淡江大學統計學系
數據科學組碩一

第一次作業

613890176 魏祺紘
指導教授:高君豪教授

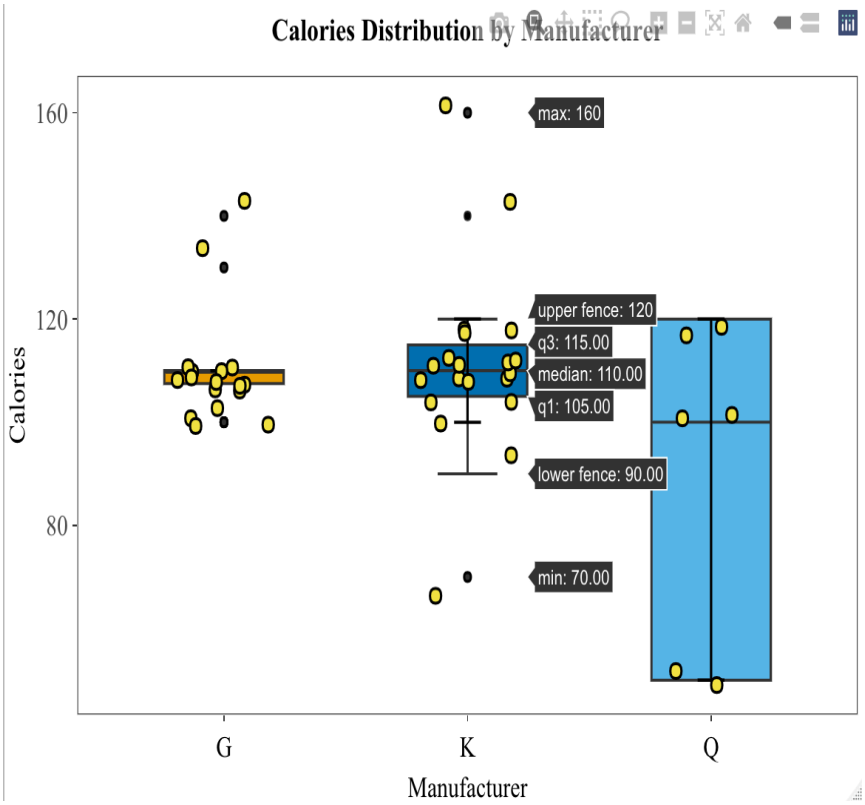
繳交時間:9/28

使用程式 : R、Python
互動式圖表連結 : R、Python

目錄

S1. 每個變數的箱型圖

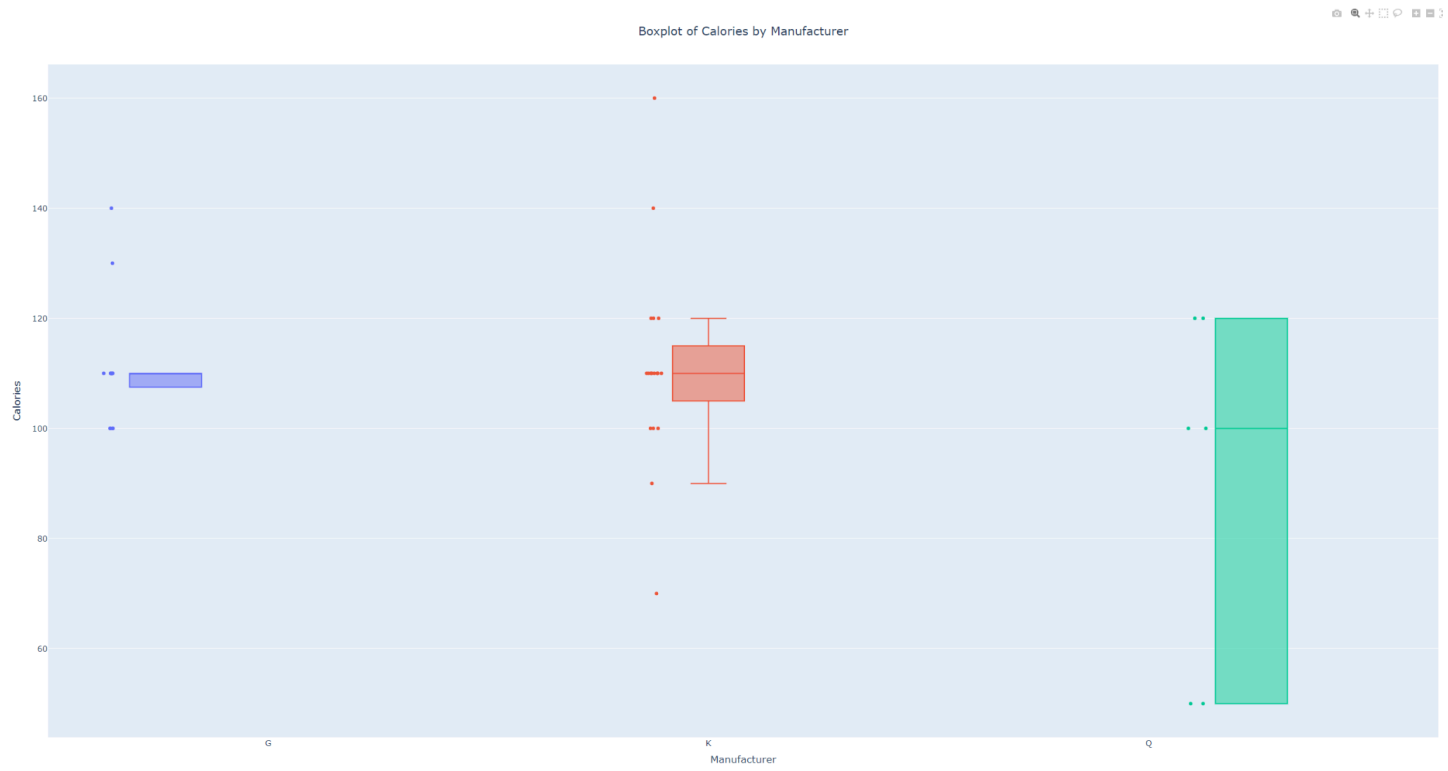
- Calories



此圖中可看出G廠商的卡路里是非常集中的變異性不大，但還是有些許的outlier在圖的上下部分。K廠商是稍微左偏分佈，其中的變異性也不高但也是有出現些許的outlier。Q廠商可看出也為左偏分佈，但也是因為有2個特別低的outlier導致圖被拉的特別長，變異較大。

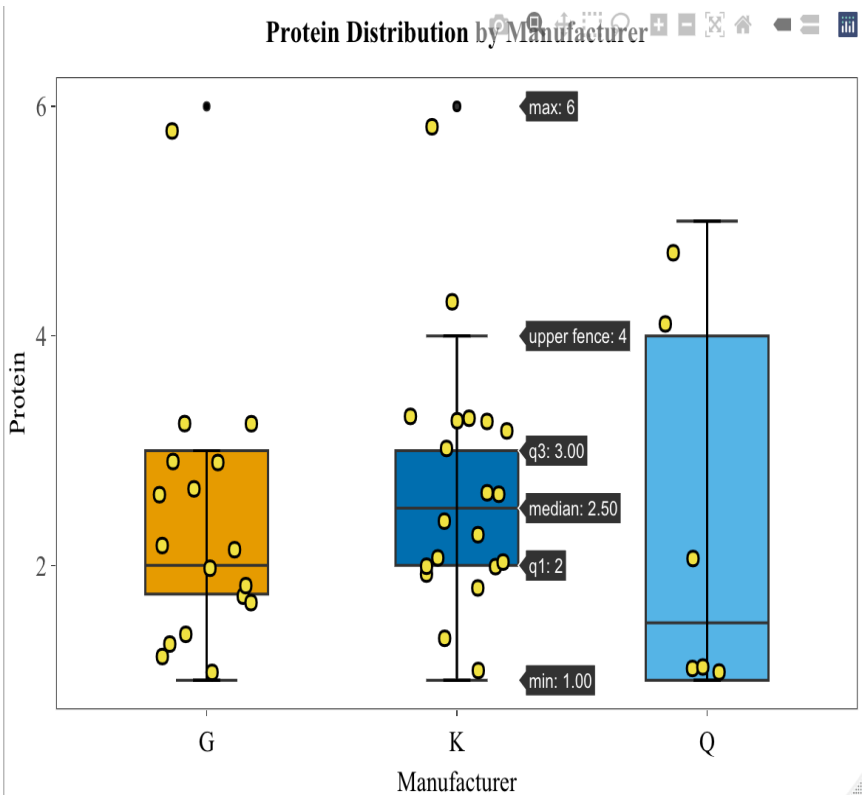
G 和 K 製造商的平均值相似（111 kcal），而 Q 較低（90 kcal）但變異較大。

