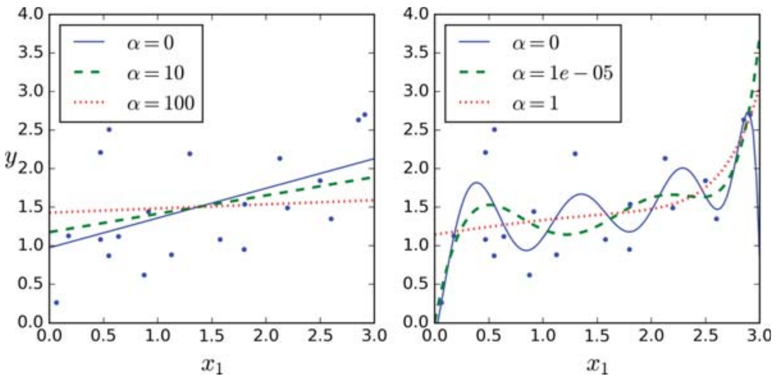
- `m = 20`
- `np.random.seed(42)`
- `X = 3 * np.random.rand(m, 1)`
- `y = 1 + 0.5 * X + np.random.randn(m, 1) / 1.5`
- `plot_model(Ridge, polynomial=False, alphas=(0, 10, 100), random_state=42)`





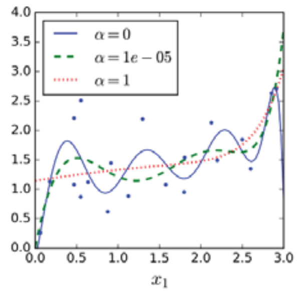
Example

- Left: Linear predictions. Right: polynomial degree 10
- increasing α leads to flatter (較平坦的) (i.e., less extreme, more reasonable (合理)) predictions; this reduces the model's variance but increases its bias



Coefficient of polynomial degree 10

- `lin_reg = LinearRegression()`
- `lin_reg.intercept_, lin_reg.coef_`
- `(array([-0.19742123]),`
- `array([[   7.15337503,    9.2726931 , -34.92184598,  -85.8481434 , 336.97477809, -423.07867457,  271.03883476,  -95.65944018, 17.72280598,   -1.34881442]]))`
- `lin_reg = Ridge(1)`
- `(array([1.13459578]),`
- `array([[ 0.26161686,  0.05341538, -0.04483643, -0.06758984, -0.04696913,-0.00157853, 0.03752267,  0.02210197, -0.02438387,  0.00464292]]))`



5.5.4 Lasso Regression (迴歸)

- Robert Tibshirani, 1996, pronounces it as "LAS-so"
- Least Absolute Shrinkage and Selection Operator (最小絕對緊縮與選擇算子) Regression
- Lasso: $J(\theta) = MSE(\theta) + \alpha \sum_{i=1}^{n} |\theta_i|$, $\ell 1$ norm
  - Ex: $\|[-1 \quad 0 \quad 2]\| = 1 + 0 + 2$
  - Popular in image processing applications
- Ridge Regression: $J(\theta) = MSE(\theta) + \frac{\alpha}{2} \sum_{i=1}^{n} {\theta_i}^2$, $\ell 2$ norm

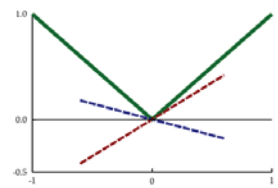
Subgradient (次梯度) for Gradient Descent (梯度下降法)

- Lasso $\sum_{i=1}^{n} |\theta_i|$ not differentiable (可微分) at $\theta_i = 0$
- Lasso Regression subgradient vector (向量)

$$g(\theta, J) = \nabla_\theta MSE(\theta) + \alpha \begin{pmatrix} sign(\theta_1) \\ sign(\theta_2) \\ \vdots \\ sign(\theta_n) \end{pmatrix},$$

$$sign(\theta_i) = \begin{cases} -1 & \text{if } \theta_i < 0 \\ 0 & \text{if } \theta_i = 0 \\ +1 & \text{if } \theta_i > 0 \end{cases}$$

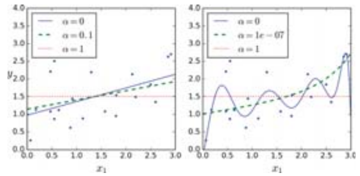
- 中間的 0 可以使用 [−1, 1] 間的數字



https://www.stats.ox.ac.uk/~lienart/_figs/ex_subgrad_plot1_g.png

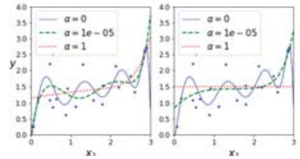
5.5.5 Lasso Regression Example

- Left: Linear predictions.
- Right: polynomial degree 10
  - With $\alpha = 10^{-7}$: looks quadratic
  - With $\alpha = 1$: almost linear: all the weights for the high-degree polynomial features = 0
- Lasso Regression automatically performs feature selection (特徵選擇) and outputs a *sparse* (稀疏的) *model* (i.e., with few nonzero (非 0) feature weights).



Comparison of green lines

```
• lin_reg = Ridge(10**-5) # left
• (array([1.49749138]),
• array([[ 1.39369937,  1.50417034, -0.65117538, -
  0.76882359,  0.28687638,0.25429796, -0.04664781, -
  0.0353996 ,  0.00248975,  0.00172478]]))
• lin_reg = Lasso(10**-5)
• (array([1.64723684]),
• array([ 9.37548949e-01,  7.10847597e-01,
  2.67625703e-02, -1.48714345e-02,-1.30853780e-03, -
  8.36937239e-04, -8.54116985e-04, -2.06159135e-04,
  5.82120123e-05,  3.17487568e-05]))
```



5.5.6 Elastic Net (彈性網)

- $J(\theta) = MSE(\theta) + r\alpha \sum_{i=1}^{n} |\theta_i| + \frac{1-r}{2} \alpha \sum_{i=1}^{n} {\theta_i}^2$
  - If r = 0, Ridge Regression (脊迴歸)
  - If r = 1, Lasso Regression
- In general, Elastic Net is preferred over Lasso since Lasso may behave erratically when the number of features (n) is greater than the number of training instances (m) or when several features are strongly correlated (強相關).
  - The Elements of Statistical Learning, chapter 18 (genomics (基因組學) and other areas of computational biology.)

