

$$(X, O) = e^{-\frac{x^2}{2\sigma^2}}$$

$$x(X, O) = -\frac{x}{\sigma^2} G(X, O) = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

$$xx(X, O) = \frac{x^2 - \sigma^2}{\sigma^4} G(X, O) = \frac{x^2 - \sigma^2}{\sigma^4} e^{-\frac{x^2}{2\sigma^2}}$$

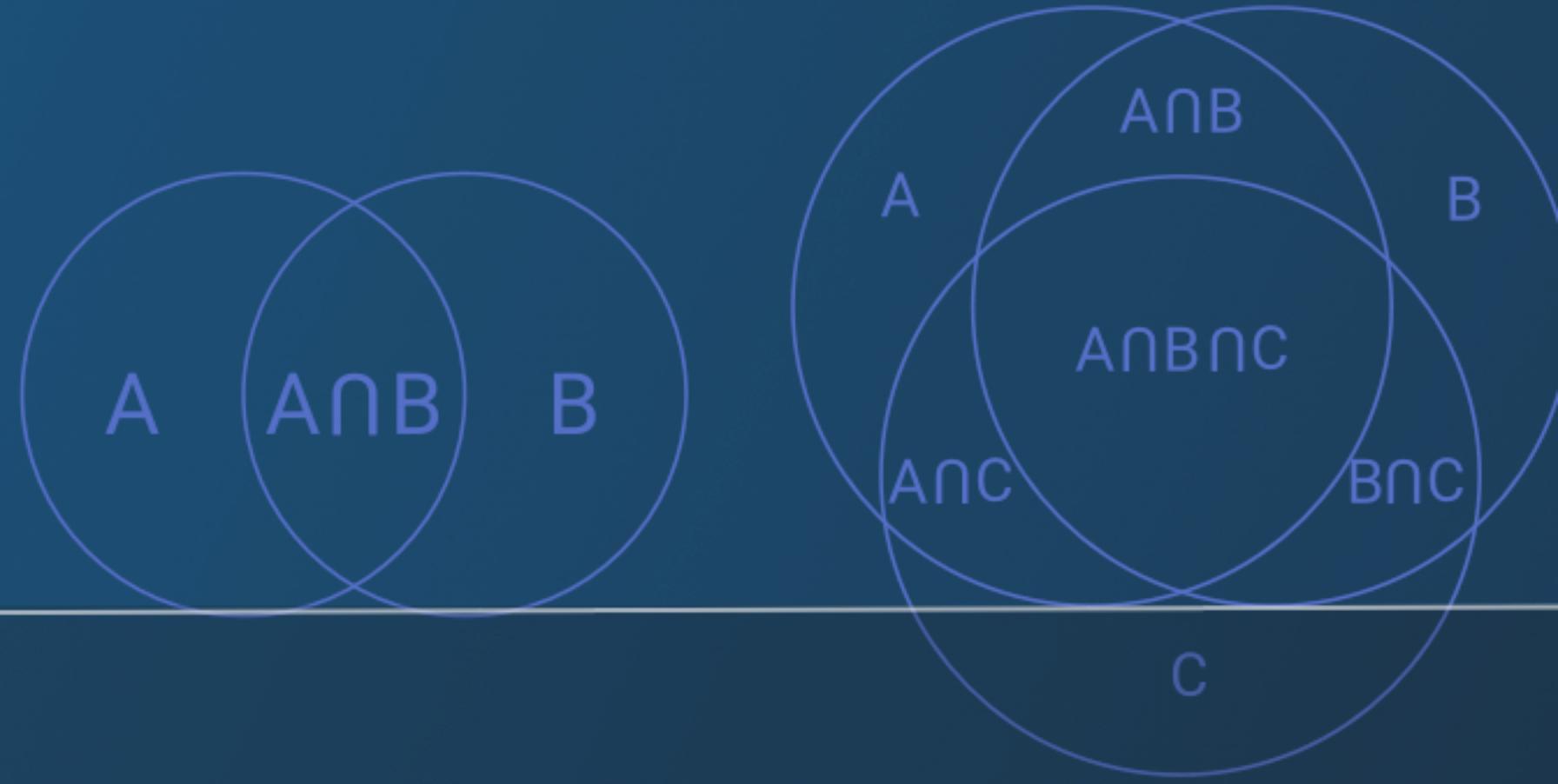
$$xxx(X, O) = -\frac{x^3 - x\sigma^2}{\sigma^6} G(X, O) = -\frac{x^3 - x\sigma^2}{\sigma^6} e^{-\frac{x^2}{2\sigma^2}}$$

