

A108222040Data_Visulization_Week14&15&16

Week14

Show dataset(國家快樂指數)

The screenshot shows the Spyder Python IDE interface. On the left, there is a code editor with several tabs open, including 'L4_IDV_Exercise_A108222040.py'. The code in this tab is related to data processing and visualization. A red box highlights a specific section of the code where a CSV file is read and its properties are printed. On the right, there is a Jupyter notebook cell output window titled 'In [1]'. It displays the command 'runfile('E:/Data_Visualization/Class/Pan_Lib_Python/L4_IDV_Exercise_A108222040.py', wdir='E:/Data_Visualization/Class/Pan_Lib_Python')' and the resulting DataFrame information. A red box highlights the 'Numerical Data Info' section of the output, which provides statistical details for the 'HPI Rank' column.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue May 24 21:37:00 2022
4
5 @date:2022/05/24
6 @author: A108222040
7 @subject:
8 #Environment
9 import os
10 import sys
11 outputPath = os.path.join("../","Output")
12 InputPath = os.path.join("../","Input")
13 imagePath = os.path.join("../","Image")
14 libPath = os.path.join("Python")
15
16 ##Packages
17 import pandas as pd
18 import numpy as np
19 import seaborn as sns
20 from matplotlib import pyplot as plt
21
22 import altair as alt
23 from vega_datasets import data
24
25 #####Data Process]
26 def L4_dataset(dataset=InputPath + "/hpi1_data_countries.csv", show=True):
27     df = pd.read_csv(dataset, sep=',')
28     if show:
29         print("Columns info. -->\n", df.info(), sep="")
30         print("\nNumerical Data info. -->\n", df.describe(), sep="")
31         print("\nCategorical Data info. -->\n", df.describe(include=[object]), sep="")
32         print("\nData -->\n", df.head(5), sep="")
33     return df
34
35 #change實心
36 def ex27_SDV_Zoom():
37     chart = alt.Chart(df, title="Wellbeing vs. HPI").mark_point().encode(
38         x = ('Wellbeing (0~10):Q'),
39         y = ('Happy Planet Index:Q'),
40         color='Region:N',
41         tooltip=['Country', "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
42     ).interactive(#zoom in/out
43     ).properties(width=400, height=300
44     ).configure_title(color="green", fontSize=24
45     ).configure_axis(labelFontSize=14,
46                     labelColor="red",
47                     titleFontSize=20,
48                     titleColor="blue",
49     )
50     chart.show()
51
```

```
In [1]: runfile('E:/Data_Visualization/Class/Pan_Lib_Python/L4_IDV_Exercise_A108222040.py', wdir='E:/Data_Visualization/Class/Pan_Lib_Python')
Out[1]:
```

Column	Non-Null Count	Dtype
---	---	---
0 HPI Rank	140 non-null	int64
1 Country	140 non-null	object
2 Region	140 non-null	object
3 Life Expectancy (years)	140 non-null	float64
4 Wellbeing (0~10)	140 non-null	float64
5 International of outcomes	140 non-null	object
6 Ecological Footprint (gha/capita)	140 non-null	float64
7 Happy Planet Index	140 non-null	float64

Numerical Data Info. -->

HPI Rank	Happy Planet Index
count	140.000000
mean	26.407857
std	7.119404
min	1.0000
25%	12.8000
50%	21.1700
75%	31.5500
max	44.7000

[8 rows x 5 columns]

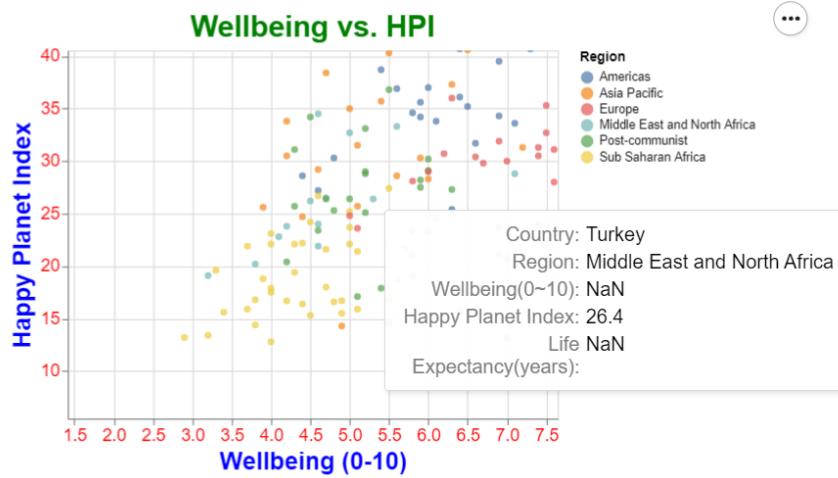
Ex27 circle 是實心 point 是空心

The screenshot shows the Spyder Python IDE interface. On the left, there is a code editor with several tabs open, including 'L4_IDV_Exercise_A108222040.py'. The code in this tab is related to data processing and visualization. A red box highlights a specific section of the code where a chart is created using Altair's mark_circle() function. On the right, there is a Jupyter notebook cell output window titled 'In [1]'. It displays the command 'runfile('E:/Data_Visualization/Class/Pan_Lib_Python/L4_IDV_Exercise_A108222040.py', wdir='E:/Data_Visualization/Class/Pan_Lib_Python')' and the resulting chart. A red box highlights the chart area, showing a scatter plot with data points represented by circles.

```
33     return df
34
35 #change實心
36 def ex27_SDV_Zoom(df):
37     chart = alt.Chart(df, title="Wellbeing vs. HPI").mark_circle().encode(
38         x = ('Wellbeing (0~10):Q'),
39         y = ('Happy Planet Index:Q'),
40         color='Region:N',
41         tooltip=['Country', "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
42     ).interactive(#zoom in/out
43     ).properties(width=400, height=300
44     ).configure_title(color="green", fontSize=24
45     ).configure_axis(labelFontSize=14,
46                     labelColor="red",
47                     titleFontSize=20,
48                     titleColor="blue",
49     )
50     chart.show()
51
```

```
In [1]: runfile('E:/Data_Visualization/Class/Pan_Lib_Python/L4_IDV_Exercise_A108222040.py', wdir='E:/Data_Visualization/Class/Pan_Lib_Python')
Out[1]:
```

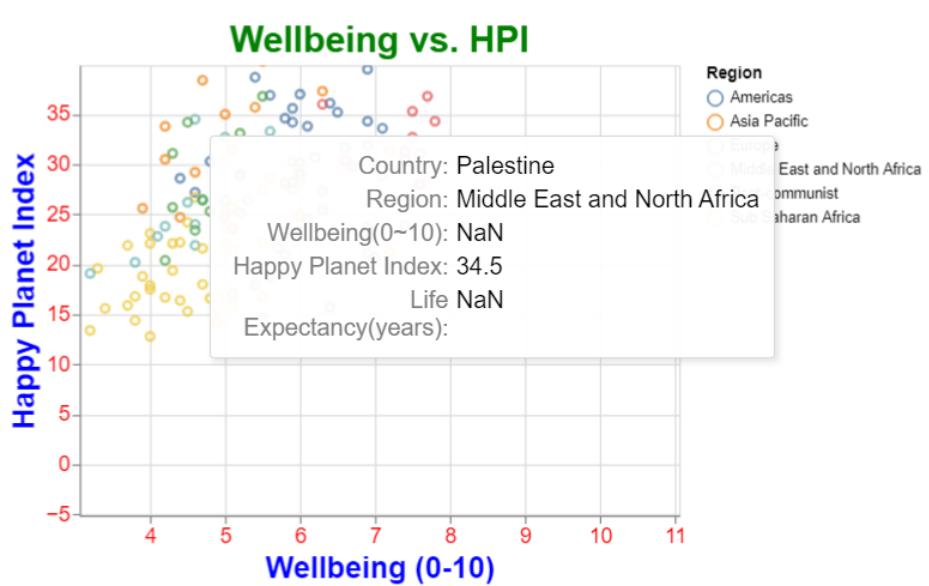
Result



Ex28

```
50 chart.show()
51
52 def ex28_SDV_Zoom_Hover(df):
53     chart = alt.Chart(df, title="Wellbeing vs. HPI").mark_circle().encode(
54         x = 'Wellbeing (0-10):Q',
55         y = 'Happy Planet Index:Q',
56         color='Region:N',
57         tooltip=[ "Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q" ]
58     ).interactive( #zoom in/out
59     ).properties(width=400, height=300
60     ).configure_title(color='green', fontSize=24
61     ).configure_axis(labelFontSize=14,
62                     labelColor="red",
63                     titleFontSize=20,
64                     titleColor="blue",
65                 )
66 chart.show()
67
```

Result

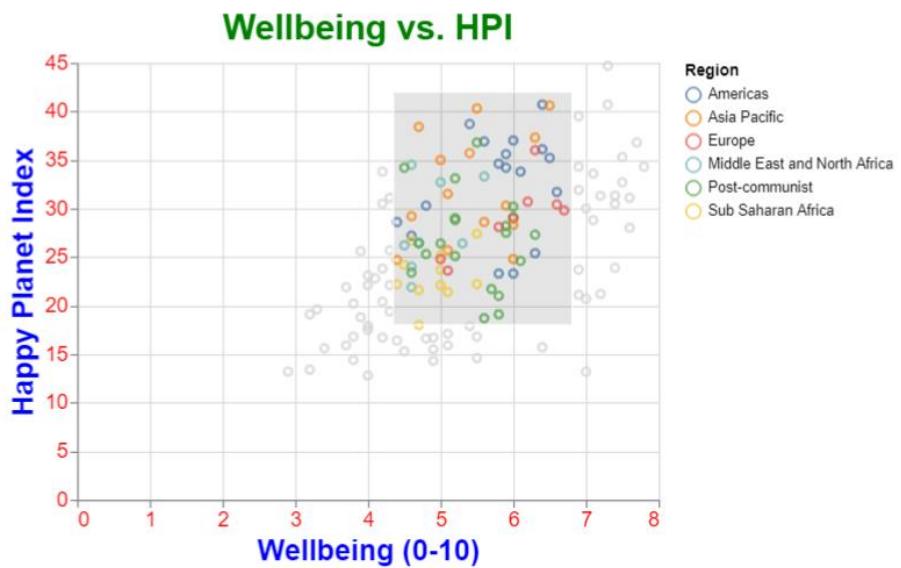


Ex29

可以選擇一塊區域做觀看