5.5.7 Optimization (最佳化)

- D. Bertsimas, A. King and R. Mazumder, Best Subset Selection (最佳子集合選擇) via a Modern Optimization Lens, *Annals of Statistics*, 2016.
- Mixed Integer (混合整數) Optimization (MIO)
- solves problems with m in the 1000s and n in the 100s in minutes to provable optimality (可證明的最佳化)
- We also establish via numerical experiments (數值實驗) that the MIO approach performs better than Lasso and other popularly used sparse learning procedures (稀疏的學習程序), in terms of achieving sparse solutions (稀疏解) with good predictive power (預測能力).

5.6 Multi Asset Trend Following Strategy (多資產趨勢跟踪策略) by J.P. Morgan (摩根大通)

- J.P. Morgan Global Quantitative (定量) & Derivatives (衍生性金融商品) Strategy Team, Big Data and AI Strategies - Machine Learning and Alternative Data Approach to Investing, May 29, 2017
- predict the returns of 4 assets: S&P (標準普爾) 500, 7-10Y Treasury Bond Index (國債指數) (美國公債 IEF), US dollar (DXY 美元指數) and Gold
- For predictor variables (預測變量), we choose lagged (滯後) 1M, 3M, 6M and 12M returns (回報) of these same 4 assets, yielding a total of 16 variables

Multi Asset Trend Following Strategy (多資產趨勢跟踪策略)

- To calibrate the model (校準模型), we used a rolling window (滾動窗口) of 500 trading days (~2y); re-calibration was performed once every 3 months.
- The model was used to predict the next day's return.
- If the next day predicted return was positive, we went long the asset (長期資產) (看多，代表買), otherwise we shorted (看空，代表賣) it.
- Prior to regression, all inputs were standardized (標準化) to avoid the problem of input features (輸入特徵) being of different scales.

Performance Analytics (績效分析) (1)

- S&P 500: Correlation 41.3%

	Annualized Return (年度回報) (%)	Sharpe Ratio (夏普比)
S&P 500	7.52	0.36
S&P 500- Lasso	8.92	0.42

- https://www.investopedia.com/terms/s/sharperatio.asp
- Sharpe Ratio = $\frac{R_p - R_f}{\sigma_p}$
  - R_p: Return of portfolio (投資組合的回報)
  - R_f: Risk-Free rate (無風險利率)
  - σ_p: Standard deviation (標準差) of portfolio's excess return (超額收益)

Performance Analytics (績效分析) (2)

- Result for IEF (美國公債) - Lasso (α = 0.001)
- Result for DXY (美元指數) - Lasso (α = 0.05)
- Result for GLD (SPDR 黃金 ETF) - Lasso (α = 0.05)
- Exchange Traded Fund (ETF，交易所交易基金)

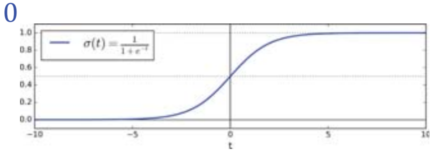
	Annualized Return (年度回報) (%)	Sharpe Ratio (夏普比)
IEF	2.30	0.32
IEF - Lasso	4.86	0.67
DXY	3.22	0.38
DXY - Lasso	4.20	0.49
Gold	6.12	0.31
Gold – Lasso	9.49	0.48

Lecture 6 Logistic Regression (邏輯斯迴歸)

- 6.1 Logistic Function (邏輯斯函數)
- 6.2 Cost Function and Training (成本函數與訓練)
- 6.3 Iris (鳶ㄩㄢ尾花) dataset
- 6.4 Softmax Regression
- Geron, Hands-On Machine Learning With Scikit-Learn and Tensorflow
- 長榮大學資設院資管系許志華

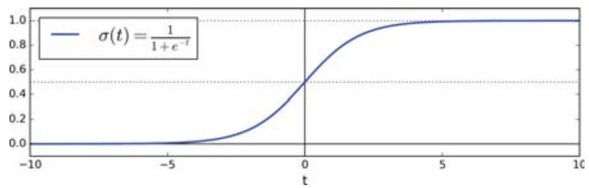
6.1 Logistic Function (邏輯斯函數)

- (sigma) $\sigma(t) = \frac{1}{1+\exp(-t)} = \frac{\exp(t)}{\exp(t)+1}$, $\exp(t) = e^t$
  - Euler number $e = \lim_{n\to\infty}(1 + 1/n)^n \approx 2.718 \ldots$
- Logistic function $\hat{p} = h_\theta(x) = \sigma(\theta^T x) = \frac{\exp(\theta^T x)}{\exp(\theta^T x)+1}$
- Model prediction (模型預測): Binary classifier (二元分類器)
  - $\hat{y} = \begin{cases} 0 & \text{if } \hat{p} < 0.5 \text{ or } \theta^T x < 0 \\ 1 & \text{if } \hat{p} \geq 0.5 \text{ or } \theta^T x \geq 0 \end{cases}$
  - Linear decision function! (線性決策)



Logit (邏輯) $\ln\frac{\hat{p}}{1-\hat{p}}$ of $\hat{p}$

- $0 \leq \hat{p} = \frac{\exp(\theta^T x)}{\exp(\theta^T x)+1} \leq 1$
- $\hat{p}(\exp(\theta^T x) + 1) = \exp(\theta^T x) \Rightarrow \exp(\theta^T x)(1-\hat{p}) = \hat{p} \Rightarrow \exp(\theta^T x) = \frac{\hat{p}}{1-\hat{p}} \Rightarrow \theta^T x = \ln\frac{\hat{p}}{1-\hat{p}}$
  - Probability (機率) of success (成功) p, probability of failure (失敗) $1-p$, odds of success (勝算) $p/(1-p)$
  - 多變量迴歸 $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 = \theta^T x = \hat{y}$



6.1.1 Applications (應用)

- profit (獲利) or loss (虧損)
- 心室性震顫所引起的心跳停止：生或死
- Customer churn (客戶流失)
  - customers that churned (i.e. left the company, value 1), did not churn (value 0)
  - Use possible predictor variables (預測變量) for churning behavior
- Paul Kvam and Joel S. Sokol, A logistic regression /Markov chain (馬可夫鏈) model for NCAA basketball (籃球), *Naval Research Logistics*, 2006

5

線上廣告

- R. Mookerjee, et al., To Show or Not Show, *Interfaces*, 2012, 42:449-464.
- http://chhsu135.blogspot.tw/2013/03/blog-post_6.html
- 利用瀏覽者使用的作業系統、瀏覽的時間點、廣告的大小位置等五十幾個變數，估計瀏覽者點閱廣告的機率 (logit)；使用卡方檢定 (chi-square test) 得知其點閱的機率是貝他分佈 (beta distribution)
- 作業研究設定臨界值 (threshold value) alpha，如果機率大於此臨界值，則提供廣告給瀏覽者看
  – Dynamic programming (動態規劃)
- 增加每天三千美元的收入 (一年約三千萬，1:30)

