

亞東學校財團法人亞東科技大學
114 年度教材編纂暨教具製作
結案報告

資料分析方法與應用

申請人：賴宜弘
單位：醫務管理系

114 年 教材編纂與教具製作補助 結案報告(113-2 學期)

教 師 姓 名	賴宜弘	系所、單位	醫務管理系
課 程 名 稱	資料分析方法與應用		
申 請 類 別 (詳 細 類 別)	靜態式數位教材	☐ 題庫編纂 ☒ PPT、講義之靜態式自學教材	
	動態影音式數位教材	☐ 線上題庫系統 ☐ 電子書 ☐ PPT 自學教材(錄音講解) ☐ 串流影音式自學教材	
	教案設計式教材開發	☐ 創新創意教案設計 ☐ 即時互動教學教案設計	
	教具製作	☐ 實體教具(教師自行開發)	
重 點 發 展 教 材 特 色	☒ 專業課程之 全英語教學 或 EMI 教學		
發 展 特 色 (需 擇 一)	☐ 資通訊 ☒ 健康照護 ☐ 紡織產業與創新育成 ☐ 教師著作出版品 ISBN_____		
☐ 五創(創新、創意、創造、創客、創業) ☐ PBL 教材導向			
提升教學品質之量化成果 (與舊課程比較)			
1、新教材主要以英文為授課媒介 (English Medium Instruction)。 2、舊教材偏重於相關學理解說。 3、新教材新增相關範例說明。			
提升教學品質之質性成果 (與舊課程比較)			
1、問題導向學習 (Problem-based learning)。 2、著重學生上機實作。 3、整合國外大學相關課程之教材，提供學生最新知識。			
學習成效評估檢討與後續補充事項			
1、問題導向學習係以問題作為核心，配合教師所設計之英文教學教材，提供學習者進行問題相關資料的蒐集、思考與討論等合作式學習互動，進而整合問題的相關資訊，以達解決問題之目的，提升學生學習興趣。 2、問題導向學習係以小組學習模式進行，組內的學習者之間必須透過各種合作式的互動來解決學習問題，藉由小組學習的歷程，學習者之間可以有效的學習問題解決方法，並分享彼此的學習心得，降低學生對英文內容的恐懼。			
補充附件 (教材檔案網址、活動紀錄、教具放置地點...等補充成果，視情況可另附)			
相關教材將置放於學校課程資料網站，學生可隨時下載複習。			
繳交附件列表(含電子檔內容目錄) 請依照規定繳交			
相關教材如附檔所示。			

成果照片(教具照片 請再附照片原始檔案)



小組討論



小組討論



課堂講解



課堂講解)



期末測驗



期末測驗

114 年 教材編纂與教具製作補助 結案資料 自審表(113-2 學期)

請依照申請類別 自審結案相關資料，相關定義與分類，請參閱 **教材編纂、教具製作 之 分類與補助金額表(113-2)**

類別	細項類別	繳交資料 (方框 ☐ 為 必交之結案資料)
所有類別皆需繳交 結案報告		a. ☒ 結案報告(需含學習成效問卷) 附註：可自行設計與統計 或 提供每學期課程學生學習評量
靜態式數位教材 (上限 20000 元)	☐ 題庫編纂	b. ☐ 題庫電子檔 (100 題以上) c. ☐ 題庫解答(含解說) 附註：b. c. 項可合併
	☒ 自學式教材 (靜態講義)	b. ☒ PPT 或 講義電子檔 (12 週以上自學教材講義，並提供週次與章節列表)
動態影音式數位教材 (上限 40000 元)	☐ 線上題庫系統	b. ☐ 動態式題庫 或 線上題庫系統檔案(需可單獨運作，系統程式檔需提供存查) c. ☐ 線上題庫系統：網址 _____ 附註：b. c. 可擇一，如為線上題庫網址，需可連結並執行，且必須維持系統 3 年以上運作。
	☐ PPT 自學教材 (錄音講解)	b. ☐ PPT 自學教材檔案(9 週以上課程，教師錄音講解總時長需滿 3 小時，並提供週次與章節列表)
	☐ 串流影音式自學教材	b. ☐ 串流影音式自學教材檔案(教材影片總時長需滿 6 小時以上，並需剪輯整理，非上課錄影)
	☐ 電子書	b. ☐ 電子書(須可以獨立執行檔案) (9 週以上課程，須為獨立執行檔案，非影片)
教案設計式教材開發 (上限 40000 元)	☐ 創新創意教案設計	b. ☐ 教案開發 設計文件 或 搭配教案課程之數位教材(擇一) c. ☐ 執行教案開發之 活動照片 (10 張以上，請提供原始檔) e. ☐ 執行教案開發之 影音資料 (5 分鐘以上)
	☐ 即時互動教學教案設計	f. ☐ 教案發展之其他資料 與 電子檔(可自行提供)。 附註：請勿直接提供學生成果或作品，如提供雲端資料，需可連結並執行，且必須維持系統 3 年以上運作
教具製作 (上限 60000 元)	☐ 實體教具 (教師自行開發)	b. ☐ 教具成品，保存地點： _____ (需為實體教具，由老師自行保管 3 年，訪視、成果發表或展示時須提供) c. ☐ 教具照片 (10 張以上，請提供原始檔案) d. ☐ 搭配教具之教學教材 或 手冊、說明書 f. ☐ 教具介紹 或 操作教學影片 (5 分鐘以上) 附註：請勿直接提供學生成果或作品
重點發展教材特色	☒ 全英語教材	☒ 全英語動態影音式數位教材，

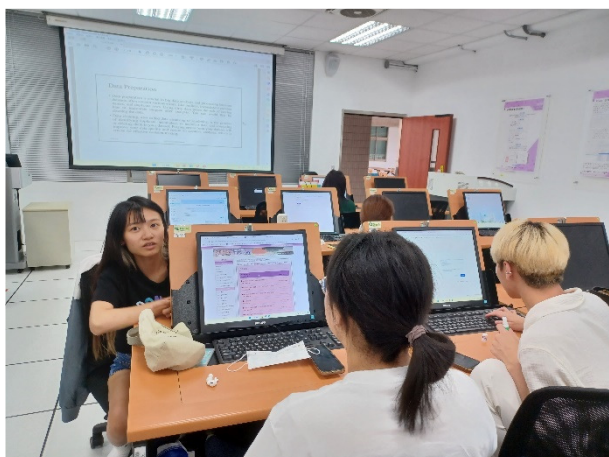
其他補充說明 ○ 其他補充說明或資料，如篇幅不敷使用時，請另增列。

※請 自行審查 結案資料是否完善，並將此表附於 結案報告 內繳交。

學生學習評量

學號	姓名	系所名稱	年級	學期成績	期中考	期末考	平時考	出席	作業
112110101	沈喬昕	日間部四技醫務管理系	2	99	100	100	98	95	100
112110102	解巽翔	日間部四技醫務管理系	2	71	100	100	18	45	0
112110103	洪紹鈞	日間部四技醫務管理系	2	99	100	100	100	100	100
112110104	游云禎	日間部四技醫務管理系	2	77	90	100	38	95	0
112110106	盧奕亘	日間部四技醫務管理系	2	99	100	100	98	95	100
112110107	黃宜柔	日間部四技醫務管理系	2	99	100	100	100	100	100
112110109	游琇芸	日間部四技醫務管理系	2	99	100	100	98	95	100
112110110	秦恩惠	日間部四技醫務管理系	2	90	90	100	72	90	60
112110111	劉子瑜	日間部四技醫務管理系	2	71	100	100	18	45	0
112110112	蘇恩俐	日間部四技醫務管理系	2	99	100	100	98	95	100
112110113	施宥亘	日間部四技醫務管理系	2	99	100	100	100	100	100
112110114	張晉維	日間部四技醫務管理系	2	95	100	90	88	100	80
112110115	鄭泳麒	日間部四技醫務管理系	2	92	100	100	76	70	80
112110116	伊君慧	日間部四技醫務管理系	2	93	90	100	82	85	80
112110117	陳鈺晴	日間部四技醫務管理系	2	91	90	100	74	95	60
112110118	王于萍	日間部四技醫務管理系	2	91	100	100	72	90	60
112110119	吳尚安	日間部四技醫務管理系	2	98	100	100	96	90	100
112110120	李品潔	日間部四技醫務管理系	2	99	90	100	100	100	100
112110122	林文晨	日間部四技醫務管理系	2	92	100	100	76	100	60
112110123	王苡婷	日間部四技醫務管理系	2	99	100	100	100	100	100
112110124	邱子榕	日間部四技醫務管理系	2	90	90	100	72	90	60
112110125	謝佳蓁	日間部四技醫務管理系	2	92	100	100	76	70	80
112110126	陳繹竣	日間部四技醫務管理系	2	99	100	100	98	95	100
112110127	張富暄	日間部四技醫務管理系	2	84	100	90	54	75	40
112110128	黃薰儀	日間部四技醫務管理系	2	99	100	100	100	100	100
112110130	顏正堯	日間部四技醫務管理系	2	78	100	100	36	60	20
112110131	楊姍樺	日間部四技醫務管理系	2	99	90	100	100	100	100
112110132	吳昀緹	日間部四技醫務管理系	2	99	100	100	100	100	100
112110133	柳冠宇	日間部四技醫務管理系	2	71	100	90	20	50	0
112110136	陳璽弘	日間部四技醫務管理系	2	86	100	100	58	85	40
112110138	劉之霖	日間部四技醫務管理系	2	75	80	100	34	55	20
112110141	陳玟伶	日間部四技醫務管理系	2	60	0	0	34	55	20
112110142	鄭凱薰	日間部四技醫務管理系	2	99	100	100	100	100	100
111110136	廖心妤	日間部四技醫務管理系	3	91	80	60	92	80	100
111110138	許翊純	日間部四技醫務管理系	3	75	90	90	36	60	20

附件一、上課照片



小組討論



小組討論



課堂講解



課堂講解



期末測驗



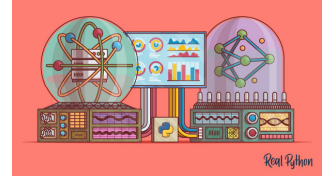
期末測驗

附件二、上課教材

1. Data Preparation

Course Outline

- Data Preparation
- Case Study 1: Titanic Sinking
- Case Study 2: Iris



Data Preparation

- Data preparation is crucial in big data analysis and processing because datasets often contain various errors, like outliers, inconsistent column names, and duplicate rows.
- Using dirty data poses the risk of having bias or inaccurate outputs after analysis. User can avoid this by cleaning the data.
- Data cleaning, also called data cleansing or scrubbing, is the process of identifying duplicate, incomplete, or incorrect data and correcting or deleting them in the dataset.
- Purging errors from the dataset will improve the data quality and ensure an accurate analysis, which is crucial for effective decision-making.

Data Cleaning Steps for Preparing the Data

- Remove duplicate and incomplete cases
- Remove oversamples
- Ensure answers are formatted correctly
- Identify and review outliers
- Code open-ended data
- Check for data consistency

Remove duplicate and incomplete cases

- Identify and remove duplicate data
 - Duplicate errors describe situations where you have repeated entries in a dataset. Such entries can artificially inflate the dataset's size, making it more difficult to work with.
- Identify and remove incomplete cases in datasets
 - Incomplete cases are data points that are missing one or more important values. For example, it's common for some respondents to skip certain questions when they fill out surveys, meaning some fields of your data will be blank.

Remove oversamples

- Consider this scenario: You send out a survey to analyze the annual income of people in a population. After the results return, you get a total response from 650 entries (350 who identify as male and another 300 who identify as female). In data science, we call that imbalanced data because one group has more representation than the other.
- To create balanced data, user may have to randomly remove 50 male entries, bringing the total of each set to 300. The image below shows a similar scenario, but with a smaller dataset.

Ensure answers are formatted correctly

- In data science, we often gather input from multiple data sources, like surveys and questionnaires. It's common to see structural errors where people present similar answers in different formats.
- For example, some people may write phone numbers using the format (999) 999 9999, while others may use dashes in the format (999) 999-9999. Or you may have varying spellings when referring to the same things, like writing "December" as "Dec." Neglecting these errors will cause the algorithm to interpret such entries as separate, introducing errors to the analysis.

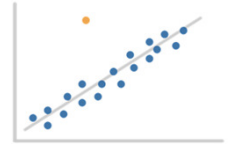
© 賴宏弘

資料分析方法與應用2025E

7

Identify and Review Outliers

- An outlier is a data point that is significantly different from the rest of the data. Data collection errors and natural data variations can cause outliers. Identifying outliers in a dataset before performing any statistical analysis is important, as they can skew the results.
- There are several methods for identifying outliers in a dataset. One common method is to calculate the interquartile range (the range of values in the middle) of the data and identify any data points that are more than two standard deviations away from the mean. Another method is to create a graphical visualization with box plots or the data points on a graph and look for any points that are far away from the rest of the data.



© 賴宏弘

資料分析方法與應用2025E

8

Code Open-ended Data

- Analyze the entries.
 - People typically analyze open-ended data in two ways. One way is to read through each entry manually. This way, you can analyze the intent of the respondent and categorize the entry accordingly.
- Create relevant categories.
 - As you analyze each entry and identify the respondent's intent, you can classify varied categories and tag each entry with a matching category.
 - For instance, if you're analyzing a customer complaints form for a bank, you can create categories like "deposit," "mobile transfer," or "mortgage" based on the subject of the complaints.

© 賴宏弘

資料分析方法與應用2025E

9

Check for Data Consistency

- Checking for data consistency is an important validation step in filtering out bad data. It ensures high-quality data for proper analysis.
- Running a data consistency check requires that you critically observe the modified or cleaned dataset and compare it with the original entries. The goal here is to ensure there are no contradictions between the original entries and the refined data. If you notice any inconsistencies, you may correct the error manually or remove the affected row from the dataset.

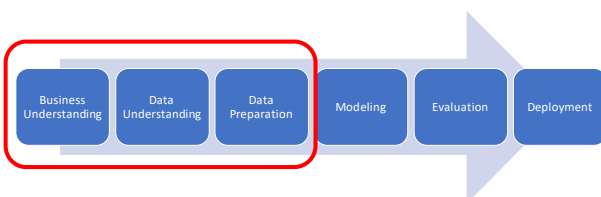
© 賴宏弘

資料分析方法與應用2025E

10

Process for Data Mining

- Process for Data Mining



© 賴宏弘

資料分析方法與應用2025E

11

Case Study 1: Did Chivalry Go Down With the Titanic? (1)

- The RMS Titanic sank in the early morning hours of 15 April 1912 in the North Atlantic Ocean, four days into her maiden voyage from Southampton to New York City.
- The largest ocean liner in service at the time, Titanic had an estimated 2,224 people on board when she struck an iceberg at around 23:40 (ship's time) on Sunday, 14 April 1912.
- Her sinking two hours and forty minutes later at 02:20 (ship's time; 05:18 GMT) on Monday, 15 April, resulted in the deaths of more than 1,500 people, making it one of the deadliest peacetime maritime disasters in history.