圖一. 三個廠商卡路里含量箱型圖(R程式)



圖二. 三個廠商卡路里含量箱型圖(Python程式)

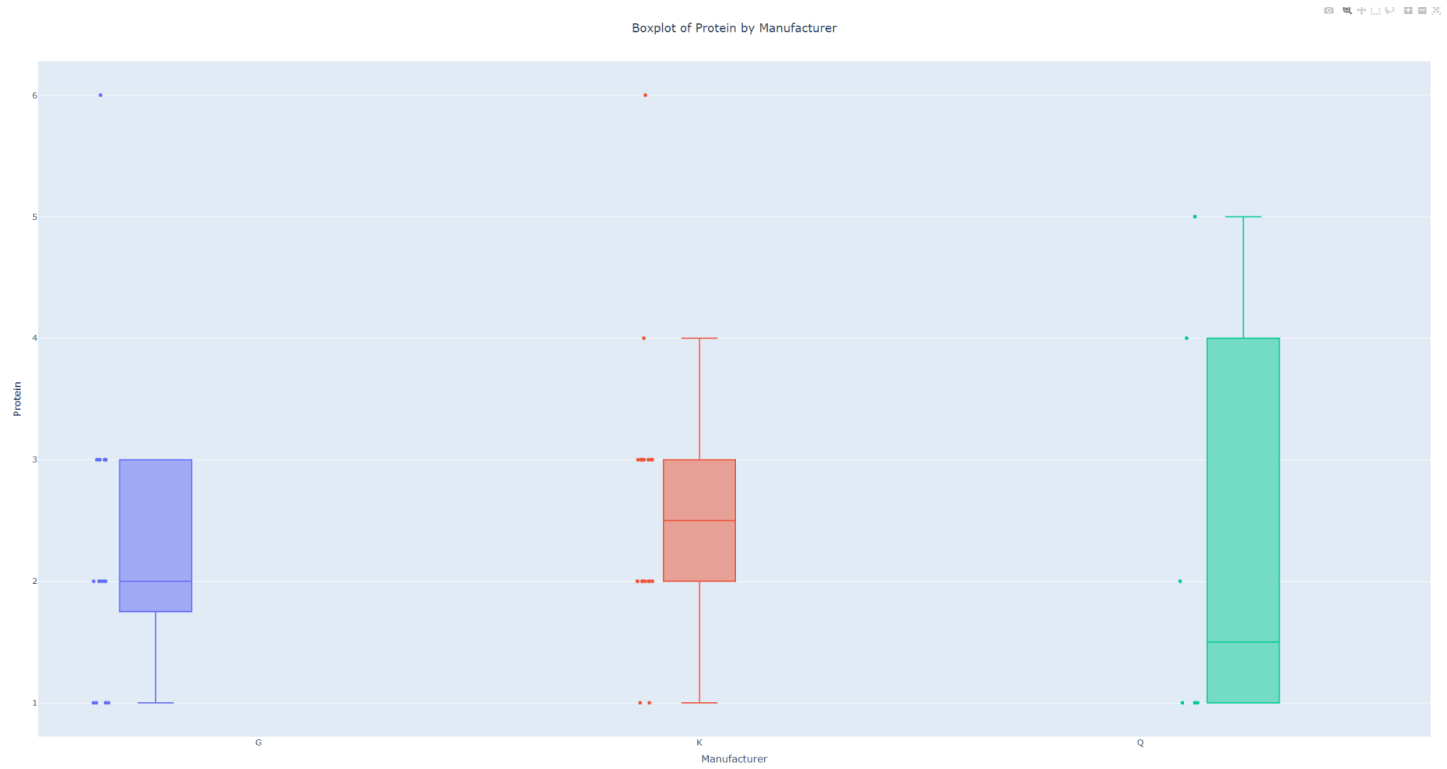
- Protein



此圖中可看出G廠商是一個右偏分佈，無特別變異性也是有出現些許的outlier。K廠商接近是鐘形分佈，其中的變異性也不高但也是有出現些許的outlier。Q廠商可看出也為左偏分佈，但可能因為樣本數少的關係，看出其箱型圖變異性非常大。

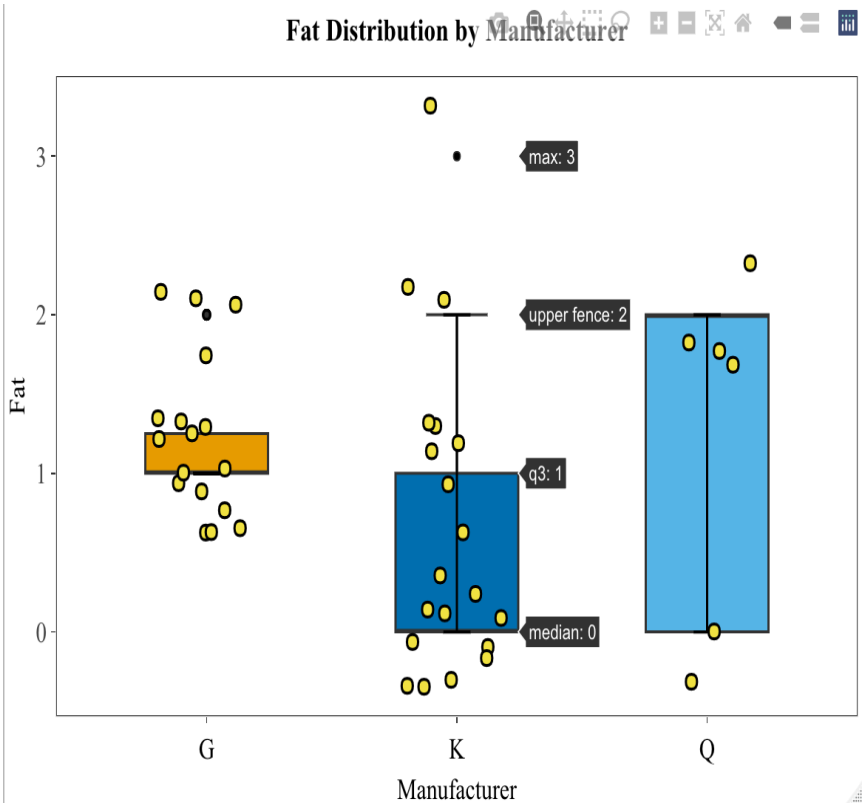
可以看到3個廠商的中位數(G:2 , K:2.5 , Q:1.5)三家製造商的平均值都在 2-3g 範圍內，可以推測各廠商在蛋白質營養物的添加上是差距不大的。

圖三. 三個廠商蛋白質含量箱型圖(R程式)



圖四. 三個廠商卡路里含量箱型圖(Python程式)

