MLDM Monday x Julia | Julia Tutorial III

Data science

杜岳華

Outline

- DataFrame.jl
- 資料清理
- 資料視覺化
    - Gadfly.jl
    - Plots.jl

DataFrame

Install

當我們要處理表格類的資料的時候，DataFrame 無疑是首選的資料結構

比起使用 Array，他讓存取更便利，支援類似資料庫的 join 運算，也有好用的 aggregate function

請先安裝套件：

```
using Pkg
Pkg.add("DataFrames")
Pkg.add("RDatasets")
```

In [1]:
```
using DataFrames
```

Missing

- missing 是用來表示一個遺失的資料點
- missing 是 Missing 的唯一物件實體
- 包含 missing 的運算，當結果無法確定的時候都會回傳 missing

Calculation

In [2]:
```
true && missing
```

Out[2]:
```
missing
```

In [3]:
```
typeof(missing)
```

Out[3]:
```
Missing
```

In [4]:
```
1 + missing
```

Out[4]:
```
missing
```

Calculation

短路邏輯

In [5]:
```
false && missing
```

Out[5]: false

In [6]:
```
true || missing
```

Out[6]: true

分辨是否為 missing

In [7]:
```
ismissing(missing)
```

Out[7]: true

DataFrame

- DataFrame 是一種表格狀的資料結構
- 每個行必須是相同長度，由 Array{T, 1} 構成

In [8]:
```
df = DataFrame(A = 1:4, B = ["M", "F", "F", "M"])
```

Out[8]:

4 rows × 2 columns

	A	B
	Int64	String
1	1	M
2	2	F
3	3	F
4	4	M

Select columns

In [9]:
```
df.A
```

Out[9]:
```
4-element Array{Int64,1}:
 1
 2
 3
 4
```

In [10]:
```
df[:A]
```

Out[10]:
```
4-element Array{Int64,1}:
 1
 2
 3
 4
```

In [11]:
```
df[1]
```

Out[11]:
```
4-element Array{Int64,1}:
 1
 2
 3
 4
```

Construct DataFrame by adding new columns

In [12]:
```
df = DataFrame()  # 先初始化一個空的，再放入資料
df[:A] = 1:8
df[:B] = ["M", "F", "F", "M", "F", "M", "M", "F"]
df
```

Out[12]:

8 rows × 2 columns

	A	B
	Int64	String
1	1	M
2	2	F
3	3	F
4	4	M
5	5	F
6	6	M
7	7	M
8	8	F

Index

可以用數字索引

In [13]:
```
df[1, 1]
```

Out[13]: 1

也可以用欄位（Symbol type）索引

In [14]:
```
df[1, :A]
```

Out[14]: 1

Index

In [15]:
```
df[1:3, :]
```

Out[15]:

3 rows × 2 columns

	A	B
	Int64	String
1	1	M
2	2	F
3	3	F

In [16]:
```
df[1:3, [:B, :A]]
```

Out[16]:

3 rows × 2 columns

	B	A
	String	Int64
1	M	1
2	F	2
3	F	3

DataFrame information

In [17]:
```
size(df)  # 取得維度資訊
```

Out[17]:
```
(8, 2)
```

In [18]:
```
nrow(df)  # 列數
```

Out[18]:
```
8
```

In [19]:
```
ncol(df)  # 行數
```

Out[19]:
```
2
```

DataFrame information

In [20]:
```
names(df)  # 取得欄位名稱
```

Out[20]:
```
2-element Array{Symbol,1}:
 :A
 :B
```

In [21]:
```
eltypes(df)
```

Out[21]:
```
2-element Array{DataType,1}:
 Int64
 String
```

Data selection

In [22]:
```
df[df[:A] .% 2 .== 0, :]  # 取符合條件的列，跟所有欄位
```

Out[22]:

4 rows × 2 columns

	A	B
	Int64	String
1	2	F
2	4	M
3	6	M
4	8	F

Constructing row by row

In [23]:
```
df = DataFrame(A = Int[], B = String[])
```

Out[23]:

0 rows × 2 columns

	A	B
	Int64	String

In [24]:
```
push!(df, (1, "M"))
```

Out[24]:

1 rows × 2 columns

	A	B
	Int64	String
1	1	M

In [25]:
```
push!(df, [2, "N"])
```

Out[25]:

2 rows × 2 columns

	A	B
	Int64	String
1	1	M
2	2	N

Constructing row by row

In [26]:
```
push!(df, Dict(:B => "F", :A => 3))
```

Out[26]:

3 rows × 2 columns

	A	B
	Int64	String
1	1	M
2	2	N
3	3	F

Load

In [27]:
```
using CSV
```

In [28]:
```
file = "data.csv"
df = CSV.read(file)
```

Out[28]:

4 rows × 3 columns

	name	age	squidPerWeek
	String⍰	Int64⍰	Float64⍰
1	Alice	36	3.14
2	Bob	24	0.0
3	Carol	58	2.71
4	Eve	49	7.77

Save

In [29]:
```
CSV.write("data.tsv", df, delim='\t')
```

Out[29]:
```
"data.tsv"
```

In [30]:
```
df |> CSV.write("data.tsv", delim='\t')
```

Out[30]:
```
"data.tsv"
```

Categorical data

In [31]:
```
using CategoricalArrays
```

In [32]:
```
df = DataFrame()
df[:A] = 1:8
df[:B] = ["M", "F", "F", "M", "F", "M", "M", "F"]
```

Out[32]:
```
8-element Array{String,1}:
 "M"
 "F"
 "F"
 "M"
 "F"
 "M"
 "M"
 "F"
```

Categorical array

In [33]:
```
ca = CategoricalArray(df[:B])
```

Out[33]:
```
8-element CategoricalArray{String,1,UInt32}:
 "M"
 "F"
 "F"
 "M"
 "F"
 "M"
 "M"
 "F"
```

In [34]:
```
levels(ca)
```

Out[34]:
```
2-element Array{String,1}:
 "F"
 "M"
```

Categorical array

用在 dataframe 的 column 上

In [35]:
```
df[:B]
```

Out[35]:
```
8-element Array{String,1}:
 "M"
 "F"
 "F"
 "M"
 "F"
 "M"
 "M"
 "F"
```

In [36]:
```
levels(df[:B])
```

Out[36]:
```
2-element Array{String,1}:
 "F"
 "M"
```

Transform to categorical array

In [37]:
```
categorical!(df, :B)
```

Out[37]:

8 rows × 2 columns

	A	B
	Int64	Categorical...
1	1	M
2	2	F
3	3	F
4	4	M
5	5	F
6	6	M
7	7	M
8	8	F

In [38]:
```
eltypes(df)
```

Out[38]:
```
2-element Array{DataType,1}:
 Int64
 CategoricalString{UInt32}
```

Transform to categorical array

In [39]:
```
categorical!(df)
```

Out[39]:

8 rows × 2 columns

	A	B
	Int64	Categorical...
1	1	M
2	2	F
3	3	F
4	4	M
5	5	F
6	6	M
7	7	M
8	8	F

Practice!

那我們就來實際用DataFrame玩玩看資料吧！

In [40]:
```
using RDatasets
```

挑選你想要的資料集

In [41]:
```
RDatasets.packages()[1:10, :]
```

```
warning: failed parsing String on row=8, col=2, error=INVALID: OK, QUOTED, NEWLINE, INVALID_DE
LIMITER
```

Out[41]:

10 rows × 2 columns

	Package	Title
	String⍰	String⍰
1	COUNT	Functions, data and code for count data.
2	Ecdat	Data sets for econometrics
3	HSAUR	A Handbook of Statistical Analyses Using R (1st Edition)
4	HistData	Data sets from the history of statistics and data visualization
5	ISLR	Data for An Introduction to Statistical Learning with Applications in R
6	KMsurv	Data sets from Klein and Moeschberger (1997), Survival Analysis
7	MASS	Support Functions and Datasets for Venables and Ripley's MASS
8	SASmixed	Data sets from \\
9	Zelig	Everyone's Statistical Software
10	adehabitatLT	Analysis of Animal Movements

這邊我們選的是iris這個資料集

In [42]:
```
iris = dataset("datasets", "iris")
first(iris, 10)
```

Out[42]:

10 rows × 5 columns

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species
	Float64	Float64	Float64	Float64	Categorical...
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa

看一下他有幾列幾行

In [43]:
```
size(iris)
```

Out[43]:
```
(150, 5)
```

把兩個相同的DataFrame垂直地併起來

In [44]:
```
size(vcat(iris, iris))
```

Out[44]:
```
(300, 5)
```

把兩個相同的 DataFrame 水平地併起來可以用 `hcat`

Dealing with missing data

In [45]:
```
messy_data = [1, 2, 3, missing, 4, 5, missing]
```

Out[45]:
```
7-element Array{Union{Missing, Int64},1}:
 1
 2
 3
  missing
 4
 5
  missing
```

In [46]:
```
sum(messy_data)
```

Out[46]:
```
missing
```

Skip missing data

In [47]:
```
skipmissing(messy_data)
```

Out[47]:
```
Base.SkipMissing{Array{Union{Missing, Int64},1}}(Union{Missing, Int64}[1, 2, 3, missing, 4, 5, missing])
```

In [48]:
```
sum(skipmissing(messy_data))
```

Out[48]:
```
15
```

Impute missing data

把 missing 换成 10

In [49]:
```
map(x -> ismissing(x) ? 10 : x, messy_data)
```

Out[49]:
```
7-element Array{Int64,1}:
  1
  2
  3
 10
  4
  5
 10
```

看看每列是不是有出現 missing

true 代表沒有出現

In [50]:
```
completecases(iris)
```

Out[50]:
```
150-element BitArray{1}:
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
    ⋮
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
 true
```

選出那些完整的列

跟 completecases!(iris) 一樣

In [51]:
```
iris[completecases(iris), :]
```

Out[51]:

150 rows × 5 columns

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species
	Float64	Float64	Float64	Float64	Categorical...
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1.0	0.2	setosa
24	5.1	3.3	1.7	0.5	setosa
25	4.8	3.4	1.9	0.2	setosa
26	5.0	3.0	1.6	0.2	setosa
27	5.0	3.4	1.6	0.4	setosa
28	5.2	3.5	1.5	0.2	setosa

或是你想要唯一的列元素組合

In [52]:
```
unique(iris)
```

Out[52]:

149 rows × 5 columns

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species
	Float64	Float64	Float64	Float64	Categorical...
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1.0	0.2	setosa
24	5.1	3.3	1.7	0.5	setosa
25	4.8	3.4	1.9	0.2	setosa
26	5.0	3.0	1.6	0.2	setosa
27	5.0	3.4	1.6	0.4	setosa
28	5.2	3.5	1.5	0.2	setosa

計算後變成一個新的一行

In [53]:
```
iris[:PetalArea] = iris[:PetalLength] .* iris[:PetalWidth]
first(iris, 10)
```

Out[53]:

10 rows × 6 columns

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species	PetalArea
	Float64	Float64	Float64	Float64	Categorical...	Float64
1	5.1	3.5	1.4	0.2	setosa	0.28
2	4.9	3.0	1.4	0.2	setosa	0.28
3	4.7	3.2	1.3	0.2	setosa	0.26
4	4.6	3.1	1.5	0.2	setosa	0.3
5	5.0	3.6	1.4	0.2	setosa	0.28
6	5.4	3.9	1.7	0.4	setosa	0.68
7	4.6	3.4	1.4	0.3	setosa	0.42
8	5.0	3.4	1.5	0.2	setosa	0.3
9	4.4	2.9	1.4	0.2	setosa	0.28
10	4.9	3.1	1.5	0.1	setosa	0.15

排序

In [54]:
```
sort!(iris, :SepalLength)
```

Out[54]:

150 rows × 6 columns

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species	PetalArea
	Float64	Float64	Float64	Float64	Categorical...	Float64
1	4.3	3.0	1.1	0.1	setosa	0.11
2	4.4	2.9	1.4	0.2	setosa	0.28
3	4.4	3.0	1.3	0.2	setosa	0.26
4	4.4	3.2	1.3	0.2	setosa	0.26
5	4.5	2.3	1.3	0.3	setosa	0.39
6	4.6	3.1	1.5	0.2	setosa	0.3
7	4.6	3.4	1.4	0.3	setosa	0.42
8	4.6	3.6	1.0	0.2	setosa	0.2
9	4.6	3.2	1.4	0.2	setosa	0.28
10	4.7	3.2	1.3	0.2	setosa	0.26
11	4.7	3.2	1.6	0.2	setosa	0.32
12	4.8	3.4	1.6	0.2	setosa	0.32
13	4.8	3.0	1.4	0.1	setosa	0.14
14	4.8	3.4	1.9	0.2	setosa	0.38
15	4.8	3.1	1.6	0.2	setosa	0.32
16	4.8	3.0	1.4	0.3	setosa	0.42
17	4.9	3.0	1.4	0.2	setosa	0.28
18	4.9	3.1	1.5	0.1	setosa	0.15
19	4.9	3.1	1.5	0.2	setosa	0.3
20	4.9	3.6	1.4	0.1	setosa	0.14
21	4.9	2.4	3.3	1.0	versicolor	3.3
22	4.9	2.5	4.5	1.7	virginica	7.65
23	5.0	3.6	1.4	0.2	setosa	0.28
24	5.0	3.4	1.5	0.2	setosa	0.3
25	5.0	3.0	1.6	0.2	setosa	0.32
26	5.0	3.4	1.6	0.4	setosa	0.64
27	5.0	3.2	1.2	0.2	setosa	0.24
28	5.0	3.5	1.3	0.3	setosa	0.39

排序多個欄位，並指定是否倒序

In [55]:
```
sort!(iris, [:Species, :SepalLength, :SepalWidth], rev=(true, false, false))
```

Out[55]:

150 rows × 6 columns

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species	PetalArea
	Float64	Float64	Float64	Float64	Categorical...	Float64
1	4.9	2.5	4.5	1.7	virginica	7.65
2	5.6	2.8	4.9	2.0	virginica	9.8
3	5.7	2.5	5.0	2.0	virginica	10.0
4	5.8	2.7	5.1	1.9	virginica	9.69
5	5.8	2.7	5.1	1.9	virginica	9.69
6	5.8	2.8	5.1	2.4	virginica	12.24
7	5.9	3.0	5.1	1.8	virginica	9.18
8	6.0	2.2	5.0	1.5	virginica	7.5
9	6.0	3.0	4.8	1.8	virginica	8.64
10	6.1	2.6	5.6	1.4	virginica	7.84
11	6.1	3.0	4.9	1.8	virginica	8.82
12	6.2	2.8	4.8	1.8	virginica	8.64
13	6.2	3.4	5.4	2.3	virginica	12.42
14	6.3	2.5	5.0	1.9	virginica	9.5
15	6.3	2.7	4.9	1.8	virginica	8.82
16	6.3	2.8	5.1	1.5	virginica	7.65
17	6.3	2.9	5.6	1.8	virginica	10.08
18	6.3	3.3	6.0	2.5	virginica	15.0
19	6.3	3.4	5.6	2.4	virginica	13.44
20	6.4	2.7	5.3	1.9	virginica	10.07
21	6.4	2.8	5.6	2.1	virginica	11.76
22	6.4	2.8	5.6	2.2	virginica	12.32
23	6.4	3.1	5.5	1.8	virginica	9.9
24	6.4	3.2	5.3	2.3	virginica	12.19
25	6.5	3.0	5.8	2.2	virginica	12.76
26	6.5	3.0	5.5	1.8	virginica	9.9
27	6.5	3.0	5.2	2.0	virginica	10.4
28	6.5	3.2	5.1	2.0	virginica	10.2

Join 運算

In [56]:
```
name = DataFrame(ID=[1, 2, 3], Name=["John Doe", "Jane Doe", "Andy Doe"])
```

Out[56]:

3 rows × 2 columns

	ID	Name
	Int64	String
1	1	John Doe
2	2	Jane Doe
3	3	Andy Doe

In [57]:
```
job = DataFrame(ID=[1, 2, 4], Job=["Lawyer", "Doctor", "Chief"])
```

Out[57]:

3 rows × 2 columns

	ID	Job
	Int64	String
1	1	Lawyer
2	2	Doctor
3	4	Chief

inner join會輸出左表跟右表都包含的列

In [58]:
```
full = join(name, job, on=:ID)  # 預設的 join 是 inner join
```

Out[58]:

2 rows × 3 columns

	ID	Name	Job
	Int64	String	String
1	1	John Doe	Lawyer
2	2	Jane Doe	Doctor

left join會輸出以左表為主包含的列

In [59]:
```
left_join = join(name, job, on=:ID, kind=:left)
```

Out[59]:

3 rows × 3 columns

	ID	Name	Job
	Int64	String	String⍰
1	1	John Doe	Lawyer
2	2	Jane Doe	Doctor
3	3	Andy Doe	missing

right join會輸出右表為主包含的列

In [60]:
```
right_join = join(name, job, on=:ID, kind=:right)
```

Out[60]:

3 rows × 3 columns

	ID	Name	Job
	Int64	String⍰	String
1	1	John Doe	Lawyer
2	2	Jane Doe	Doctor
3	4	missing	Chief

outer join會輸出左表或是右表包含的列

In [61]:
```
outer_join = join(name, job, on=:ID, kind=:outer)
```

Out[61]:

4 rows × 3 columns

	ID	Name	Job
	Int64	String⍰	String⍰
1	1	John Doe	Lawyer
2	2	Jane Doe	Doctor
3	3	Andy Doe	missing
4	4	missing	Chief

其他的join:

- Semi join: 類似inner join，但輸出只限於左表的行而已
- Anti join: 輸出的列是key在左表存在，但右表不存在
- Cross join: 輸出是左表跟右表所有列的所有排列組合 (Cartesian product)

In [62]:
```
cross_join = join(name, job, kind=:cross, makeunique=true)  # 不需要key
```

Out[62]:

9 rows × 4 columns

	ID	Name	ID_1	Job
	Int64	String	Int64	String
1	1	John Doe	1	Lawyer
2	1	John Doe	2	Doctor
3	1	John Doe	4	Chief
4	2	Jane Doe	1	Lawyer
5	2	Jane Doe	2	Doctor
6	2	Jane Doe	4	Chief
7	3	Andy Doe	1	Lawyer
8	3	Andy Doe	2	Doctor
9	3	Andy Doe	4	Chief

The Split-Apply-Combine Strategy

除了可以像資料庫一樣join，還有運算的策略：

- 拆分：把資料拆分成幾個群
- 應用：在每群資料上運算一些函式
- 合併：合併運算結果

By

In [63]:
```
by(iris, :Species, size)
```

Out[63]:

3 rows × 2 columns

	Species	x1
	Categorical...	Tuple...
1	setosa	(50, 6)
2	versicolor	(50, 6)
3	virginica	(50, 6)

根據不同:Species，去計算iris上的size()

參數:

1. *DataFrame*
2. *針對一個column去拆分DataFrame*
3. *會對每個群進行運算的一個function或是expression*

可以支援 do 語法:

In [64]:
```
using StatsBase
```

```
WARNING: using StatsBase.df in module Main conflicts with an existing identifier.
```

適合用在複雜的function的情況

In [65]:
```
by(iris, :Species) do df
    DataFrame(m = StatsBase.mean(df[:, :PetalLength]), s² = StatsBase.var(df[:, :PetalLength]))
end
```

Out[65]:

3 rows × 3 columns

	Species	m	s²
	Categorical...	Float64	Float64
1	setosa	1.462	0.0301592
2	versicolor	4.26	0.220816
3	virginica	5.552	0.304588

Aggregate

In [66]:
```
aggregate(iris, :Species, [sum, StatsBase.mean])
```

Out[66]:

3 rows × 11 columns

	Species	SepalLength_sum	SepalWidth_sum	PetalLength_sum	PetalWidth_sum	PetalArea_sum	SepalLength_mean	SepalWidth_
	Categorical...	Float64	Float64	Float64	Float64	Float64	Float64	Float64
1	setosa	250.3	171.4	73.1	12.3	18.28	5.006	3.428
2	versicolor	296.8	138.5	213.0	66.3	286.02	5.936	2.77
3	virginica	329.4	148.7	277.6	101.3	564.81	6.588	2.974

以:Species分群，計算總和跟平均

arguments:

1. *DataFrame*
2. *針對一個或多個column去拆分DataFrame*
3. *會對每個群進行運算的多個function或是expression*

Groupby

單純拆分資料

In [67]:
```
for subdf in groupby(iris, :Species)
    println(size(subdf, 1))
end
```

```
50
50
50
```