6

6.1.2 Framingham Heart Study (心臟研究) (1948)

- D. Bertsimas, et al., *The Analytics Edge*, 2016.
- Franklin Delano Roosevelt (FDR,羅斯福): Died while president (總統), April 12, 1945
- 5,209 patients aged 30-59 enrolled (參加)
- Patients given questionnaire (問卷) and exam every 2 years
  – Physical characteristics (身體特徵)
  – Behavioral (行為的) characteristics
  – Test results
- Exams and questions expanded over time

7

Coronary Heart Disease (CHD) (冠狀動脈性心臟病) (1)

- 7.3 million people died from CHD in 2008
- Framingham Heart Study (1948)
- Risk Factors (風險因素) by Logistic regression (部分)

$$logit(CHD) = -7.7013 + 0.0524(Age) + 0.6555(Male) + 0.0205(SystolicBP) + 0.6723(Diabetes) + 0.2991(smoker)$$

  – 特徵 (feature) x：年紀，男性，收縮壓，糖尿病，抽煙者
  – 參數 (parameter) θ_i：–7.7013, 0.0524, …
  – $logit \equiv \theta^T x = \log\left(\frac{p}{1-p}\right) \Rightarrow \hat{p} = \frac{\exp(\theta^T x)}{\exp(\theta^T x)+1}$
  – 65, male (1), 145, dia (1), not (0) ⟹ logit = 0.005 ⟹ probability (機率) of CHD = 0.501 (risk of 10-year period)

http://tupian.baike.com/a2_57_80_01300000162256121239809536613_jpg.html

8

Coronary Heart Disease (CHD) (冠狀動脈性心臟病) (2)

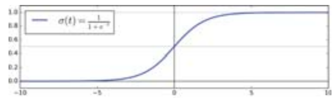
$$logit(CHD) = -7.7013 + 0.0524(年紀) + 0.6555(男性) + 0.0205(收縮壓) + 0.6723(糖尿病) + 0.2991(抽煙者)$$

- 年紀 65，收縮壓 145，糖尿病，不抽煙
  – 男 logit = 0.005 ⟹ 機率 $\hat{p} = \frac{\exp(logit)}{\exp(logit)+1} \approx 0.501$
  – 女 logit = –0.650 ⟹ 機率 $\hat{p} \approx 0.343$
  – 增加 (0.501 – 0.343) / 0.343 ≈ 0.462
- 年紀 65，男性，收縮壓 145，糖尿病
  – 抽煙 logit = 0.304 ⟹ 機率 $\hat{p} \approx 0.575$
  – 增加 (0.575 – 0.501) / 0.501 ≈ 0.148

輸入與輸出變數的種類與範圍

$logit(CHD) = -7.7013 + 0.0524(\text{年紀}) + 0.6555(\text{男性})$
$+0.0205(\text{收縮壓}) + 0.6723(\text{糖尿病}) + 0.2991(\text{抽煙者})$

- 特徵 (feature)：
  - — 年紀 (0 到 122) (wiki Maximum life span 最長壽命)
  - — 收縮壓 (90 到 200)
  - — 男性，糖尿病，抽煙者：1 或 0
    - ✓ 沒有分等級或抽煙的支數
- 邏輯斯迴歸輸出 (Output): [0, 1]



  - — 線性迴歸：正負值，基本上沒有範圍限制

6.2 Cost Function and Training (成本函數與訓練)

- Minimize Cost Function (最小化成本函數)

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(\hat{p}^{(i)}\right) + \left(1 - y^{(i)}\right)\log\left(1 - \hat{p}^{(i)}\right)\right]$$

  - – natural logarithm (自然對數) with base (底數) e
  - – for classification (分類) $y^{(i)}$
- $\hat{p} = \sigma(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)} = \frac{\exp(\theta^T x)}{\exp(\theta^T x)+1} \in [0, 1], y_i = 0/1$
  - – 例子：$\theta^T x = -7.7013 + 0.0524(\text{年紀}) + 0.6555(\text{男性})$ $+0.0205(\text{收縮壓}) + 0.6723(\text{糖尿病}) + 0.2991(\text{抽煙者})$
- If $y^{(i)} = 1$, then [...] $= \log\left(\hat{p}^{(i)}\right)$
- If $y^{(i)} = 0$, then [...] $= \log\left(1 - \hat{p}^{(i)}\right)$

Minimize Cost Function (最小化成本函數)

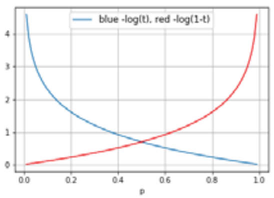
$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(\hat{p}^{(i)}\right) + \left(1 - y^{(i)}\right)\log\left(1 - \hat{p}^{(i)}\right)\right]$$

– for classification (分類) $y^{(i)} \in \{0,1\}, \hat{p} \in [0, 1]$

- If $m = 1$, $y_i = 1 \Rightarrow J(\theta) = -\log\left(\hat{p}^{(i)}\right)$ (similar for $y_i = 0$)

$\hat{p}^{(i)}$	`array([[0.55      , 0.65      , 0.75      , 0.85      , 0.95      ],`
$J(\theta)$	`[0.597837  , 0.43078292, 0.28768207, 0.16251893, 0.05129329]])`

– It penalizes the model (懲罰模型) when it estimates (估計) a low probability (概率低) for a target class (目標類別)



6.2.1 Training (訓練)

- Minimize Cost Function (for classification (分類))

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(\hat{p}^{(i)}\right) + \left(1 - y^{(i)}\right)\log\left(1 - \hat{p}^{(i)}\right)\right]$$

- $\hat{p} = \sigma(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)} = \frac{\exp(\theta^T x)}{\exp(\theta^T x)+1}, y_i = 0/1$
- By calculus (微積分): Column vector (行向量) x

$$\nabla_\theta J(\theta) = -\frac{1}{m}y^{(i)}x + \frac{1}{m}\frac{\exp(\theta^T x)}{\exp(\theta^T x) + 1}x$$

  - – No known closed-form equation (解析解) to compute the value of θ that minimizes (最小化) this cost function ($\nabla_\theta J(\theta) = 0$)

Training (訓練)