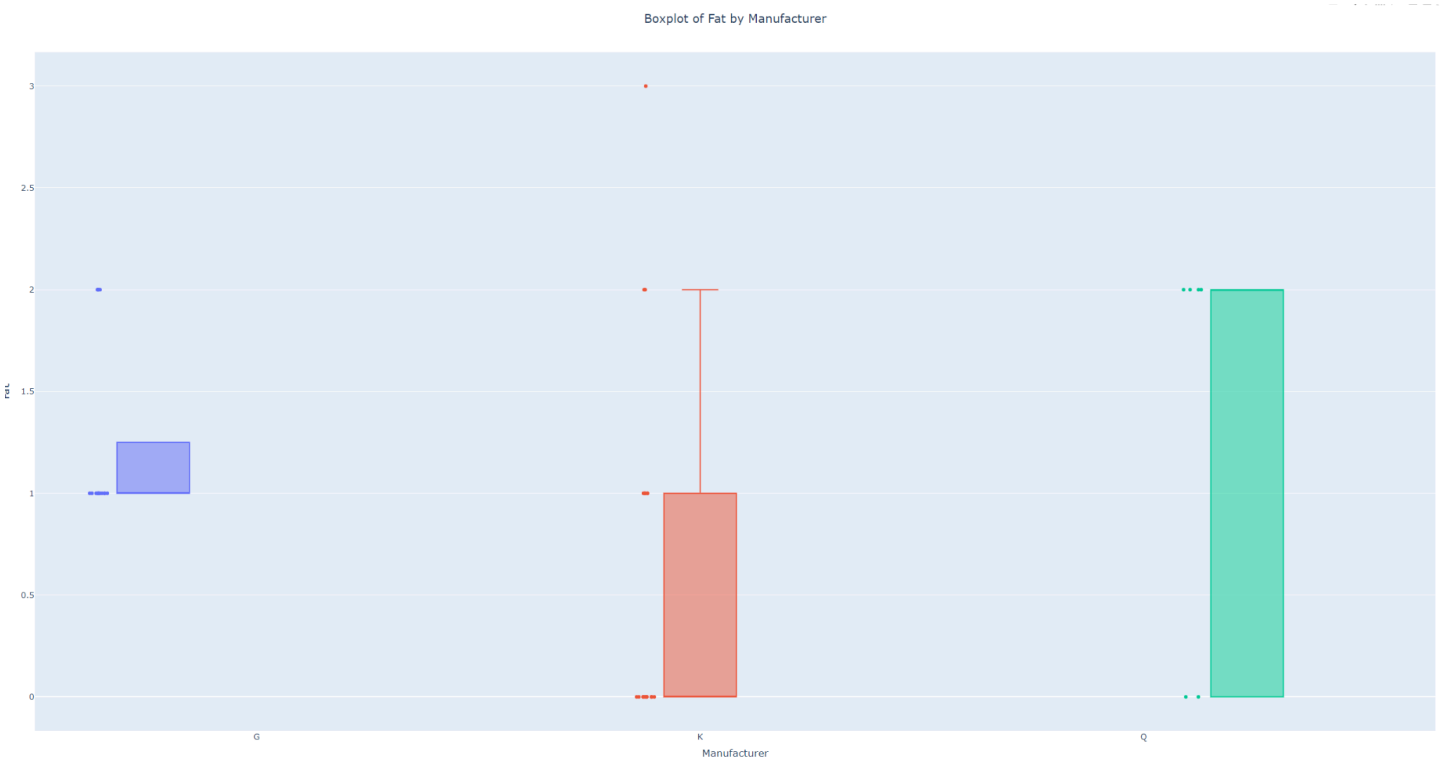
- Fat



此圖中可看出G廠商的脂肪含量是非常集中的變異性不大，但還是有些許的outlier在圖的上下部分。K廠商是右偏分佈，其中大多數的資料都集中在0、1，而變異性也相對較少但也是有出現些許的outlier。Q廠商可看出為左偏分佈，但因為大部分料都集中在0、2所以導致圖為長方形。

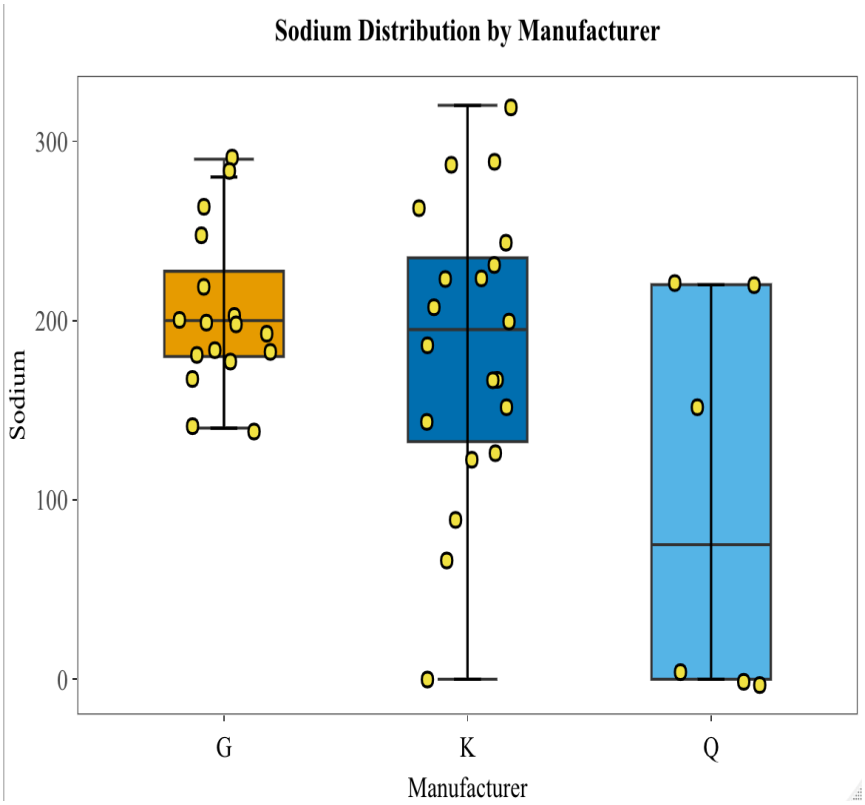
可以看到3個廠商的K 製造商的平均值最低（0.65g），G 和 Q 較高（分別為 1.24g 和 1.33g），這表明不同製造商的穀物在脂肪含量上可能存在實質性差異。

圖五. 三個廠商脂肪含量箱型圖(R程式)



圖六. 三個廠商卡路里含量箱型圖(Python程式)