Julia 程式語言學習馬拉松

Day 17



cupay 陪跑專家 : James Huang



DataFrames.jl 介紹

(一): 入門操作





重要知識點



- `DataFrames.jl` 是一個處理及操作表格式 (tabular) 資料的套件，資料的部分以 `DataFrame` 型別載入、操作、與儲存。
- `DataFrames.jl` 的介紹將分為四個部分：
 - 入門操作
 - Joins 與 Split-Apply-Combine Strategy
 - Reshaping 與 Sorting
 - Categorical Data 與 Missing Data
- 資料軸的描述是以直“行 (column)”、橫”列 (row)”的用語表示。Column 也會以”欄位”稱之。



DataFrame：建立

- 透過 DataFrame 建構子，有兩種方式可以建立 DataFrame：
 - 使用向量 (vector)：將每一行資料以向量的方式指定。
例如：`df=DataFrame(coll=1:5,col2=["M", "F", "F", missing, "M"])`
 - 使用 "Column by Column" 的方式，逐行建立 DataFrame
例如：`df.coll=1:5; df.col2=["M", "F", "F", missing, "M"]`





DataFrame：新增及刪除 Row



- 呼叫 `push!()` 函式可以新增 row 到 DataFrame，例如：`push!(df,[1,"M"])`，資料的部分可以使用 tuple, vector, 或是 dictionary。
- 刪除 row 的函式是 `deleterows!()`，例如要刪除 7 – 8 行，可以呼叫 `deleterows!(df,7:8)`。





DataFrame：載入資料、儲存及複製



- DataFrame 是廣受支援的資料型別 (Type)，很多套件均預設支援，以讀取 CSV 檔案為例，配合 CSV.jl 套件可以將檔案直接讀取為 DataFrame。
- 透過 CSV.jl，可直接將 DataFrame 儲存為 CSV 檔案。
- 呼叫 DataFrames.copy() 函式可以複製並建立一個新的 DataFrame，例如：`df2 = copy(df)`。



DataFrame：操作



- 下列是常見用來檢視及操作 DataFrame 的函式。

函式名稱	功能	範例
size()	檢視 DataFrame 的尺寸	<code>size(df)</code>
describe()	彙總 DataFrame 資訊	<code>describe(df)</code>
show()	顯示 DataFrame	<code>show(df, allcols=true, allrows=true)</code>
first()	顯示前 n 筆資料	<code>first(df, 5)</code>
last()	顯示最後 n 筆資料	<code>last(df, 5)</code>



DataFrame：子集 (Slicing)



- 要查看 DataFrame 子集，可以使用 `df[<row index>, <column index>]`，範例：

語法	說明
<code>df</code> <code>df[:, :]</code> <code>df[:, :1]</code>	所有 row 與 column
<code>df[1:5, :]</code>	前 5 個 row 與所有 column
<code>df[[1, 3, 5], [1, 2, 9]]</code>	可以指定特定要查看的 row / column
<code>df[1:5, [:mpg, :displacement, :horsepower]]</code>	指定 column 可以使用 index, 也可以使用 column 名稱, 使用的方式為 ":" 加上 column 名稱.



DataFrame：篩選 (Select)



- 如果要篩選 DataFrame 中的 column, 可以使用 select() 和 select!() 函式. 例如：
`select(df2, 1:3)`，篩選出第 1 – 3 個 column 的資料。
- `select()` 不會變更原 DataFrame 而會傳回傳變更後的 DataFrame,但是呼叫 `select!()` 會變更原 DataFrame，例如：`select!(df2, 1:3)`。



DataFrame：行 (Column) 的操作



- Aggregate
 - aggregate() 函式可以套用到 column 中的每一個值, 例如計算該 column 資料值的平均數, 在範例中示範搭配 Statistics 模組中的 mean() 和 median() 函式, 得到 column 的平均值和中位數。
- Sort (排序)
 - sort() 和 sort!() 函式可用來排序, 不同點在於 sort!() 會改變原 DataFrame, 而 sort() 不會。
 - 排序時可以指定 column 名稱或是使用 index 值, 也可以指定要正排還是逆排, 例如 : sort!(df3, :displacement, rev=true)。

知識點 回顧

- 在今天的內容中介紹了 `DataFrames.jl` 套件，以及 `DataFrame` 型別與各種行、列的操作，這些都是在資料處理和資料分析中，常會用到的功能和技巧。
- Tabular 資料是常見資料型式之一，透過 `DataFrame` 能讓我們更容易操作資料。
- 另外搭配其他常用套件，例如 `CSV.jl`，透過支援 `DataFrames.jl`，我們可以讀取或寫入資料到不同的資料來源。



推薦閱讀

- [DataFrames.jl 官方網頁](#)
- [CSV.jl 官方文件](#)
- [Statistics 模組官方頁面](#)





解題時間

請跳出 PDF 至官網 Sample Code
& 作業開始解題