```
65     )
66     chart.show()
67
68 def ex29_SelectedArea(df):
69     selected_area = alt.selection_interval()
70
71     chart = alt.Chart(df, title="Wellbeing vs. HPI").mark_point().encode(
72         x="Wellbeing (0-10):Q",
73         y="Happy Planet Index:Q",
74         color=alt.condition(selected_area, "Region:N", alt.value('lightgray'))
75     ).configure_tooltips(["Country", "Region", "Wellbeing(0-10):Q", "Happy Planet Index", "Life Expectancy(years):Q"])
76     .add_selection(selected_area)
77     .properties(width=600, height=300)
78     .configure_title(color="green", fontSize=24)
79     .configure_axis(labelFontSize=14,
80                      titleFontSize=20,
81                      titleColor="red")
82     .chart.show()
83
84 def ex30_SelectedArea_Zoom_Hover(df):
85     selected_area = alt.selection_interval()
```

In [3]: runfile('E:/Data_Visualization/Class/Pan_Lib_Python/L4_IDV_Exercise_A108222040.py', wdir='E:/Data_Visualization/Class/Pan_Lib_Python')
Displaying chart at http://localhost:16734/
In [4]: runfile('E:/Data_Visualization/Class/Pan_Lib_Python/L4_IDV_Exercise_A108222040.py', wdir='E:/Data_Visualization/Class/Pan_Lib_Python')
Displaying chart at http://localhost:16734/
In [5]: runfile('E:/Data_Visualization/Class/Pan_Lib_Python/L4_IDV_Exercise_A108222040.py', wdir='E:/Data_Visualization/Class/Pan_Lib_Python')
Displaying chart at http://localhost:16734/
In [6]: runfile('E:/Data_Visualization/Class/Pan_Lib_Python/L4_IDV_Exercise_A108222040.py', wdir='E:/Data_Visualization/Class/Pan_Lib_Python')

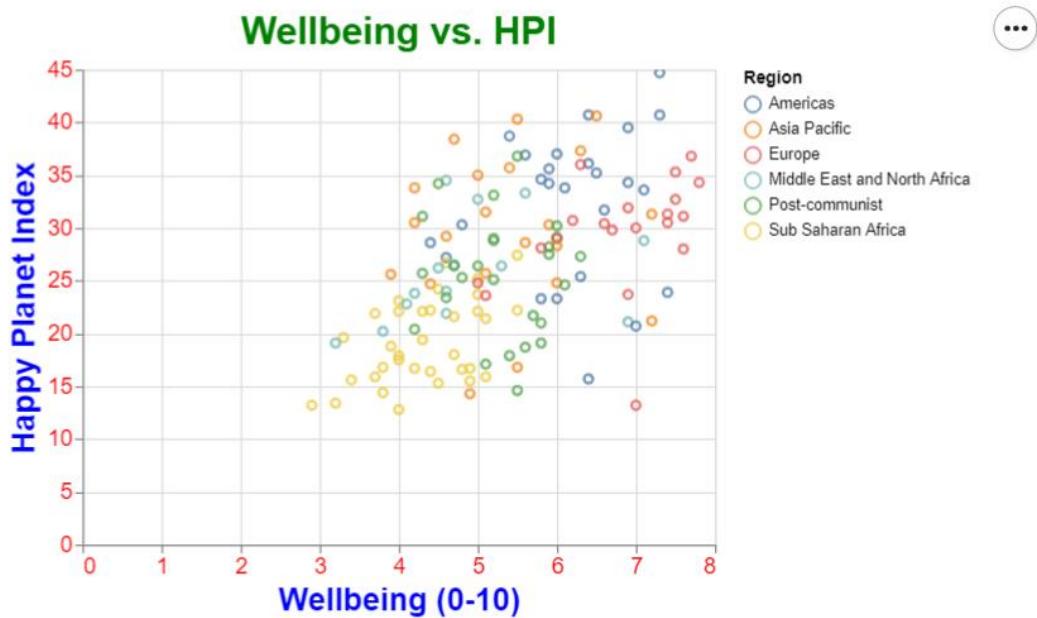


Ex30

可以選擇一塊區域做觀看 Zoom 移動

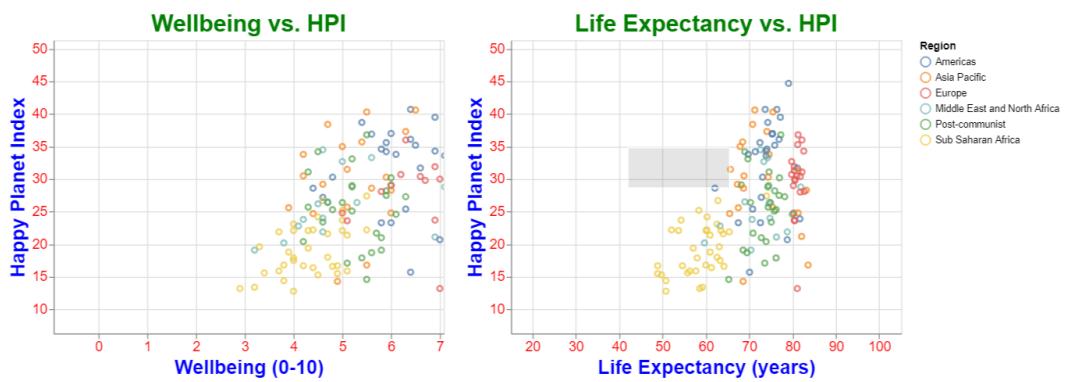
紅框為與 29 不同之處

```
84     chart.show()
85
86 def ex30_SelectedArea_Zoom_Hover(df):
87     selected_area = alt.selection_interval()
88
89     chart = alt.Chart(df, title="Wellbeing vs. HPI").mark_point().encode(
90         x='Wellbeing (0-10):Q',
91         y='Happy Planet Index:Q',
92         color=alt.condition(selected_area, 'Region:N', alt.value('lightgray')),
93         tooltip=['Country', "Region", "Wellbeing(0-10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
94     ).interactive()
95     .add_selection(selected_area)
96     .properties(width=400, height=300)
97     .configure_title(color='green', fontSize=24)
98     .configure_axis(labelFontSize=14,
99                     labelColor="red",
100                    titleFontSize=20,
101                    titleColor="blue",
102    )
```



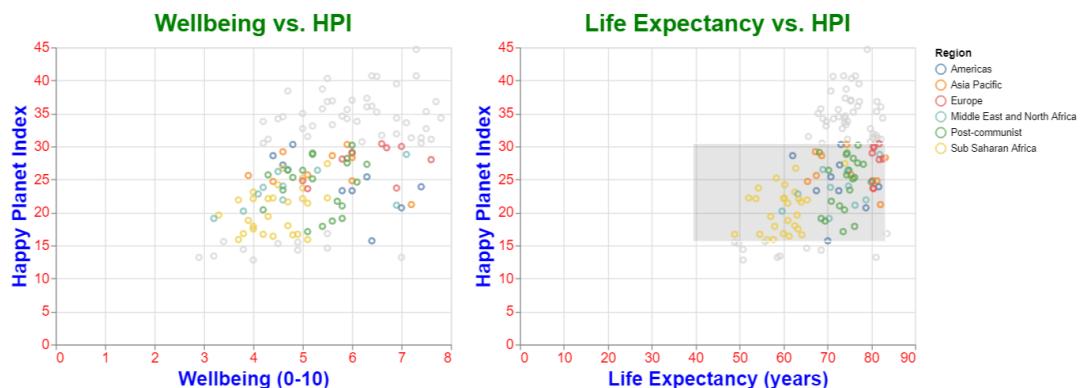
Ex31 左右兩個區域 可以同時移動

```
104
105 def ex31_MultiplePlot(df):
106     selectedArea = alt.selection_interval()
107
108     chart = alt.Chart(df, title="HPI").mark_point().encode(
109         x='Wellbeing (0-10):Q',
110         y='Happy Planet Index:Q',
111         color='Region:N',
112         tooltip=["Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
113     ).interactive(
114     ).add_selection(selectedArea)
115     .properties(width=400, height=300)
116     # .configure_title(color='green', fontSize=24
117     # ).configure_axis(labelFontSize=14,
118     #                 labelColor="red",
119     #                 titleFontSize=20,
120     #                 titleColor="blue",
121     # )
122     chart1 = chart.encode(x='Wellbeing (0-10)').properties(title="Wellbeing vs. HPI")
123     chart2 = chart.encode(x='Life Expectancy (years)').properties(title="Life Expectancy vs. HPI")
124     chart = alt.hconcat(chart1, chart2
125                         ).configure_title(color='green', fontSize=24
126                         ).configure_axis(labelFontSize=14,
127                                         labelColor="red",
128                                         titleFontSize=20,
129                                         titleColor="blue",
130                         )
131     chart.show()
```



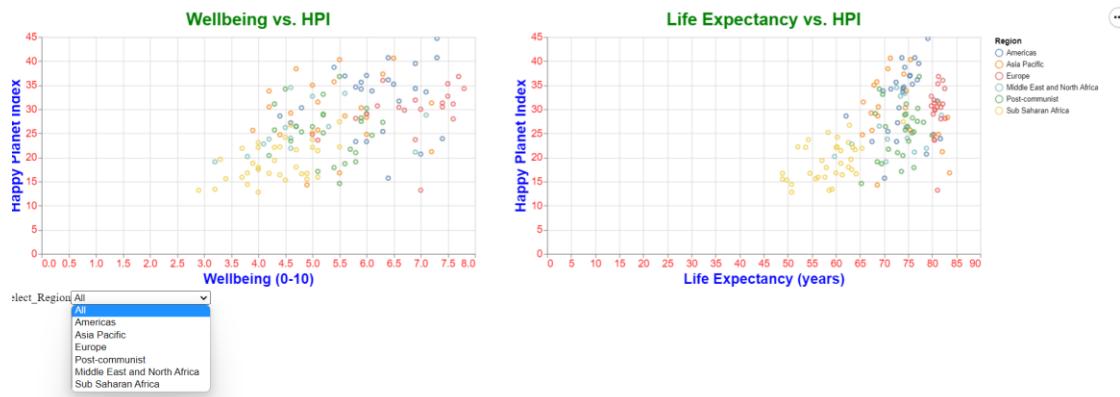
Ex31 右邊選擇區域 左邊就會亮該處