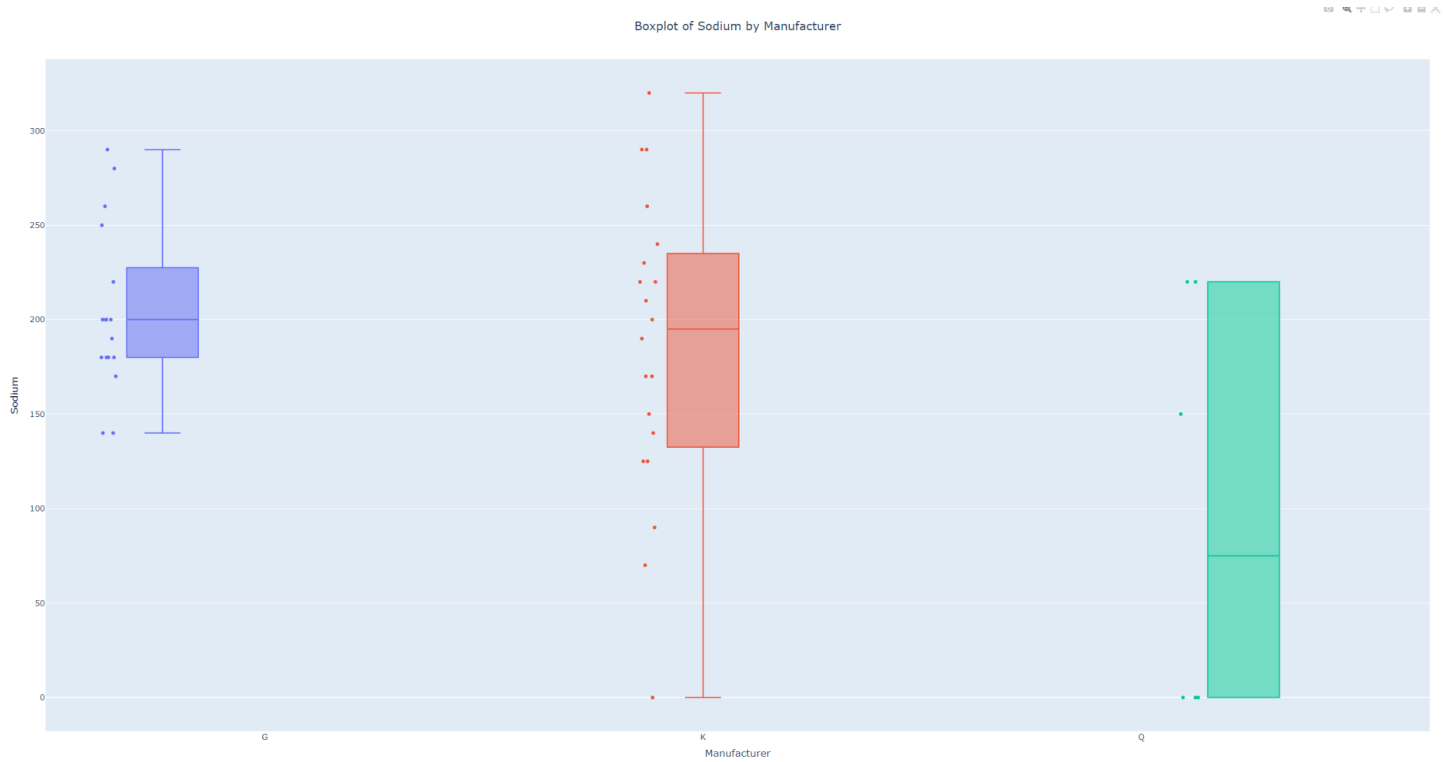
- Sodium



此圖中可看出G廠商的鈉含量是已經有點接近鐘型分配，並且資料集中。K廠商是稍微左偏分佈，資料較為分散所以變異性也相對較大。Q廠商可看出為右偏分佈，但因為半數資料都為0所以導致圖被拉得很開，變異性看起來也較大。

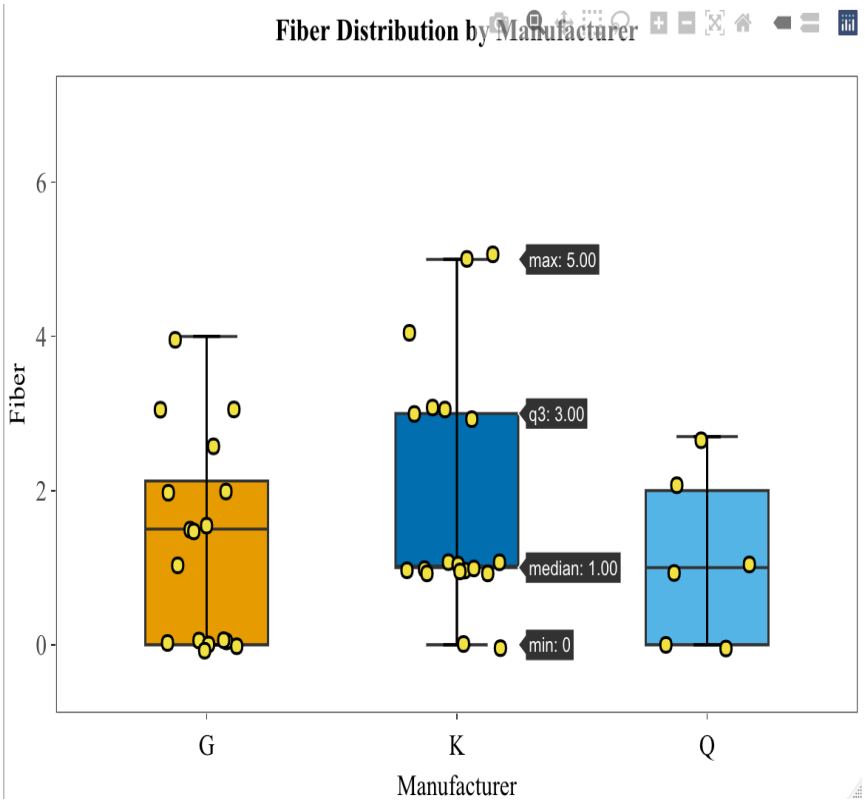
可以看到3個廠商的中位數(G:200，K:195，Q:75)是有差異，可以推測沒有顯著差異，但有一些趨勢。

圖七. 三個廠商鈉含量箱型圖(R程式)



圖八. 三個廠商卡路里含量箱型圖(Python程式)

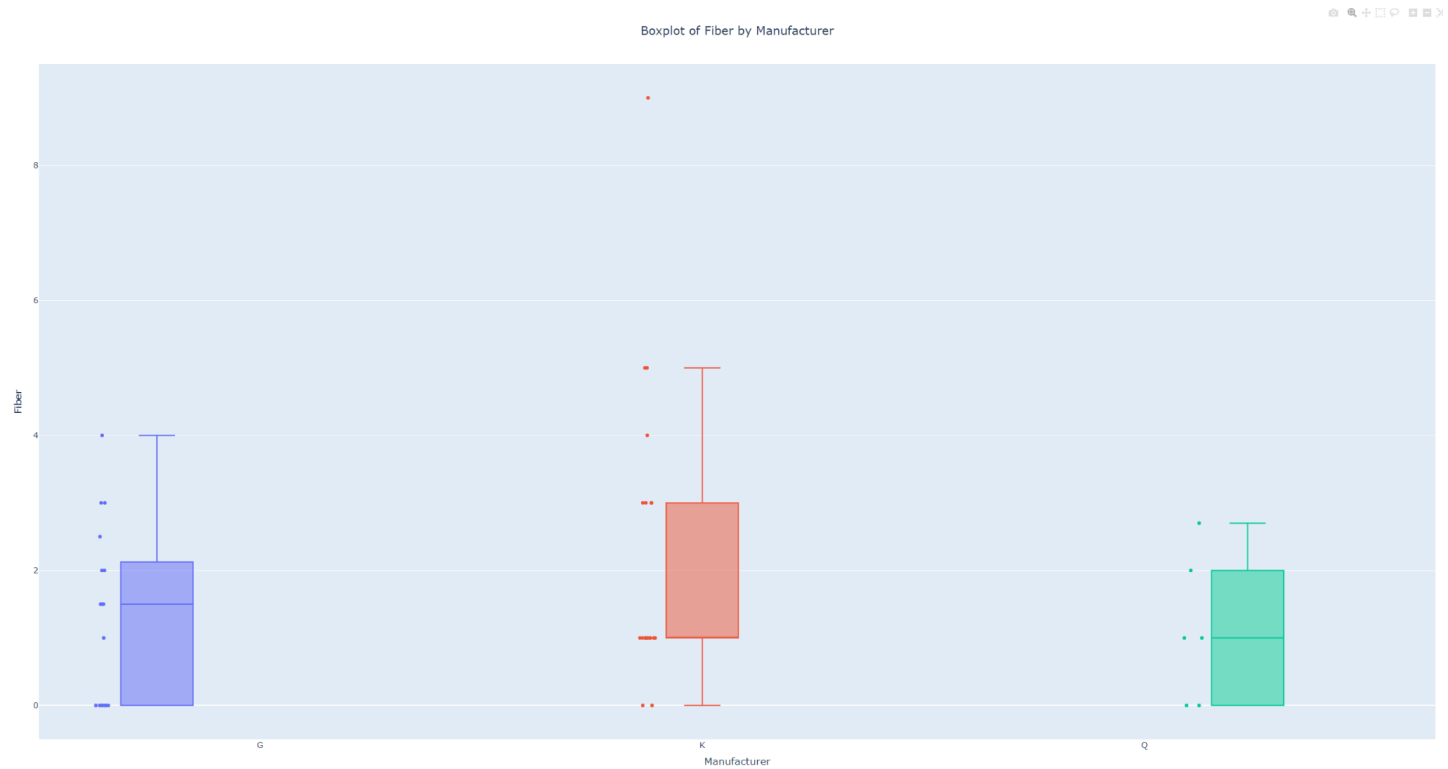
- Fiber



此圖中可看出G廠商的纖維含量是已經有點接近鐘型分配，但也因為有大部分資料集中在0導致圖型分佈較開。K廠商是右偏分佈，資料較為分散所以變異性也相對較大，另外其中有一個數值為9的outlier，因爲差距太大沒有顯示在圖中。Q廠商因為樣本數較少，看起來也接近鐘型分配。