© 賴宏弘

資料分析方法與應用2025E

Case Study 1: Did Chivalry Go Down With the Titanic? (2)

- Titanic Sinking



© 顧立弘

資料分析方法與應用2025E

13

Case Study 1: Did Chivalry Go Down With the Titanic? (3)

- titanic_ugly.csv
- 891 passengers

	Column	
1	PassengerId	
2	Survived	0: Dead, 1: Survival
3	Pclass	1: class 1, 2: class 2, 3: class3
4	Name	
5	Sex	male, female
6	Age	
7	SibSp	Sibling, & spouse
8	Parch	Parents, & children
9	Ticket	
10	Fare	
11	Cabin	
12	Embarked	C: Cherbourg, Q: Queenstown, S: Southampton

© 顧立弘

資料分析方法與應用2025E

14

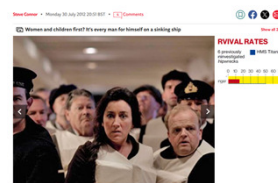
Case Study 1: Did Chivalry Go Down With the Titanic? (4)

It is the unwritten law of the sea but the idea of women and children first when a ship is sinking is a myth, according to a study of 18 maritime disasters where the prevailing attitude is best summarised as "every man for himself".

With the notable exception of the Titanic disaster in 1912, women and children were far less likely to survive a sinking ship than men, and the male members of a ship's crew almost invariably had the best chances of survival compared with male and female passengers, the study found.

- <https://www.independent.co.uk/news/world/world-history/women-and-children-first-it-s-every-man-for-himself-on-a-sinking-ship-7987975.html>

Women and children first? It's every man for himself on a sinking ship



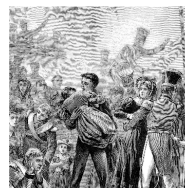
© 顧立弘

資料分析方法與應用2025E

15

Case Study 1: Did Chivalry Go Down With the Titanic? (5)

- Whether gender determines survival chance?
- Whether social class determines survival chance?



資料分析方法與應用2025E

16

Import the File to DataFrame

- from google.colab import drive
- drive.mount('/content/MyGoogleDrive')
- import pandas
- df=pandas.read_csv('/content/MyGoogleDrive/My Drive/titanic_ugly.csv')
- df

© 顧立弘

資料分析方法與應用2025E

17

Description of the Data in the DataFrame

- df.info()
- df.describe()

© 顧立弘

資料分析方法與應用2025E

18

Duplicate Values in a DataFrame

- `df[df.duplicated()]`
- `df=df.drop_duplicates()`

NULL values in a DataFrame (1): Continuous Data

- `df['Age2']=df['Age'].fillna(df['Age'].mean())`

NULL values in a DataFrame (2): Categorical Data

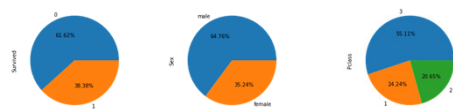
- #Find Max one
- `df['Embarked'].value_counts()`
- `df['Embarked']=df['Embarked'].fillna('S')`

To Recode Values in a DataFrame

- `sex_code={'female':0, 'male':1}`
- `df['Sex2']=df['Sex'].map(sex_code)`
- `port_code={'S':0, 'C':1, 'Q':2}`
- `df['Embarked2']=df['Embarked'].map(port_code)`

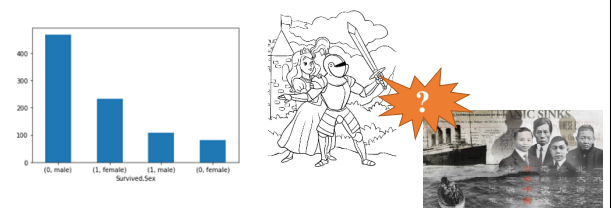
Descriptive Analysis (1): Data Visualization

- `df['Survived'].value_counts().plot(kind='pie', autopct='%1.2f%%')`
- `df['Sex'].value_counts().plot(kind='pie', autopct='%1.2f%%')`
- `df['Pclass'].value_counts().plot(kind='pie', autopct='%1.2f%%')`



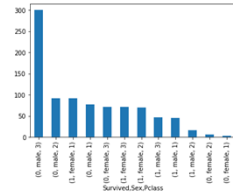
Descriptive Analysis (2): Data Visualization

- `df[['Sex','Survived']].value_counts().plot(kind='bar', rot=0)`



Descriptive Analysis (3): Data Visualization

- `df[['Survived','Sex','Pclass']].value_counts().plot(kind='bar')`



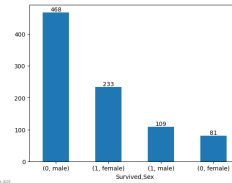
© 顧宜弘

資料分析方法與應用2025E

25

Descriptive Analysis (4): Data Visualization

- `myplot=df[['Survived','Sex']].value_counts().plot(kind='bar',rot=0)`
- `myplot.bar_label(myplot.containers[0])`



© 顧宜弘

資料分析方法與應用2025E

26

Example 1. Three Species of Iris (1)

- The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor).
- The famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.

- Dataset: iris_ugly.csv



© 顧宜弘

資料分析方法與應用2025E

27

Example 1. Three Species of Iris (2)

- Code Book

	Column	Note
1	Length of Sepals	
2	Width of Sepals	
3	Length of Petals	
4	Width of Petals	
5	class	Iris setosa Iris virginica Iris versicolor

© 顧宜弘

資料分析方法與應用2025E

28

Import the File to DataFrame

- from google.colab import drive
- `drive.mount('/content/MyGoogleDrive')`
- import pandas
- `df=pandas.read_csv('/content/MyGoogleDrive/My Drive/iris_ugly.csv')`
- df

© 顧宜弘

資料分析方法與應用2025E

29

Description of the Data in the DataFrame

- `df.info()`
- `df.describe()`

© 顧宜弘

資料分析方法與應用2025E

30

Duplicate Values in a DataFrame

- `df[df.duplicated()]`
- `df=df.drop_duplicates()`

NULL values in a DataFrame: Continuous Data

- `df['Length of Sepals']=df['Length of Sepals'].fillna(df['Length of Sepals'].mean())`

To Recode Values in a DataFrame (1)

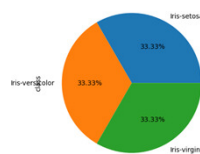
- `class_code={'Iris-setosa':1, 'Iris-virginica':2, 'Iris-versicolor':3}`
- `df['class1']=df['class'].map(class_code)`

To Recode Values in a DataFrame (2)

- `#The color in plot`
- `color_code={'Iris-setosa': 'r', 'Iris-versicolor': 'g', 'Iris-virginica': 'b'}`
- `df['class_color']=df['class'].map(color_code)`

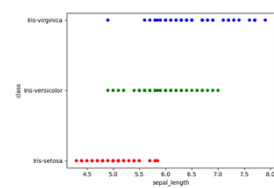
Descriptive Analysis : Data Visualization (1)

- `df['class'].value_counts()`
- `df['class'].value_counts().plot(kind='pie', autopct='%1.2f%%')`



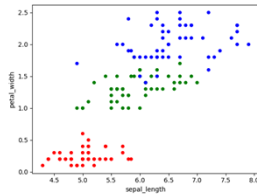
Descriptive Analysis : Data Visualization (2)

- `df.plot(kind='scatter',x='sepal_length',y='class', c='class_color')`



Descriptive Analysis : Data Visualization (3)

- `df.plot(kind='scatter',x='sepal_length',y='petal_width', c='class_color')`



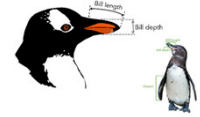
© 賴宜弘

資料分析方法與應用2025E

37

Example 2. The Type of Penguins (1)

- The dataset contains 333 instances of penguins from three species (classes), Adelie, Chinstrap, and Gentoo. The Adelie species contains 146 instances of penguins, Chinstrap has 68, and Gentoo has 119.



- Dataset: `penguins_size_ugly.csv`

© 賴宜弘

資料分析方法與應用2025E

38

Example 2. The Type of Penguins (2)

- Code Book

Feature	Description	Type
species	Adelie, Chinstrap, or Gentoo	Categorical
species1	1: Adelie, 2: Chinstrap, or 3:Gentoo	
island	The name of the island (Dream, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica) where the penguin was found and measured.	Categorical
island1	1: Dream, 2: Torgersen, or 3: Biscoe	
culmen_length_mm	The length of the penguin's culmen in millimeters.	Numerical
culmen_depth_mm	The depth of the penguin's culmen in millimeters.	Numerical
flipper_length_mm	The length of the penguin's flippers in millimeters.	Numerical
body_mass_g	The body mass of the penguin in grams.	Numerical
sex	The sex (MALE, FEMALE) of the penguin.	Categorical
species1_1	1: Adelie 0: Not Adelie	
species1_2	1: Chinstrap 0: Not Chinstrap	
species1_3	1: Gentoo 0: Not Gentoo	

© 賴宜弘

資料分析方法與應用2025E

39



© 賴宜弘

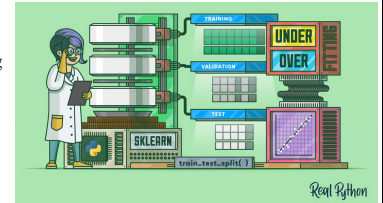
資料分析方法與應用2025E

40

2. Data Science

Course Outline

- Data Science
- Database Table
- Machine Learning
- Types of Machine Learning
- Process for Data Mining
- **Note: Split The Dataset**



Data Science

- Data Science is a combination of multiple disciplines that uses statistics, data analysis, and machine learning to analyze data and to extract knowledge and insights from it.
- Data Science is about data gathering, analysis and decision-making.
- Data Science is about finding patterns in data, through analysis, and make future predictions.
- By using Data Science, companies are able to make:
 - Better decisions (should we choose A or B)
 - Predictive analysis (what will happen next?)
 - Pattern discoveries (find pattern, or maybe hidden information in the data)

How Does a Data Scientist Work

- Backgrounds
 - Machine Learning
 - Statistics
 - Programming (Python or R)
 - Mathematics
 - Databases
- Data Scientist works
 - Ask the right questions: To understand the business problem.
 - Explore and collect data: From database, web logs, customer feedback, etc.
 - Extract the data: Transform the data to a standardized format.
 - Clean the data: Remove erroneous values from the data.
 - Find and replace missing values: Check for missing values and replace them with a suitable value.
 - Normalize data: Scale the values in a practical range.
 - Analyze data, find patterns and make future predictions.
 - Represent the result: Present the result with useful insights in a way the "company" can understand.

What is Data

- Data is a collection of information.
- One purpose of Data Science is to structure data, making it interpretable and easy to work with.
- Data can be categorized into two groups:
 - Unstructured data: Unstructured data is not organized. We must organize the data for analysis purposes.
 - Structured data: Structured data is organized and easier to work with.



Example of Unstructured data

Database Table

- A database table is a table with structured data
- A database table consists of column(s) and row(s)
 - A row is a horizontal representation of data
 - A column is a vertical representation of data
- Variables
 - A variable is defined as something that can be measured or counted

total_length	total_weight	total_length	total_weight	class
50	3.5	14	0.2	fin-sword
40	3	14	0.2	fin-sword
40	2.2	14	0.2	fin-sword
7	2.2	4.5	1.4	fin-sword
40	2.6	40	1.4	fin-sword
40	3.1	40	1.5	fin-sword
40	3.3	40	2.5	fin-sword
50	2.7	5.1	1.9	fin-sword
7	3	5.0	2.1	fin-sword

Machine Learning

- Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.
- A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

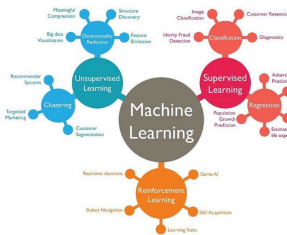


Features of Machine Learning

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

Types of Machine Learning

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning



Supervised Learning Algorithm

- In Supervised learning, an AI system is presented with data which is labeled, which means that each data tagged with the correct label.
- The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data.
- Types of Supervised learning
 - Classification: A classification problem is when the output variable is a category.
 - Regression: A regression problem is when the output variable is a real value.

Unsupervised Learning Algorithm

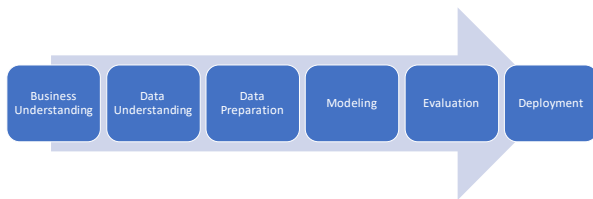
- In unsupervised learning, an AI system is presented with unlabeled, uncategorized data and the system's algorithms act on the data without prior training.
- The output is dependent upon the coded algorithms. Subjecting a system to unsupervised learning is one way of testing AI.
- Types of Unsupervised learning
 - Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
 - Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Reinforcement Learning Algorithm

- A reinforcement learning algorithm, or agent, learns by interacting with its environment.
- The agent receives rewards by performing correctly and penalties for performing incorrectly.
- The agent learns without intervention from a human by maximizing its reward and minimizing its penalty.
- It is a type of dynamic programming that trains algorithms using a system of reward and punishment.

Process for Data Mining

- Process for Data Mining



1. Business Understanding

- What does the business need?
- The Business Understanding phase focuses on understanding the objectives and requirements of the project.
- Task
 - Determine business objectives
 - Assess situation
 - Determine data mining goals
 - Produce project plan

2. Data Understanding

- What data do we have / need? Is it clean?
- Data Understanding phase drives the focus to identify, collect, and analyze the data sets that can help people accomplish the project goals.
- Task
 - Collect initial data
 - Describe data
 - Explore data
 - Verify data quality

3. Data Preparation

- How do we organize the data for modeling?
- Data Understanding phase drives the focus to identify, collect, and analyze the data sets that can help people accomplish the project goals.
- Task
 - Select data
 - Clean data
 - Construct data
 - Integrate data
 - Format data

4. Modeling

- What modeling techniques should we apply?
- People build and assess various models based on several different modeling techniques.
- Task
 - Select modeling techniques
 - Generate test design
 - Build model
 - Assess model

5. Evaluation

- Which model best meets the business objectives?
- Whereas the Assess Model task of the Modeling phase focuses on technical model assessment, the Evaluation phase looks more broadly at which model best meets the business and what to do next.
- Task
 - Evaluate results
 - Review process
 - Determine next steps

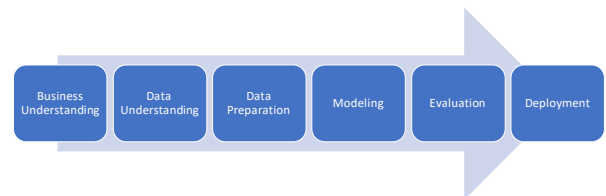
6. Deployment

- How do stakeholders access the results?
- The deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process across the enterprise.