- $J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(\hat{p}^{(i)}) + (1-y^{(i)})\log(1-\hat{p}^{(i)})]$
- $\hat{p} = \sigma(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)} = \frac{\exp(\theta^T x)}{\exp(\theta^T x)+1}, y_i = 0/1$
- this cost function is convex
$$\frac{\partial^2 f(\theta)}{\partial \theta^2} = \frac{1}{m}\frac{\exp(\theta^T x)}{(1+\exp(\theta^T x))^2} x^T x$$
$-x^T x \in R^{n\times n}$: Positive semidefinite (半正定)
- so Gradient Descent is guaranteed (保證) to find the global minimum (廣域極小)
$$\theta^{(next\ step)} = \theta - \eta \nabla_\theta J(\theta) \ (\eta \text{ eta})$$

6.2.2 Statistical Inference (統計推論)

- Ref: J. Ledolter, *Data Mining and Business Analytics with R*
- Consider a single regressor (迴歸自變數) variable x.
- Assume n pairs of observations (觀察): x_i and the success (成功) indicator (指示) $y_i = 0/1$
- Bernoulli (伯努利) model: Outcome (結果) of case i is either 1 or 0 with probabilities $p_i = \frac{exp(\alpha+\beta x_i)}{1+exp(\alpha+\beta x_i)}$
- parameter estimation (參數估計): α (alpha) and β (beta)
- In general, $\theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$, 參數 θ_i

Likelihood function (似然函數)

$$\text{Max} \prod_{i=1}^{n} p(y_i|x_i) = \prod_{i=1}^{n}(p_i)^{y_i}(1-p_i)^{1-y_i}$$
$$= \prod_{i=1}^{n}\left[\frac{exp(\alpha+\beta x_i)}{1+exp(\alpha+\beta x_i)}\right]^{y_i}\left[\frac{1}{1+exp(\alpha+\beta x_i)}\right]^{1-y_i}$$

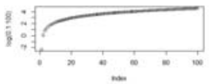
- Find α and β to maximize the product (乘積) function
- If $y_i = 1$, then $(p_i)^{y_i}(1-p_i)^{1-y_i} = p_i$
- If $y_i = 0$, then $(p_i)^{y_i}(1-p_i)^{1-y_i} = 1 - p_i$
- If $y_1 = 1$ and $y_2 = 0$, then $\prod_{i=1}^{2} p(y_i|x_i) = p_1\,(1-p_2)$

Maximum likelihood estimation (最大似然估計) (1)

- Find α and β to maximize the product (乘積) function
$$\text{Max} \prod_{i=1}^{n} p(y_i|x_i) = \prod_{i=1}^{n}(p_i)^{y_i}(1-p_i)^{1-y_i}$$
$$= \prod_{i=1}^{n}\left[\frac{exp(\alpha+\beta x_i)}{1+exp(\alpha+\beta x_i)}\right]^{y_i}\left[\frac{1}{1+exp(\alpha+\beta x_i)}\right]^{1-y_i}$$
- $\log(ab) = \log a + \log b, \log a^b = b\log a$
- $\log\left(\frac{b}{a}\right) = \log b - \log a$
$\Rightarrow \log(c^d f^g) = \log(c^d) + \log(f^g) = d\log c + g\log f$

Maximum likelihood estimation (最大似然 (概似) 估計) (2)

- Log(x)：遞增函數

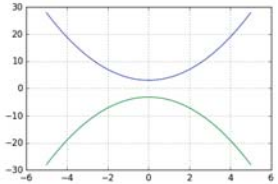


$$\text{Max} \prod_{i=1}^{n} p(y_i|x_i) = \prod_{i=1}^{n} (p_i)^{y_i}(1-p_i)^{1-y_i}$$

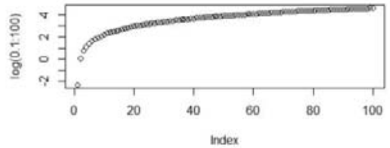
$$\underbrace{\Leftrightarrow}_{\log} \max \sum [y_i \log(p_i) + (1-y_i)\log(1-p_i)]$$

$$\Leftrightarrow \min -\left[\sum_{i=1}^{n} y_i \log(p_i) + \sum_{i=1}^{n} (1-y_i)\log(1-p_i)\right] \equiv D$$

- D: Deviance (偏異值)
  - Entropy (熵ㄉㄧ)



Log(x)：遞增函數



$$\text{Max} \prod_{i=1}^{n} p(y_i|x_i) = \prod_{i=1}^{n} (p_i)^{y_i}(1-p_i)^{1-y_i}$$

$$\underbrace{\Leftrightarrow}_{\log} \max \sum [y_i \log(p_i) + (1-y_i)\log(1-p_i)]$$

- Increasing (遞增): If $x_1 > x_2$, then $\log x_1 > \log x_2$.
- Max x^*: $f(x^*) > f(x), \forall x \Rightarrow \log f(x^*) > \log f(x)$

6.2.3 Entropy (熵ㄉㄧ)

- originated from information theory (資訊理論) by Claude E. Shannon in 1948
- How to efficiently transmit information (有效傳遞資訊)
- Entropy: $-\sum_x p(x)\log_2 p(x)$
  - Why add –: $p(x) \le 1 \Rightarrow \log_2 p(x) \le 0 \Rightarrow -\log_2 p(x) \ge 0$
- Weather
  - always sunny p = 1, entropy = $-\log 1 = 0$. Certainty (確定)
  - Sunny (晴朗) 0.5, rainy (多雨的) 0.5, entropy = 1. One bit to transmit. For example, sunny 1 and rainy 0

http://static2.businessinsider.com/image/597a478fb50ab12a008b4fc7-824/screen%20shot%202017-07-27%20at%2035625%20pm.png

Cross Entropy (交叉熵) (1)

- Cost Function

$$\min -\left[\sum_{i=1}^{n} y_i \log(p_i) + \sum_{i=1}^{n} (1-y_i)\log(1-p_i)\right]$$

- Cross entropy: If guess p as q, $H(p,q) = -\sum_x p(x)\log q(x)$
- Kullback–Leibler divergence = Cross entropy – Entropy

21

Cross Entropy (交叉熵) (2)

	A	B	C	D
P	0.4	0.1	0.25	0.25
M1	0.25	0.25	0.25	0.25
M2	0.4	0.1	0.1	0.4

- Entropy $H(p) = -\sum_x p(x) \log_2 p(x)$ =1.86
- Cross-entropy $H(p, M1) = -\sum_x p(x) \log q(x) = 2$
  - Uniform distribution (均匀分佈): typically use when we have no information about the behavior of X
- Cross-entropy $H(p, M2) = -\sum_x p(x) \log q(x) = 2.02$
  - Guessing after seeing
- M1 is a slightly better model of X than M2
- http://www.cs.rochester.edu/u/james/CSC248/Lec6.pdf