```
132
133     def ex31_Selection(df):
134         selectedArea = alt.selection_interval()
135
136         chart = alt.Chart(df, title="HPI").mark_point().encode(
137             #x='Wellbeing (0-10):Q',
138             y='Happy Planet Index:Q',
139             color=alt.condition(selectedArea, 'Region', alt.value('lightgray')),
140             tooltip=['Country', "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
141         ),interactive()
142         .add_selection(selectedArea)
143         .properties(width=400, height=300)
144         # .configure_title(color='green', fontSize=24
145         # ).configure_axis(labelFontSize=14,
146         #     labelColor="red",
147         #     titleFontSize=20,
148         #     titleColor="blue",
149         # )
150         chart.encode(x='Wellbeing (0-10)').properties(title="Wellbeing vs. HPI")
151         chart2 = chart.encode(x='Life Expectancy (years)').properties(title="Life Expectancy vs. HPI")
152         charts = alt.hconcat(chart1, chart2
153             ).configure_title(color='green', fontSize=24
154             ).configure_axis(labelFontSize=14,
155                 labelColor="red",
156                 titleFontSize=20,
157                 titleColor="blue",
158             )
159         charts.show()
```



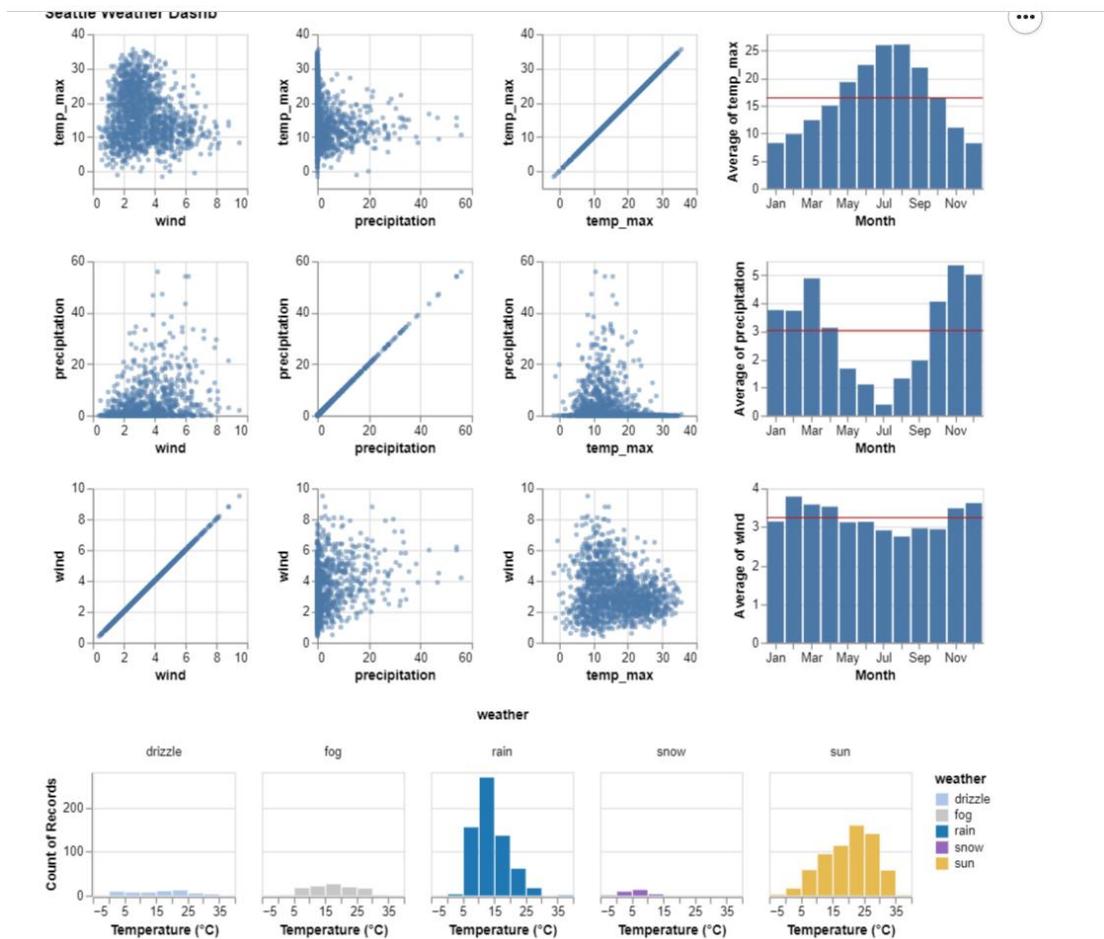
Ex32 有下拉式選單

```
160
161 def ex32_Selection_FeatureValue_MP(df):
162     #input_dropdown = alt.binding_select(options=list(r))
163     input_dropdown = alt.binding_select(
164         options=[None] + list(df.Region.unique()), labels = [ 'All' ] + list(df.Region.unique()))
165     selected_points = alt.selection_single(fields=['Region'],
166                                             bind=input_dropdown, name='Select')
167
168     #chart
169     chart = alt.Chart(df, title="Wellbeing vs. HPI").mark_point().encode(
170         x='Wellbeing (0~10):Q',
171         y='Happy Planet Index:Q',
172         color=alt.condition(selected_points, alt.Color("Region:N"), alt.value("lightgray")),
173         tooltip=["Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
174     ).interactive(
175     ).add_selection(selected_points
176     ).properties(width=600, height=300)
177
178     chart1 = chart.encode(x='Wellbeing (0~10)').properties(title="Wellbeing vs. HPI")
179     chart2 = chart.encode(x='Life Expectancy (years)').properties(title="Life Expectancy vs. HPI")
180     charts = alt.hconcat(chart1, chart2, spacing=50
181                         ).configure_title(color='green', fontSize=24
182                         ).configure_axis(labelFontSize=14,
183                                         labelColor='red',
184                                         titleFontSize=20,
185                                         titleColor='blue',
186                         )
187     charts.show()
```



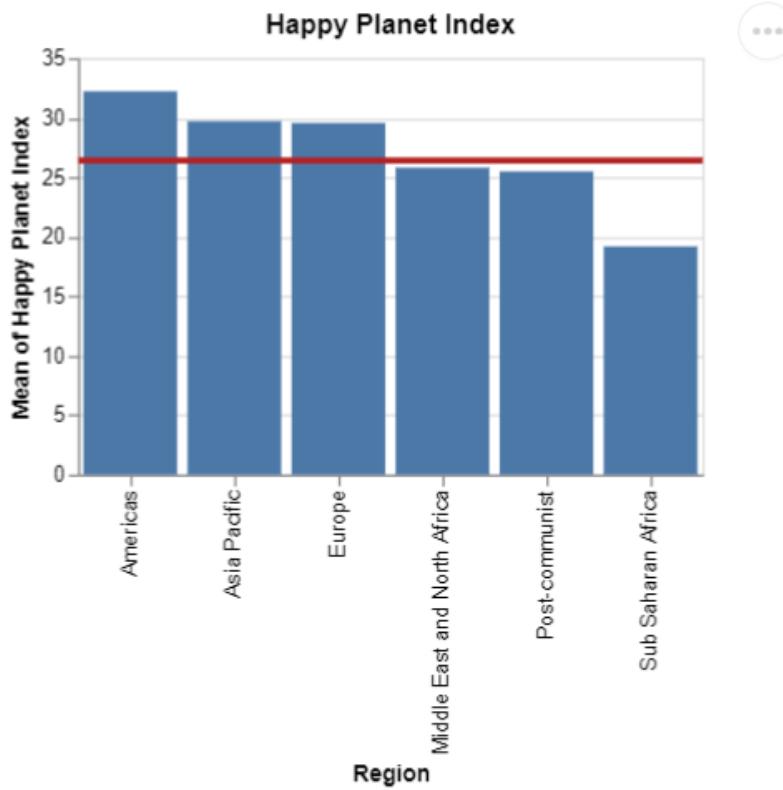
範例 weather