Process

- Process



Example 1. Three Species of Iris (1)

- The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor).
- The famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.
- Dataset: iris_1.csv



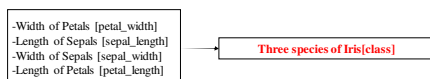
Example 1. Three Species of Iris (2)

- Code Book

	Column	Note
1	Length of Sepals	
2	Width of Sepals	
3	Length of Petals	
4	Width of Petals	
5	class	Iris setosa Iris virginica Iris versicolor
6	class1	1: Iris setosa 2: Iris virginica 3: Iris versicolor
8	class1_1	1: Iris setosa 0: Not Iris setosa
9	class1_2	1: Iris virginica 0: Not Iris virginica
10	Class1_3	1: Iris versicolor 0: Not Iris versicolor

Example 1. Three Species of Iris (Step 1. Business Understanding)

- Business Understanding



Example 1. Three Species of Iris (Step 2. Data Understanding 1)

- #Open File
- from google.colab import drive
- drive.mount('/content/MyGoogleDrive')
- import pandas
- df=pandas.read_csv('/content/MyGoogleDrive/My Drive/iris_1.csv')
- df

Example 1. Three Species of Iris (Step 3. Data Preparation 1)

- `df.drop_duplicates()`
- `df.dropna(how='any')`
- `df['sepal_length'].fillna(value=df['sepal_length'].mean(), inplace=True)`

Example 1. Three Species of Iris (Step 3. Data Preparation 2)

- `## Split the dataset into training and testing sets`
- `from sklearn.model_selection import train_test_split`
- `x=df[['sepal_length', 'sepal_width', 'petal_length']]`
- `y=df['petal_width']`
- `x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)`
- `data_train=pandas.concat([x_train,y_train],axis=1)`
- `data_test=pandas.concat([x_test,y_test],axis=1)`
- `data_train`

Note: Split The Dataset With scikit-learn's `train_test_split()` (1)

- With `train_test_split()` from scikit-learn, user can efficiently divide the dataset into training and testing subsets to ensure unbiased model evaluation in machine learning.
- This process helps prevent overfitting and underfitting by keeping the test data separate from the training data, allowing users to assess the model's predictive performance accurately.

- Source: <https://realpython.com/train-test-split-python-data/>

Note: Split The Dataset with scikit-learn's `train_test_split()` (2)

- `train_test_split()` is a function in sklearn that divides datasets into training and testing subsets.
- `x_train` and `y_train` represent the inputs and outputs of the training data subset, respectively, while `x_test` and `y_test` represent the input and output of the testing data subset.
- By specifying `test_size=0.2`, user use 20% of the dataset for testing, leaving 80% for training.
- `train_test_split()` can handle imbalanced datasets using the `stratify` parameter to maintain class distribution.

Note: Split The Dataset with scikit-learn's `train_test_split()` (3)

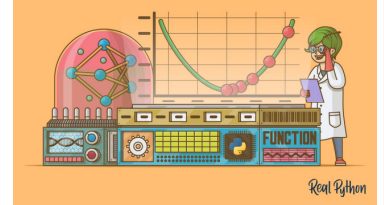
- The training set is applied to train or fit your model. For example, you use the training set to find the optimal weights, or coefficients, for linear regression, logistic regression, or neural networks.
- The validation set is used for unbiased model evaluation during hyperparameter tuning. For example, when you want to find the optimal number of neurons in a neural network or the best kernel for a support vector machine, you experiment with different values. For each considered setting of hyperparameters, you fit the model with the training set and assess its performance with the validation set.
- The test set is needed for an unbiased evaluation of the final model. You shouldn't use it for fitting or validation.



3. Linear Regression

Course Outline

- Multiple Linear Regression
- Estimation of Parameters
- Assumptions for MLR
- Note: Evaluation



Linear Regression

- Beginning with the definition of regression, for determining the significance and potential of the relationships between a dependent variable and a series of independent variables, a statistical method is used, known as regression.
- Two basic types of regression are;
 - Simple Linear Regression
 - Multiple Linear Regression
- Linear regression attempts to identify the connection amid the two variables along a straight line.
- Simply, this model is used to predict or show the relationship between a dependent variable and an independent variable.

Multiple Linear Regression (1)

- Multiple linear regression is simply the extension of simple linear regression, that predicts the value of a dependent variable (sometimes it is called as the outcome, target or criterion variable) on the basis of two or more independent variables (or sometimes, the predictor, explanatory or regressor variables).

Multiple Linear Regression (2)

- It is a method of statistical analysis that provides the statistical significance to explanatory variables, or which potential explanatory variables are crucial predictors for a given response (target) variable.
- It can be used to determine the impact of changes, i.e. to understand the changes in the dependent variable while making changes in the independent variables. For example, reviewing the health of a person to check how much blood pressure goes up and down with a unit change in the body mass index of that person, keeping other factors constant.

Multiple Linear Regression (3)

- Multiple linear regression is a mathematical technique that deploys the relationship among multiple independent predictor variables and a single dependent outcome variable.
- The methodology also involves the various means of determining which variables are important and can be implemented to make a regression model for prediction considerations.

Multiple linear regression Formula

- The equation of multiple linear regression is expressed as

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p + \epsilon$$
- Where for
- $i = n$ observations;
- y = dependent variable,
- x = explanatory variables, here we have "p" predictor variables and "p+1" as total regression parameters.
- β_0 = y-intercept which is a constant term,
- β_p = Slope coefficient for each explanatory variable, and
- ϵ = residuals (model's error term), having a normal distribution with mean 0 and constant variance,
- In multiple linear regression, the word linear signifies that the model is linear in parameters, $\beta_0, \beta_1, \beta_2$ and so on.

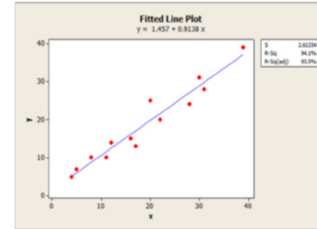
© 賴立弘

資料分析方法與應用 2025E

7

Linear Relationship - Scatter Plot

- Linear Regression



X	Y
4	5
5	7
8	10
11	10
12	14
16	15
17	13
20	25
22	20
28	24
30	31
31	28
39	39

© 賴立弘

8

Estimation of Parameters (1)

- Given a collection of pairs (x, y) of numbers (in which not all the x -values are the same), there is a line $\hat{y} = \hat{\beta}_1 x + \hat{\beta}_0$ that best fits the data in the sense of minimizing the sum of the squared errors.
- It is called the least squares regression line. Its slope $\hat{\beta}_1$ and y-intercept $\hat{\beta}_0$ are computed using the formulas

© 賴立弘

資料分析方法與應用 2025E

9

Estimation of Parameters (2)

- The Maximum Likelihood Regression Line
- $\hat{y} = \hat{\beta}_1 x + \hat{\beta}_0$

$$\hat{\beta}_0 = \frac{\left| \frac{\sum_{i=1}^n y_i}{\sum_{i=1}^n x_i y_i} \quad \frac{\sum_{i=1}^n x_i}{\sum_{i=1}^n x_i^2} \right|}{\left| \frac{n}{\sum_{i=1}^n x_i} \quad \frac{\sum_{i=1}^n x_i}{\sum_{i=1}^n x_i^2} \right|} = \frac{\sum_{i=1}^n y_i \sum_{i=1}^n x_i^2 - \sum_{i=1}^n x_i \sum_{i=1}^n x_i y_i}{n \sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2} = \bar{y} - \hat{\beta}_1 \bar{x}$$

$$\hat{\beta}_1 = \frac{\left| \frac{n}{\sum_{i=1}^n x_i} \quad \frac{\sum_{i=1}^n y_i}{\sum_{i=1}^n x_i y_i} \right|}{\left| \frac{n}{\sum_{i=1}^n x_i} \quad \frac{\sum_{i=1}^n x_i}{\sum_{i=1}^n x_i^2} \right|} = \frac{\sum_{i=1}^n x_i y_i - n \bar{x} \bar{y}}{\sum_{i=1}^n x_i^2 - n \bar{x}^2} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{\sum_{i=1}^n (x_i - \bar{x}) y_i}{\sum_{i=1}^n (x_i - \bar{x}) x_i}$$

© 賴立弘

資料分析方法與應用 2025E

10

Assumption 1: Linear Relationship (1)

- There is a linear relationship between the independent variable x , and the independent variable y .
- The easiest way to detect if this assumption is met is to create a scatter plot of x vs. y .
 - This allows you to visually see if there is a linear relationship between the two variables.
 - If it looks like the points in the plot could fall along a straight line, then there exists some type of linear relationship between the two variables and this assumption is met.



© 賴立弘

資料分析方法與應用 2025E

11

Assumption 1: Linear Relationship (2)

- If you create a scatter plot of values for x and y and see that there is not a linear relationship between the two variables, then you have a couple options:
 - Apply a nonlinear transformation to the independent and/or dependent variable. Common examples include taking the log, the square root, or the reciprocal of the independent and/or dependent variable.
 - Add another independent variable to the model. For example, if the plot of x vs. y has a parabolic shape then it might make sense to add x^2 as an additional independent variable in the model.

© 賴立弘

資料分析方法與應用 2025E

12

Assumption 2: Independence (1)

- The residuals are independent.
 - This is mostly relevant when working with time series data. Ideally, we don't want there to be a pattern among consecutive residuals. For example, residuals shouldn't steadily grow larger as time goes on.
- The simplest way to test if this assumption is met is to look at a residual time series plot, which is a plot of residuals vs. time.
 - Ideally, most of the residual autocorrelations should fall within the 95% confidence bands around zero, which are located at about ± 2 -over the square root of n , where n is the sample size.
 - You can also formally test if this assumption is met using the Durbin-Watson test.

Assumption 2: Independence (2)

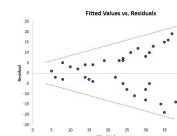
- Depending on the nature of the way this assumption is violated:
 - For positive serial correlation, consider adding lags of the dependent and/or independent variable to the model.
 - For negative serial correlation, check to make sure that none of your variables are over differenced.
 - For seasonal correlation, consider adding seasonal dummy variables to the model.

Assumption 3: Homoscedasticity (1)

- The residuals have constant variance at every level of x . This is known as homoscedasticity. When this is not the case, the residuals are said to suffer from heteroscedasticity.
- When heteroscedasticity is present in a regression analysis, the results of the analysis become hard to trust. Specifically, heteroscedasticity increases the variance of the regression coefficient estimates, but the regression model doesn't pick up on this. This makes it much more likely for a regression model to declare that a term in the model is statistically significant, when in fact it is not.

Assumption 3: Homoscedasticity (2)

- The simplest way to detect heteroscedasticity is by creating a fitted value vs. residual plot.
- Once you fit a regression line to a set of data, you can then create a scatterplot that shows the fitted values of the model vs. the residuals of those fitted values. The scatterplot below shows a typical fitted value vs. residual plot in which heteroscedasticity is present.
- Notice how the residuals become much more spread out as the fitted values get larger. This "cone" shape is a classic sign of heteroscedasticity:



Assumption 3: Homoscedasticity (3)

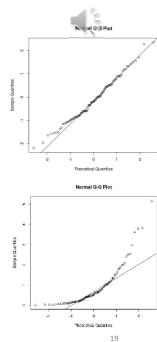
- The common ways to fix heteroscedasticity:
 - Transform the dependent variable.
 - One common transformation is to simply take the log of the dependent variable. For example, if we are using population size (independent variable) to predict the number of flower shops in a city (dependent variable), we may instead try to use population size to predict the log of the number of flower shops in a city. Using the log of the dependent variable, rather than the original dependent variable, often causes heteroscedasticity to go away.
 - Redefine the dependent variable.
 - One common way to redefine the dependent variable is to use a rate, rather than the raw value. For example, instead of using the population size to predict the number of flower shops in a city, we may instead use population size to predict the number of flower shops per capita. In most cases, this reduces the variability that naturally occurs among larger populations since we're measuring the number of flower shops per person, rather than the sheer amount of flower shops.
 - Use weighted regression.
 - Another way to fix heteroscedasticity is to use weighted regression. This type of regression assigns a weight to each data point based on the variance of its fitted value. Essentially, this gives small weights to data points that have higher variances, which shrinks their squared residuals. When the proper weights are used, this can eliminate the problem of heteroscedasticity.

Assumption 4: Normality (1)

- The next assumption of linear regression is that the residuals are normally distributed.

Assumption 4: Normality (2)

- The common ways to check if this assumption is met:
 - Check the assumption visually using Q-Q plots.
 - A Q-Q plot, short for quantile-quantile plot, is a type of plot that we can use to determine whether or not the residuals of a model follow a normal distribution. If the points on the plot roughly form a straight diagonal line, then the normality assumption is met.
 - The following Q-Q plot shows an example of residuals that roughly follow a normal distribution.
 - The Q-Q plot below shows an example of when the residuals clearly depart from a straight diagonal line, which indicates that they do not follow normal distribution.
 - You can also check the normality assumption using formal statistical tests like Shapiro-Wilk, Kolmogorov-Smirnov, Jarque-Barre, or D'Agostino-Pearson. However, keep in mind that these tests are sensitive to large sample sizes – that is, they often conclude that the residuals are not normal when your sample size is large. This is why it's often easier to just use graphical methods like a Q-Q plot to check this assumption.



© 賴立弘

資料分析方法與應用2025E

19

Assumption 4: Normality (3)

- If the normality assumption is violated:
 - Verify that any outliers aren't having a huge impact on the distribution. If there are outliers present, make sure that they are real values and that they aren't data entry errors.
 - You can apply a nonlinear transformation to the independent and/or dependent variable. Common examples include taking the log, the square root, or the reciprocal of the independent and/or dependent variable.