資料表的變形

”寬型”的表格型態

In [68]:
```
iris[:id] = 1:size(iris, 1)
iris
```

Out[68]:

150 rows × 7 columns

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species	PetalArea	id
	Float64	Float64	Float64	Float64	Categorical...	Float64	Int64
1	4.9	2.5	4.5	1.7	virginica	7.65	1
2	5.6	2.8	4.9	2.0	virginica	9.8	2
3	5.7	2.5	5.0	2.0	virginica	10.0	3
4	5.8	2.7	5.1	1.9	virginica	9.69	4
5	5.8	2.7	5.1	1.9	virginica	9.69	5
6	5.8	2.8	5.1	2.4	virginica	12.24	6
7	5.9	3.0	5.1	1.8	virginica	9.18	7
8	6.0	2.2	5.0	1.5	virginica	7.5	8
9	6.0	3.0	4.8	1.8	virginica	8.64	9
10	6.1	2.6	5.6	1.4	virginica	7.84	10
11	6.1	3.0	4.9	1.8	virginica	8.82	11
12	6.2	2.8	4.8	1.8	virginica	8.64	12
13	6.2	3.4	5.4	2.3	virginica	12.42	13
14	6.3	2.5	5.0	1.9	virginica	9.5	14
15	6.3	2.7	4.9	1.8	virginica	8.82	15
16	6.3	2.8	5.1	1.5	virginica	7.65	16
17	6.3	2.9	5.6	1.8	virginica	10.08	17
18	6.3	3.3	6.0	2.5	virginica	15.0	18
19	6.3	3.4	5.6	2.4	virginica	13.44	19
20	6.4	2.7	5.3	1.9	virginica	10.07	20
21	6.4	2.8	5.6	2.1	virginica	11.76	21
22	6.4	2.8	5.6	2.2	virginica	12.32	22
23	6.4	3.1	5.5	1.8	virginica	9.9	23
24	6.4	3.2	5.3	2.3	virginica	12.19	24
25	6.5	3.0	5.8	2.2	virginica	12.76	25
26	6.5	3.0	5.5	1.8	virginica	9.9	26
27	6.5	3.0	5.2	2.0	virginica	10.4	27

將指定的欄位併縮到資料中，變成”長型”的表格

In [69]:
```
d = stack(iris, [:SepalLength, :SepalWidth, :PetalLength, :PetalWidth])
```

Out[69]:

600 rows × 5 columns

	variable	value	Species	PetalArea	id
	Symbol	Float64	Categorical...	Float64	Int64
1	SepalLength	4.9	virginica	7.65	1
2	SepalLength	5.6	virginica	9.8	2
3	SepalLength	5.7	virginica	10.0	3
4	SepalLength	5.8	virginica	9.69	4
5	SepalLength	5.8	virginica	9.69	5
6	SepalLength	5.8	virginica	12.24	6
7	SepalLength	5.9	virginica	9.18	7
8	SepalLength	6.0	virginica	7.5	8
9	SepalLength	6.0	virginica	8.64	9
10	SepalLength	6.1	virginica	7.84	10
11	SepalLength	6.1	virginica	8.82	11
12	SepalLength	6.2	virginica	8.64	12
13	SepalLength	6.2	virginica	12.42	13
14	SepalLength	6.3	virginica	9.5	14
15	SepalLength	6.3	virginica	8.82	15
16	SepalLength	6.3	virginica	7.65	16
17	SepalLength	6.3	virginica	10.08	17
18	SepalLength	6.3	virginica	15.0	18
19	SepalLength	6.3	virginica	13.44	19
20	SepalLength	6.4	virginica	10.07	20
21	SepalLength	6.4	virginica	11.76	21
22	SepalLength	6.4	virginica	12.32	22
23	SepalLength	6.4	virginica	9.9	23
24	SepalLength	6.4	virginica	12.19	24
25	SepalLength	6.5	virginica	12.76	25
26	SepalLength	6.5	virginica	9.9	26
27	SepalLength	6.5	virginica	10.4	27
28	SepalLength	6.5	virginica	10.2	28

併縮指定的欄位，並選擇其他欄位

In [70]:
```
d = stack(iris, [:SepalLength, :SepalWidth], :Species)
```

Out[70]:

300 rows × 3 columns

	variable	value	Species
	Symbol	Float64	Categorical...
1	SepalLength	4.9	virginica
2	SepalLength	5.6	virginica
3	SepalLength	5.7	virginica
4	SepalLength	5.8	virginica
5	SepalLength	5.8	virginica
6	SepalLength	5.8	virginica
7	SepalLength	5.9	virginica
8	SepalLength	6.0	virginica
9	SepalLength	6.0	virginica
10	SepalLength	6.1	virginica
11	SepalLength	6.1	virginica
12	SepalLength	6.2	virginica
13	SepalLength	6.2	virginica
14	SepalLength	6.3	virginica
15	SepalLength	6.3	virginica
16	SepalLength	6.3	virginica
17	SepalLength	6.3	virginica
18	SepalLength	6.3	virginica
19	SepalLength	6.3	virginica
20	SepalLength	6.4	virginica
21	SepalLength	6.4	virginica
22	SepalLength	6.4	virginica
23	SepalLength	6.4	virginica
24	SepalLength	6.4	virginica
25	SepalLength	6.5	virginica
26	SepalLength	6.5	virginica
27	SepalLength	6.5	virginica
28	SepalLength	6.5	virginica

將長型表格轉為寬型

第2個參數是可辨識列的欄位，第3、4個參數分別是併縮時的欄位名跟值

In [71]:
```
d = stack(iris, [:SepalLength, :SepalWidth, :PetalLength, :PetalWidth])
unstack(d, :id, :variable, :value)
```

Out[71]:

150 rows × 5 columns

	id	PetalLength	PetalWidth	SepalLength	SepalWidth
	Int64	Float64?	Float64?	Float64?	Float64?
1	1	4.5	1.7	4.9	2.5
2	2	4.9	2.0	5.6	2.8
3	3	5.0	2.0	5.7	2.5
4	4	5.1	1.9	5.8	2.7
5	5	5.1	1.9	5.8	2.7
6	6	5.1	2.4	5.8	2.8
7	7	5.1	1.8	5.9	3.0
8	8	5.0	1.5	6.0	2.2
9	9	4.8	1.8	6.0	3.0
10	10	5.6	1.4	6.1	2.6
11	11	4.9	1.8	6.1	3.0
12	12	4.8	1.8	6.2	2.8
13	13	5.4	2.3	6.2	3.4
14	14	5.0	1.9	6.3	2.5
15	15	4.9	1.8	6.3	2.7
16	16	5.1	1.5	6.3	2.8
17	17	5.6	1.8	6.3	2.9
18	18	6.0	2.5	6.3	3.3
19	19	5.6	2.4	6.3	3.4
20	20	5.3	1.9	6.4	2.7
21	21	5.6	2.1	6.4	2.8
22	22	5.6	2.2	6.4	2.8
23	23	5.5	1.8	6.4	3.1
24	24	5.3	2.3	6.4	3.2
25	25	5.8	2.2	6.5	3.0
26	26	5.5	1.8	6.5	3.0
27	27	5.2	2.0	6.5	3.0

若是其餘的欄位不重複，也可以不指定辨識欄位

In [72]:
```
unstack(d, :variable, :value)
```

Out[72]:

150 rows × 7 columns

	Species	PetalArea	id	PetalLength	PetalWidth	SepalLength	SepalWidth
	Categorical...	Float64	Int64	Float64⍰	Float64⍰	Float64⍰	Float64⍰
1	virginica	7.5	8	5.0	1.5	6.0	2.2
2	virginica	7.65	1	4.5	1.7	4.9	2.5
3	virginica	7.65	16	5.1	1.5	6.3	2.8
4	virginica	7.84	10	5.6	1.4	6.1	2.6
5	virginica	8.64	9	4.8	1.8	6.0	3.0
6	virginica	8.64	12	4.8	1.8	6.2	2.8
7	virginica	8.82	11	4.9	1.8	6.1	3.0
8	virginica	8.82	15	4.9	1.8	6.3	2.7
9	virginica	9.18	7	5.1	1.8	5.9	3.0
10	virginica	9.28	40	5.8	1.6	7.2	3.0
11	virginica	9.5	14	5.0	1.9	6.3	2.5
12	virginica	9.69	4	5.1	1.9	5.8	2.7
13	virginica	9.69	5	5.1	1.9	5.8	2.7
14	virginica	9.8	2	4.9	2.0	5.6	2.8
15	virginica	9.9	23	5.5	1.8	6.4	3.1
16	virginica	9.9	26	5.5	1.8	6.5	3.0
17	virginica	10.0	3	5.0	2.0	5.7	2.5
18	virginica	10.07	20	5.3	1.9	6.4	2.7
19	virginica	10.08	17	5.6	1.8	6.3	2.9
20	virginica	10.2	28	5.1	2.0	6.5	3.2
21	virginica	10.4	27	5.2	2.0	6.5	3.0
22	virginica	10.44	29	5.8	1.8	6.7	2.5
23	virginica	10.8	41	6.0	1.8	7.2	3.2
24	virginica	11.34	43	6.3	1.8	7.3	2.9
25	virginica	11.34	36	5.4	2.1	6.9	3.1
26	virginica	11.55	34	5.5	2.1	6.8	3.0
27	virginica	11.59	44	6.1	1.9	7.4	2.8
28	virginica	11.73	37	5.1	2.3	6.9	3.1

應用之前學過的就可以將資料任意變形與運算

對不同的:Species，分別計算他們各項特徵的平均

In [73]:
```
d = stack(iris)
x = by(d, [:variable, :Species], df -> DataFrame(vmean = StatsBase.mean(df[:, :value])))
unstack(x, :Species, :vmean)
```

Out[73]:

5 rows × 4 columns

	variable	virginica	versicolor	setosa
	Symbol	Float64⍰	Float64⍰	Float64⍰
1	PetalArea	11.2962	5.7204	0.3656
2	PetalLength	5.552	4.26	1.462
3	PetalWidth	2.026	1.326	0.246
4	SepalLength	6.588	5.936	5.006
5	SepalWidth	2.974	2.77	3.428