```
187     charts.show()
188
189 def testAltair_MP_weather():
190     #---Load Data
191     weather = 'https://vega.github.io/vega-datasets/data/weather.csv'
192     df = pd.read_csv(weather)
193
194     #--plot
195     splom = alt.Chart().mark_point(filled=True, size=15, opacity=0.5
196         ).encode(alt.X(alt.repeat('column'), type='quantitative'),
197                 alt.Y(alt.repeat('row'), type='quantitative')
198             ).properties(width=125, height=125
199             ).repeat(row=['temp_max', 'precipitation', 'wind'],
200                     column=[ 'wind', 'precipitation', 'temp_max'])
201
202     dateHist = alt.layer(alt.Chart().mark_bar().encode(
203         alt.X('month(date):O', title='Month'),
204         alt.Y(alt.repeat('row'), aggregate='average', type='quantitative'),
205         alt.Chart().mark_rule(stroke='firebrick').encode(
206             alt.Y(alt.repeat('row'), aggregate='average', type='quantitative'))
207         ).properties(width=175, height=125
208         ).repeat(row=[ 'temp_max', 'precipitation', 'wind'])
209
210     tempHist = alt.Chart(weather).mark_bar().encode(
211         alt.X('temp_max:Q', bin=True, title='Temperature (°C)'),
212         alt.Y('count():Q'),
213         alt.Color('weather:N', scale=alt.Scale(
214             domain=['drizzle', 'fog', 'rain', 'snow', 'sun'],
215             range=[ '#aec7e8', '#c7c7c7', '#1f77b4', '#9467bd', '#e7ba52' ]))
216         ).properties(width=115, height=100
217         ).facet(column='weather:N')
218
219     chart = alt.vconcat(alt.hconcat(splom, dateHist), tempHist, data=weather, title='Seattle Weather Dashb'
220         ).transform_filter('datum.location == "Seattle"'
221         ).resolve_legend(color='independent'
222         ).configure_axis(labelAngle=0)
223
224     chart.show()
```



Ex33 bar 一樣是用快樂指數

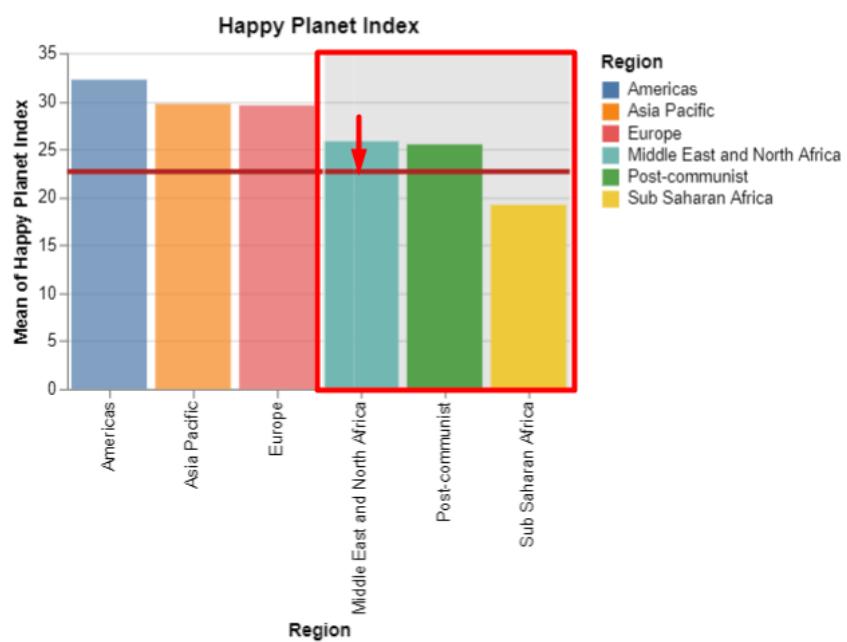
```
225
226 def ex33_Bar(df):
227     columns = df.columns
228     xData = columns[2] + ":N"
229     yData = "mean(" + columns[7] + "):Q"
230     bars = alt.Chart(df, title="Happy Planet Index").mark_bar().encode(
231         x=xData, # x=alt.X('x', axis=alt.Axis(format='%', title='Wellbeing'))
232         y=yData, # y=alt.Y('y', axis=alt.Axis(labels=False))
233         #color='Region:N',
234         #color=alt.condition(selectedArea, "Region:N", alt.value("lightgray")),
235         #tooltip=["Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
236     ).properties(width=300, height=200)
237     #).interactive()
238     #).add_selection(selected_points
239
240     line = alt.Chart(df).mark_rule(color='firebrick').encode(
241         y=yData,
242         size=alt.SizeValue(3))
243
244     #-- Concat 1: horizontal two figures
245     #charts = alt.hconcat(chart1, chart2, spacing=50
246     #    ).configure_title(color='green', fontSize=24
247     #    ).configure_axis(labelFontSize=14,
248     #        labelColor="red",
249     #        titleFontSize=20,
250     #        titleColor="blue",
251     #    )
252
253     #--Concat 2 : overlap
254     chart = alt.layer(bars, line, data=df).interactive()
255     chart.show()
```



Ex34 用 bar 呈現 從上面選擇區域呈現 bar

```
256 def ex34_Bar_Selection(df):
257     selectedBars = alt.selection(type="interval", encodings=["x"])
258
259     columns = df.columns
260     xData = columns[2] + ":N"
261     yData = "mean(" + columns[7] + "):Q"
262     bars = alt.Chart(df, title="Happy Planet Index").mark_bar().encode(
263         x=xData, # x=alt.X('x', axis=alt.Axis(format='%', title='Wellbeing'))
264         y=yData, # y=alt.Y('y', axis=alt.Axis(labels=False))
265         opacity=alt.condition(selectedBars, alt.OpacityValue(1), alt.OpacityValue(0.7)),
266         color="Region:N",
267         #color=alt.condition(selectedArea, "Region:N", alt.value("lightgray")),
268         #tooltip=["Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
269     ).properties(width=300, height=200
270     ).add_selection(selectedBars
271     ).interactive()
272
273
274     line = alt.Chart(df).mark_rule(color='firebrick').encode(
275         y=yData,
276         size=alt.SizeValue(3)
277     ).transform_filter(selectedBars)
278
279
280     chart = alt.layer(bars, line, data=df)
281     chart.show()
```

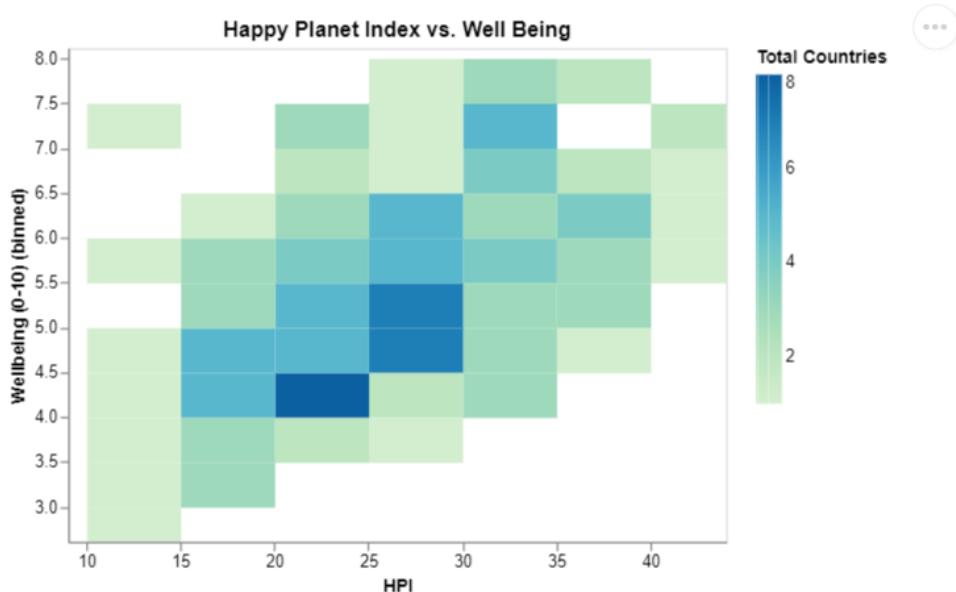
選擇區域數值較低 線也較往下



Ex35 heatmap

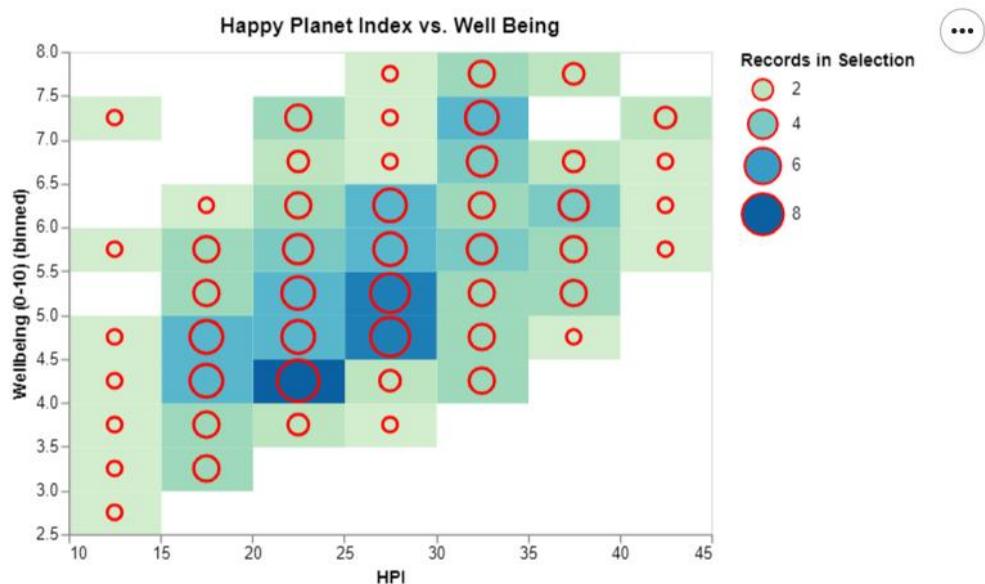
```
282
283 def ex35_Heatmap(df):
284     columns = df.columns
285     xData = columns[7] + ":Q"
286     yData = columns[4] + ":Q"
287     chart = alt.Chart(df, title="Happy Planet Index vs. Well Being").mark_rect().encode(
288         x=alt.X(xData, title="HPI", bin=True),
289         y=alt.Y(yData, bin=True),
290         color=alt.Color('count()', scale=alt.Scale(scheme="greenblue"),
291                         legend=alt.Legend(title="Total Countries")),
292         #color=alt.condition(selectedArea, "Region:N", alt.value("lightgray")),
293         #tooltip=["Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
294     ).properties(width=400, height=300
295     ).interactive()
296
297     chart.show()
298
```

可以看到熱度相關



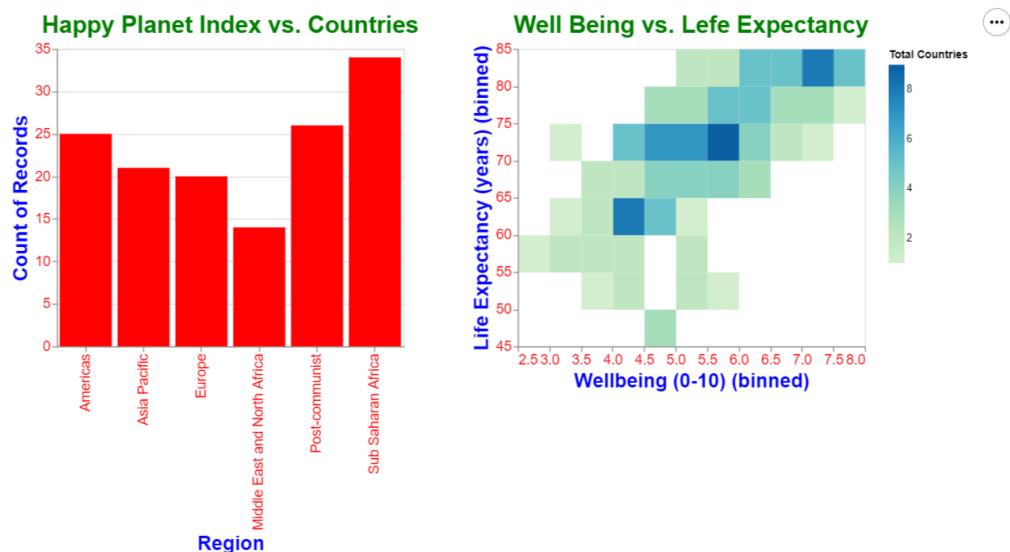
Ex35 熱度圖上面有圈圈代表