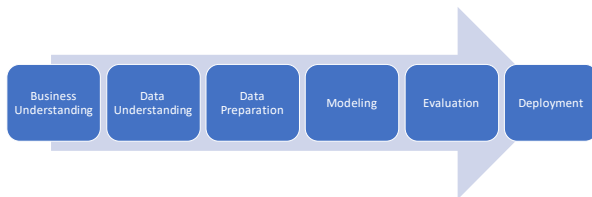
© 賴立弘

資料分析方法與應用2025E

20

Process

- Process



© 賴立弘

資料分析方法與應用2025E

21

Example 1. Three Species of Iris (1)

- The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor).
- The famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.
- Dataset: iris_1.csv



© 賴立弘

資料分析方法與應用2025E

22

Example 1. Three Species of Iris (2)

- Code Book

	Column	Note
1	Length of Sepals	
2	Width of Sepals	
3	Length of Petals	
4	Width of Petals	
5	class	Iris setosa Iris virginica Iris versicolor
6	class1	1: Iris setosa 2: Iris virginica 3: Iris versicolor
7	class1_1	1: Iris setosa 0: Not Iris setosa
8	class1_2	1: Iris virginica 0: Not Iris virginica
9	Class1_3	1: Iris versicolor 0: Not Iris versicolor

© 賴立弘

資料分析方法與應用2025E

23

Example 1. Three Species of Iris (Step 1. Business Understanding)

- Business understanding



© 賴立弘

資料分析方法與應用2025E

24

Example 1. Three species of Iris (Step 2. Data Understanding 1)

- #Open File
- from google.colab import drive
- drive.mount('/content/MyGoogleDrive')
- import pandas
- df=pandas.read_csv('/content/MyGoogleDrive/My Drive/iris_1.csv')
- df

Example 1. Three species of Iris (Step 2. Data Understanding 2)

- df.info()
- df.describe()

Example 1. Three species of Iris (Step 3. Data Preparation 1)

- df.drop_duplicates()
- df.dropna(how='any')
- df['sepal_length'].fillna(value=df['sepal_length'].mean(), inplace=True)

Example 1. Three species of Iris (Step 3. Data Preparation 2)

- # Split the dataset into training and testing sets
- from sklearn.model_selection import train_test_split
- x=df[['sepal_length', 'sepal_width', 'petal_length']]
- y=df['petal_width']
- x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
- data_train=pandas.concat([x_train,y_train],axis=1)
- data_test=pandas.concat([x_test,y_test],axis=1)

Example 1. Three species of Iris (Step 4. Modeling) Type 1

- ### TYPE 1
- import statsmodels.formula.api as smf
- model=smf.ols(formula='petal_width~sepal_length+sepal_width+petal_length', data=df)
- results=model.fit()
- results.summary()

Example 1. Three species of Iris (Step 5. Evaluation) Type 1

- Evaluation

```
OLS Regression Results
Dep. Variable: petal_width    R-squared: 0.938
Model: OLS                  Adj. R-squared: 0.937
Method: Least Squares       F-statistic: 726.3
Date: Mon, 17 Mar 2025      Prob (F-statistic): 6.20e-48
Time: 06:44:48              Log-Likelihood: 36.809
No. Observations: 150       AIC: -65.62
DF Residuals: 146          BIC: -53.57
DF Model: 3
Covariance Type: nonrobust

coef    std err    t    Pr>|t| [0.025 0.975]
Intercept    -0.2487    0.178    -1.396    0.165    -0.601    0.103
sepal_length    0.2103    0.048     4.428    0.000    0.308    0.116
sepal_width    0.2288    0.049     4.669    0.000    0.132    0.326
petal_length    0.0261    0.024     1.124    0.264    -0.021    0.073
Omnibus:    5.603    Durbin-Watson:    1.577
Prob(Omnibus): 0.061    Jarque-Bera (JB): 6.817
Skew:    0.222    Prob(JB):    0.0331
Kurtosis:    3.945    Cond. No.    99.0
```

Example 1. Three species of Iris (Step 6. Deployment) **Type 1**

- $\text{pw} = (-0.199) * \text{sl} + (0.208) * \text{sw} + (0.523) * \text{pl} + (-0.227)$
- `x_new=pandas.DataFrame({'sepal_length':[5.1],'sepal_width':[3.5],'petal_length':[1.4]})`
- `results.predict(x_new)`

sepal_length	sepal_width	petal_length	petal_width
5.1	3.5	1.4	?

Example 1. Three species of Iris (Step 4. Modeling) **Type 2**

- `### TYPE 2`
- `from sklearn import linear_model`
- `lm2=linear_model.LinearRegression()`
- `lm2.fit(x_train,y_train)`
- `print(lm2.coef_)`
- `print(lm2.intercept_)`

Example 1. Three species of Iris (Step 5. Evaluation) **Type 2**

- # the coefficient of determination, or R2
- `r2 = lm2.score(x_train, y_train)`
- `pp = lm2.score(x_test, y_test)`
- `print("R2:", r2)`
- `print("The model's prediction performance:", pp)`

Example 1. Three species of Iris (Step 6. Deployment) **Type 2**

- $\text{pw} = (-0.199) * \text{sl} + (0.208) * \text{sw} + (0.523) * \text{pl} + (-0.227)$
- `x_new=pandas.DataFrame({'sepal_length':[5.1],'sepal_width':[3.5],'petal_length':[1.4]})`
- `lm2.predict(x_new)`

sepal_length	sepal_width	petal_length	petal_width
5.1	3.5	1.4	?

Example 2. The Heights of Children and Their Parents

- The table below gives data based on the famous 1885 study of Francis Galton exploring the relationship between the heights of adult children and the heights of their parents.

- Dataset: Galton.csv
 - Family: The family that the child belongs to, labeled from 1 to 204 and 136A
 - Father: The father's height, in inches
 - Mother: The mother's height, in inches
 - Gender: The gender of the child, male (M) or female (F)
 - Height: The height of the child, in inches
 - Kids: The number of kids in the family of the child
- Source: <http://www.randomservices.org/random/data/index.html>



Example 2. The Heights of Children and Their Parents (Step 1. Business understanding)

- Business understanding

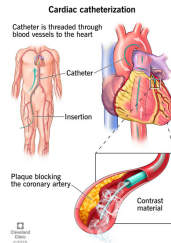
-The father's height, in inches [Father]
-The mother's height, in inches [Mother]
-The gender of the child [Gender]
-The number of kids in the family of the child [Kids]

The height of the child, in inches [Height]



Example 3. Heart Catherization Data (1)

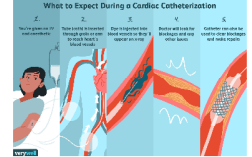
- This data set was analyzed by Weisberg (1980) and Chambers et al. (1983).
 - A catheter is passed into a major vein or artery at the femoral region and moved into the heart.
 - The proper length of the introduced catheter has to be guessed by the physician. The aim of the data set is to describe the relation between the catheter length and the patient's height (X1) and weight (X2).
 - This data sets is used to demonstrate the effects caused by collinearity. The correlation between height and weight is so high that either variable almost completely determines the other.
- Dataset: heart01.csv



Example 3. Heart Catherization Data (2)

- Code Book

	Description	Type
height	X1: Patient's height in inches	Numerical
weight	X2: Patient's weight in pounds	Numerical
length	Y: Catheter Length (in centimeters)	Numerical



Example 3. Heart Catherization Data (3) (Step 1. Business understanding)

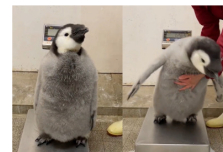
- Business Understanding

-Patient's height in inches [height]
-Patient's weights in pounds [weight] → Catheter Length (in centimeters)[c.length]



Example 4. The Weight of Penguins (1)

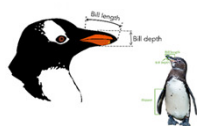
- The dataset contains 333 instances of penguins from three species (classes), Adelie, Chinstrap, and Gentoo. The Adelie species contains 146 instances of penguins, Chinstrap has 68, and Gentoo has 119.



- Dataset: penguins_size1.csv

Example 4. The Weight of Penguins (2)

- Code Book

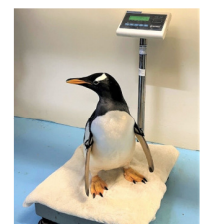


Feature	Description	Type
species	Adelie, Chinstrap, or Gentoo	Categorical
species_1	1: Adelie, 2: Chinstrap, or 3: Gentoo	
island	The name of the island (Dumont, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica) where the penguin was found and measured.	Categorical
island_1	1: Dumont, 2: Torgersen, or 3: Biscoe	
culmen_length_mm	The length of the penguin's culmen in millimeters.	Numerical
culmen_depth_mm	The depth of the penguin's culmen in millimeters.	Numerical
flipper_length_mm	The length of the penguin's flipper in millimeters.	Numerical
body_mass_g	The body mass of the penguin in grams.	Numerical
sex	The sex (MALE, FEMALE) of the penguin.	Categorical
sex_1	1: MALE, 0: FEMALE	
species_1_1	1: Adelie	
species_1_2	0: Not Adelie	
species_1_2	1: Chinstrap	
species_1_3	0: Not Chinstrap	
species_1_3	1: Gentoo	
species_1_3	0: Not Gentoo	

Example 4. The Weight of Penguins (Step 1. Business understanding)

- Business Understanding

-culmen_length_mm
-culmen_depth_mm
-flipper_length_mm → -body_mass_g



Exercise 5. Air temperature and Pearl milk tea

- Pearl milk tea
 - milktea01.csv



Note: Evaluation

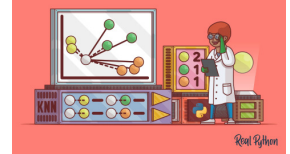
- `.score()` returns the coefficient of determination, or R^2 , for the data passed.
- Its maximum is 1.
- The higher the R^2 value, the better the fit.
- In this case, the **training data** yields a slightly higher coefficient.
- The R^2 calculated with **test data** is an unbiased measure of your model's prediction performance.
- Source: <https://realpython.com/train-test-split-python-data/>



4. Nearest Neighbor Analysis

Course Outline

- K-Nearest Neighbor
- The Advantages of KNN Algorithm
- The Disadvantages of KNN Algorithm
- Significance of k
- How to decide the value of k



K-Nearest Neighbor (1)

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K-NN algorithm.

K-Nearest Neighbor (2)

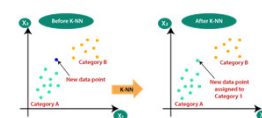
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

K-Nearest Neighbor (3)

- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
- Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

Why do we need a K-NN Algorithm?

- Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x_1 , so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm.
- With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



How does K-NN work? (1)

- Step 1: Select the number K of the neighbors
- Step 2: Calculate the Euclidean distance of K number of neighbors
- Step 3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step 4: Among these k neighbors, count the number of the data points in each category.
- Step 5: Assign the new data points to that category for which the number of the neighbor is maximum.
- Step 6: Our model is ready.

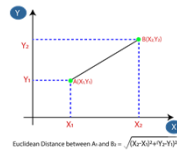
How does K-NN work? (2)

- Suppose we have a new data point and we need to put it in the required category. Consider the below image:



How does K-NN work? (3)

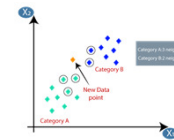
- Firstly, we will choose the number of neighbors, so we will choose the $k=5$.
- Next, we will calculate the Euclidean distance between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:



Euclidean Distance between A and B is $= \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$

How does K-NN work? (4)

- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B.
- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.



How to select the value of K in the K-NN Algorithm?

- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
- A very low value for K such as $K=1$ or $K=2$, can be noisy and lead to the effects of outliers in the model.
- Large values for K are good, but it may find some difficulties.

Advantages of KNN Algorithm

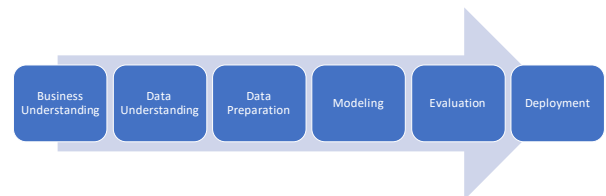
- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

Disadvantages of KNN Algorithm

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

Process

- Process



Example 1. Three Species of Iris (1)

- The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor).
- The famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.

- Dataset: iris_1.csv



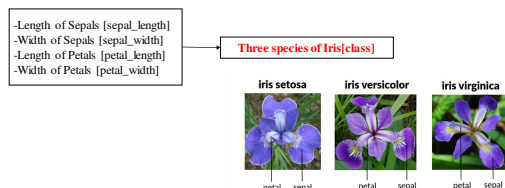
Example 1. Three Species of Iris (2)

- Code Book

	Column	Note
1	Length of Sepals	
2	Width of Sepals	
3	Length of Petals	
4	Width of Petals	
5	class	Iris setosa Iris virginica Iris versicolor
6	class1	1: Iris setosa 2: Iris virginica 3: Iris versicolor
7	class1_1	1: Iris setosa 0: Not Iris setosa
8	class1_2	1: Iris virginica 0: Not Iris virginica
9	Class1_3	1: Iris versicolor 0: Not Iris versicolor

Example 1. Three Species of Iris (Step 1. Business Understanding)

- Business Understanding



Example 1. Three Species of Iris (Step 2. Data Understanding 1)

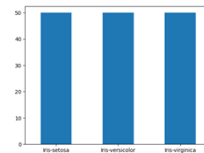
- #Open File
- from google.colab import drive
- drive.mount('/content/MyGoogleDrive')
- import pandas
- df=pandas.read_csv('/content/MyGoogleDrive/My Drive/iris_1.csv')
- df

Example 1. Three Species of Iris (Step 2. Data Understanding 2)

- `df.info()`
- `df.describe()`
- `df['class'].value_counts()`

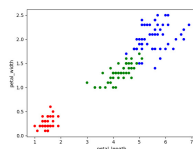
Example 1. Three Species of Iris (Step 2. Data Understanding 3)

- `df['class'].value_counts().plot(kind='bar',rot=0)`



Example 1. Three Species of Iris (Step 2. Data Understanding 4)

- `color_code={'Iris-setosa': 'r', 'Iris-versicolor': 'g', 'Iris-virginica': 'b'}`
- `df['class_color']=df['class'].map(color_code)`
- `df.plot(kind='scatter', x='petal_length', y='petal_width', c='class_color')`



Example 1. Three Species of Iris (Step 3. Data Preparation 1)

- `df.drop_duplicates()`
- `df.dropna(how='any')`
- `df['sepal_length'].fillna(value=df['sepal_length'].mean(), inplace=True)`



Example 1. Three Species of Iris (Step 3. Data Preparation 2)

- `# Split the dataset into training and testing sets`
- `from sklearn.model_selection import train_test_split`
- `x=df[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']]`
- `y=df['class']`
- `x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)`
- `data_train=pandas.concat([x_train,y_train],axis=1)`
- `data_test=pandas.concat([x_test,y_test],axis=1)`

Example 1. Three Species of Iris (Step 4. Modeling 1)

- `from sklearn.neighbors import KNeighborsClassifier`
- `my_knn=KNeighborsClassifier(n_neighbors=3)`
- `my_knn.fit(x_train,y_train)`
- `my_knn.score(x_test,y_test)`

Example 1. Three Species of Iris (Step 4. Modeling 2)