22

6.3 Iris (鳶ㄩㄢ尾花) dataset

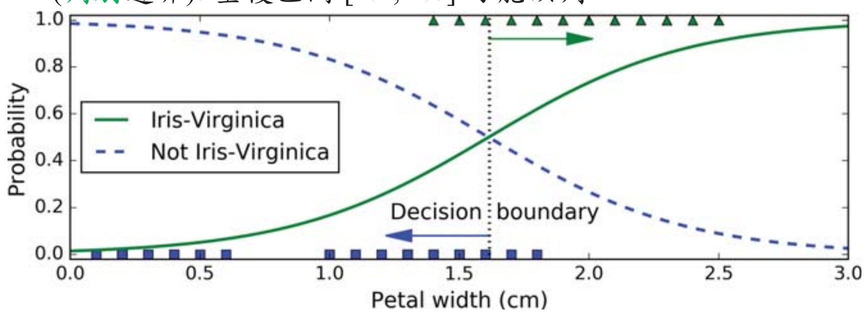
- sepal (花萼) and petal (花瓣) length (長度) and width (寬度) of 150 iris flowers of three different species (種)
  - classification (分類): use 4 features (特徵) (單位公分) to predict class
  - 如果有類別特徵，處理方法如前
  - 監督式學習 (supervised learning)：(3) 類別已知

```
                Min  Max   Mean    SD   Class Correlation
============== ==== ==== ======= ===== ====================
sepal length:   4.3  7.9   5.84   0.83    0.7826
sepal width:    2.0  4.4   3.05   0.43   -0.4194
petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)
petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)
```

23

A classifier based only on the petal width feature (花瓣寬度特徵)

- Why: Highest correlation
- Triangles (三角形): Petal width of Iris-Virginica flowers, 1.4 to 2.5 cm (Assume class 1 if Iris-Virginica, the others class 0)
- Squares: The other iris flowers, 0.1 to 1.8 cm.
- Estimated probabilities (估計的機率) and decision boundaries (判別邊界): 重複區間 [1.4, 1.8] 可能誤判



24

6.3.1 Python for class transformation (類別轉換)

```
from sklearn import datasets
iris = datasets.load_iris()
iris["target"] #目標
[0 … 0 1…1 2…2] # 50 each, 2 for Iris-Virginica
X = iris["data"][:, 3:]  # petal width (花瓣寬度)
y = (iris["target"] == 2).astype(np.int)
# iris["target"] == 2 是否是 Iris-Virginica
# 1 if Iris-Virginica, else 0
y
[0 … 0 1 .. 1] # 100s 0, 50s 1
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	class
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0

25

Python for Logistic Regression

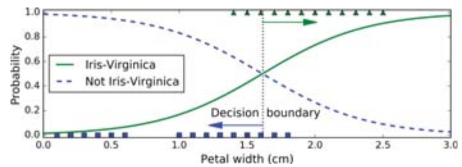
```
X_train, X_test, Y_train, Y_test =
train_test_split(X, y, test_size=0.20)
```

– sample size = 150, then training samples = 120
– Test samples (測試樣品) 30, error interval (錯誤區間) $1/30 \approx 3.33\%$

```
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression(random_state=42)
log_reg.fit(X_train, Y_train)

training score: 0.967 # 116/120, 116 correct, 4 errors
Testing score: 0.933 # 28/30, 28 correct, 2 errors
```

26

Example



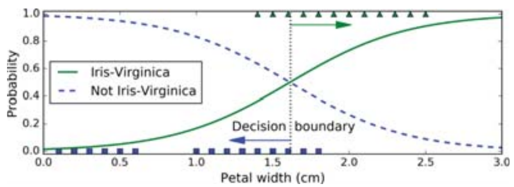
- Triangles (三角形): Petal width (花瓣寬度) of Iris-Virginica, 1.4 to 2.5 cm
- Squares: The other iris flowers, 0.1 to 1.8 cm.
- 重疊區域 [1.4, 1.8] 可能誤判
- Assume class 1 if Iris-Virginica, the others class 0
- Decision (決策) function $-3.88 + 2.40x_i$ (> 0 or <0) (page 2)

x_i	Decision function	Predicted class	Real class
1.6 (test)	–0.04	0 (正確)	0
1.8 (test)	0.44	1 (正確)	1
1.6 (test)	–0.04	0 (錯誤)	1
1.7 (train)	0.20	1 (錯誤)	0

27

Decision Boundary (決策邊界)

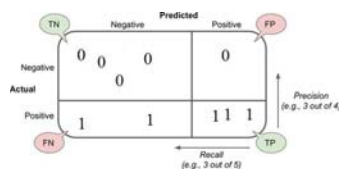
```
log_reg = LogisticRegression(random_state=42)
X_new = np.linspace(0, 3, 1000).reshape(-1, 1) #
# 取 1000 點。-1: 大小由矩陣決定，3×4 則為 12
y_proba = log_reg.predict_proba(X_new)
array([[0.97983051, 0.02016949], # 類別 0 和 1，x = 0
       [0.97968751, 0.02031249],
        …
       [0.02874159, 0.97125841]]) # x = 3
decision_boundary = X_new[ y_proba[:, 1] >= 0.5 ][0]
decision_boundary

array([ 1.61561562])
```



28

6.3.2 Using confusion matrices (混淆矩陣) to measure performance

Aurelien Geron



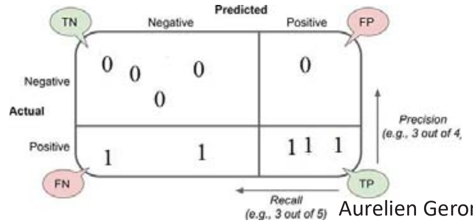
- (隨問題而變) array([4, 1], [2, 3]) (共 10)
- True negative (TN, 真陰性) 4
- False positive (FP 偽陽性) 1
- False negative (FN 偽陰性) 2
- True positive (TP, 真陽性) 3
- `from sklearn.metrics import confusion_matrix`
- `y_true = [0, 1, 0, 1, 1, 1, 0, 1, 0, 0]`
- `y_pred = [0, 0, 0, 1, 1, 0, 0, 1, 0, 1]`
- `confusion_matrix(y_true, y_pred)`

```
array([[4, 1],
       [2, 3]], dtype=int64)
```

confusion matrices (混淆矩陣)

- (隨問題而變) array([4, 1], [2, 3]) (共 10)
- True negative (TN, 真陰性) 4
- False positive (FP 偽陽性) 1
- False negative (FN 偽陰性) 2
- True positive (TP, 真陽性) 3