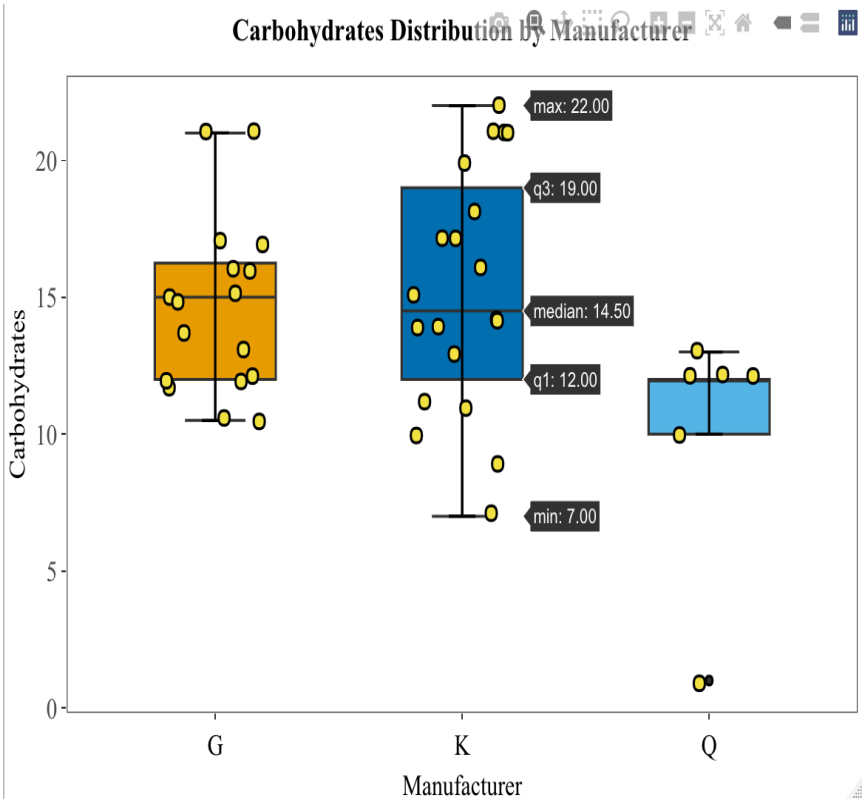
可以看到3個廠商的中位數(G:1.5 , K:1 , Q:1)是有差異，可以推測各廠商在纖維的部分差距不大。

圖九. 三個廠商纖維含量箱型圖(R程式)



圖十. 三個廠商卡路里含量箱型圖(Python程式)

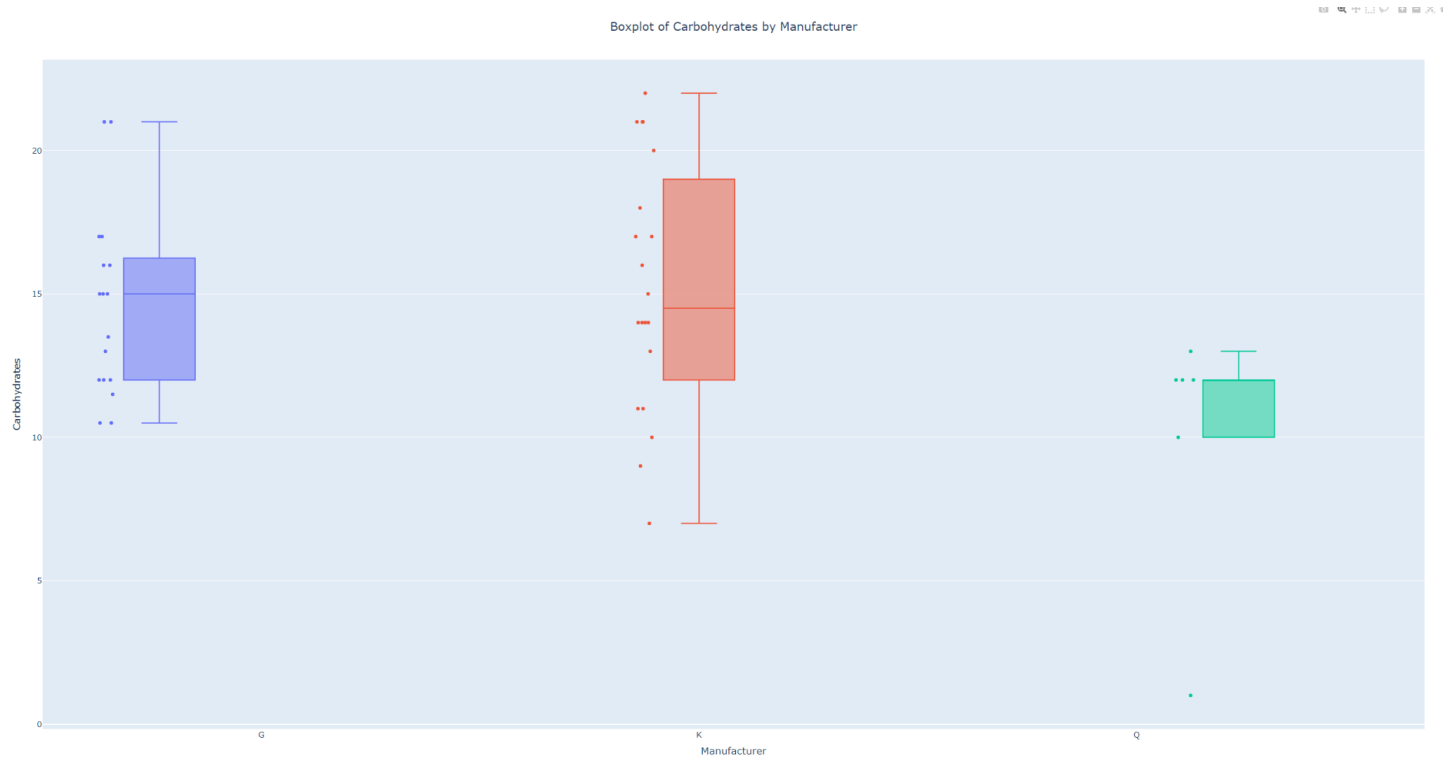
- Carbohydrates



此圖中可看出G廠商的碳水化合物含量是偏右偏分配，資料也較為集中所以變異性不大。K廠商已經有點接近鐘型分配，資料較為分散所以變異性也相對較大。Q廠商因為樣本數較少，看是偏左偏分配，而且有一個明顯的outlier。

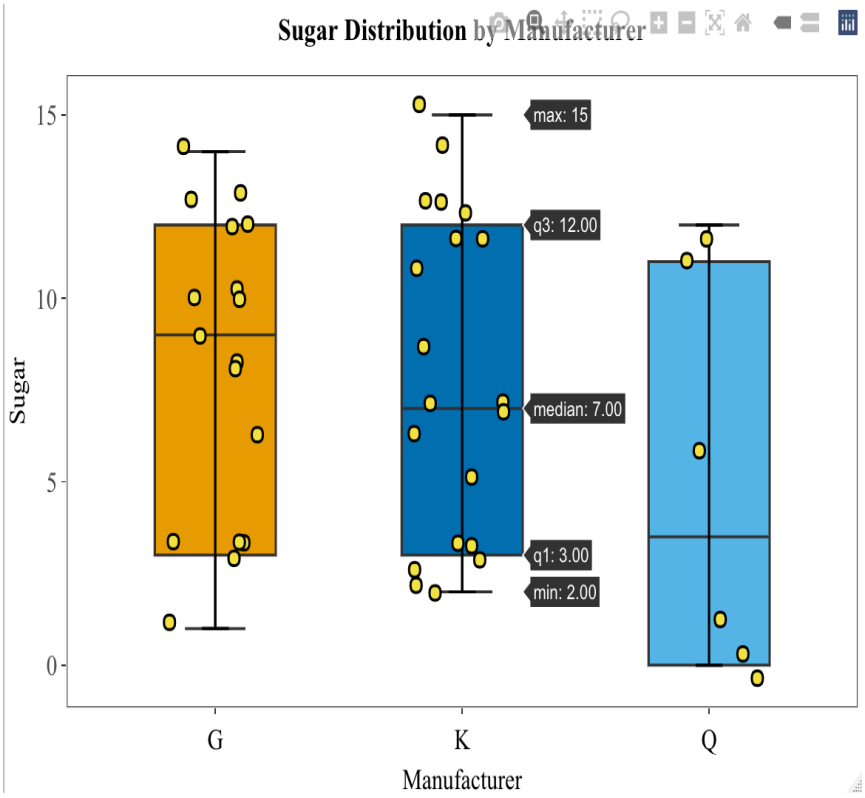
可以看到3個廠商的Q 製造商的平均值較低（10g），G 和 K 較高（分別為 14.6g 和 15.2g），可看出Q 製造商明顯低於其餘兩個。