```
298 def ex35_Heatmap_Circle(df):
299     #-- chart 1
300     columns = df.columns
301     xData = columns[7] + ":Q"
302     yData = columns[4] + ":Q"
303
304     heatmap = alt.Chart(df, title="Happy Planet Index vs. Well Being").mark_rect().encode(
305         x=alt.X(xData, title="HPI", bin=True),
306         y=alt.Y(yData, bin=True),
307         color=alt.Color('count()', scale=alt.Scale(scheme="greenblue")),
308         #color=alt.condition(selectedArea, "Region:N", alt.value("lightgray")),
309         #tooltip=["Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
310     ).properties(width=400, height=300)
311
312     #-- chart2
313     circles = heatmap.mark_point().encode(
314         alt.ColorValue("red"),
315         alt.Size("count()", legend=alt.Legend(title="Records in Selection"))
316     ).interactive()
317
318     #-- overlap
319     chart = alt.layer(heatmap, circles)
320     chart.show()
321
```



Ex36 bar & heatmap

```
321 def ex36_Bar_Heatmap(df):
322     #-- chart 1
323     bars = alt.Chart(df, title="Happy Planet Index vs. Countries").mark_bar().encode(
324         x="Region:N",
325         y="count():Q",
326         color=alt.ColorValue("red")
327     ).properties(width=350, height=300).interactive()
328
329     #-- chart2
330     columns = df.columns
331     xData = columns[4] + ":Q"
332     yData = columns[3] + ":Q"
333     heatmap = alt.Chart(df, title="Well Being vs. Life Expectancy").mark_rect().encode(
334         x=alt.X(xData, bin=True),
335         y=alt.Y(yData, bin=True),
336         color=alt.Color('count()', scale=alt.Scale(scheme="greenblue"),
337                         legend=alt.Legend(title="Total Countries")),
338         #color=alt.condition(selectedArea, "Region:N", alt.value("lightgray")),
339         #tooltip=["Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
340     ).properties(width=350, height=300).interactive()
341
342     #-- overlap
343     chart = alt.hconcat(bars, heatmap, spacing=50, title="",
344                         ).configure_title(color='green', fontSize=24, align="center", anchor="middle"
345                         ).configure_axis(labelFontSize=14,
346                                         labelColor="red",
347                                         titleFontSize=20,
348                                         titleColor="blue",
349                                         )
350
351     chart.show()
```

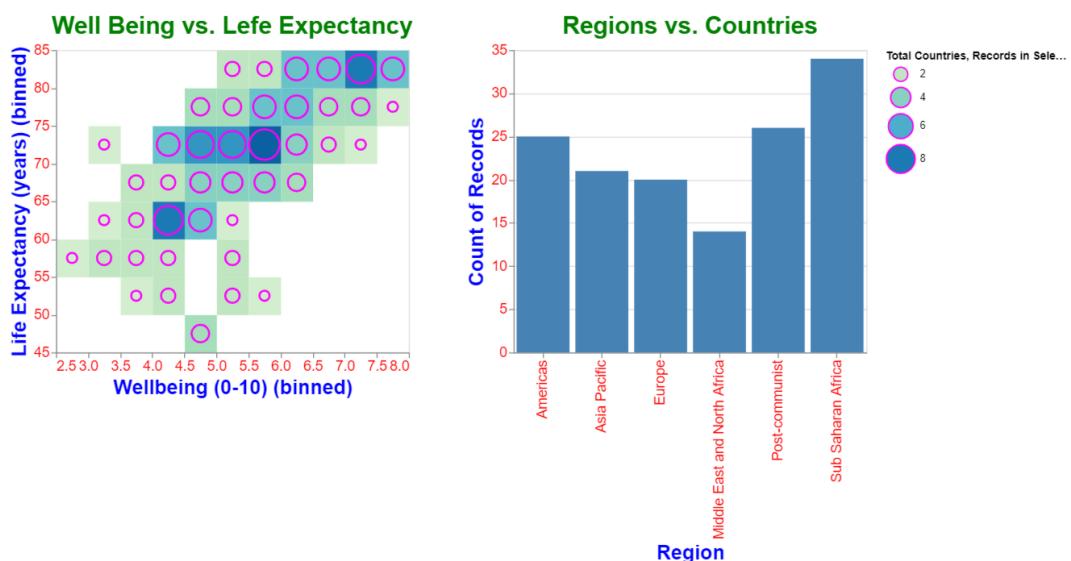


Ex37 bar heatmap bar 在右

```

352
353     def ex37_Bar_Link_Heatmap(df):
354         selectedRegion = alt.selection(type="interval", encodings=[ "x"])
355
356         #-- chart 1
357         bars = alt.Chart(df, title="Regions vs. Countries").mark_bar().encode(
358             x="Region:N",
359             y="count():Q",
360             color=alt.condition(selectedRegion,
361                                 alt.ColorValue("steelblue"), alt.ColorValue("grey"))
362         ).properties(width=350, height=300)
363         .add_selection(selectedRegion)
364         .interactive()
365
366         #-- chart 2
367         columns = df.columns
368         xData = columns[4] + ":Q"
369         yData = columns[3] + ":Q"
370         heatmap = alt.Chart(df, title="Well Being vs. Lefe Expectancy").mark_rect().encode(
371             x=alt.X(xData, bin=True),
372             y=alt.Y(yData, bin=True),
373             color=alt.Color('count()', scale=alt.Scale(scheme="greenblue"),
374                             legend=alt.Legend(title="Total Countries")),
375             #color=alt.condition(selectedArea, "Region:N", alt.value("lightgray")),
376             #tooltip=["Country", "Region", "Wellbeing(0~10):Q", "Happy Planet Index", "Life Expectancy(years):Q"]
377         ).properties(width=350, height=300)
378
379         circles = heatmap.mark_point().encode(
380             alt.ColorValue("magenta"),
381             alt.Size("count()", legend=alt.Legend(title="Records in Selection"))
382         ).interactive()
383
384         heatmapCircle = alt.layer(heatmap, circles)
385
386         #--overlap
387         chart = alt.hconcat(heatmapCircle, bars, spacing=50, title="",
388             ).configure_title(color='green', fontSize=24, align="center", anchor="middle"
389             ).configure_axis(labelFontSize=14,
390                             labelColor="red",
391                             titleFontSize=20,
392                             titleColor="blue",
393             )
394
395         chart.show()

```

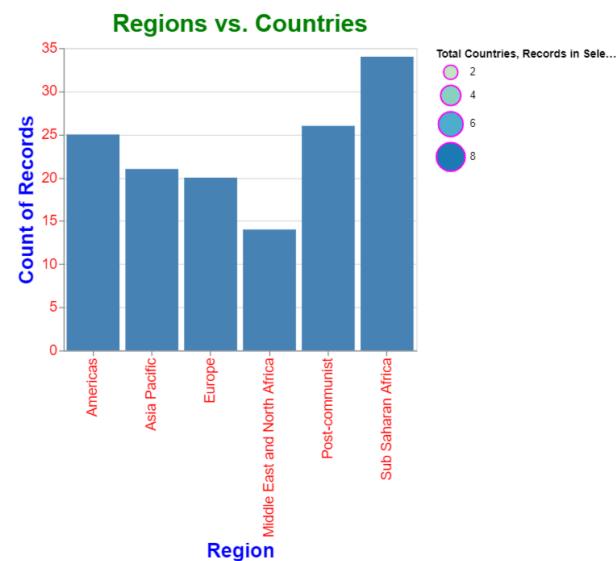
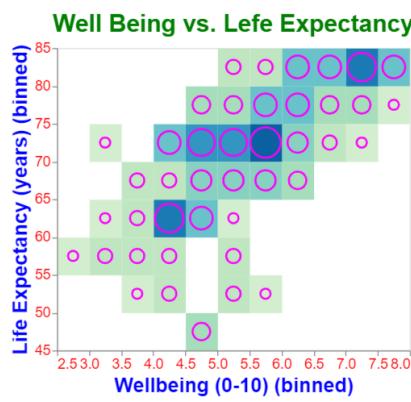


L4_Activity Google Play Store Rating

右邊是年紀評價 app 數量的 bar

左邊是幾個 app 與評價的熱度圖