- `from sklearn.metrics import accuracy_score`
- `pred= my_knn.predict(x_test)`
- `accuracy_score(y_test,pred)`

Example 1. Three Species of Iris (Step 4. Modeling 3)

- #Confusion Matrix
- `from sklearn.metrics import confusion_matrix`
- `pred= my_knn.predict(x_test)`
- `confusion_matrix(y_test,pred)`

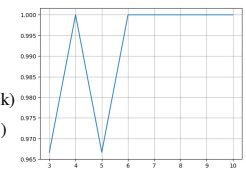
		0	1	
Actual label	0	TN	FP	0 : Negative 1 : Positive
	1	FN	TP	
		Predict label		

Example 1. Three Species of Iris (Step 5. Evaluation 1)

- `from sklearn.neighbors import KNeighborsClassifier`
- `for k in range(3,11):`
- `my_knn=KNeighborsClassifier(n_neighbors=k)`
- `my_knn.fit(x_train,y_train)`
- `acc= my_knn.score(x_test,y_test)`
- `print('k=',k,' ',acc)`

Example 1. Three Species of Iris (Step 5. Evaluation 2)

- `from sklearn.neighbors import KNeighborsClassifier`
- `import pandas`
- `import pandas`
- `myscore=[]`
- `for k in range(3,11):`
- `my_knn=KNeighborsClassifier(n_neighbors=k)`
- `my_knn.fit(x_train,y_train)`
- `myscore.append(my_knn.score(x_test,y_test))`
- `df_knn=pandas.DataFrame(myscore)`
- `df_knn.index=[3,4,5,6,7,8,9,10]`
- `df_knn.plot(kind='line', grid=True, legend=False)`

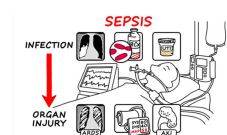


Example 1. Three Species of Iris (Step 6. Deployment)

- # k=4 is best!
- `from sklearn.neighbors import KNeighborsClassifier`
- `best_knn=KNeighborsClassifier(n_neighbors=4)`
- `best_knn.fit(x_train, y_train)`
- `x_new=[[5.1, 3.5, 1.4, 0.2],[7. 3.2, 4.7, 1.4],[6.3, 3.3, 6, 2.5]]`
- `best_knn.predict(x_new)`

Example 2. The Type of Sepsis (1)

- Predict the type of sepsis (urosepsis, bile duct sepsis, and airway sepsis) based on the laboratory screenings of 45 patients.



- Dataset: sepsis_1.csv
- Source: https://rstudio-pubs-static.s3.amazonaws.com/200568_9c836d5f0337451c8841e707395c1ae1.html

Example 2. The Type of Sepsis (2)

• Code Book

Column	Note
-gammaagt	gammaagt
-asat	asat
-alat	alat
-bili	bilirubin
-ureum	ureum
-creatinine	creatinine
-creatinineclearance	creatinine clearance
-erythrocytesr	erythrocyte sedimentation rate
-reactiveprotein	reactive protein
-leucos	leucocyte count
diagnosisgroup	1: unsepsis 2: bile duct sepsis 3: airway sepsis
diagnosisgroup1	11: unsepsis 12: bile duct sepsis 13: airway sepsis
diagnosisgroup2	unsepsis bile duct sepsis airway sepsis
11	1: unsepsis 0: Not unsepsis
12	1: bile duct sepsis 0: Not bile duct sepsis
13	1: airway sepsis 0: Not airway sepsis

© 賴宏弘

資料分析方法與應用 2025E

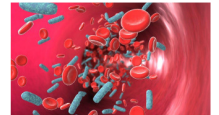
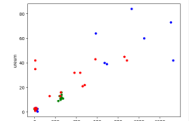
31

Example 2. The Type of Sepsis (Step 1. Business understanding)

• Business Understanding

-gammaagt
-asat
-alat
-bili
-ureum
-creatinine
-creatinineclearance
-erythrocytesr
-reactiveprotein
-leucos

The type of sepsis [diagnosisgroup2]



© 賴宏弘

資料分析方法與應用 2025E

32

Example 3. Indian Liver Patient Dataset (1)

- Death by liver cirrhosis continues to increase, given the increase in alcohol consumption rates, chronic hepatitis infections, and obesity-related liver disease.
- Notwithstanding the high mortality of this disease, liver diseases do not affect all sub-populations equally.
- The early detection of pathology is a determinant of patient outcomes, yet female patients appear to be marginalized when it comes to early diagnosis of liver pathology.
- The dataset comprises 584 patient records collected from the NorthEast of Andhra Pradesh, India.
- The prediction task is to determine whether a patient suffers from liver disease based on the information about several biochemical markers, including albumin and other enzymes required for metabolism.
- This data set contains records of 416 patients diagnosed with liver disease and 167 patients without liver disease. This information is contained in the class label named 'Selector'. There are 10 variables per patient: age, gender, total Bilirubin, direct Bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT and Alkphos. Of the 583 patient records, 441 are male, and 142 are female.

- Dataset: IndianLiver_1.csv
- Source: <https://archive.ics.uci.edu/dataset/225/indian+live+patient+dataset>



© 賴宏弘

資料分析方法與應用 2025E

33

Example 3. Indian Liver Patient Dataset (2)

• Code Book

Feature	Description	Type
Age	Age of the patient. Any patient whose age exceeded 89 is listed as being of age "90".	Numerical
Gender	Gender of the patient	Categorical
TB	Total Bilirubin	Numerical
DB	Direct Bilirubin	Numerical
Alkphos	Alkaline Phosphatase	Numerical
Sgpt	Alanine Aminotransferase	Numerical
Sgot	Aspartate Aminotransferase	Numerical
TP	Total Proteins	Numerical
ALB	Albumin	Numerical
A/G Ratio	Albumin and Globulin Ratio	Numerical
Label	1: with liver disease 2: without liver disease	Categorical
Gender1	1: Male 0: Female	
Label1	1: with liver disease 0: without liver disease	

© 賴宏弘

資料分析方法與應用 2025E

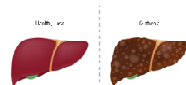
34

Example 3. Indian Liver Patient Dataset (3) (Step 1. Business understanding)

• Business Understanding

-Age of the patient [Age]
-Gender of the patient [Gender1]
-Total Bilirubin [TB]
-Direct Bilirubin [DB]
-Alkaline Phosphatase [Alkphos]
-Alanine Aminotransferase [Sgpt]
-Aspartate Aminotransferase [Sgot]
-Total Proteins [TP]
-Albumin [ALB]
-Albumin and Globulin Ratio [A/G Ratio]

Label1



© 賴宏弘

資料分析方法與應用 2025E

35

Example 4. The Type of Penguins (1)

- The dataset contains 333 instances of penguins from three species (classes), Adelie, Chinstrap, and Gentoo. The Adelie species contains 146 instances of penguins, Chinstrap has 68, and Gentoo has 119.



- Dataset: penguins_size1.csv

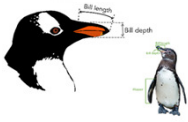
© 賴宏弘

資料分析方法與應用 2025E

36

Example 4. The Type of Penguins (2)

- Code Book



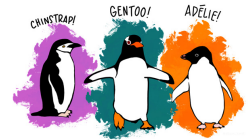
Feature	Description	Type
species	Adelie, Chinstrap, or Gentoo	
species1	1: Adelie, 2: Chinstrap, or 3: Gentoo	Categorical
island	The name of the island (Dream, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica) where the penguin was found and measured.	
island1	1: Dream, 2: Torgersen, or 3: Biscoe	
culmen_length_mm	The length of the penguin's culmen in millimeters.	Numerical
culmen_depth_mm	The depth of the penguin's culmen in millimeters.	Numerical
flipper_length_mm	The length of the penguin's flippers in millimeters.	Numerical
body_mass_g	The body mass of the penguin in grams.	Numerical
sex	The sex (MALE, FEMALE) of the penguin.	Categorical
sex1	1: MALE, 0: FEMALE	
species1_1	1: Adelie	
species1_2	0: Not Adelie	
species1_2	1: Chinstrap	
species1_3	0: Not Chinstrap	
species1_3	1: Gentoo	
species1_3	0: Not Gentoo	

Example 4. The Type of Penguins (3) (Step 1. Business understanding)

- Business Understanding

-culmen_length_mm
-culmen_depth_mm
-flipper_length_mm
-body_mass_g

→ The type of penguins [species]



5. K-means Cluster Analysis

Course Outline

- K-means Clustering
- Key Features of K-means Clustering
- Limitations of K-means Clustering
- Disadvantages of K-means Clustering
- Expectation-Maximization: K-means Algorithm
- Working of K-means Algorithm
- Stopping Criteria for K-Means Clustering
- K-means vs Hierarchical Clustering



K-means Cluster Analysis

- K-means algorithm explores for a preplanned number of clusters in an un-labelled multidimensional dataset, it concludes this via an easy interpretation of how an optimized cluster can be expressed.
- Primarily the concept would be in two steps:
 - Firstly, the cluster center is the arithmetic mean (AM) of all the data points associated with the cluster.
 - Secondly, each point is adjoint to its cluster center in comparison to other cluster centers. These two interpretations are the foundation of the k-means clustering model.

K-means Cluster Analysis

- You can take the centre as a data point that outlines the means of the cluster, also it might not possibly be a member of the dataset.
- In simple terms, k-means clustering enables us to cluster the data into several groups by detecting the distinct categories of groups in the unlabelled datasets by itself, even without the necessity of training of data.
- This is the centroid-based algorithm such that each cluster is connected to a centroid while following the objective to minimize the sum of distances between the data points and their corresponding clusters.
- As an input, the algorithm consumes an unlabelled dataset, splits the complete dataset into k-number of clusters, and iterates the process to meet the right clusters, and the value of k should be predetermined.

The K-means Algorithm

- Specifically performing two tasks, the k-means algorithm
 - Calculates the correct value of K-center points or centroids by an iterative method
 - Assigns every data point to its nearest k-center, and the data points, closer to a particular k-center, make a cluster. Therefore, data points, in each cluster, have some similarities and far apart from other clusters.

Key Features of K-means Clustering

- It is very smooth in terms of interpretation and resolution.
- For a large number of variables present in the dataset, K-means operates quicker than Hierarchical clustering.
- While redetermining the cluster centre, an instance can modify the cluster.
- K-means reforms compact clusters.
- It can work on unlabeled numerical data.
- Moreover, it is fast, robust and uncomplicated to understand and yields the best outcomes when datasets are well distinctive (thoroughly separated) from each other.

Limitations of K-means Clustering



- Sometimes, it is quite tough to forecast the number of clusters, or the value of k .
- The output is highly influenced by original input, for example, the number of clusters.
- An array of data substantially hits the concluding outcomes.
- In some cases, clusters show complex spatial views, then executing clustering is not a good choice.
- Also, rescaling is sometimes conscious, it can't be done by normalization or standardization of data points, the output gets changed entirely.

Disadvantages of K-means Clustering



- The algorithm demands for the inferred specification of the number of cluster/centers.
- An algorithm goes down for non-linear sets of data and unable to deal with noisy data and outliers.
- It is not directly applicable to categorical data since only operatable when mean is provided.
- Also, Euclidean distance can weight unequally the underlying factors.
- The algorithm is not variant to non-linear transformation, i.e provides different results with different portrayals of data.

Expectation-Maximization: K-means Algorithm (1)



- K-Means is just the Expectation-Maximization (EM) algorithm, It is a persuasive algorithm that exhibits a variety of context in data science, the E-M approach incorporates two parts in its procedure;
 - To assume some cluster centres,
 - Re-run as far as transformed;
 - E-Step: To appoint data points to the closest cluster centre,
 - M-Step: To introduce the cluster centres to the mean.

Expectation-Maximization: K-means Algorithm (2)



- Where the E-step is the Expectation step, it comprises upgrading forecasts of associating the data point with the respective cluster.
- And, M-step is the Maximization step, it includes maximizing some features that specify the region of the cluster centres, for this maximization, is expressed by considering the mean of the data points of each cluster.
- In account with some critical possibilities, each iteration of E-step and M-step algorithm will always yield in terms of improved estimation of clusters' characteristics.
- K-means utilize an iterative procedure to yield its final clustering based on the number of predefined clusters, as per need according to the dataset and represented by the variable K .
- For instance, if K is set to 3 (k_3), then the dataset would be categorized in 3 clusters if k is equal to 4, then the number of clusters will be 4 and so on.

Expectation-Maximization: K-means Algorithm (3)



- The fundamental aim is to define k centres, one for each cluster, these centres must be located in a sharp manner because of the various allocation causes different outcomes. So, it would be best to put them as far away as possible from each other.
- Also, The maximum number of plausible clusters will be the same as the total number of observations/features present in the dataset.

Working of K-means Algorithm



- K-centres are modelled randomly in accordance with the present value of K .
- K-means assigns each data point in the dataset to the adjacent centre and attempts to curtail Euclidean distance between data points. Data points are assumed to be present in the peculiar cluster as if it is nearby to centre to that cluster than any other cluster centre.
- After that, k-means determines the centre by accounting the mean of all data points referred to that cluster centre. It reduces the complete variance of the intra-clusters with respect to the prior step. Here, the "means" defines the average of data points and identifies a new centre in the method of k-means clustering.
- The algorithm gets repeated among the steps 2 and 3 till some paradigm will be achieved such as the sum of distances in between data points and their respective centres are diminished, an appropriate number of iterations is attained, no variation in the value of cluster centre or no change in the cluster due to data points.

Stopping Criteria for K-Means Clustering

- If the centroids of the newly built clusters are not changing
 - An algorithm can be brought to an end if the centroids of the newly constructed clusters are not altering. Even after multiple iterations, if the obtained centroids are same for all the clusters, it can be concluded that the algorithm is not learning any new pattern and gives a sign to stop its execution/training to a dataset.
- If data points remain in the same cluster
 - The training process can also be halt if the data points stay in the same cluster even after the training the algorithm for multiple iterations.
- If the maximum number of iterations have achieved
 - At last, the training on a dataset can also be stopped if the maximum number of iterations is attained, for example, assume the number of iterations has set as 200, then the process will be repeated for 200 times (200 iterations) before coming to end.