圖十一. 三個廠商碳水化合物含量箱型圖(R程式)



圖十二. 三個廠商卡路里含量箱型圖(Python程式)

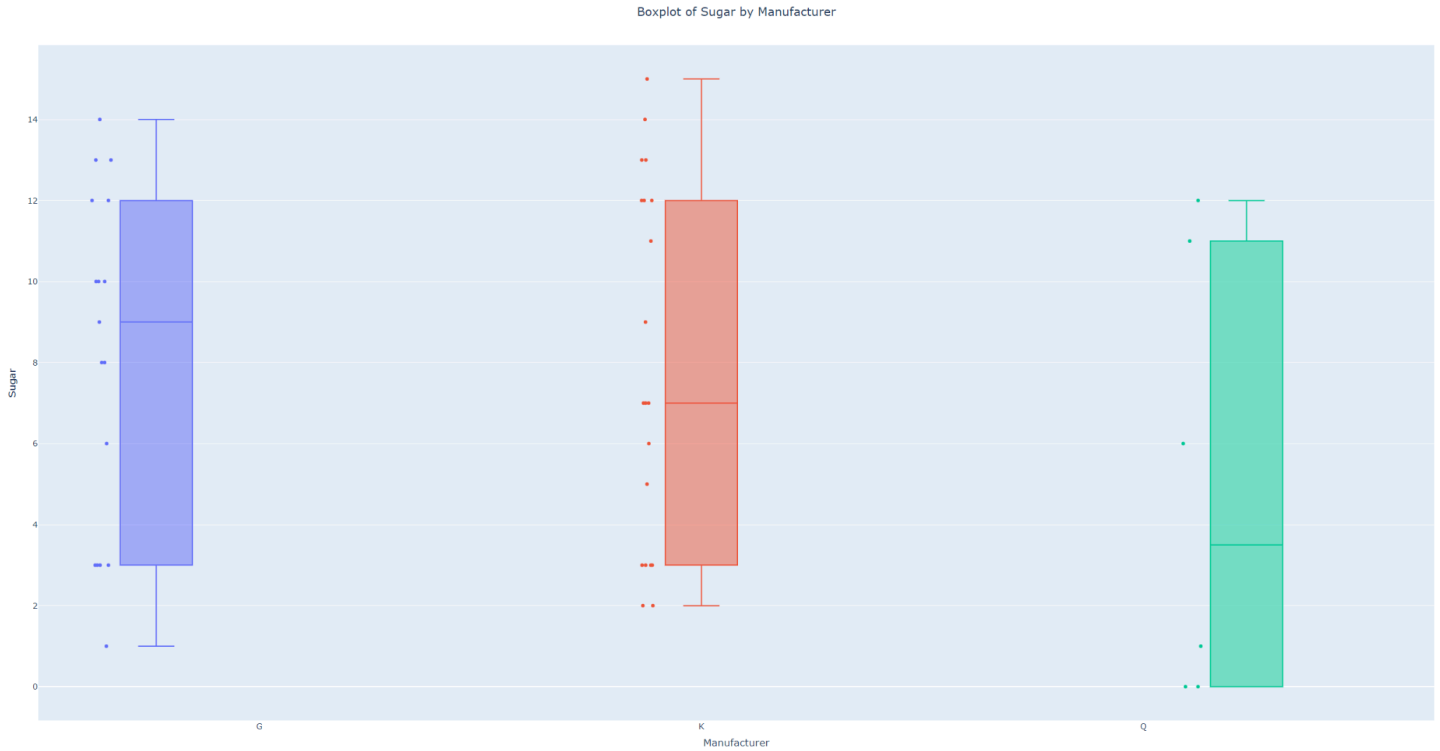
- Sugar



此圖中可看出G廠商的糖含量是左偏分佈，資料較為分散所以變異性也相對較大。K廠商是右偏分佈，資料較為分散所以變異性也相對較大，Q廠商是左偏分佈，且資料較為分散所以變異性也相對較大。

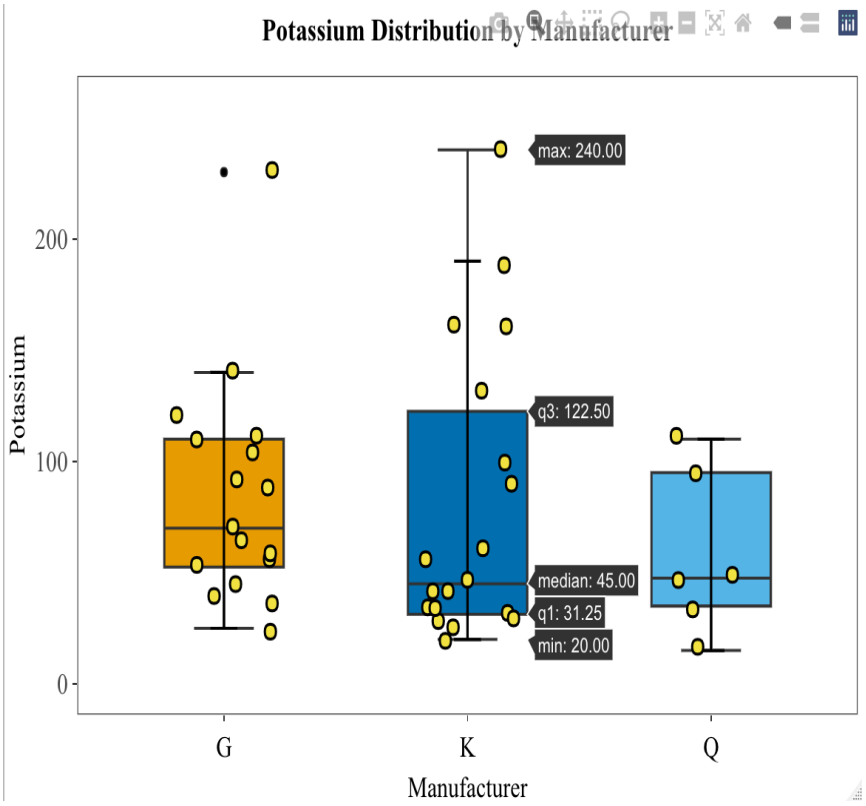
可以看到3個廠商的中位數(G:9 , K:7 , Q:3.5)是有差異，可以推測Q廠商在糖分營養物的添加上明顯是少於G、K廠商的，但變異也較大。

圖十三. 三個廠商糖含量箱型圖(R程式)



圖十四. 三個廠商卡路里含量箱型圖(Python程式)