```
cm = confusion_matrix(y_true = Y_test, y_pred =
log_reg.predict(X_test))
```



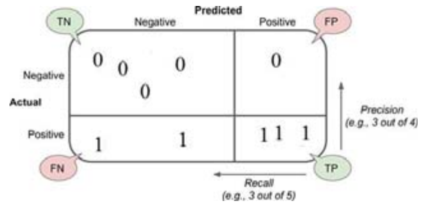
Aurelien Geron

Precision and Recall (查全率)

- (準確度) accuracy = $\frac{TP+TN}{TP+TN+FP+FN} = 0.7$
- (錯誤率) error rate = $\frac{FP+FN}{TP+TN+FP+FN} = 1 - accuracy$
- (查準率) precision = $\frac{TP}{TP+FP} = 0.75$ (猜對 1)
- (敏感度) sensitivity = recall $= \frac{TP}{TP+FN} = 0.6$ (找到)
- (特異性) specificity = $\frac{TN}{TN+FP} = 0.8$

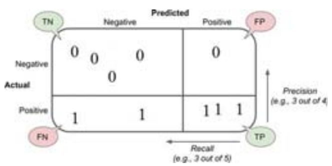
```
classification_report(Y_test,
log_reg.predict(X_test))
```

- Linear regression:
  mean square error
  (for numerical data)



F1 score

- (查準率) precision = $\frac{TP}{TP+FP}$ (猜對 1)
- (查全率) recall $= \frac{TP}{TP+FN}$ (找到)

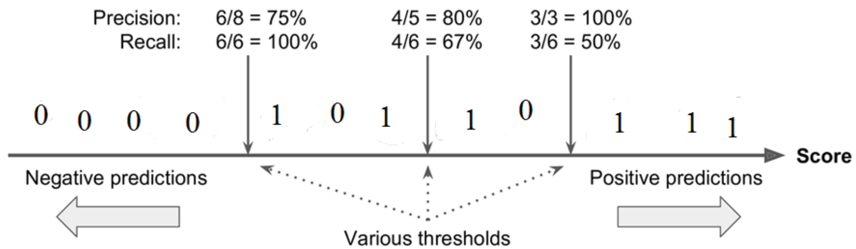


- *Harmonic mean* (調和平均數) $F_1 = \frac{2}{\frac{1}{precision}+\frac{1}{recall}}$
  - Both good ⇔ F1 good.
  - 單一指標，較容易比較不同的演算法

Precision	0.9	0.9	0.9	0.9	0.9
recall	0.9	0.7	0.5	0.3	0.1
F1	0.9	0.79	0.64	0.45	0.18
Regular mean	0.9	0.8	0.7	0.6	0.5

6.3.3 Precision/Recall Tradeoff (查準率與查全率的取捨)

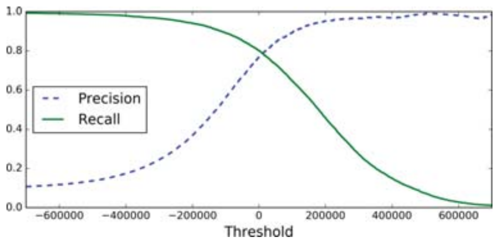
- For each instance (例子) x
  - compute a score = $\theta_1 x + \theta_0$ (*decision function*),
  - if score > a threshold (閾值), the instance positive class, or else to the negative.
- (查準率) p= $\frac{TP}{TP+FP}$(猜對 1), (查全率)r $= \frac{TP}{TP+FN}$ (找到)

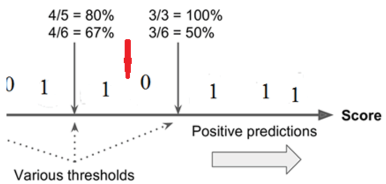


Aurelien Geron

Precision and recall versus the decision threshold (閾值)

- (查準率) p= $\frac{TP}{TP+FP}$(猜對 1), (查全率)r $= \frac{TP}{TP+FN}$(找到)
- precision curve bumpier: start from the central threshold and move it just one digit to the right: precision goes from 4/5 (80%) down to ¾ (75%).



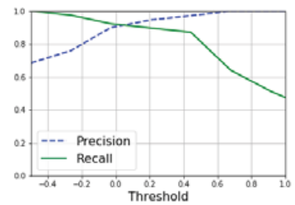


Precision versus recall: Python for Iris

```
from sklearn.metrics import precision_recall_curve
log_reg.fit(X_train, Y_train)
Y_scores = log_reg.decision_function(X_train)

precisions, recalls, thresholds =
precision_recall_curve(Y_train, Y_scores)
```

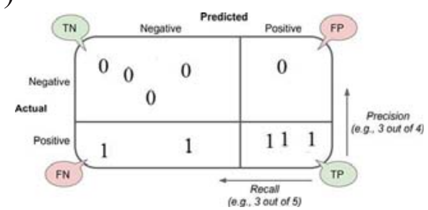
- 90% precision (查準率), then threshold -0.02841664, recall (查全率) 0.92307692