Gadfly

我們使用的是 Gadfly.jl 這個 module，所以還沒安裝的快去安裝吧~

In [74]:
```
using Gadfly
```

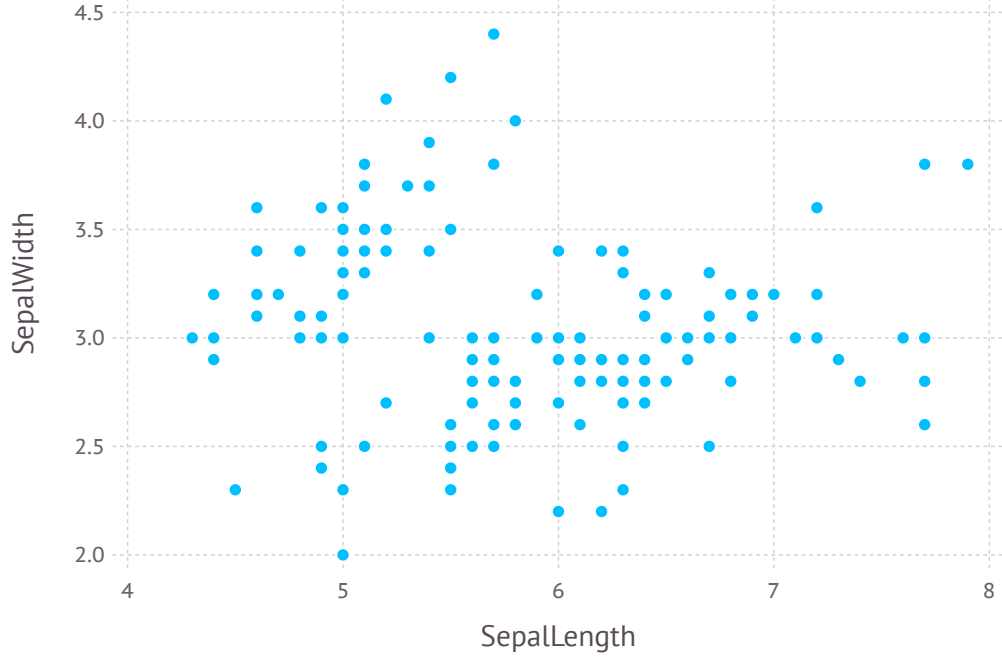
```
┌ Info: Loading DataFrames support into Gadfly.jl
└ @ Gadfly /home/pika/.julia/packages/Gadfly/ew1SM/src/mapping.jl:228
```

Tastes

In [75]:
```
p = plot(iris, x=:SepalLength, y=:SepalWidth, Geom.point)
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[75]:



In [76]:
```
img = SVG("iris_plot.svg", 14cm, 8cm)
draw(img, p)
```

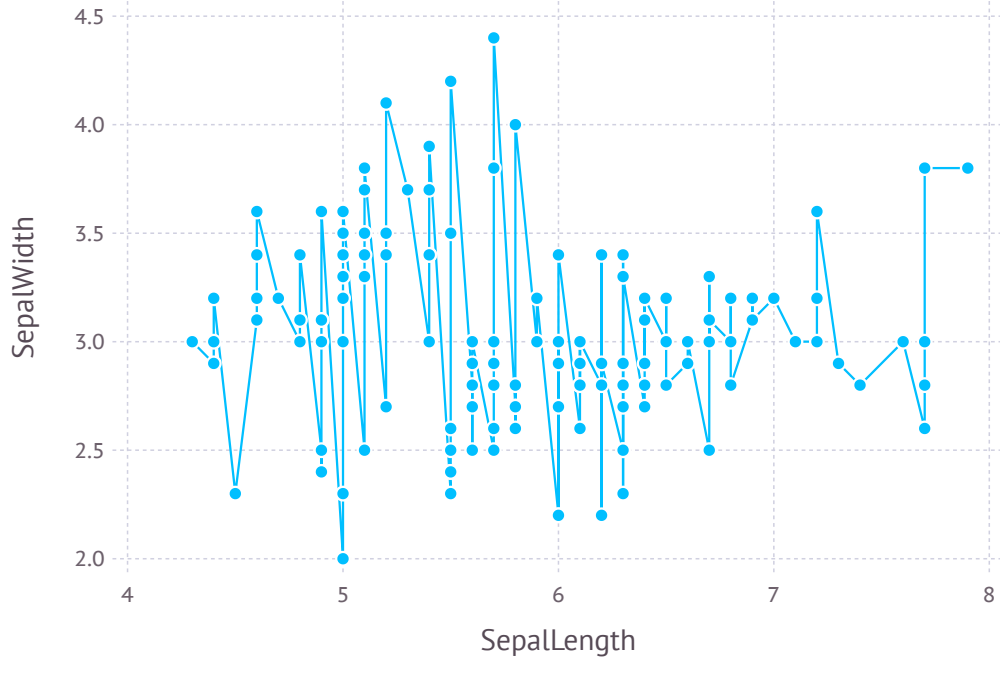
```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[76]:
```
false
```

With lines

In [77]:
```
plot(iris, x=:SepalLength, y=:SepalWidth, Geom.point, Geom.line)
```

Out[77]:




```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

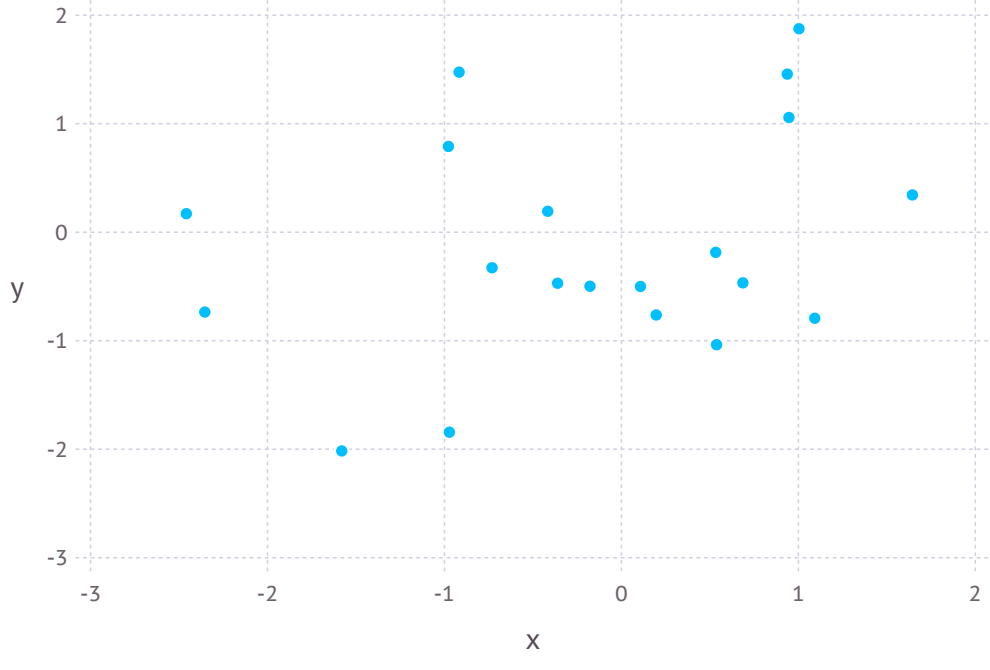
Plotting arrays

我們先從最簡單的Array開始，只要分別指定X跟Y的座標給他，他就可以幫你畫出圖來

In [78]:
```
plot(x=randn(20), y=randn(20))
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

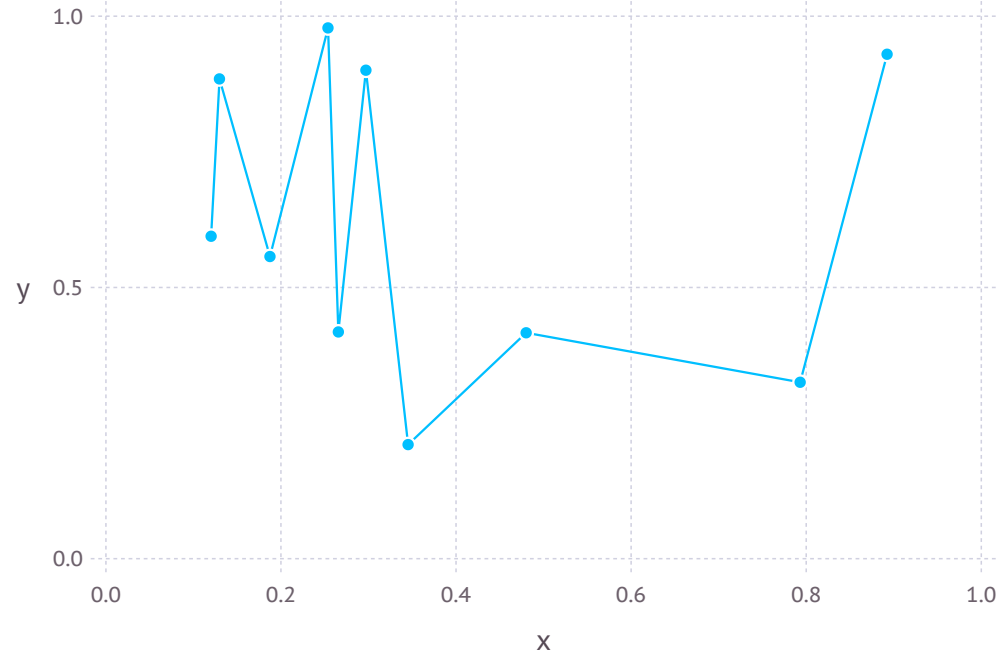
Out[78]:



Aesthetics

In [79]:
```
plot(x=rand(10), y=rand(10), Geom.point, Geom.line)
```

Out[79]:




```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

藉著我們要在上面加一些線，把這些點連起來，所以我們會用到兩個參數：

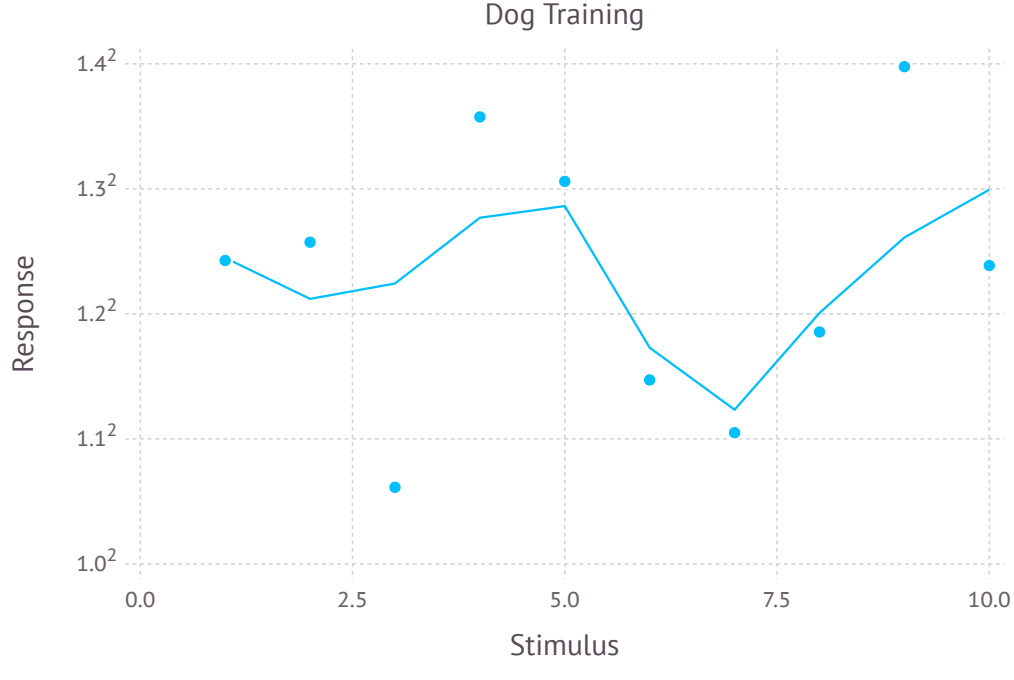
- Geom.point: 繪製點
- Geom.line: 繪製線，變成折線圖

Scale and guide

In [80]:
```
plot(x=1:10, y=2 .^rand(10), Scale.y_sqrt, Geom.point, Geom.smooth,
     Guide.xlabel("Stimulus"), Guide.ylabel("Response"), Guide.title("Dog Training"))
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[80]:



如果有需要調整軸的尺度或是操作標題跟軸線，可以用以下的參數：

- Scale: 操作軸的尺度
- Geom: 操作點跟線或是內容的呈現
- Guide: 增加標題跟軸的標示

Save

當你把你要的圖畫好之後，就可以存成不同的檔案了

```
myplot = plot(..)

draw(SVG("myplot.svg", 4inch, 3inch), myplot)
draw(PNG("myplot.png", 4inch, 3inch), myplot)
draw(PDF("myplot.pdf", 4inch, 3inch), myplot)
draw(PS("myplot.ps", 4inch, 3inch), myplot)
draw(D3("myplot.js", 4inch, 3inch), myplot)
```

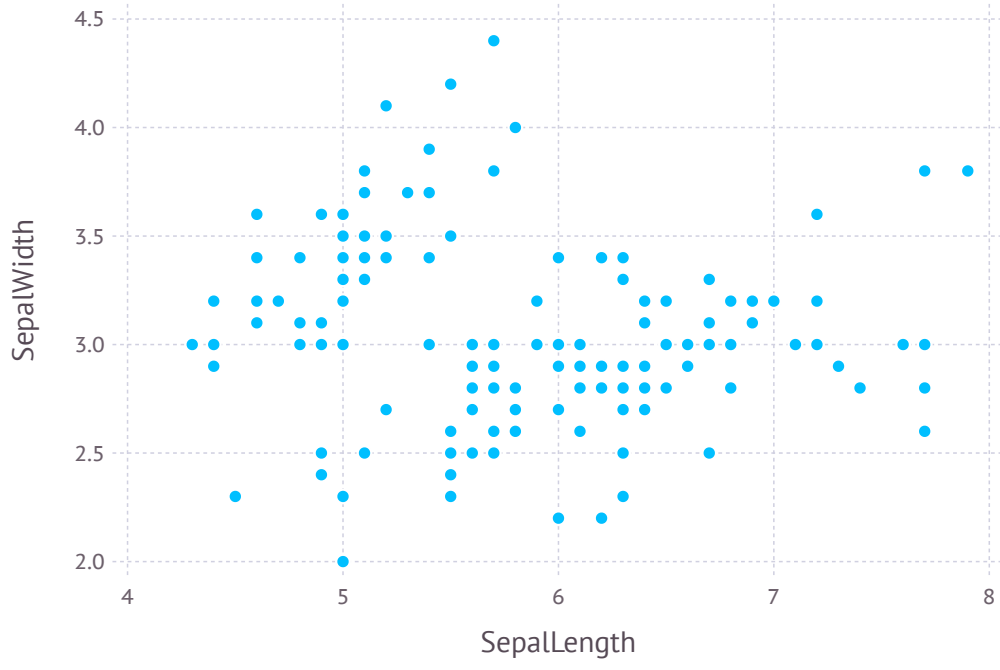
Plotting data frames

當你的資料裝在DataFrame裡的時候，只要指定欄位名稱就可以了

注意，第一個是放DataFrame

In [81]:
```
plot(iris, x="SepalLength", y="SepalWidth", Geom.point)
```

Out[81]:




```
 ┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
 └ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
 ┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
 └ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Histogram

In [82]:
```
plot(dataset("car", "SLID"), x="Wages", color="Language", Geom.histogram)
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[82]:

Functions and Expressions

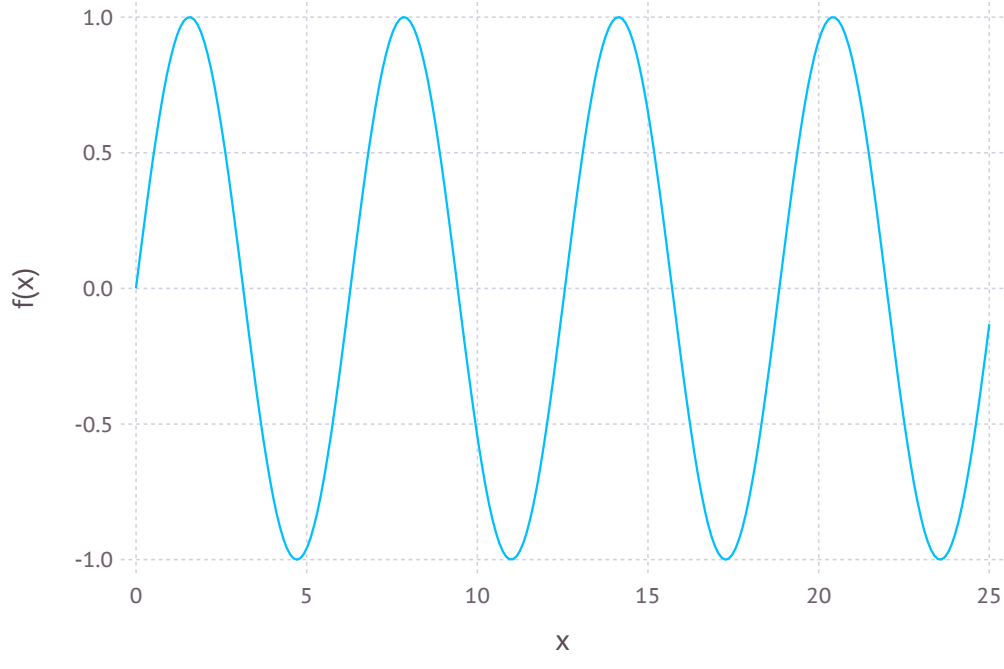
當你要畫的東西是function的話，那你就需要用到這邊的語法了

```
plot(f::Function, a, b, ...)
```

a跟b分別是圖形的起始位置跟終點位置，以X軸計算

In [83]:
```
plot(sin, 0, 25)
```

Out[83]:




```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

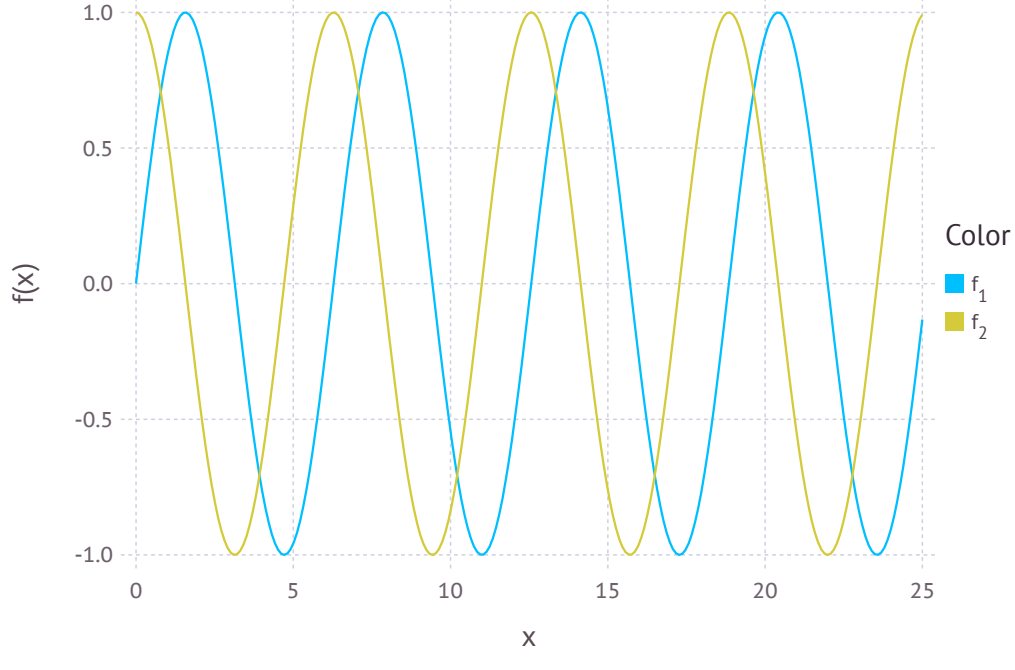
另一種是畫多個function：