```
34 if (__name__ == "__main__"):
35     df = pd.read_csv(InputPath + "googleplaystore.csv")
36
37     gps_apps_df = df.dropna()
38
39     selectedCategory = alt.selection(type="single", encodings=['x'])
40
41     alt.data_transformers.enable('default', max_rows=None)
42     bars = alt.Chart(gps_apps_df, title="Content Rating").mark_bar().encode(
43         x='Content Rating:N',
44         y='count():Q',
45         color=alt.condition(selectedCategory, alt.ColorValue("steelblue"), alt.ColorValue("grey"))
46     ).properties(width=100, height=100)
47     .add_selection(selectedCategory)
48
49     #-- Heatmap indicating number of apps across app Category and Rating ranges.
50     heatmap = alt.Chart(gps_apps_df, title="Rating of Category").mark_rect().encode(
51         alt.X('Category:N'),
52         alt.Y('Rating:Q', bin=True),
53         alt.Color('count()'),
54         scale=alt.Scale(scheme='greenblue'),
55         legend=alt.Legend(title='Total Apps')
56     )
57     .properties(width=400, height=100)
58
59     #-- Circle stick on heatmap
60     circles = heatmap.mark_point().encode(
61         alt.ColorValue("magenta"),
62         alt.Size('count()', scale=alt.Scale(domain=(1, 600), range=(1, 200)),
63         legend=alt.Legend(title='Apps in Selection'))
64     ).transform_filter(selectedCategory)
65
66     heatmapCircles = alt.layer(heatmap, circles)
67
68     chart = alt.hconcat(heatmapCircles, bars, spacing=50, title="Google Play Store Apps Rating"
69                         ).configure_title(color='green', font_size=22, align="center", anchor="middle"
70                         ).configure_axis(label_font_size=14,
71                             label_color="red",
72                             title_font_size=20,
73                             title_color="blue",
74                         )
75
76     chart.show()
```

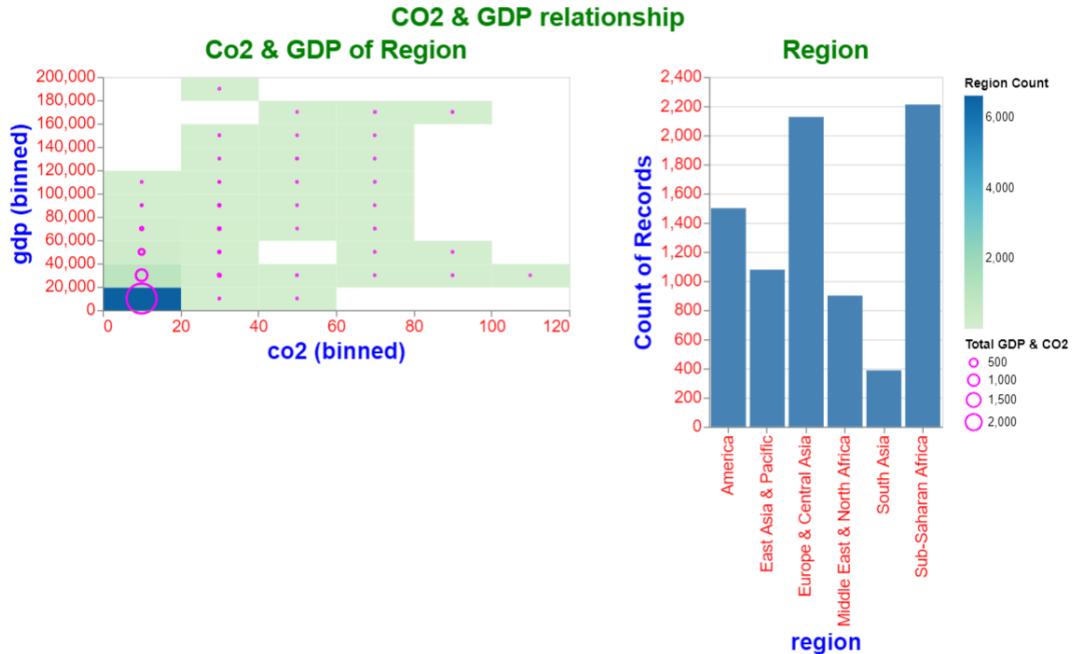


HW Co2 & gdp

紅色框為與 Google app 不同之處

```
Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
E:\Data_Visualization\ClassPan_Lib_Python
L5_IDV_Activity_A108222040.py APP_IDV_Weather.py L4_HW_Olympics_A108222040.py L4_HW_Co2Gdp_A108222040.py

33
34 if __name__ == "__main__":
35     df = l4_dataset(InputPath + "/new_co2_gdp.csv")
36
37     gdp_aco2_df = df.dropna()
38
39     selectedRegion = alt.selection(type="single", encodings=['x'])
40
41     alt.data_transformers.enable('default', max_rows=None)
42     bars = alt.Chart(gdp_aco2_df, title="Region").mark_bar().encode(
43         x="region:N",
44         y="count():Q",
45         color=alt.condition(selectedRegion, alt.ColorValue("steelblue"), alt.ColorValue("grey"))
46     ).properties(width=200
47     ).add_selection(selectedRegion)
48
49     #-- Heatmap indicating number of apps across app category and Rating ranges.
50     heatmap = alt.Chart(gdp_aco2_df, title="Co2 & GDP of Region").mark_rect().encode(
51         alt.X('co2', bin=True),
52         alt.Y('gdp', bin=True),
53         alt.Color('count()',
54             scale=alt.Scale(scheme='greenblue'),
55             legend=alt.Legend(title="Region Count")
56         )
57     ).properties(width=400, height=200)
58
59     #-- Circle stick on heatmap
60     circles = heatmap.mark_point().encode(
61         alt.ColorValue('magenta'),
62         alt.Size('count()', scale=alt.Scale(domain=(1, 2000), range=(1, 2000)),
63         legend=alt.Legend(title="Total GDP & CO2")
64     ).transform_filter(selectedRegion)
65
66     heatmapCircles = alt.layer(heatmap, circles)
67
68     chart = alt.hconcat(heatmapCircles, bars, spacing=50, title="CO2 & GDP relationship"
69         ).configure_title(color="green", fontSize=22, align="center", anchor="middle"
70         ).configure_axis(labelFontSize=14,
71                         labelColor="red",
72                         titleFontSize=20,
73                         titleColor="blue",
74                     )
75
chart.show()
```



HW Olympics

紅色框為與 Google app 不同之處

Spyder (Python 3.8)

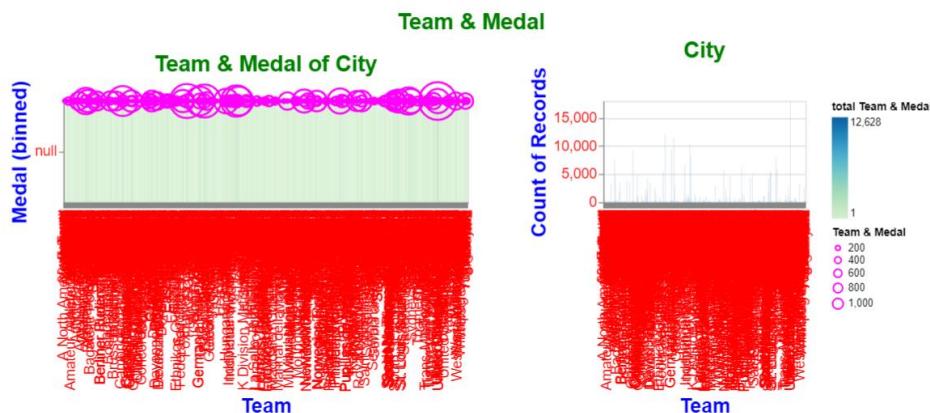
File Edit Search Source Run Debug Consoles Projects Tools View Help

E:\Data_Visualization\Class\Pan_Lib_Python\

L5_IDV_Activity_A108222040.py APP_IDV_Weather.py L4_HW_Olympics_A108222040.py

```
33
34 if (__name__ == "__main__"):
35     df = 14_dataset(InputPath + "/athlete_events.csv")
36
37     medal_df = df.dropna(how='all')
38
39     selectedCategory = alt.selection(type="single", encodings=[ 'x'])
40
41     alt.data_transformers.enable('default', max_rows=None)
42     bars = alt.Chart(medal_df, title="City").mark_bar().encode(
43         x=[Team:N],
44         y= count():Q,
45         color=alt.condition(selectedCategory, alt.ColorValue("steelblue"), alt.ColorValue("grey"))
46     ).properties(width=200, height=100
47     ).add_selection(selectedCategory)
48
49     #-- Heatmap indicating number of apps across app Category and Rating ranges.
50     heatmap = alt.Chart(medal_df, title="Team & Medal of City").mark_rect().encode(
51         alt.X([Team:N]),
52         alt.Y([Medal], bin=True),
53         alt.Color('count()',
54             scale=alt.Scale(scheme='greenblue'),
55             legend=alt.Legend(title='total Team & Medal')
56         )
57     ).properties(width=400, height=100)
58
59     #-- Circle stick on heatmap
60     circles = heatmap.mark_point().encode(
61         alt.ColorValue('magenta'),
62         alt.Size('count()', scale=alt.Scale(domain=(1, 1000), range=(1, 110)),
63         legend=alt.Legend(title='Team & Medal'))
64     ).transform_filter(selectedCategory)
65
66     heatmapCircles = alt.layer(heatmap, circles)
67
68     chart = alt.hconcat(heatmapCircles, bars, spacing=50, title="Team & Medal"
69         ).configure_title(color="green", fontSize=22, align="center", anchor="middle"
70         ).configure_axis(labelFontSize=14,
71             labelColor="red",
72             titleFontSize=20,
73             titleColor="blue",
74         )
75
chart.show()
```

很多國家導致都聚在一起



Weather Week16

Dataset -> weather

The screenshot shows the Spyder Python IDE interface. On the left, the code editor displays a Python script named `LS_IDV_Activity_A109222040.py`. The script imports various modules like `os`, `pandas`, and `altair`, and performs data loading and analysis. A red box highlights the data loading section of the code. On the right, the variable explorer shows a table for the `df` DataFrame. The table includes columns for `Name`, `Type`, `Count`, and `Dtype`. It lists 7 columns: `location` (object), `date` (float64), `precipitation` (float64), `temp_max` (float64), `temp_min` (float64), `wind` (float64), and `weather` (object). Below the table, numerical and categorical data info sections provide summary statistics.

Name	Type	Count	Dtype
location	object	2922	non-null
date	float64	2922	non-null
precipitation	float64	2922	non-null
temp_max	float64	2922	non-null
temp_min	float64	2922	non-null
wind	float64	2922	non-null
weather	object	2922	non-null

Numerical Data info. -->

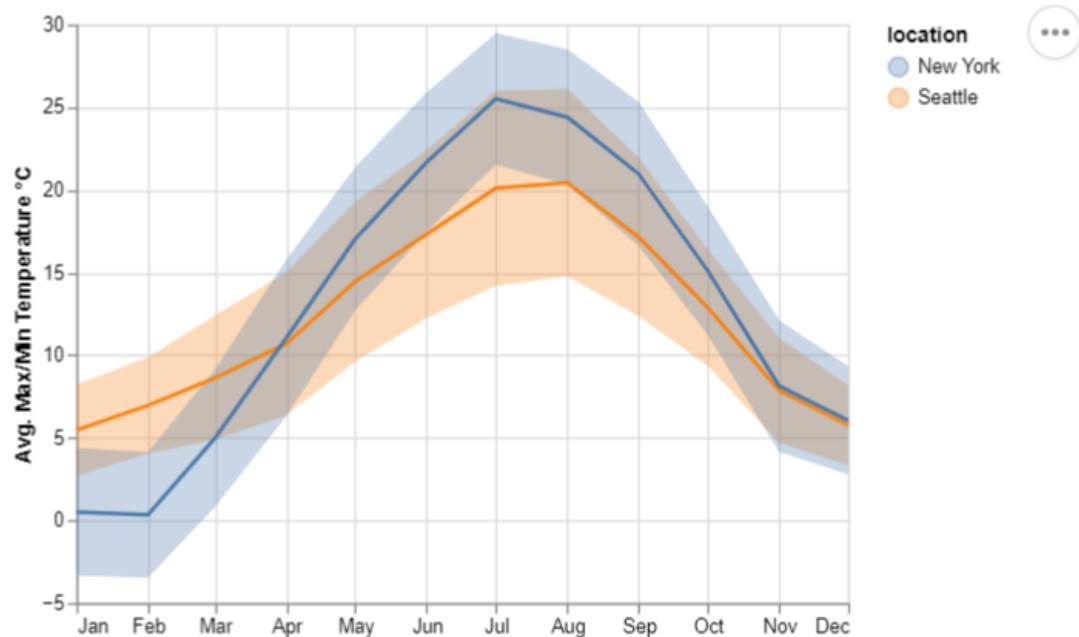
	precipitation	temp_max	temp_min	wind
count	2922.000000	2922.000000	2922.000000	2922.000000
mean	2.944764	16.769131	8.612320	4.101129
std	7.695286	8.644596	7.511776	1.888791
min	0.000000	-7.700000	-16.000000	0.400000
25%	0.000000	10.700000	3.300000	2.700000
50%	0.000000	15.100000	8.700000	5.100000
75%	1.800000	23.900000	13.900000	5.100000
max	118.900000	37.800000	26.700000	16.200000

Categorical Data info. -->

	location	date	weather
count	2922	2922	2922
unique	2	1461	5
top	Seattle	2012-01-01	sun
freq	1461	2	1466

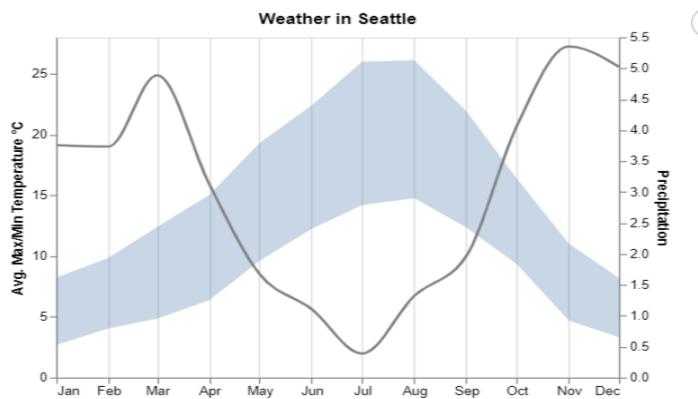
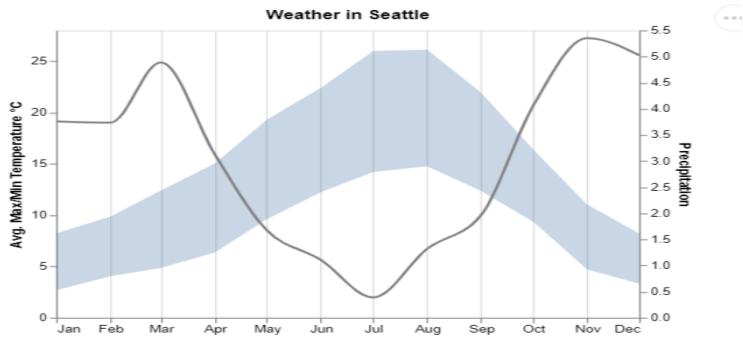
Case1 New York Seattle 平均高低溫