```
(array([0.68421053, 0.76      , 0.9       , 0.94594595, 0.97142857,
        1.        , 1.        , 1.        , 1.        , 1.        ,
        1.        , 1.        , 1.        ]),
 array([1.        , 0.97435897, 0.92307692, 0.8974359 , 0.87179487,
        0.64102564, 0.51282051, 0.41025641, 0.28205128, 0.25641026,
        0.15384615, 0.07692308, 0.        ]),
 array([-0.49943374, -0.26392519, -0.02841664,  0.20709191,  0.44260047,
         0.67810902,  0.91361757,  1.14912612,  1.38463467,  1.62014322,
         1.85565177,  2.09116032]))
```



Tradeoff depends on your problem

- (查準率) precision = $\frac{TP}{TP+FP}$ = 0.75 (猜對 1)
- (敏感度) sensitivity = recall $= \frac{TP}{TP+FN}$ = 0.6 (找到)
- Video safe for kids
  - high precision (保留安全)
  - low recall (拒絕好)
- Detect shoplifters (偷竊商店)
  - high rec (根據錄影)
  - low pre

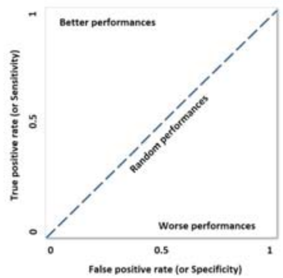


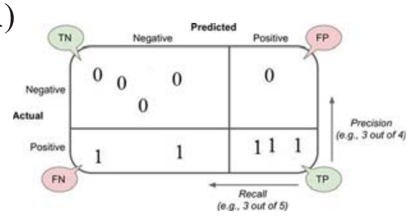
6.3.4 ROC curve (曲線)

- The Receiver Operating Characteristic (ROC) Curve (接收者操作特徵曲線) is a popular alternative (供選擇的東西) to the density plot.
  - First used during World War II for the analysis of radar (雷達) signals (信號)
  - Increase the prediction of correctly detected enemy aircraft (飛機) from their radar signals
  - Signal detection theory (信號偵測理論)
- Tom Fawcett, An introduction to ROC analysis, *Pattern Recognition Letters*, 27 (2006), 861–874

ROC Curve

- True positive rate (真陽性率) = $\frac{TP}{TP+FN}$ = Sensitivity (敏感度)
  - Proportion (比例) of 1s are correctly identified (正確鑑別)
- True negative rate (真陰性率) = $\frac{TN}{TN+FP}$ = Specificity (特異性)
  - Proportion of 0s are correctly classified (正確分類)

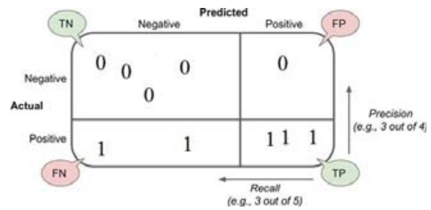


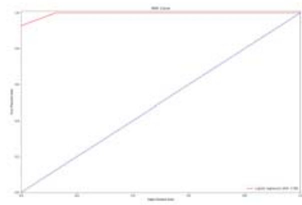


Tradeoff (取捨) under ROC

- Tradeoff: Higher TPR (recall (查全率)), then higher FPR
  - $TPR = \frac{TP}{TP+FN}, FPR = \frac{FP}{TN+FP} = 1 - \frac{TN}{TN+FP}$
- TPR = FPR = 0.5 ⇒ TP = FN, FP = TN
  - dotted line: a purely random (隨機) classifier
- a good classifier: Toward the top-left corner

```
fpr, tpr, thresholds = roc_curve(Y_test, Y_score)
```

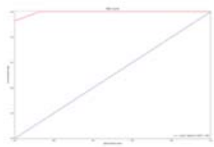


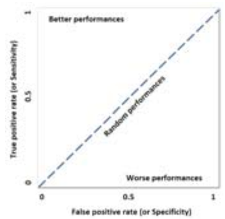


Area under the curve (AUC) (曲線下面積)

- value is bounded between 0 (worst performances 最差的性能) and 1 (best performances)
- perfectly random value (完全隨機值) 0.5

```
auc(fpr, tpr)
0.9955357142857143
```





- Another application: Framingham Risk Score (風險評分)
  - D. Bertsimas, et al., The Analytics Edge, 2016, sec 7.3

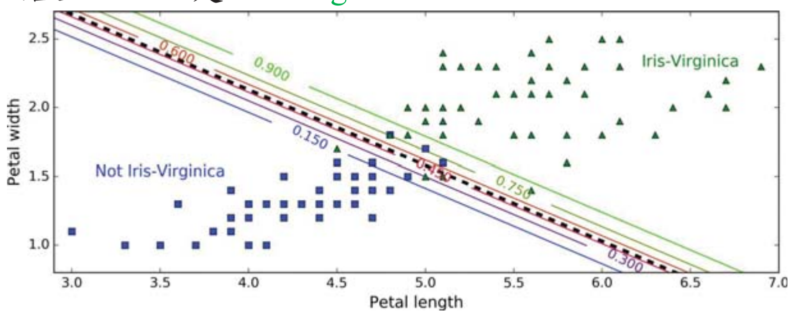
6.3.5 Python with 2 features

- `X = iris["data"][:, (2, 3)]`
- `#` petal width (花瓣寬度) and length (長度)
- `y = (iris["target"] == 2).astype(np.int)`
- `log_reg = LogisticRegression(C=10**10, random_state=42)`
- `# C` Inverse of regularization (正規化) strength ($1/\alpha$)
- `log_reg.fit(X_train, Y_train)`
- `print('Logistic regression score: %.3f' % log_reg.score(X_test, Y_test))`
- `Logistic regression score: 0.967 (> 0.933, 2 errors with petal width )`

Decision boundary (決策邊界)