K-means vs Hierarchical Clustering

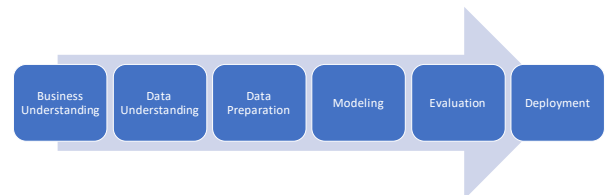
- K-means clustering produces a specific number of clusters for the disarranged and flat dataset, where Hierarchical clustering builds a hierarchy of clusters, not for just a partition of objects under various clustering methods and applications.
- K-means can be used for categorical data and first converted into numeric by assigning rank, where Hierarchical clustering was selected for categorical data but due to its complexity, a new technique is considered to assign rank value to categorical features.
- K-means are highly sensitive to noise in the dataset and perform well than Hierarchical clustering where it is less sensitive to noise in a dataset.
- Performance of the K-Means algorithm increases as the RMSE decreases and the RMSE decreases as the number of clusters increases so the time of execution increases, in contrast to this, the performance of Hierarchical clustering is less.
- K-means are good for a large dataset and Hierarchical clustering is good for small datasets.

Final Words

- K-means clustering is the unsupervised machine learning algorithm that is part of a much deep pool of data techniques and operations in the realm of Data Science. It is the fastest and most efficient algorithm to categorize data points into groups even when very little information is available about data.
- More on, similar to other unsupervised learning, it is necessary to understand the data before adopting which technique fits well on a given dataset to solve problems. Considering the correct algorithm, in return, can save time and efforts and assist in obtaining more accurate results.

Process

- Process



Example 1. Three Species of Iris (1)

- The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor).
- The famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.
- Dataset: iris_1.csv



Example 1. Three Species of Iris (2)

- Code Book

	Column	Note
1	Length of Sepals	
2	Width of Sepals	
3	Length of Petals	
4	Width of Petals	
5	class	0-Iris-setosa 1-Iris-virginica 2-Iris-versicolor
6	class_1	1-Iris-setosa 2-Iris-virginica 3-Iris-versicolor
7	class_1_1	1-Iris-setosa 0-Not-Iris-setosa
8	class_1_2	1-Iris-virginica 0-Not-Iris-virginica
9	class_1_3	1-Iris-versicolor 0-Not-Iris-versicolor

Example 1. Three Species of Iris (Step 1. Business Understanding)

- Business Understanding

```
-Width of Petals [petal_width]  
-Length of Petals [petal_length]  
-Width of Sepals [sepal_width]  
-Length of Sepals [sepal_length]
```

classify



Example 1. Three Species of Iris (Step 2. Data Understanding 1)

- #Open File
- from google.colab import drive
- drive.mount('/content/MyGoogleDrive')
- import pandas
- df=pandas.read_csv('/content/MyGoogleDrive/My Drive/iris_1.csv')
- df

Example 1. Three Species of Iris (Step 2. Data Understanding 2)

- df.info()
- df.describe()

Example 1. Three Species of Iris (Step 3. Data Preparation 1)

- df.drop_duplicates()
- df.dropna(how='any')
- df['sepal_length'].fillna(value=df['sepal_length'].mean(), inplace=True)

Example 1. Three Species of Iris (Step 3. Data Preparation 2)

- x_train=df[['sepal_length','sepal_width','petal_length','petal_width']]
- x_train

Example 1. Three Species of Iris (Step 4. Modeling)

- from sklearn.cluster import KMeans
- my_km=KMeans(n_clusters=3)
- my_km.fit(x_train)
- my_km.inertia_ **#small is good**

Example 1. Three Species of Iris (Step 5. Evaluation 1)

- from sklearn.cluster import KMeans
- for k in range(1,11):
- my_km=KMeans(n_clusters=k)
- my_km.fit(x_train)
- print('k=',k,'==>',my_km.inertia_) #small is good

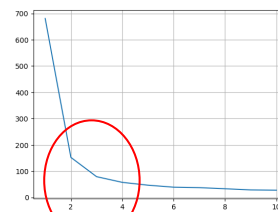
© 赖克弘

资料分析方法与案例应用2025E

25

Example 1. Three Species of Iris (Step 5. Evaluation 2)

- from sklearn.cluster import KMeans
- import pandas
- myscore=[]
- for k in range(1,11):
- my_km=KMeans(n_clusters=k)
- my_km.fit(x_train)
- myscore.append(my_km.inertia_)
- df_km=pandas.DataFrame(myscore)
- df_km.index=[1,2,3,4,5,6,7,8,9,10]
- df_km.plot(kind='line', grid=True, legend=False)



© 赖克弘

资料分析方法与案例应用2025E

26

Example 1. Three Species of Iris (Step 6. Deployment 1)

- # k=3 is better!
- best_km=KMeans(n_clusters=3)
- pred_y=best_km.fit_predict(x_train)
- pred_y
- fit() : Computes the centroids and assigns each data point to a cluster, but does not return the labels for each point.
- fit_predict() : Performs the same operations as fit() and also returns the labels for each data point.

© 赖克弘

资料分析方法与案例应用2025E

27

Example 1. Three Species of Iris (Step 6. Deployment 2)

- df1=x_train.copy()
- df1['pred_y']=pred_y
- df1

	sepal_length	sepal_width	petal_length	petal_width	pred_y
0	5.1	3.5	1.4	0.2	1
1	4.9	3.0	1.4	0.2	1
2	4.7	3.2	1.3	0.2	1
3	4.6	3.1	1.5	0.2	1
4	5.0	3.6	1.4	0.2	1
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	0
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	0
148	6.2	3.4	5.4	2.3	0
149	5.9	3.0	5.1	1.8	2

150 rows x 6 columns

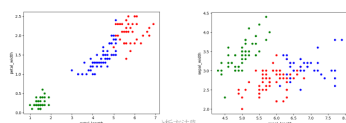
© 赖克弘

资料分析方法与案例应用2025E

28

Example 1. Three Species of Iris (Step 6. Deployment 3)

- pred_y_color={0:'r',1:'g',2:'b'}
- df1['y_color']=df1['pred_y'].map(pred_y_color)
- df1.plot(kind='scatter',x='petal_length',y='petal_width',c=df1['y_color'])
- df1.plot(kind='scatter',x='sepal_length',y='sepal_width',c=df1['y_color'])



© 赖克弘

资料分析方法

29

Example 1. Three Species of Iris (Step 6. Deployment 4)

- x_new=[[6.7,3.0,5.2,2.3],[5.1,3.5,1.4,0.2],[5.9,3.0,5.1,1.8]]
- best_km.predict(x_new)



© 赖克弘

资料分析方法与案例应用2025E

30

Example 2. The Type of Sepsis (1)

- Predict the type of sepsis (urosepsis, bile duct sepsis, and airway sepsis) based on the laboratory screenings of 45 patients.



- Dataset: sepsis_1.csv
- Source: https://rstudio-pubs-static.s3.amazonaws.com/200568_9c836d5f0337451c8841e707395c1ae1.html

© 顧宜弘

資料分析方法與應用 2025E

31

Example 2. The Type of Sepsis (2)

- Code Book



Column	Note
gammagt	gammagt
asat	asat
alat	alat
bili	bilirubin
ureum	ureum
creatinine	creatinine
creatinineclearance	creatinine clearance
erythrocytes	erythrocytes (red blood cell count)
creactiveprotein	creactive protein
leucos	leucocyte count
diagnosis.group	1: urosepsis 2: bile duct sepsis 3: airway sepsis
diagnosis.group1	1: urosepsis 2: bile duct sepsis 3: airway sepsis
diagnosis.group2	urosepsis bile duct sepsis airway sepsis
t1	1: urosepsis 2: Not urosepsis
t2	1: bile duct sepsis 2: Not bile duct sepsis
t3	1: airway sepsis 2: Not airway sepsis

© 顧宜弘

資料分析方法與應用 2025E

32

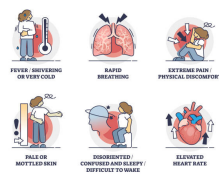
Example 2. The Type of Sepsis (Step 1. Business understanding)

- Business Understanding

-gammagt
-asat
-alat
-bili
-ureum
-creatinine
-creatinineclearance
-erythrocytes
-creactiveprotein
-leucos

classify

SIGNS OF SEPSIS



© 顧宜弘

資料分析方法與應用 2025E

33

Example 3. The Type of Penguins (1)

- The dataset contains 333 instances of penguins from three species (classes), Adelie, Chinstrap, and Gentoo. The Adelie species contains 146 instances of penguins, Chinstrap has 68, and Gentoo has 119.



- Dataset: penguins_size1.csv

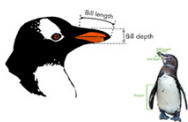
© 顧宜弘

資料分析方法與應用 2025E

34

Example 3. The Type of Penguins (2)

- Code Book



Feature	Description	Type
species	Adelie, Chinstrap, or Gentoo	Categorical
species1	1: Adelie, 2: Chinstrap, or 3: Gentoo	C
island	The name of the island (Dream, Torgersen, or Raccoon) in the Palmer Archipelago (Antarctica) where the penguin was found and measured	Categorical
island1	1: Dream, 2: Torgersen, or 3: Raccoon	C
culmen_length_mm	The length of the penguin's culmen in millimeters.	Numerical
culmen_depth_mm	The depth of the penguin's culmen in millimeters.	Numerical
flipper_length_mm	The length of the penguin's flippers in millimeters.	Numerical
body_mass_g	The body mass of the penguin in grams.	Numerical
sex	The sex (MALE, FEMALE) of the penguin.	Categorical
sex1	1: MALE, 2: FEMALE	C
species1_1	1: Adelie	
species1_2	2: Not Adelie	
species1_3	1: Chinstrap	
species1_4	2: Not Chinstrap	
species1_5	1: Gentoo	
species1_6	2: Not Gentoo	

© 顧宜弘

資料分析方法與應用 2025E

35

Example 3. The Type of Penguins (Step 1. Business understanding)

- Business Understanding

-culmen_length_mm
-culmen_depth_mm
-flipper_length_mm
-body_mass_g

PENGUIN SPECIES



© 顧宜弘

資料分析方法與應用 2025E

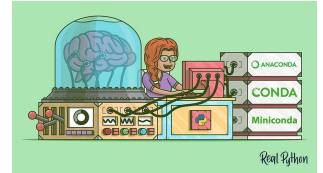
36



6. Multilayer Perceptron

Course Outline

- Multilayer Perceptron
- MLP Diagram
- Neural Network Models in Keras
- Model Compilation

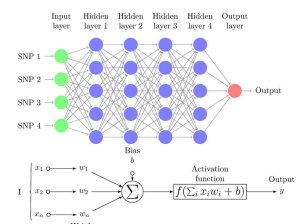


Multilayer Perceptron

- A multilayer perceptron (MLP) is a feedforward artificial neural network that generates a set of outputs from a set of inputs.
- An MLP is characterized by several layers of input nodes connected as a directed graph between the input and output layers.
- MLP uses backpropagation for training the network.
- MLP is a deep learning method.

MLP Diagram

- Multi-Layer Perceptron (MLP) diagram with four hidden layers and a collection of single nucleotide polymorphisms (SNPs) as input and illustrates a basic "neuron" with n inputs.
- One neuron is the result of applying the nonlinear transformations of linear combinations (x_i , w_i , and biases b).



Neural Network Models in Keras

- Create a Sequential model and add the layers in the order of the computation you wish to perform.
 - `from keras.models import Sequential`
 - `model = Sequential()`
 - `model.add(...)`
 - `model.add(...)`
 - `model.add(...)`

Model Layers: Layer Types

- Dense: Fully connected layer and the most common type of layer used on multi-layer perceptron models
 - The input is XXX features, and the output is XXX neurons, that is, XXX regression lines.
 - Output XXX neurons and convert them into probabilities through the softmax activation function, that is, predicted probabilities from 0 to 9. The one with the highest probability is selected as the predicted value.
- Dropout: Apply dropout to the model, setting a fraction of inputs to zero in an effort to reduce overfitting
 - Randomly discard XXX% of neurons during the training cycle to correct the overfitting phenomenon.
- Concatenate: Combine the outputs from multiple layers as input to a single layer

Model Layers: Weight Initialization

- The type of initialization used for a layer is specified in the `kernel_initializer` argument.
- `random_uniform`: Weights are initialized to small uniformly random values between -0.05 and 0.05.
- `random_normal`: Weights are initialized to small Gaussian random values (zero mean and standard deviation of 0.05).
- `zeros`: All weights are set to zero values.

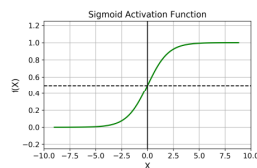
Model Layers: Activation Function

- `deserialize(...)`: Returns activation function given a string identifier.
- `elu(...)`: Exponential Linear Unit.
- `exponential(...)`: Exponential activation function.
- `gelu(...)`: Applies the Gaussian error linear unit (GELU) activation function.
- `get(...)`: Returns function.
- `hard_sigmoid(...)`: Hard sigmoid activation function.
- `linear(...)`: Linear activation function (pass-through).
- `relu(...)`: Applies the rectified linear unit activation function.
- `selu(...)`: Scaled Exponential Linear Unit (SELU).
- `serialize(...)`: Returns the string identifier of an activation function.
- `sigmoid(...)`: Sigmoid activation function, $\text{sigmoid}(x) = 1 / (1 + \exp(-x))$.
- `softmax(...)`: Softmax converts a vector of values to a probability distribution.
- `softplus(...)`: Softplus activation function, $\text{softplus}(x) = \log(\exp(x) + 1)$.
- `softsign(...)`: Softsign activation function, $\text{softsign}(x) = x / (\text{abs}(x) + 1)$.
- `swish(...)`: Swish activation function, $\text{swish}(x) = x * \text{sigmoid}(x)$.
- `tanh(...)`: Hyperbolic tangent activation function.

Model Layers: Activation Function (sigmoid)

- `sigmoid`
 - Values are between [0,1] and the distribution is polarized, with most being either 0 or 1.
 - Suitable for binary classification.

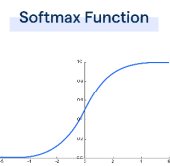
$$f(x) = \frac{1}{1 + e^{-x}}$$



Model Layers: Activation Function (softmax)

- `softmax`
 - Values are between [0,1] and the probabilities sum to 1.
 - Suitable for use in multiple categories.

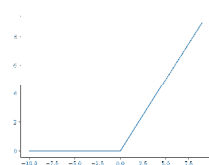
$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^J e^{x_j}}$$



Model Layers: Activation Function (relu)

- `relu`
 - Negative values between $[0, \infty]$ are ignored.