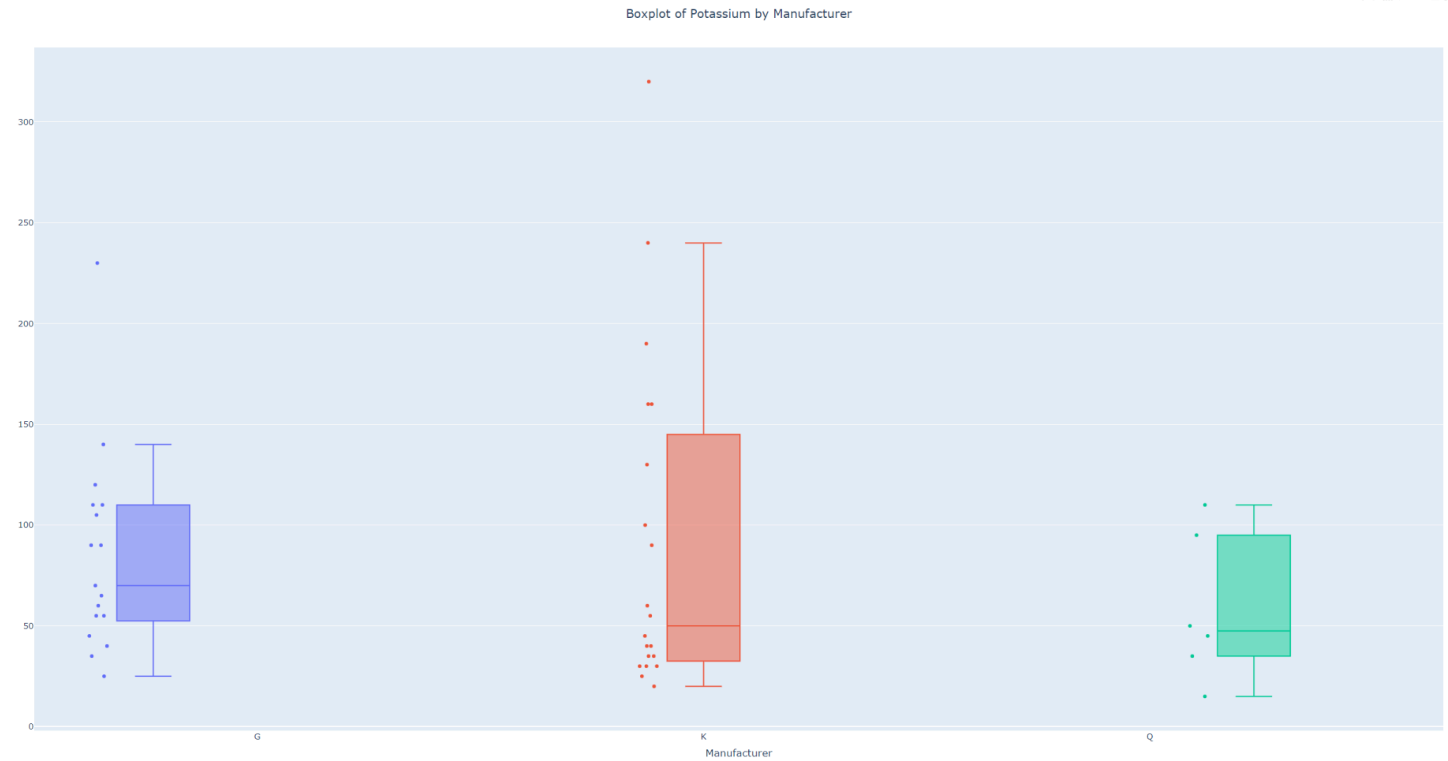
- Potassium



此圖中可看出G廠商的鉀含量是左偏分佈，資料也較為集中所以變異性不大。K廠商是右尾分佈，資料較為分散所以變異性也相對較大，另外其中有一個數值為320的outlier，因爲差距太大沒有顯示在圖中。Q廠商左偏分佈，資料分散程度正常。

可以看到3個廠商的中位數(G:70，K:50，Q:47.5)是有差異，可以推測G廠商在鉀營養物的添加上明顯是略高於K、Q廠商的。

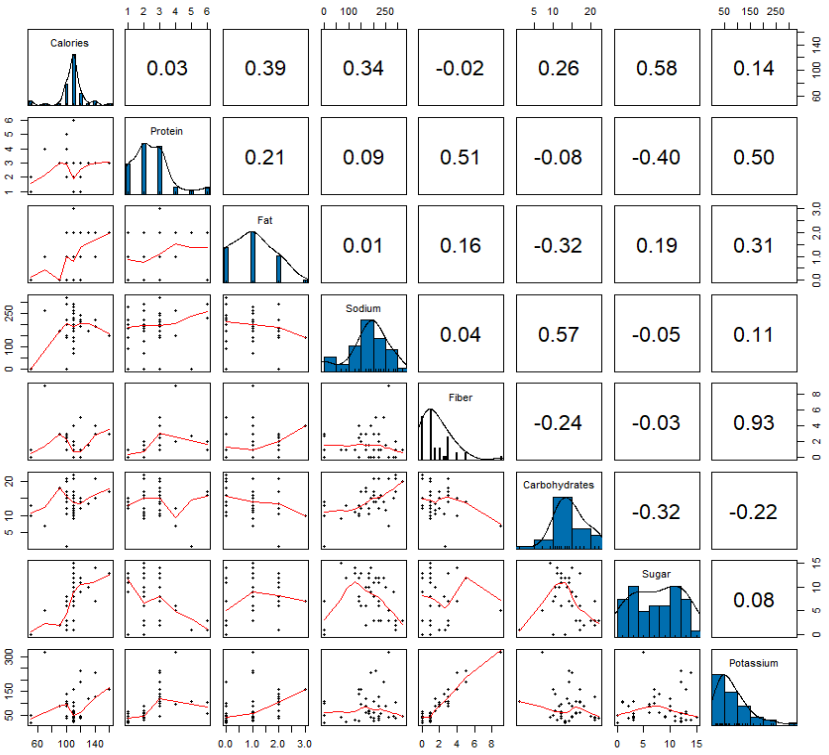
圖十五. 三個廠商鉀含量箱型圖(R程式)



圖十六. 三個廠商卡路里含量箱型圖(Python程式)