```
X = iris["data"][:, (2, 3)]
```

- Linear decision boundary: $\theta_0 + \theta_1 x_1 + \theta_2 x_2 =$ threshold
- Dashed line (虛線): the model estimates a 50% probability
- 右上：90% 是 Iris-Virginica



6.4 Softmax Regression (迴歸)

- can be generalized to support multiple classes (多個類別), without having to train and combine multiple binary classifiers (組合多個二元分類器)
- Given an instance (例子) x
  - first compute $s_k(x) = {\theta_k}^T x$ for each class $k \in K$
    - 係數 θ_k 待定
  - then estimates the probability (估計機率) of each class by applying the *softmax function* (or *normalized exponential* (正規化指數))

$$\hat{p}_k = \frac{\exp(s_k(x))}{\sum_{j=1}^{K} \exp(s_j(x))}$$

  - Ex: $\exp(s_k(x))$ =[1, 2, 4], then $\hat{p} = \begin{bmatrix} \frac{1}{7} & \frac{2}{7} & \frac{4}{7} \end{bmatrix}$

Classifier prediction (分類器預測)

- softmax function $\hat{p}_k = \frac{\exp(s_k(x))}{\sum_{j=1}^{K} \exp(s_j(x))}$, $s_k(x) = {\theta_k}^T x$
- Ex:$s_k(x)$ =[0, 1, $\underline{2}$], then $\hat{p}$ = [0.090, 0.245, $\underline{0.665}$] $\Rightarrow$ Predicted class (預測類別) $\hat{y} = 3$
- classifier prediction (one class at a time)

$$\hat{y} = \underset{k}{\operatorname{argmax}} \frac{\exp(s_k(x))}{\sum_{j=1}^{K} \exp(s_j(x))}$$
$$= \underset{k}{\operatorname{argmax}} \exp\big(s_k(x)\big) = \underset{k}{\operatorname{argmax}}\, {\theta_k}^T x$$

  - 分母相同，exp(x) = e^x 遞增函數
  - $f(x) = -x^2 + 1,\ \max f(x) = 1 = f(0)$
    - $\underset{k}{\operatorname{argmax}} f(x) = 0$. arg: Argument (引數)
  - Choose $\underline{2}$ of $s_k(x)$, class 3, easier computation

Cost Function and Training (成本函數與訓練)

- Cross entropy cost $J(\Theta) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} {y_k}^{(i)} \log({\hat{p}_k}^{(i)})$
  - 參數矩陣 (Theta) $\Theta = \begin{bmatrix} {\theta_1}^T \\ \vdots \\ {\theta_K}^T \end{bmatrix} = \begin{bmatrix} \theta_{10} & \cdots & \theta_{1n} \\ & \vdots & \\ \theta_{K0} & \cdots & \theta_{Kn} \end{bmatrix}$

$$\sum_{k=1}^{K} {y_k}^{(i)} \log({\hat{p}_k}^{(i)}) = {y_1}^{(i)} \log({\hat{p}_1}^{(i)}) + \cdots + {y_K}^{(i)} \log({\hat{p}_K}^{(i)})$$

  - ${y_k}^{(i)} = 1$ if the target class (目標類別) for the ith instance (例子) x is k; otherwise, it is equal to 0
- Gradient (梯度) $\nabla_{\theta_k} J(\Theta) = \frac{1}{m} \sum_{i=1}^{m} \left({\hat{p}_k}^{(i)} - {y_k}^{(i)}\right) x^{(i)}$

45

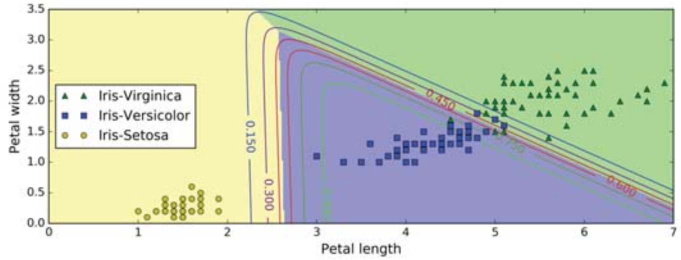
6.4.1 Three-class (三種類別) Iris

- `X = iris["data"][:, (2, 3)]`
- # petal width (花瓣寬度) and length (長度), 2 features
- `y = iris["target"]`
- `X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size=0.20, random_state = 42)`
- `softmax_reg = LogisticRegression (multi_class="multinomial",solver="lbfgs", C=10, random_state=42)`
- # C Inverse of regularization (正規化) strength ($1/\alpha$)
- `softmax_reg.fit(X_train, Y_train)`
- `y_predict = softmax_reg.predict(X_test)`

46

Decision boundary (決策邊界)

- (背景) decision boundaries: linear between any two classes
- (曲線) probability for the Iris-Versicolor class
- `softmax_reg.predict([[5, 2]]) # X`
- `array([2]) # class` $\hat{y}$
- `softmax_reg.predict_proba([[5, 2]])`
- `array([[9.48936932e-07, 6.91779715e-02, 9.30821080e-01]])`



47

Confusion matrix and mean accuracy (混淆矩陣和平均準確度)

- `import pandas as pd`
- `pd.crosstab(Y_test,softmax_reg.predict(X_test), rownames=['label'], colnames=['predict'])`
- `Random state: 42, 46 for LogisticRegression`

label \ predict	0	1	2
0	10	0	0
1	0	9	0
2	0	0	11

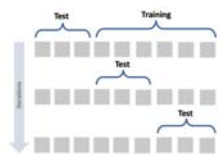
label \ predict	0	1	2
0	12	0	0
1	0	7	2
2	0	2	7

- `print('Logistic regression score: %.3f' % softmax_reg.score(X_test, Y_test))`
- `Logistic regression score: 1.000, 0.867`

48

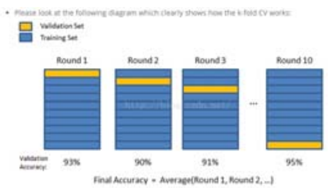
6.4.2 Cross-validation (交叉驗證)

- G. James et al., An Introduction to Statistical Learning (統計學習), section 5.1
- When the original data set isn't large enough, splitting it into (分成) training and test sets may reduce the number of samples that can be used for fitting the model.
- k-fold (K等分) cross-validation: The whole dataset is split into k folds using always k-1 folds for training and the remaining one to validate the model (驗證模型).
- K iterations will be performed, using always a different validation fold.
- an example with 3 folds/iterations:



cross-validation (交叉驗證)

- Use 4 features to predict 3 classes
- `from sklearn.model_selection import cross_val_score`
- `lr = LogisticRegression()`
- `scores = cross_val_score(lr, X, y, scoring='accuracy', cv = 10)`
- `array([0.8 , 0.86666667, 0.86666667, 0.8, 0.86666667, 0.8, 0.73333333, 0.86666667, 1., 1.])`
- `scores.mean()`
- `0.86`
- Could we do better?



https://ithelp.ithome.com.tw/articles/10197461

6.4.3 Grid Search (網格搜索)

- Giuseppe Bonaccorso, Finding the optimal hyperparameters (最佳超參數) through grid search
- Use 4 features to predict 3 classes
- `from sklearn.model_selection import GridSearchCV, cross_val_score`
- `param_grid = [{`

```
'penalty': ['l1', 'l2'], # 懲罰, ℓ1 norm (範數)
'C': [1e-5, 1e-4, 5e-4, 1e-3, 2.3e-3, 5e-3,
     1e-2, 1, 5, 10, 15, 20, 100] }]
```

- `# C:` Inverse of regularization (正規化) strength ($1/\alpha$)

Grid Search (網格搜索)

- `gs = GridSearchCV(estimator = LogisticRegression(), param_grid = param_grid, scoring ='accuracy', cv = 10) # cv:` Cross-validation (交叉驗證)
- `gs.fit(X, y)`
- `print(gs.best_estimator_)`

```
LogisticRegression(C=10, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l1', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

- `gs_scores = cross_val_score(gs.best_estimator_, X, y, scoring='accuracy',  cv=10)`
- `print('Best estimator CV average score: %.3f' % gs_scores.mean())`
- `Best estimator CV average score: 0.980`

Verification (驗證)

- `lr = LogisticRegression(penalty= 'l1')`
- `# C : float, default: 1.0`
- `scores = cross_val_score(lr, X, y, scoring='accuracy', cv = 10)`
- `array([1. , 1., 1. , 0.93333333, 0.93333333, 0.93333333, 0.8 , 1. , 1., 1.])`
- `scores.mean()`
- `0.9600000000000002`
- `lr = LogisticRegression(penalty= 'l1', C = 10)`
- `cross_val_score(lr, X, y, scoring='accuracy', cv = 10)`
- `array([1., 1., 1., 1., 0.93333333,1., 0.86666667, 1., 1. , 1.])`
- `cross_val_score(lr, X, y, scoring='accuracy', cv = 10).mean()`
- `0.9800000000000001 # same number on slide 51`