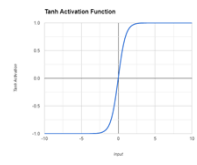
$$f(x) = \max(x, 0)$$



Model Layers: Activation Function (tanh)

- `tanh`
 - Similar to sigmoid, but the value is between [-1,1].
 - The conduction has a negative value

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$



Model Compilation: Model Loss Functions

- The loss function, also called the objective function, is the evaluation of the model used by the optimizer to navigate the weight space.
- 'mse': for mean squared error
- 'binary_crossentropy': for binary logarithmic loss (logloss)
- 'categorical_crossentropy': for multi-class logarithmic loss (logloss)

Model Compilation: Model Optimizers

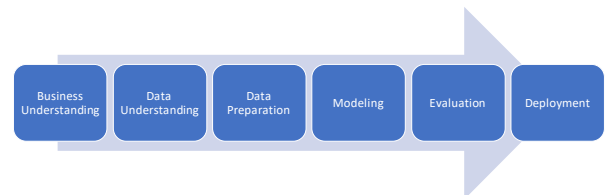
- The optimizer is the search technique used to update weights in your model.
- SGD: stochastic gradient descent, with support for momentum
- RMSprop: adaptive learning rate optimization method proposed by Geoff Hinton
- Adam: Adaptive Moment Estimation (Adam) that also uses adaptive learning rates

Model Compilation: Model Metrics

- Keras Regression Metrics
 - Mean Squared Error: mean_squared_error, MSE or mse
 - Mean Absolute Error: mean_absolute_error, MAE, mae
 - Mean Absolute Percentage Error: mean_absolute_percentage_error, MAPE, mape
 - Cosine Proximity: cosine_proximity, cosine
- Keras Classification Metrics
 - Binary Accuracy: binary_accuracy, acc
 - Categorical Accuracy: categorical_accuracy, acc
 - Sparse Categorical Accuracy: sparse_categorical_accuracy
 - Top k Categorical Accuracy: top_k_categorical_accuracy (requires you specify a k parameter)
 - Sparse Top k Categorical Accuracy: sparse_top_k_categorical_accuracy (requires you specify a k parameter)

Process

- Process



Example 1. Three Species of Iris (1)

- The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor).
- The famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.
- Dataset: iris_1.csv



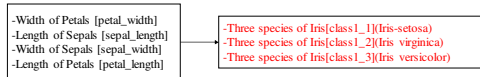
Example 1. Three Species of Iris (2)

- Code Book

	Column	Note
1	Length of Sepals	
2	Width of Sepals	
3	Length of Petals	
4	Width of Petals	
5	class	Iris setosa Iris virginica Iris versicolor
6	class1	1: Iris setosa 2: Iris virginica 3: Iris versicolor
7	class1_1	1: Iris setosa 0: Not Iris setosa
8	class1_2	1: Iris virginica 0: Not Iris virginica
9	Class1_3	1: Iris versicolor 0: Not Iris versicolor

Example 1. Three Species of Iris (Step 1. Business Understanding)

- Business Understanding



Example 1. Three species of Iris (Step 2. Data Understanding 1)

- #Open File
- from google.colab import drive
- drive.mount('/content/MyGoogleDrive')
- import pandas
- df=pandas.read_csv('/content/MyGoogleDrive/My Drive/iris_1.csv')
- df

Example 1. Three species of Iris (Step 2. Data Understanding 2)

- df.info()
- df.describe()
- df['class1_1'].value_counts()
- df['class1_1'].value_counts().plot(kind='bar',rot=0)



Example 1. Three species of Iris (Step 3. Data Preparation 1)

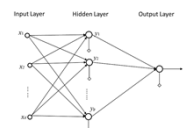
- df.drop_duplicates()
- df.dropna(how='any')
- df['sepal_length'].fillna(value=df['sepal_length'].mean(), inplace=True)

Example 1. Three species of Iris (Step 3. Data Preparation 2)

- from sklearn.model_selection import train_test_split
- y=df[['class1_1', 'class1_2','class1_3']]
- x=df[['sepal_length','sepal_width','petal_length','petal_width']]
- x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)

Example 1. Three species of Iris (Step 4. Modeling 1)

- from keras.models import Sequential
- from keras.layers import Dense, Dropout
- model=Sequential()
- #Input layer with 4 inputs neurons
- #model.add(Dense(units=40, input_dim=4, kernel_initializer='uniform', activation='relu'))
- model.add(Dense(units=40, input_shape=(4,), kernel_initializer='uniform', activation='relu'))
- #Hidden layer
- model.add(Dense(units=30, kernel_initializer='uniform', activation='relu'))
- #Output layer with 3 output neuron which will predict 1 or 0
- model.add(Dense(units=3, kernel_initializer='uniform', activation='sigmoid'))
- model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
- print(model.summary())



Example 1. Three species of Iris (Step 4. Modeling 2)

- `train_history=model.fit(x=x_train, y=y_train, validation_split=0.1, epochs=100, batch_size=30, verbose=2)`

Example 1. Three species of Iris (Step 5. Evaluation 1)

- `# test loss, test acc`
- `score=model.evaluate(x=x_test,y=y_test)`
- `score`

Example 1. Three species of Iris (Step 5. Evaluation 2: test vs validation)

- The validation set is used during the training phase of the model to provide an unbiased evaluation of the model's performance and to fine-tune the model's parameters.
- The test set is used after the model has been fully trained to assess the model's performance on completely unseen data.

Example 1. Three species of Iris (Step 5. Evaluation 2: Training loss)

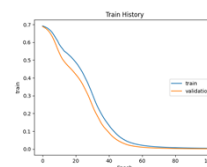
- Training loss is the calculated error when the model makes predictions on the training data.
- It is updated after every forward and backward pass of the model during the training process.
- The loss typically decreases over time as the model learns to map inputs to outputs more accurately.
- A loss function (such as Mean Squared Error, Cross-Entropy Loss, etc.) quantifies the difference between the predicted and actual labels.

Example 1. Three species of Iris (Step 5. Evaluation 2: Validation loss)

- Validation loss evaluates the model's performance on a separate dataset (validation set) that the model has never seen during training.
- This metric provides an indication of how well the model generalizes to new data.
- Validation loss is computed at the end of each epoch during training but is not used to update the model weights.

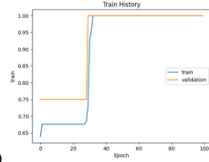
Example 1. Three species of Iris (Step 5. Evaluation 2: Training/Validation Loss)

- `import matplotlib.pyplot as plt`
- `plt.plot(train_history.history["loss"])`
- `plt.plot(train_history.history["val_loss"])`
- `plt.title('Train History')`
- `plt.ylabel('train')`
- `plt.xlabel('Epoch')`
- `plt.legend(['train', 'validation'], loc='center right')`
- `plt.show()`



Example 1. Three species of Iris (Step 5. Evaluation 3: Training/Validation Accuracy)

- `import matplotlib.pyplot as plt`
- `plt.plot(train_history.history["accuracy"])`
- `plt.plot(train_history.history["val_accuracy"])`
- `plt.title("Train History")`
- `plt.ylabel('train')`
- `plt.xlabel('Epoch')`
- `plt.legend(['train', 'validation'], loc='center right')`
- `plt.show()`



© 赖立弘

資料分析方法與應用2025E

31

Example 1. Three species of Iris (Step 5. Evaluation 4)

- `y_test_pred=model.predict(x_test)`
- `y_test_pred`

© 赖立弘

資料分析方法與應用2025E

32

Example 1. Three species of Iris (Step 6. Deployment)

- `x_new=[[5.1,3.5,1.4,0.2],[7.0,3.2,4.7,1.4],[6.3,3.3,6.0,2.5]]`
- `y_new=model.predict(x_new)`
- `y_new`

© 赖立弘

資料分析方法與應用2025E

33

Example 1b. Three Species of Iris (1)

- The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor).
- The famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.
- Dataset: iris_1.csv



© 赖立弘

資料分析方法與應用2025E

34

Example 1b. Three Species of Iris (2)

- Code Book

	Column	Note
1	Length of Sepals	
2	Width of Sepals	
3	Length of Petals	
4	Width of Petals	
5	class	Iris setosa Iris virginica Iris versicolor
6	class 1	1: Iris setosa 2: Iris virginica 3: Iris versicolor
8	class 1_1	1: Iris setosa 0: Not Iris setosa
9	class 1_2	1: Iris virginica 0: Not Iris virginica
10	Class 1_3	1: Iris versicolor 0: Not Iris versicolor

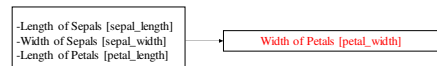
© 赖立弘

資料分析方法與應用2025E

35

Example 1b. Three Species of Iris (Step 1. Business Understanding)

- Business Understanding



© 赖立弘

資料分析方法與應用2025E

36

Example 1b. Three species of Iris (Step 2. Data Understanding 1)

- #Open File
- from google.colab import drive
- drive.mount('/content/MyGoogleDrive')
- import pandas
- df=pandas.read_csv('/content/MyGoogleDrive/My Drive/資料分析方法與應用/iris_1.csv')
- df

Example 1b. Three species of Iris (Step 2. Data Understanding 2)

- df.info()
- df.describe()

Example 1b. Three species of Iris (Step 3. Data Preparation 1)

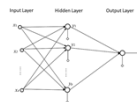
- df.drop_duplicates()
- df.dropna(how='any')
- df['sepal_length'].fillna(value=df['sepal_length'].mean(), inplace=True)

Example 1b. Three species of Iris (Step 3. Data Preparation 2)

- from sklearn.model_selection import train_test_split
- x=df[['sepal_length','sepal_width','petal_length']]
- y=df['petal_width']
- x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)

Example 1b. Three species of Iris (Step 4. Modeling 1)

- from keras.models import Sequential
- from keras.layers import Dense, Dropout
- model=Sequential()
- #Input layer with 3 inputs neurons
- model.add(Dense(units=40, input_dim=3, kernel_initializer='random_uniform', activation='relu'))
- model.add(Dropout(0.5))
- #Hidden layer
- model.add(Dense(units=30, kernel_initializer='random_uniform', activation='relu'))
- model.add(Dropout(0.5))
- #Output layer with 1 output neuron which will predict 1 or 0
- model.add(Dense(units=1, kernel_initializer='random_uniform'))
- model.compile(loss='mean_squared_error',optimizer='adam',metrics=['mse'])
- print(model.summary())



Example 1b. Three species of Iris (Step 4. Modeling 2)

- train_history=model.fit(x=x_train, y=y_train, validation_split=0.1, epochs=100, batch_size=30, verbose=2)

Example 1b. Three species of Iris (Step 5. Evaluation 1)

- # mse, mse
- score=model.evaluate(x=x_test,y=y_test)
- score

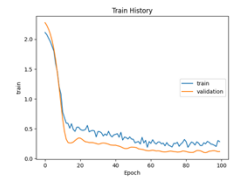
© 賴立弘

資料分析方法與應用2025E

43

Example 1b. Three species of Iris (Step 5. Evaluation 2: MSE)

- import matplotlib.pyplot as plt
- plt.plot(train_history.history["mse"])
- plt.plot(train_history.history["val_mse"])
- plt.title('Train History')
- plt.ylabel('train')
- plt.xlabel('Epoch')
- plt.legend(['train', 'validation'], loc='center right')
- plt.show()



© 賴立弘

資料分析方法與應用2025E

44

Example 1b. Three species of Iris (Step 5. Evaluation 4)

- y_test_pred=model.predict(x_test)
- y_test_pred

© 賴立弘

資料分析方法與應用2025E

45

Example 1b. Three species of Iris (Step 6. Deployment)

- x_new=[[5.1,3.5,1.4],[7.0,3.2,4.7],[6.3,3.3,6.0]]
- x_new=pandas.DataFrame(x_new)
- y_test_pred=model.predict(x_new)
- y_test_pred

© 賴立弘

資料分析方法與應用2025E

46

Example 2. The Type of Sepsis (1)

- Predict the type of sepsis (urosepsis, bile duct sepsis, and airway sepsis) based on the laboratory screenings of 45 patients.



- Dataset: sepsis_1.csv
- Source: https://rstudio-pubs-static.s3.amazonaws.com/200568_9c836d5f0337451c8841e707395c1ae1.html

© 賴立弘

資料分析方法與應用2025E

47

Example 2. The Type of Sepsis (2)

- Code Book



Column	Note
gender	gender
age	age
alt	alt
hgb	hemoglobin
ureum	ureum
creatinine	creatinine
creatinineclearance	creatinine clearance
erythrocyte	erythrocyte count
erythrocyte sedimentation rate	erythrocyte sedimentation rate
c-reactive protein	c-reactive protein
leucocyte count	leucocyte count
diagnosis group	1: urosepsis 2: bile duct sepsis 3: airway sepsis
diagnosis group1	(1: urosepsis 2: bile duct sepsis 3: airway sepsis)
diagnosis group2	urosepsis bile duct sepsis airway sepsis
t1	1: urosepsis 2: Not urosepsis
t2	1: bile duct sepsis 0: Not bile duct sepsis
t3	1: airway sepsis 0: Not airway sepsis

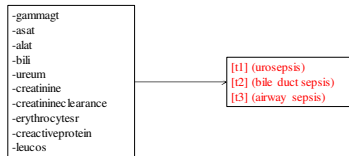
© 賴立弘

資料分析方法與應用2025E

48

Example 2. The Type of Sepsis (3) (Step 1. Business understanding)

• Business Understanding



Example 3. Indian Liver Patient Dataset (1)

- Death by liver cirrhosis continues to increase, given the increase in alcohol consumption rates, chronic hepatitis infections, and obesity-related liver disease.
- Notwithstanding the high mortality of this disease, liver diseases do not affect all sub-populations equally.
- The early detection of pathology is a determinant of patient outcomes, yet female patients appear to be marginalized when it comes to early diagnosis of liver pathology.
- The dataset comprises 584 patient records collected from the NorthEast of Andhra Pradesh, India.
- The prediction task is to determine whether a patient suffers from liver disease based on the information about several biochemical markers, including albumin and other enzymes required for metabolism.
- This data set contains records of 416 patients diagnosed with liver disease and 167 patients without liver disease. This information is contained in the class label named 'Sexclass'. There are 10 variables per patient: age, gender, total Bilirubin, direct Bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT and Alkalphos. Of the 583 patient records, 441 are male, and 142 are female.
- Dataset: IndianLiver_1.csv
- Source: <https://archive.ics.uci.edu/dataset/225/indian+liver+patient+dataset>



Example 3. Indian Liver Patient Dataset (2)

• Code Book

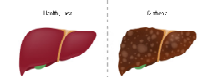
Feature	Description	Type
Age	Age of the patient. Any patient whose age exceeded 89 is listed as being of age "90".	Numerical
Gender	Gender of the patient	Categorical
TB	Total Bilirubin	Numerical
DB	Direct Bilirubin	Numerical
Alkphos	Alkaline Phosphatase	Numerical
Sgpt	Alanine Aminotransferase	Numerical
Sgot	Aspartate Aminotransferase	Numerical
TP	Total Protein	Numerical
ALB	Albumin	Numerical
A/G Ratio	Albumin and Globulin Ratio	Numerical
Label	1: with liver disease 2: without liver disease	Categorical
Gender1	1: Male 0: Female	
Label1	1: with liver disease 0: without liver disease	

Example 3. Indian Liver Patient Dataset (3) (Step 1. Business understanding)

• Business Understanding

- Age of the patient [Age]
- Gender of the patient [Gender1]
- Total Bilirubin [TB]
- Direct Bilirubin [DB]
- Alkaline Phosphatase [Alkphos]
- Alanine Aminotransferase [Sgpt]
- Aspartate Aminotransferase [Sgot]
- Total Proteins [TP]
- Albumin [ALB]
- Albumin and Globulin Ratio [A/G Ratio]

Label1



Example 4. The Type of Penguins (1)

- The dataset contains 333 instances of penguins from three species (classes), Adelie, Chinstrap, and Gentoo.
- The Adelie species contains 146 instances of penguins, Chinstrap has 68, and Gentoo has 119.