```
51
52  ##---Case1
53  def case1_transformC(df):
54      #-- left chart
55      tempMinMax = alt.Chart(df).mark_area(opacity=0.3).encode(
56          alt.X('month(date):T', title=None, axis=alt.Axis(format='%b')),
57          alt.Y('average(temp_max):Q', title='Avg. Max/Min Temperature °C'),
58          alt.Y2('average(temp_min):Q'),
59          alt.Color('location:N'))
60
61      #--right chart
62      tempMid = alt.Chart(df).mark_line().transform_calculate(
63          temp_mid='(+datum.temp_min + +datum.temp_max)/2'
64      ).encode(alt.X('month(date):T'),
65              alt.Y('average(temp_mid):Q'),
66              alt.Color('location:N'))
67
68      #-- tempMinMax + tempMid
69      chart = alt.layer(tempMinMax, tempMid)
70      charts = alt.hconcat(tempMinMax, tempMid, spacing=50, title="Weather", center=True,
71          ).configure_title(color='green', fontSize=24, anchor="middle", align="right",
72              ).configure_axis(labelFontSize=14,
73                  labelColor="red",
74                  titleFontSize=20,
75                  titleColor="blue",
76              )
77      chart.show()
```



Case2 Seattle

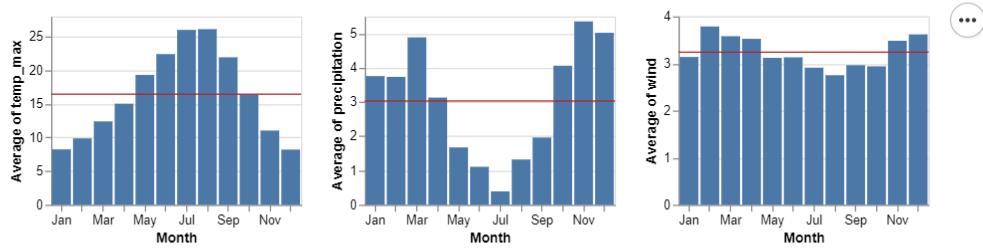
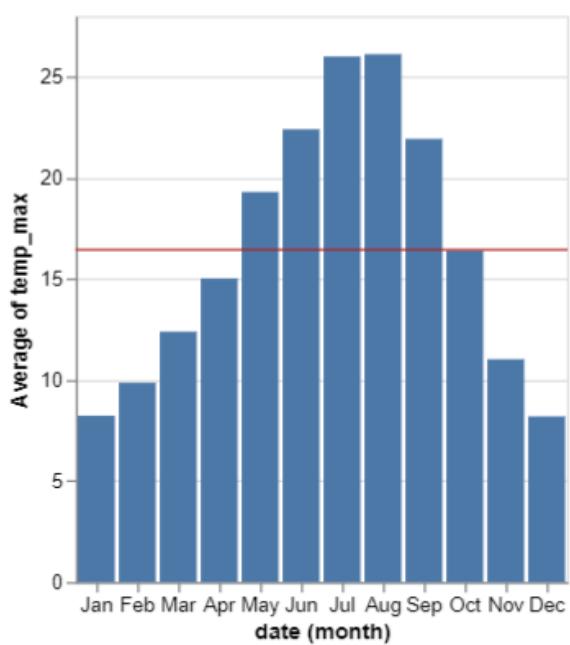
```
57     chart.show()
58
59 def case2_transformF(df):
60     #--method 1
61     precip = alt.Chart(df).transform_filter('datum.location == "Seattle"')
62         .mark_line(interpolate='monotone', stroke='grey')
63         .encode(alt.X('month(date):T', title=None, axis=alt.Axis(format='%b')),
64                 alt.Y('average(precipitation):Q', title='Precipitation'))
65
66 tempMinMax = alt.Chart(df).transform_filter('datum.location == "Seattle"')
67     .mark_area(opacity=0.3).encode(
68         alt.X('month(date):T', title=None, axis=alt.Axis(format='%b')),
69         alt.Y('average(temp_max):Q', title='Avg. Max/Min Temperature °C'),
70         alt.Y2('average(temp_min):Q'))
71
72 chart = alt.layer(tempMinMax, precip, title="Weather in Seattle").resolve_scale(y='independent')
73 chart.show()
74
75     #--method 2
76     precip = alt.Chart(df).mark_line(interpolate='monotone', stroke='grey')
77         .encode(alt.X('month(date):T', title=None),
78                 alt.Y('average(precipitation):Q', title='Precipitation'))
79
80 tempMinMax = alt.Chart().mark_area(opacity=0.3).encode(
81         alt.X('month(date):T', title=None, axis=alt.Axis(format='%b')),
82         alt.Y('average(temp_max):Q', title='Avg. Max/Min Temperature °C'),
83         alt.Y2('average(temp_min):Q'))
84
85 chart = alt.layer(tempMinMax, precip, data=df, title="Weather in Seattle"
86                     ).transform_filter('datum.location == "Seattle"')
87                     .resolve_scale(y='independent')
88 chart.show()
89
```



Case6 綜合