```
plot(fs::Array, a, b, ...)
```

In [84]:
```
plot([sin, cos], 0, 25)
```

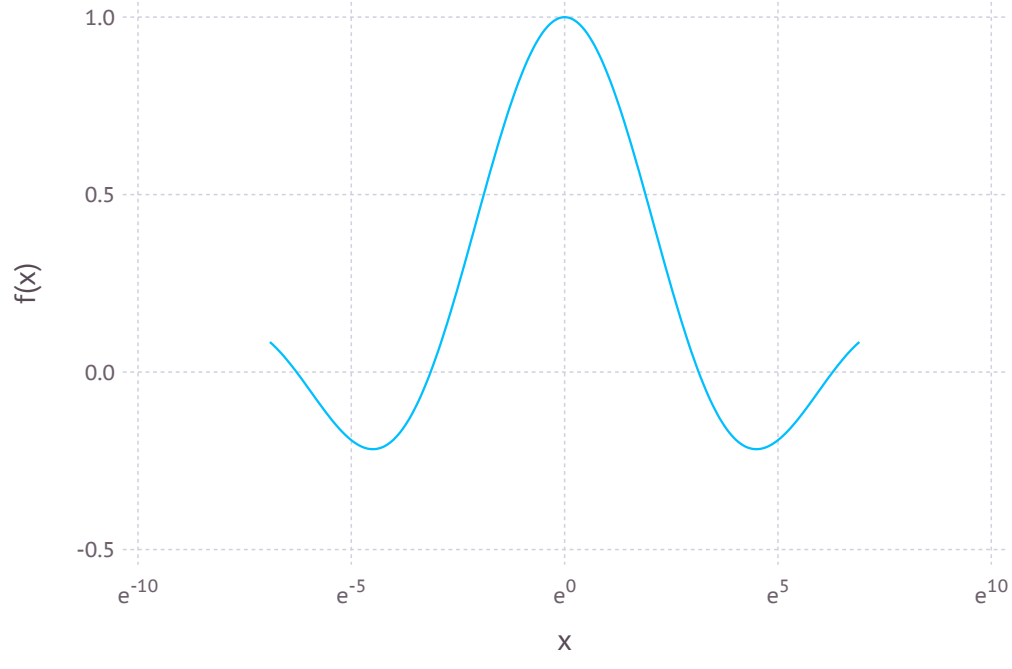
```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[84]:



In [85]:
```
plot((x) -> sin(x)/x, 0.001, 1000, Scale.x_log)
```

Out[85]:



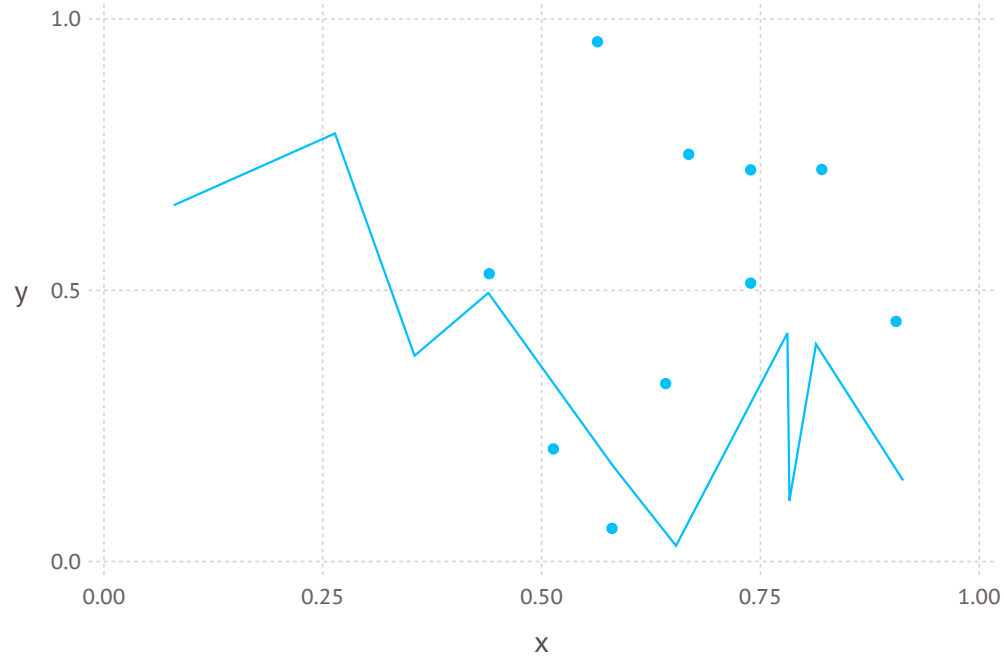

```
 ┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
 └ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
 ┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
 └ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Layers

當你需要在同一張上畫很多圖，而且他們是疊起來的，你需要用到layer這東西

In [86]:
```
plot(layer(x=rand(10), y=rand(10), Geom.point), layer(x=rand(10), y=rand(10), Geom.line))
```

Out[86]:

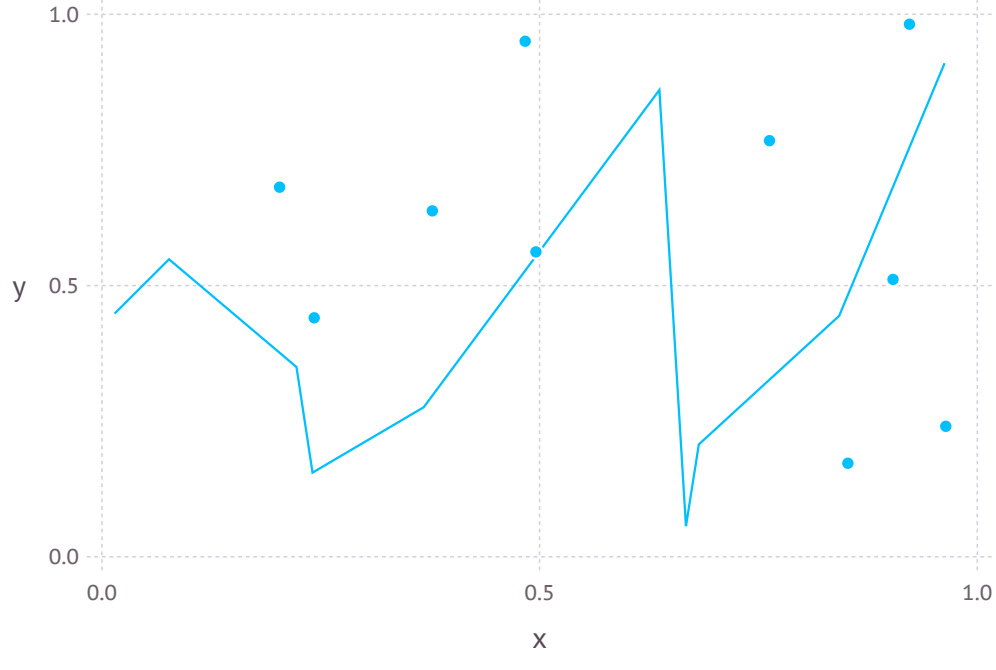



```
 ┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
 └ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
 ┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
 └ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

In [87]:
```
layer1 = layer(x=rand(10), y=rand(10), Geom.point)
layer2 = layer(x=rand(10), y=rand(10), Geom.line)
plot(layer1, layer2)  # 當有太多layer要畫的時候我就會用變數把他們分開
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[87]:



Geom

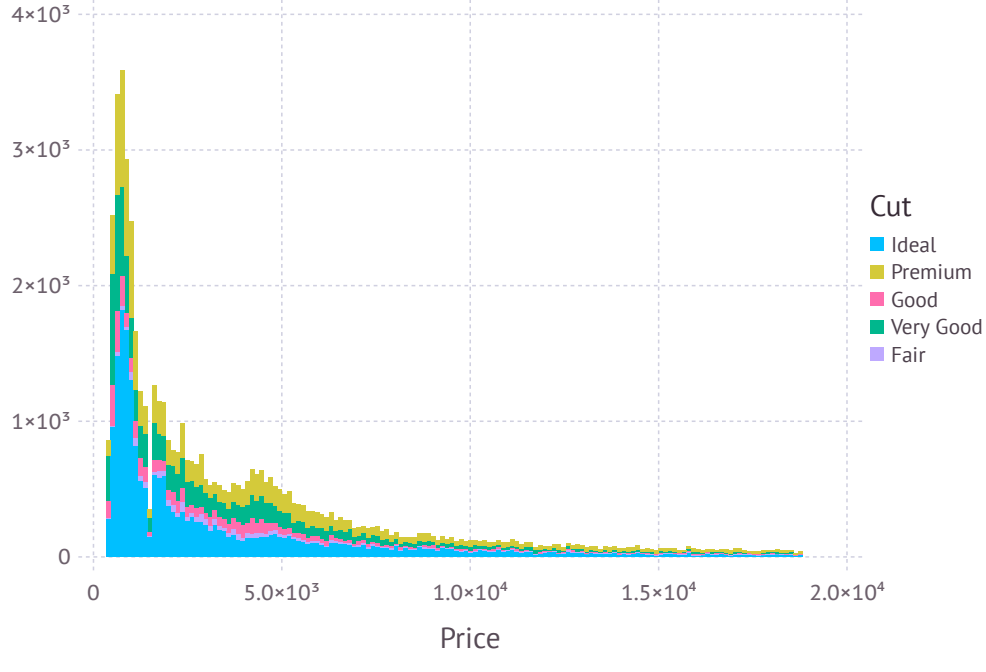
接下來介紹一下常用的圖

直方圖（Histogram）

In [88]:
```
plot(dataset("ggplot2", "diamonds"), x="Price", color="Cut", Geom.histogram)
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[88]:

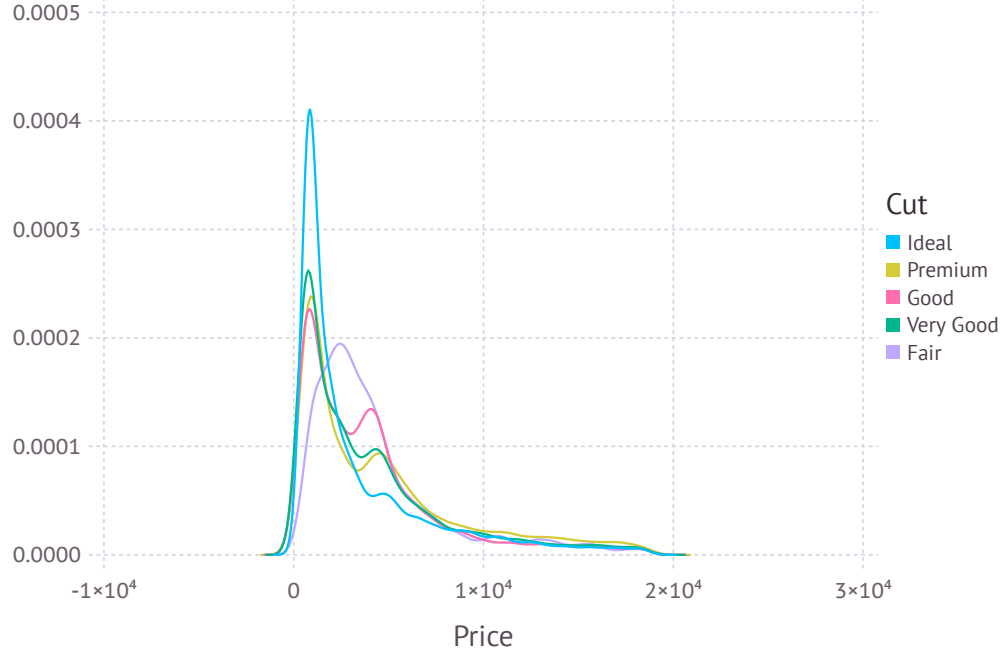


機率分佈圖（Kernel density estimation）

In [89]:
```
plot(dataset("ggplot2", "diamonds"), x="Price", color="Cut", Geom.density)
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[89]:

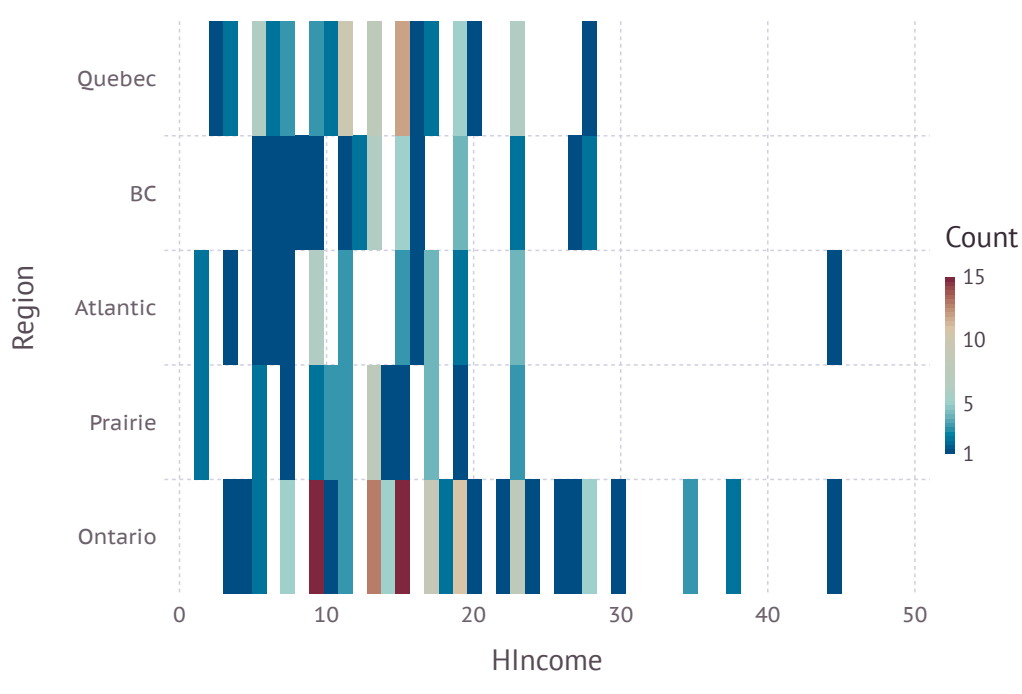


2D 直方圖（X跟Y分別是不同的維度，計數的多寡以顏色表示）

In [90]:
```
plot(dataset("car", "Womenlf"), x="HIncome", y="Region", Geom.histogram2d)
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[90]:



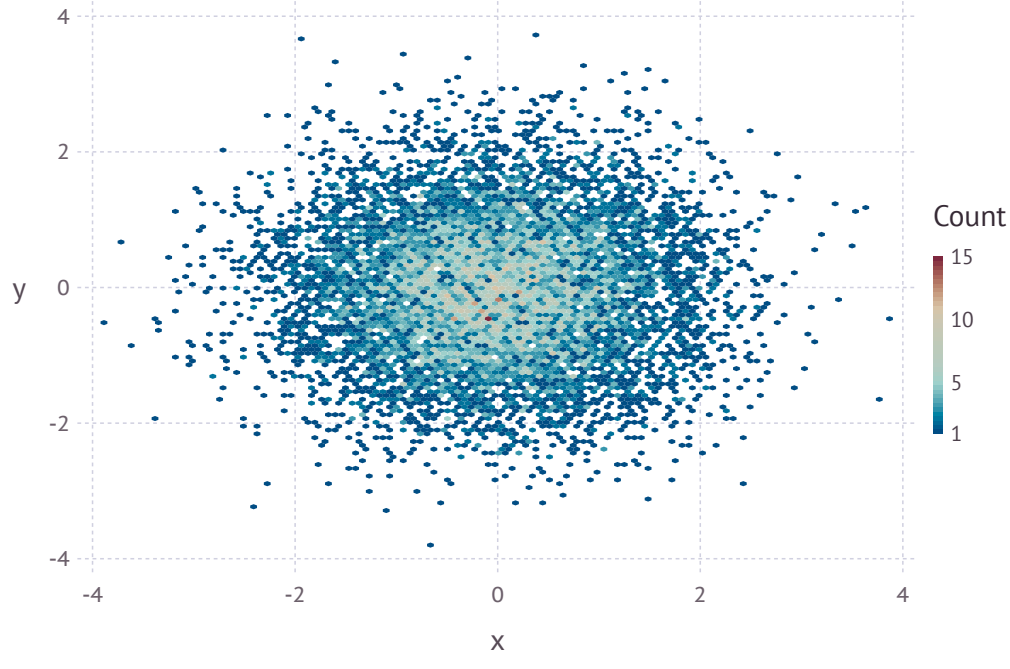
Hexbin plot

In [91]:
```
using Distributions
```

In [92]:
```
X = rand(MultivariateNormal([0.0, 0.0], [1.0 0.01; 0.01 1.0]), 10000);
plot(x=X[1,:], y=X[2,:], Geom.hexbin)
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[92]:

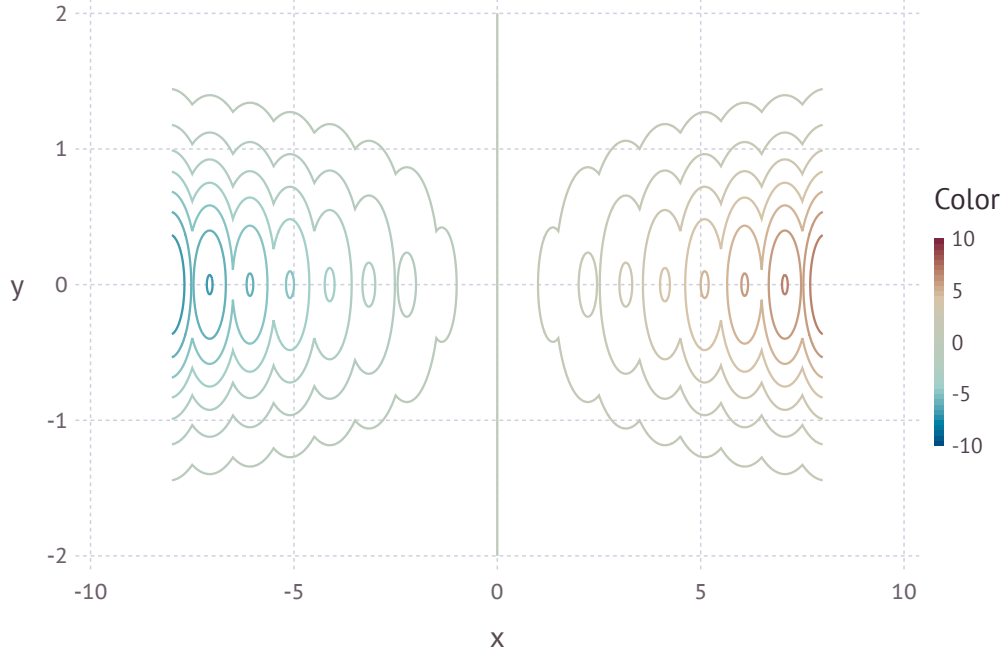


等高線圖（Contour, 適合用來表達二維的函數圖形）

In [93]:
```
plot(z=(x,y) -> x*exp(-(x-round(Int, x))^2-y^2),
     x=range(-8, stop=8, step=0.01), y=range(-2, stop=2, step=0.01), Geom.contour)
```

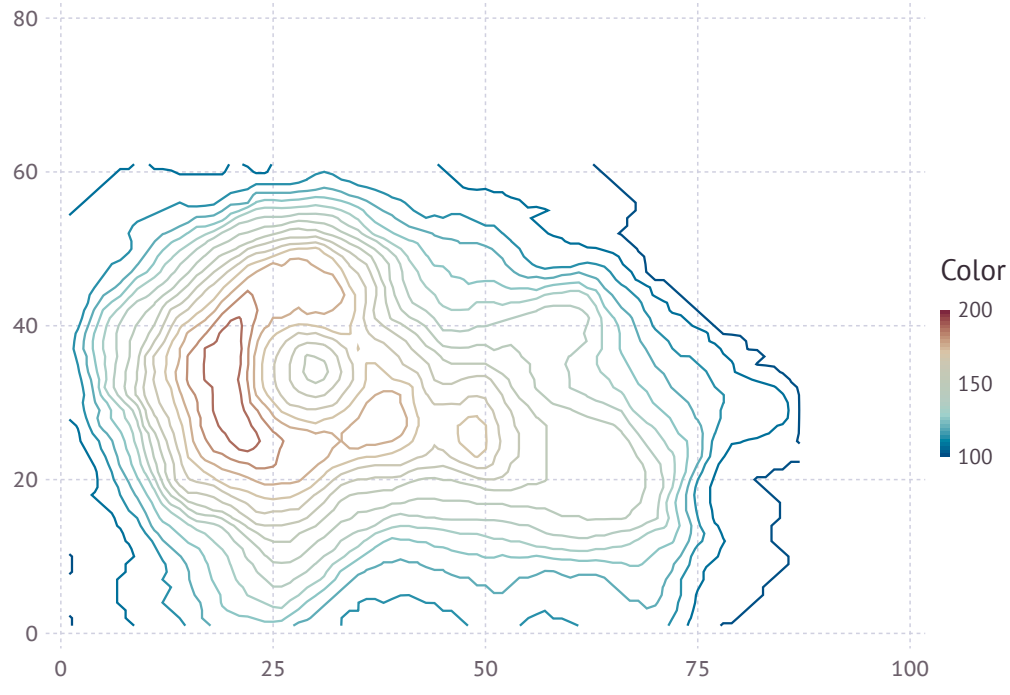
```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[93]:



In [94]:
```
volcano = convert(Array{Float64}, dataset("datasets", "volcano"))
plot(z=volcano, Geom.contour)
```

Out[94]:



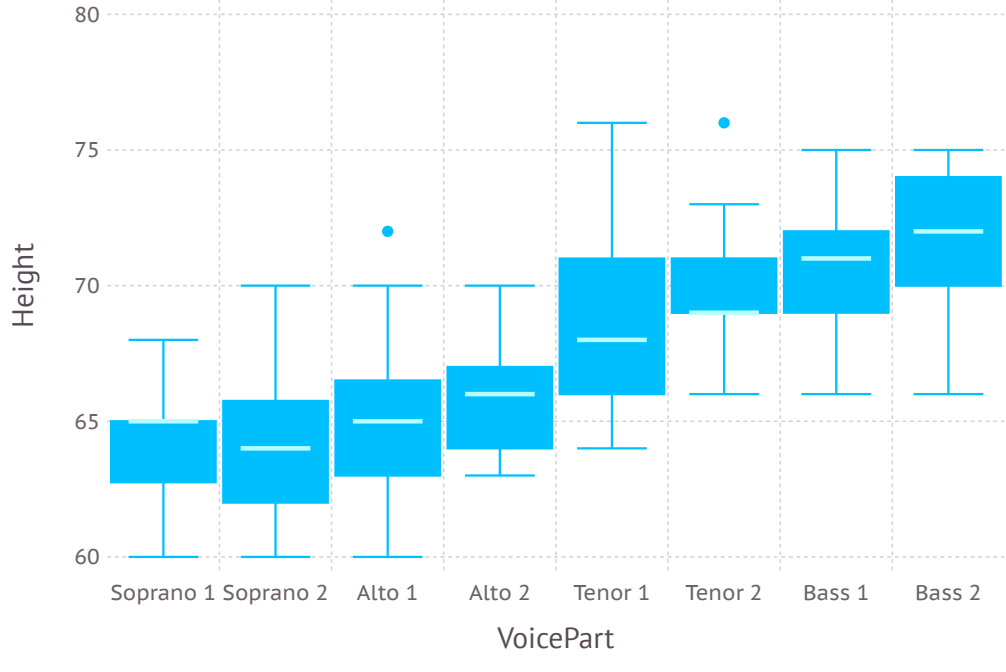

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

盒狀圖（Boxplot）

In [95]:
```
plot(dataset("lattice", "singer"), x="VoicePart", y="Height", Geom.boxplot)
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Out[95]:

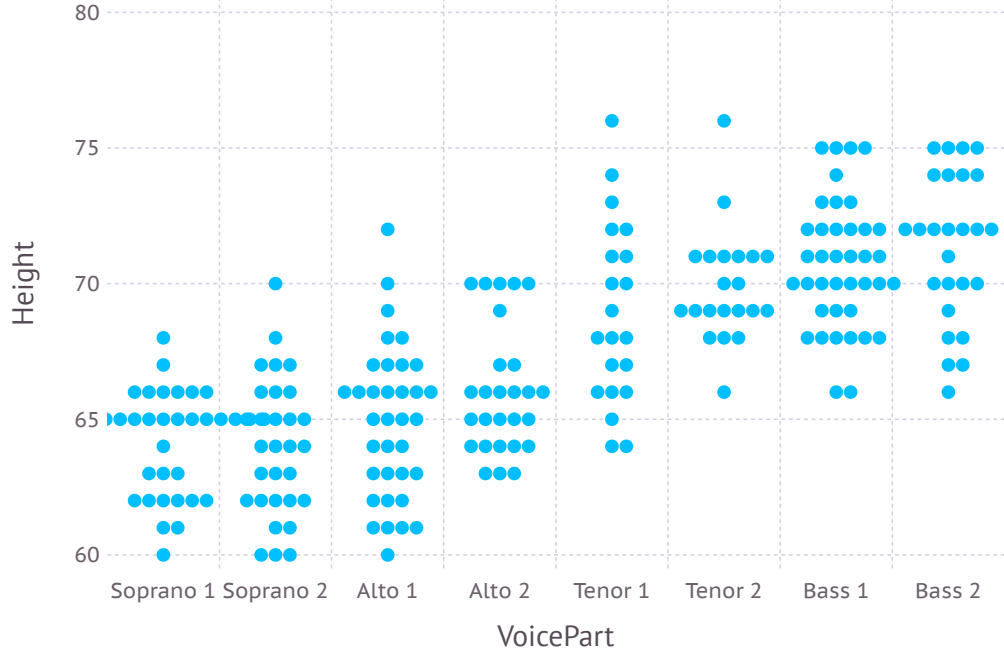


Beeswarm plot

In [96]:
```
plot(dataset("lattice", "singer"), x="VoicePart", y="Height", Geom.beeswarm)
```

```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

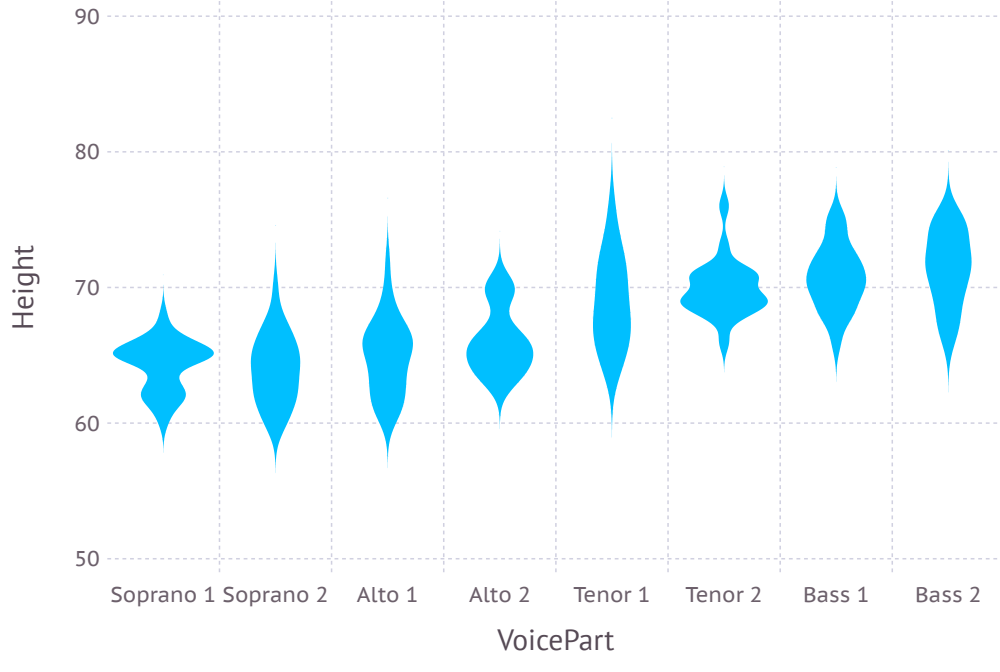
Out[96]:



Violin plot

In [97]:
```
plot(dataset("lattice", "singer"), x="VoicePart", y="Height", Geom.violin)
```

Out[97]:

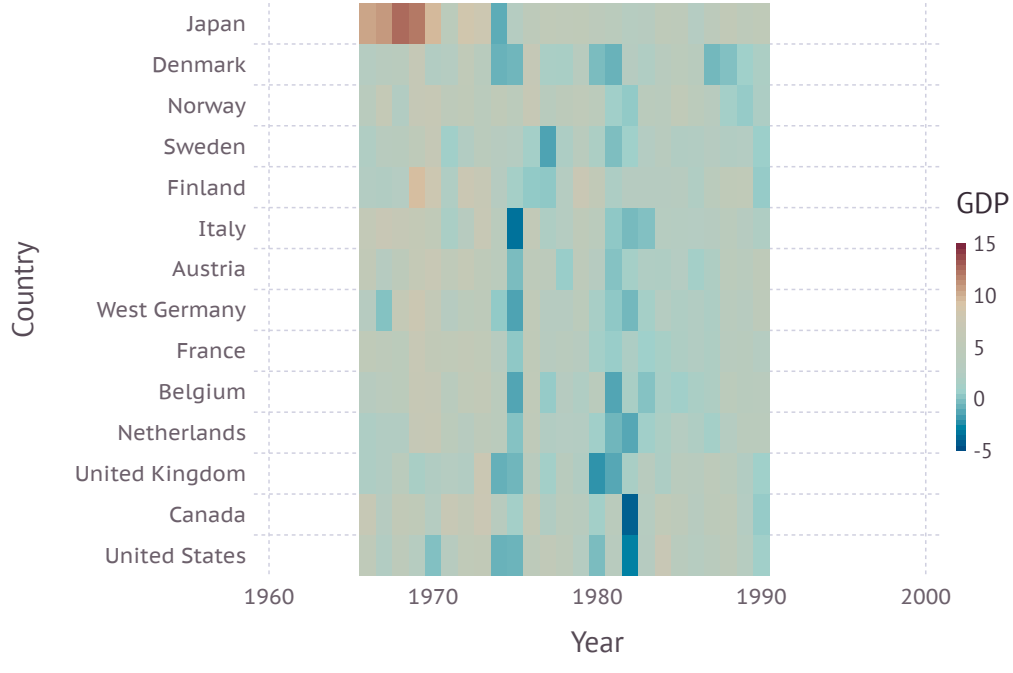



```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

熱圖（Heatmap）

In [98]:
```
plot(dataset("Zelig", "macro"), x="Year", y="Country", color="GDP", Geom.rectbin)
```

Out[98]:




```
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
┌ Warning: For svg transparent colors, use either e.g. fill(RGBA(r,g,b,a)) or fillopacity(a),
but not both.
└ @ Compose /home/pika/.julia/packages/Compose/wlPCt/src/svg.jl:1271
```

Q&A

其他套件：

- JSON.jl
- LightXML.jl
- HDF5.jl
- JLD2.jl
- SQLite.jl
- MySQL.jl