- Dataset: penguins_size1.csv

Example 4. The Type of Penguins (2)

• Code Book



Feature	Description	Type
species	Adelie, Chinstrap, or Gentoo	Categorical
species1	1: Adelie, 2: Chinstrap, or 3: Gentoo	
island	The name of the island (DREAM, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica) where the penguin was found and measured.	Categorical
island1	1: DREAM, 2: Torgersen, or 3: Biscoe	
culmen_length_mm	The length of the penguin's culmen in millimeters.	Numerical
culmen_depth_mm	The depth of the penguin's culmen in millimeters.	Numerical
flipper_length_mm	The length of the penguin's flipper in millimeters.	Numerical
body_mass_g	The body mass of the penguin in grams.	Numerical
sex	The sex (MALE, FEMALE) of the penguin.	Categorical
sex1	1: MALE, 0: FEMALE	
species1_1	1: Adelie 0: Not Adelie	
species1_2	1: Chinstrap 0: Not Chinstrap	
species1_3	1: Gentoo 0: Not Gentoo	

Example 4. The Type of Penguins (3) (Step 1. Business understanding)

- Business Understanding

-culmen_length_mm
-culmen_depth_mm
-flipper_length_mm
-body_mass_g

The type of penguins [species1_1](Adelie)
The type of penguins [species1_2](Chinstrap)
The type of penguins [species1_3](Gentoo)



7. Convolutional Neural Network

Course Outline

- Convolutional Neural Network
- The importance of CNNs
- Key Components of a CNN

Convolutional Neural Network

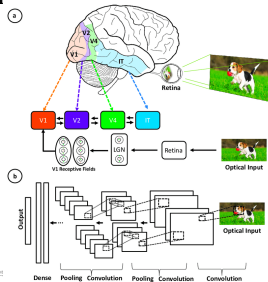
- A Convolutional Neural Network (CNN), also known as ConvNet, is a specialized type of deep learning algorithm mainly designed for tasks that necessitate object recognition, including image classification, detection, and segmentation.
- CNNs are employed in a variety of practical scenarios, such as autonomous vehicles, security camera systems, and others.

The importance of CNNs

- CNNs are distinguished from classic machine learning algorithms such as SVMs and decision trees by their ability to autonomously extract features at a large scale, bypassing the need for manual feature engineering and thereby enhancing efficiency.
- The convolutional layers grant CNNs their translation-invariant characteristics, empowering them to identify and extract patterns and features from data irrespective of variations in position, orientation, scale, or translation.
- A variety of pre-trained CNN architectures, including VGG-16, ResNet50, Inceptionv3, and EfficientNet, have demonstrated top-tier performance. These models can be adapted to new tasks with relatively little data through a process known as fine-tuning.
- Beyond image classification tasks, CNNs are versatile and can be applied to a range of other domains, such as natural language processing, time series analysis, and speech recognition.

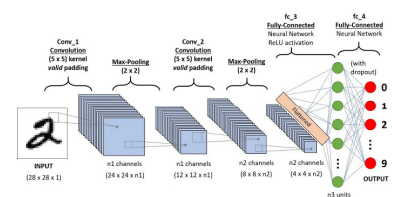
Inspiration Behind CNN and Parallels with The Human Visual System

- **Hierarchical architecture:** Both CNNs and the visual cortex have a hierarchical structure, with simple features extracted in early layers and more complex features built up in deeper layers. This allows increasingly sophisticated representations of visual inputs.
- **Local connectivity:** Neurons in the visual cortex only connect to a local region of the input, not the entire visual field. Similarly, the input volume through the convolution operation. This local connectivity enables efficiency.
- **Translation invariance:** Visual cortex neurons can detect features regardless of their location in the visual field. Pooling layers in a CNN provide a degree of translation invariance by summarizing local features.
- **Multiple feature maps:** At each stage of visual processing, there are many different feature maps extracted. CNNs mimic this through multiple filter maps in each convolution layer.
- **Non-linearity:** Neurons in the visual cortex exhibit non-linear response properties. CNNs achieve non-linearity through activation functions like ReLU applied after each convolution.



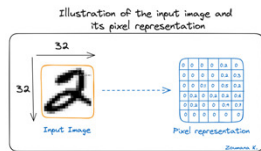
Key Components of a CNN

- Convolutional layers
- Rectified Linear Unit (ReLU for short)
- Pooling layers
- Fully connected layers



Convolution layers

- As the name suggests, the main mathematical task performed is called convolution, which is the application of a sliding window function to a matrix of pixels representing an image. The sliding function applied to the matrix is called kernel or filter, and both can be used interchangeably.
- In the convolution layer, several filters of equal size are applied, and each filter is used to recognize a specific pattern from the image, such as the curving of the digits, the edges, the whole shape of the digits, and more.
- Put simply, in the convolution layer, we use small grids (called filters or kernels) that move over the image. Each small grid is like a mini magnifying glass that looks for specific patterns in the photo, like lines, curves, or shapes. As it moves across the photo, it creates a new grid that highlights where it found these patterns.

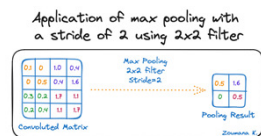


Activation function

- A ReLU activation function is applied after each convolution operation. This function helps the network learn non-linear relationships between the features in the image, hence making the network more robust for identifying different patterns.
- It also helps to mitigate the vanishing gradient problems.

Pooling layer

- The goal of the pooling layer is to pull the most significant features from the convoluted matrix. This is done by applying some aggregation operations, which reduce the dimension of the feature map (convoluted matrix), hence reducing the memory used while training the network. Pooling is also relevant for mitigating overfitting.
- The most common aggregation functions that can be applied are:
 - Max pooling, which is the maximum value of the feature map
 - Sum pooling corresponds to the sum of all the values of the feature map
 - Average pooling is the average of all the values.

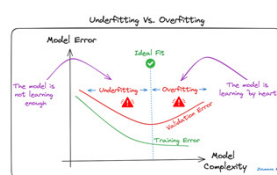


Fully connected layers

- These layers are in the last layer of the convolutional neural network, and their inputs correspond to the flattened one-dimensional matrix generated by the last pooling layer.
- ReLU activations functions are applied to them for non-linearity.
- Finally, a softmax prediction layer is used to generate probability values for each of the possible output labels, and the final label predicted is the one with the highest probability score.

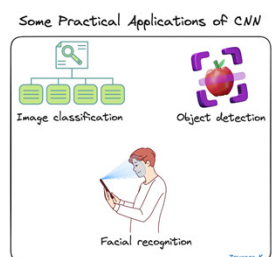
Overfitting and Regularization in CNNs

- Overfitting is a common challenge in machine learning models and CNN deep learning projects. It happens when the model learns the training data too well ("learning by heart"), including its noise and outliers. Such a learning leads to a model that performs well on the training data but badly on new, unseen data.



Practical Applications of CNNs

- Image classification: Convolutional neural networks are used for image categorization, where images are assigned to predefined categories. One use of such a scenario is automatic photo organization in social media platforms.
- Object detection: CNNs are able to identify and locate multiple objects within an image. This capability is crucial in multiple scenarios of shelf scanning in retail to identify out-of-stock items.
- Facial recognition: this is also one of the main industries of application of CNNs. For instance, this technology can be embedded into security systems for efficient control of access based on facial features.



flow_from_directory()

- **directory**: The directory must be set to the path where your 'n' classes of folders are present.
- **target_size**: The **target_size** is the size of your input images, every image will be resized to this size. Ex: **target_size**=(256, 256).
- **color_mode**: if the image is either black and white or grayscale set "grayscale" or if the image has three color channels, set "rgb". Ex: **color_mode**="rgb".
- **batch_size**: No. of images to be yielded from the generator per batch. Ex: **batch_size**=32.
- **class_mode**: Set "binary" if you have only two classes to predict, if not set to "categorical". in case if you're developing an Autoencoder system, both input and the output would probably be the same image, for this case set to "input".
- **shuffle**: Set True if you want to shuffle the order of the image that is being yielded, else set False. Ex: **shuffle**=True.
- **seed**: Random seed for applying random image augmentation and shuffling the order of the image.

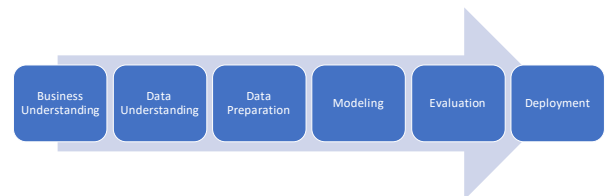
© 魏立弘

資料分析方法與應用2025E

13

Process

- Process



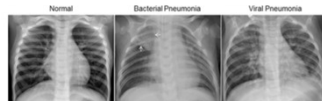
© 魏立弘

資料分析方法與應用2025E

14

Example 1. Chest X-Ray Images (Pneumonia) (1)

- The normal chest X-ray (left panel) depicts clear lungs without any areas of abnormal opacification in the image. Bacterial pneumonia (middle) typically exhibits a focal lobar consolidation, in this case in the right upper lobe (white arrows), whereas viral pneumonia (right) manifests with a more diffuse "interstitial" pattern in both lungs.
- <https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia>
- Dataset: chest_xray_quarter



© 魏立弘

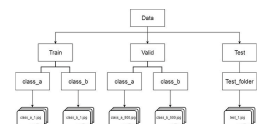
資料分析方法與應用2025E

15

Example 1. Chest X-Ray Images (Pneumonia) (2)

- The dataset is organized into 3 folders (train, test, val) and contains subfolders for each image category (Pneumonia/Normal). There are 5,863 X-Ray images (JPEG) and 2 categories (Pneumonia/Normal).
- Chest X-ray images (anterior-posterior) were selected from retrospective cohorts of pediatric patients of one to five years old from Guangzhou Women and Children's Medical Center, Guangzhou. All chest X-ray imaging was performed as part of patients' routine clinical care.

			Original	Quarter
1	train	Pneumonia	3875	969
2		Normal	1341	336
3	test	Pneumonia	390	98
4		Normal	234	59
5	val	Pneumonia	8	8
6		Normal	8	8



© 魏立弘

資料分析方法與應用2025E

16

Example 1. Chest X-Ray Images (Pneumonia) (Step 0. GPU usage)

- Edit->Notebook settings->Hardware accelerator: T4 GPU

- `import tensorflow`
- `tensorflow.test.gpu_device_name()`
- `!nvidia-smi -L`

© 魏立弘

資料分析方法與應用2025E

17

Example 1. Chest X-Ray Images (Pneumonia) (Step 1. Business Understanding)

- Business Understanding



© 魏立弘

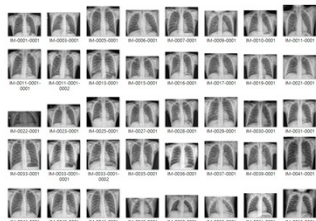
資料分析方法與應用2025E

18

Example 1. Chest X-Ray Images (Pneumonia) (Step 2. Data Understanding)

• Data Understanding

- chest_xray_quarter
 - test
 - NORMAL
 - PNEUMONIA
 - train
 - NORMAL
 - PNEUMONIA
 - val
 - NORMAL
 - PNEUMONIA



Example 1. Chest X-Ray Images (Pneumonia) (Step 3. Data Preparation 1)

```
# Open File
from google.colab import drive
drive.mount('/content/MyGoogleDrive')

from keras.api._v2.keras.preprocessing.image import ImageDataGenerator

train_generator=ImageDataGenerator(rescale=1./255).flow_from_directory('/content/MyGoogleDrive/MyDrive/chest_xray_quarter/train',
                                target_size=(256, 256),
                                batch_size=32,
                                class_mode='binary',
                                seed=123)
test_generator=ImageDataGenerator(rescale=1./255).flow_from_directory('/content/MyGoogleDrive/MyDrive/chest_xray_quarter/test',
                                target_size=(256, 256),
                                batch_size=32,
                                class_mode='binary',
                                seed=123)
val_generator = ImageDataGenerator(rescale=1./255).flow_from_directory('/content/MyGoogleDrive/MyDrive/chest_xray_quarter/val',
                                target_size=(256, 256),
                                batch_size=32,
                                class_mode='binary',
                                seed=123)
```

Example 1. Chest X-Ray Images (Pneumonia) (Step 3. Data Preparation 2)

- # Iterate through each generator to create the data sets with the train/test/val splits
- x_train, y_train = next(train_generator)
- x_test, y_test = next(test_generator)
- x_val, y_val = next(val_generator)

Example 1. Chest X-Ray Images (Pneumonia) (Step 3. Data Preparation 3)

- # Check which class is which
- train_generator.class_indices

Example 1. Chest X-Ray Images (Pneumonia) (Step 4. Modeling 1)

```
from keras.api._v2.keras.models import Sequential
from keras.api._v2.keras.layers import Conv2D, MaxPooling2D
from keras.api._v2.keras.layers import Dense, Flatten

model = Sequential()

model.add(Conv2D(16, (3,3), padding='same', input_shape=(224,224,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(32, (3,3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(64, (3,3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

model.summary()
```

Example 1. Chest X-Ray Images (Pneumonia) (Step 4. Modeling 2)

```
#from tensorflow.keras.optimizers import SGD

from keras.api._v2.keras.optimizers import SGD

model.compile(loss='mse', optimizer=SGD(learning_rate=0.087), metrics=['accuracy'])
```

Example 1. Chest X-Ray Images (Pneumonia) (Step 4. Modeling 3)

- # epochs=total//batch_size
- model.fit(x_train, y_train, batch_size=128, epochs=12)

Example 1. Chest X-Ray Images (Pneumonia) (Step 5. Evaluation)

- loss, acc = model.evaluate(x_test, y_test)
- print(f'Correct Rate= {acc*100:.2f}%')

Example 1. Chest X-Ray Images (Pneumonia) (Step 6. Deployment)

- import numpy
- y_predict = numpy.argmax(model.predict(x_test), axis=-1)
- y_predict