```
 93 #Q1
 94 def case6_composition(df):
 95     #-- method 1
 96     basic1 = alt.Chart(df).transform_filter('datum.location == "Seattle"')
 97         .mark_bar(
 98             ).encode(alt.X('month(date):O'), alt.Y('average(temp_max):Q'))
 99     basic2 = alt.Chart(df).transform_filter('datum.location == "Seattle"')
100         .mark_rule(stroke='firebrick')
101         .encode(alt.Y('average(temp_max):Q'))
102
103     chart = alt.layer(basic1, basic2)
104     chart.show()
105
106     #--method 2
107     chart = alt.layer(alt.Chart().mark_bar()).encode(
108         alt.X('month(date):O', title='Month'),
109             alt.Y(alt.repeat('column'), aggregate='average', type='quantitative'),
110             alt.Chart().mark_rule(stroke='firebrick').encode(
111                 alt.Y(alt.repeat('column'), aggregate='average', type='quantitative'))
112                 .properties(width=200, height=150)
113                 .repeat(data=df, column=['temp_max', 'precipitation', 'wind'])
114                 .transform_filter('datum.location == "Seattle"')
115     chart.show()
116
117     #--method 3
118     spлом = alt.Chart().mark_point(filled=True, size=15, opacity=0.5
119         ).encode(alt.X(alt.repeat('column'), type='quantitative'),
120             alt.Y(alt.repeat('row'), type='quantitative')
121             .properties(width=125, height=125
122             .repeat(row=['temp_max', 'precipitation', 'wind'],
123                 column=['wind', 'precipitation', 'temp_max']
124             )
125
126     dateHist = alt.layer(alt.Chart(df).mark_bar()).encode(
127         alt.X('month(date):O', title='Month'),
128             alt.Y(alt.repeat('row'), aggregate='average', type='quantitative'),
129             #alt.Color('month(date):O', scale = alt.Scale(range=[ '#aec7e8', 'c7c7c7', '#1f77b4', '#9467bd',
130             alt.Chart().mark_rule(stroke='firebrick').encode(
131                 alt.Y(alt.repeat('row'), aggregate='average', type='quantitative'))
132                 .properties(width=175, height=125
133                 .repeat(row=['temp_max', 'precipitation', 'wind'])
134
135     tempHist = alt.Chart(df).mark_bar(
136         ).encode(alt.X('temp_max:Q', bin=True, title='Temperature (°C)'),
137             alt.Y('count():Q'),
138             alt.Color('weather:N', scale=alt.Scale(
139                 domain=['drizzle', 'fog', 'rain', 'snow', 'sun'],
140                 range=['blue', 'black', 'yellow', 'pink', 'red']))
141             .properties(width=115, height=100
142             ).facet(column="weather:N")
143
144     chart = alt.vconcat(alt.hconcat(spлом, dateHist),
145         tempHist, data=df, title='Seattle Weather Dashboard', center=True
146         ).transform_filter('datum.location == "Seattle"')
147         .resolve_legend(color='independent'
148         ).configure_axis(
149             labelAngle=0,
150             labelFontSize=10,
151             labelColor="orange",
152             titleFontSize=10,
153             titleColor="green",)
154
155     chart.show()
156
157 if __name__ == "__main__":
158     df = loadData()
159     #case1_transformC(df)
160     #case2_transformB(df)
161     case6_composition(df)
```

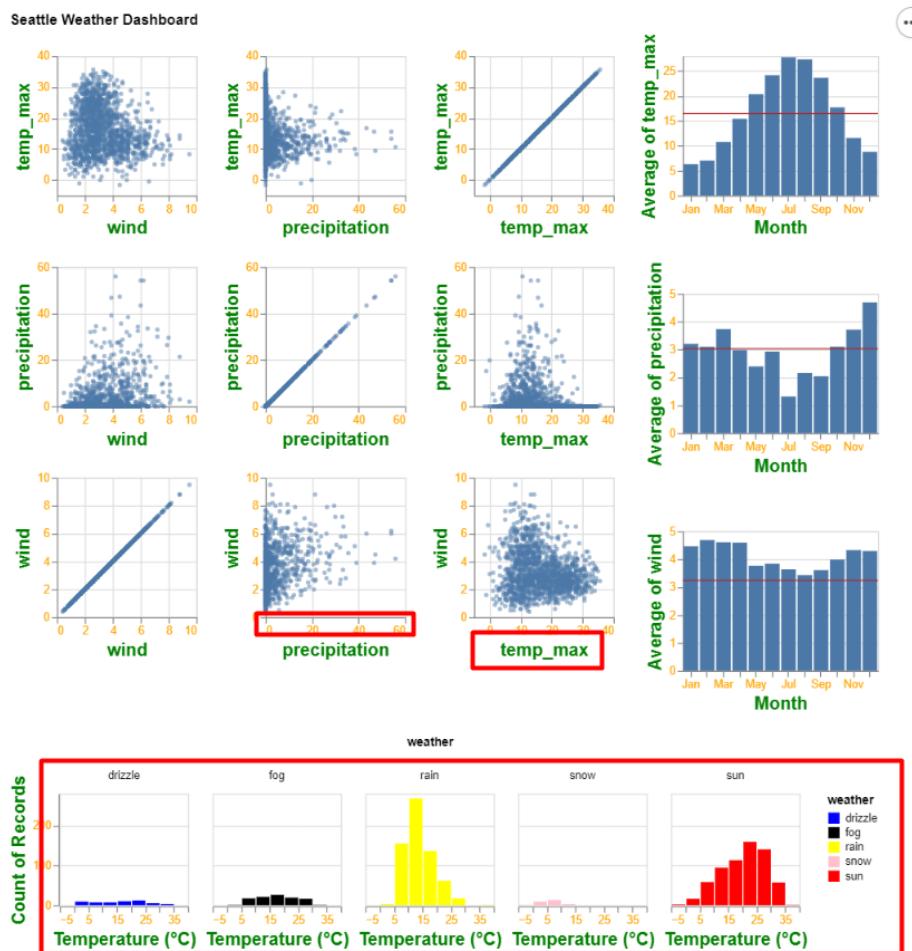
```
 120         alt.Y(alt.repeat('row'), type='quantitative')
121             .properties(width=125, height=125
122             .repeat(row=['temp_max', 'precipitation', 'wind'],
123                 column=['wind', 'precipitation', 'temp_max']
124             )
125
126     dateHist = alt.layer(alt.Chart(df).mark_bar()).encode(
127         alt.X('month(date):O', title='Month'),
128             alt.Y(alt.repeat('row'), aggregate='average', type='quantitative'),
129             #alt.Color('month(date):O', scale = alt.Scale(range=[ '#aec7e8', 'c7c7c7', '#1f77b4', '#9467bd',
130             alt.Chart().mark_rule(stroke='firebrick').encode(
131                 alt.Y(alt.repeat('row'), aggregate='average', type='quantitative'))
132                 .properties(width=175, height=125
133                 .repeat(row=['temp_max', 'precipitation', 'wind'])
134
135     tempHist = alt.Chart(df).mark_bar(
136         ).encode(alt.X('temp_max:Q', bin=True, title='Temperature (°C)'),
137             alt.Y('count():Q'),
138             alt.Color('weather:N', scale=alt.Scale(
139                 domain=['drizzle', 'fog', 'rain', 'snow', 'sun'],
140                 range=['blue', 'black', 'yellow', 'pink', 'red']))
141             .properties(width=115, height=100
142             ).facet(column="weather:N")
143
144     chart = alt.vconcat(alt.hconcat(spлом, dateHist),
145         tempHist, data=df, title='Seattle Weather Dashboard', center=True
146         ).transform_filter('datum.location == "Seattle"')
147         .resolve_legend(color='independent'
148         ).configure_axis(
149             labelAngle=0,
150             labelFontSize=10,
151             labelColor="orange",
152             titleFontSize=10,
153             titleColor="green",)
154
155     chart.show()
156
157 if __name__ == "__main__":
158     df = loadData()
159     #case1_transformC(df)
160     #case2_transformB(df)
161     case6_composition(df)
```



有改變 x y 軸的字顏色

刻度也有改變

最下面每個圖的 bar 都有改變



Stock

```
└─ L5_IDV_Activity_A108222040.py X
  37 def getStocks():
  38     #-- get data
  39     startDate = "2020-1-1"
  40     endDate = "2022-6-6"
  41     source = "yahoo"
  42     #-- get data
  43     appleStock = pdData.DataReader("AAPL", start=startDate, end=endDate, data_source=source
  44                                     ).reset_index()[["Date", "Close", "High"]]
  45     ibmStock = pdData.DataReader("IBM", start=startDate, end=endDate, data_source=source
  46                                     ).reset_index()[["Date", "Close", "High"]]
  47     googleStock = pdData.DataReader("GOOG", start=startDate, end=endDate, data_source=source
  48                                     ).reset_index()[["Date", "Close", "High"]]
  49     microsoftStock = pdData.DataReader("MSFT", start=startDate, end=endDate, data_source=source
  50                                     ).reset_index()[["Date", "Close", "High"]]
  51
  52     #--save data
  53     appleStock.to_csv(InputPath + "/AppleStock_20200101_20220606.csv")
  54     ibmStock.to_csv(InputPath + "/IbmStock_20200101_20220606.csv")
  55     googleStock.to_csv(InputPath + "/GoogleStock_20200101_20220606.csv")
  56     microsoftStock.to_csv(InputPath + "/MicrosfotStock_20200101_20220606.csv")
  57
  58     #--process data
  59     appleStock["Stock"] = "Apple"
  60     ibmStock["Stock"] = "Ibm"
  61     googleStock["Stock"] = "Google"
  62     microsoftStock["Stock"] = "Microsoft"
  63
  64     stocks = pd.concat([appleStock, ibmStock, googleStock, microsoftStock])
  65     stocks[ "Month" ] = stocks.Date.dt.month
  66     stocks[ "Year" ] = stocks.Date.dt.year
  67     stocks[ "Day" ] = stocks.Date.dt.day
  68     stocks.to_csv(InputPath + "/stocks_20200101_20220606.csv")
  69
  70     return stocks
  71
```

```
 70     return stocks
 71
 72 def IDV_Altair_Stocks(stocks):
 73     stockSelection1 = alt.selection_single(fields=[ "Stock" ], bind="legend")
 74
 75     #--rightChart
 76     rightChart = alt.Chart(stocks).mark_line().encode(
 77         x="Date", y="Close", color="Stock",
 78         opacity=alt.condition(stockSelection1, alt.value(1), alt.value(0.1)),
 79         tooltip=[ "Date", "Close" ] #hover
 80     ).properties(height=300, width=500
 81     ).configure_title(color="green", fontSize=24
 82     ).configure_axis(labelFontSize=14,
 83                     labelColor="brown",
 84                     titleFontSize=20,
 85                     titleColor="purple"
 86     ).add_selection(stockSelection1
 87     ).interactive()
 88
 89     rightChart.show()
 90
 91
 92     #-- left chart
 93     stockSelection1 = alt.selection_single(fields=[ "Stock" ], bind="legend")
 94
 95     rightChart = alt.Chart(stocks).mark_line().encode(
 96         x="Date", y="Close", color="Stock",
 97         opacity=alt.condition(stockSelection1, alt.value(1), alt.value(0.1)),
 98         tooltip=[ "Date", "Close" ] #hover
 99     ).properties(height=300, width=500
100     ).add_selection(stockSelection1
101     ).interactive()
102
103     upper = rightChart.encode(alternateX('Date:T', scale=alt.Scale(domain=stockSelection1)))
104
105     lower = rightChart.properties(height=60).add_selection(stockSelection1)
106
107     leftChart = upper & lower
108     leftChart.show()
109
```



下面與上面一樣 可以運動



只抓 microsoft 的股票

```
#-- left chart
brush = alt.selection(type='interval', encodings=[ 'x'])

base = alt.Chart(stocks.loc[stocks["Stock"] == "Microsoft"]
                  .mark_line().encode(x = 'Date:T', y = 'Close:Q'
                  ).properties(width=600, height=200)

upper = base.encode(alt.X('Date:T', scale=alt.Scale(domain=brush)))

lower = base.properties(height=60).add_selection(brush)

leftChart = upper & lower
leftChart
```