S2. 八個變數的相關性

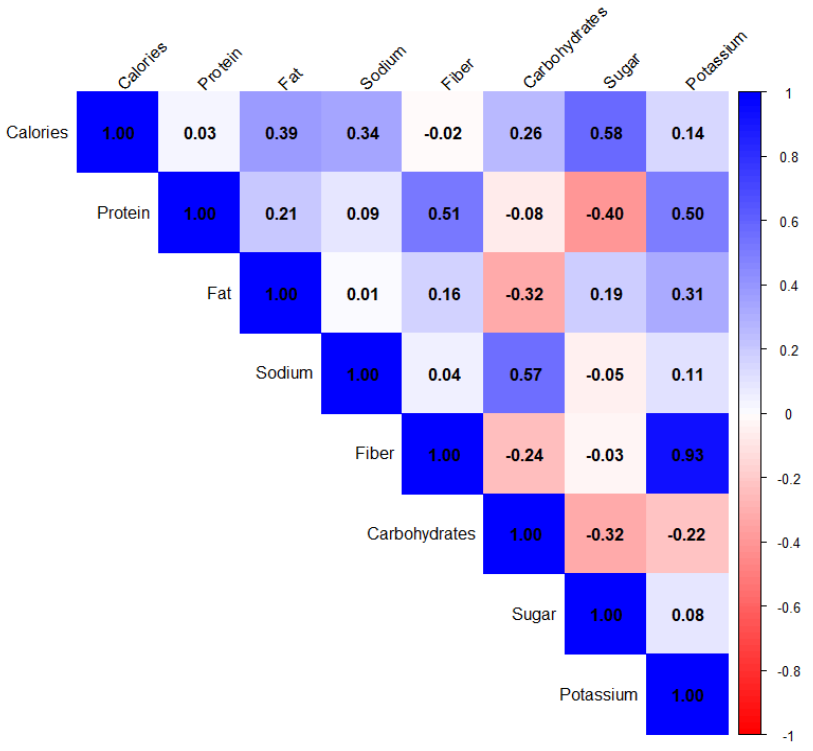
- Scatterplot Matrix



此圖可以看出八個變數兩兩之間的相關性圖中左下半部為變數間的散佈圖，由左下往右上的為正相關，左上往右下為負相關，而趨於平穩的則是無線性相關。

右上半部為變數之間的相關係數，其中可以更清楚的看出變數之間的相關性，我將這半部分輸出成另一份視覺化圖表以更好進行觀察。

圖十七. 八個變數的Scatterplot Matrix



圖中藍色部分顏色越深，代表正相關性越高。紅色顏色越深，代表負相關性越高。而顏色越淺則越趨近於無線性相關。

其中可看出幾點特別之處

1.Potassium和Fiber存在高度正相關。

2.Sugar和Calories

Potassium和Protein

Protein和Fiber

Sodium和Carbohydrates存在中度正相關。

3.Sugar和Protein

Carbohydrates和Fiber

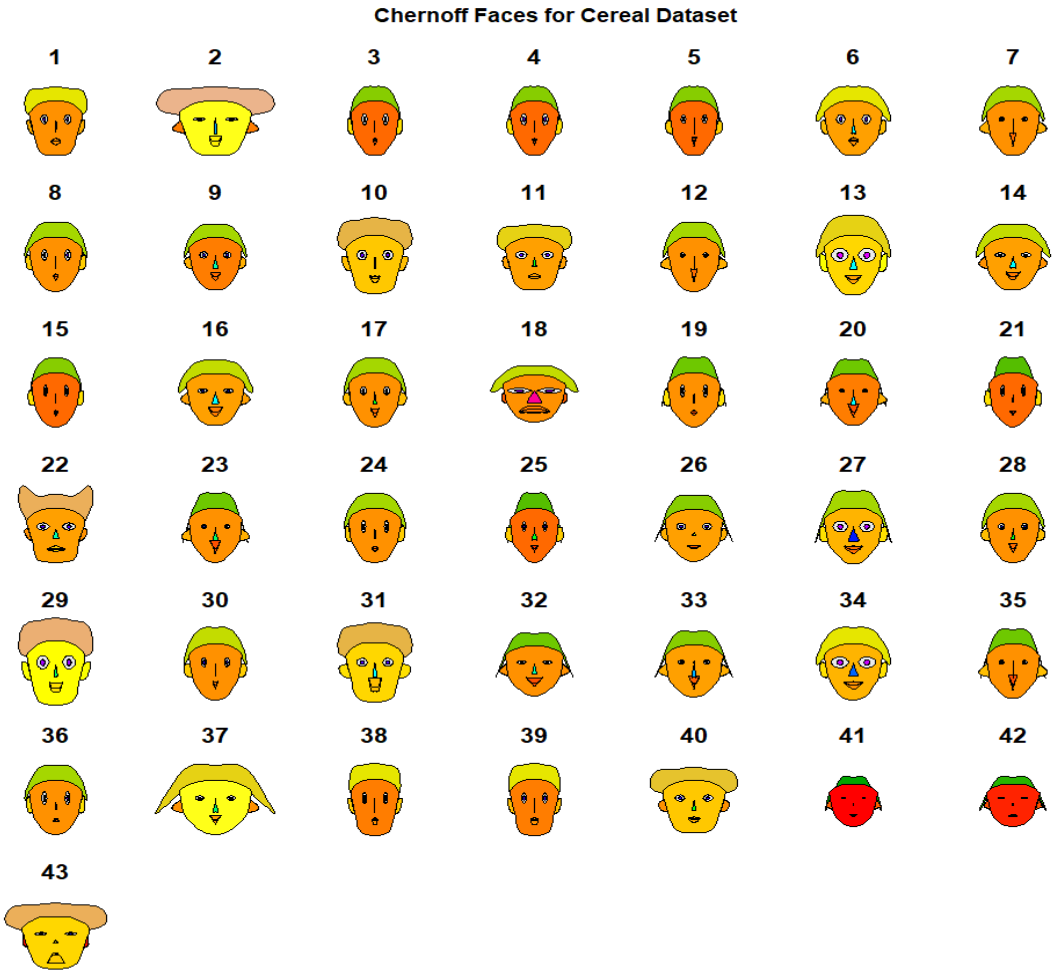
Protein和Fiber

Fat和Carbohydrates存在中度負相關。

4. 其他大多都為相關性不明顯。

圖十八. 八個變數的Scatterplot Matrix

- Chernoff Faces



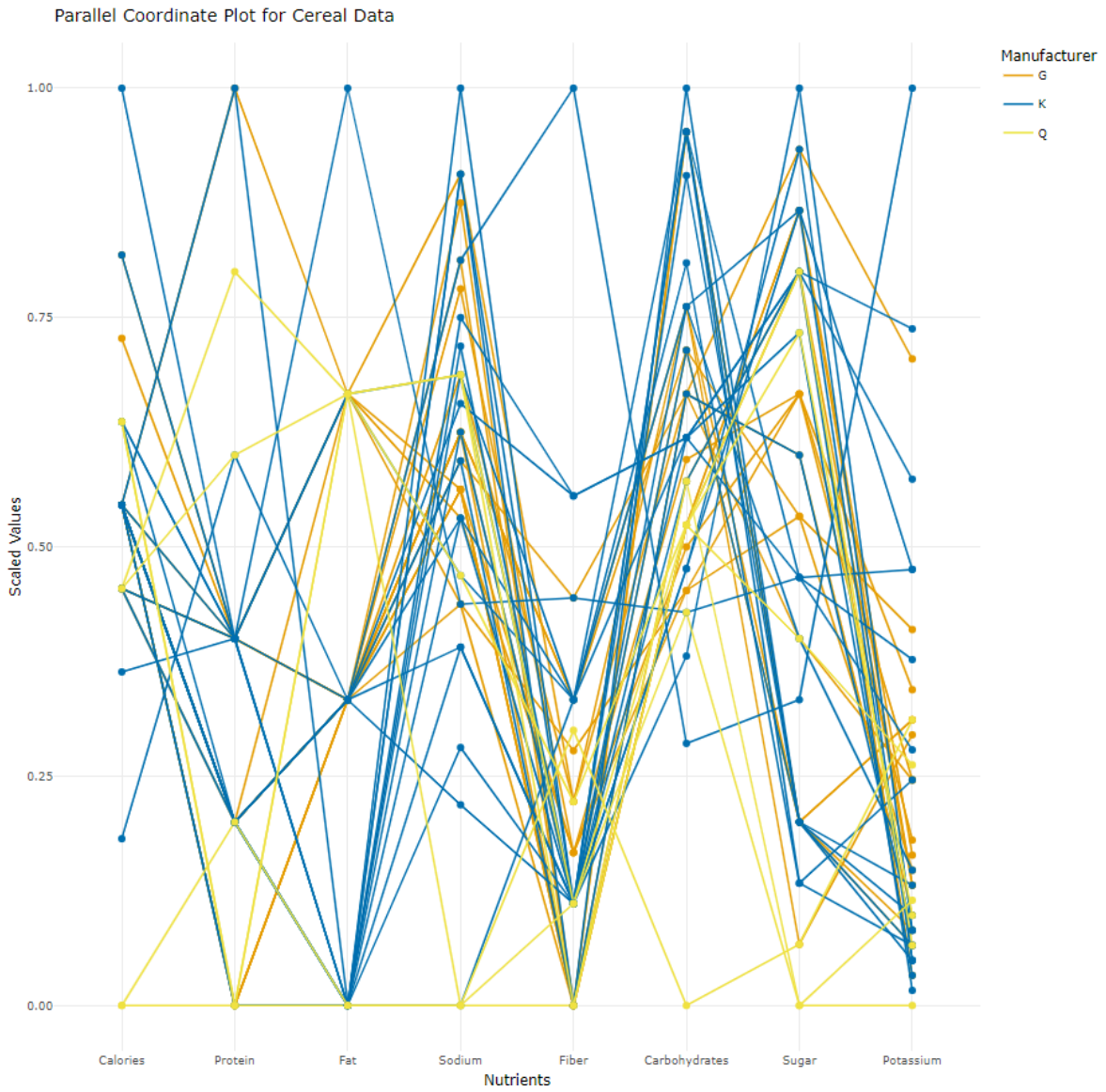
圖十九. 三十二個樣本的Chernoff Face

此圖為32個樣本的Chernoff Faces，以下是每個特徵所代表的變數:

1. height of face --> Calories
2. width of face --> Protein
3. structure of face --> Fat
4. height of mouth --> Sodium
5. width of mouth --> Fiber
6. smiling --> Carbohydrates
7. height of eyes --> Sugar
8. width of eyes --> Potassium
9. height of hair --> Calories
10. width of hair --> Protein
11. style of hair --> Fat
12. height of nose --> Sodium
13. width of nose --> Fiber
14. width of ear --> Carbohydrates
15. height of ear --> Sugar

在此圖中1-17為G廠商18-37為K廠商37-43為Q廠商，其中我們可以看出，43筆數據中有其中幾筆與其他的有顯著差距。像是第2、13、18、22、27、29、31、34、37、40、41、42、43。這13比數據都在不同的特徵上與其他樣本有明顯差距。其中Chernoff Faces也可方便我們去判斷變數中的outlier，像圖形2、37明顯在頭髮寬度上有顯著差距，對應到上面的特徵變數可以發現，樣本2、37確實在蛋白質的數據上明顯高於其他樣本。

- parallel coordinate plot



圖二十. 八個變數的parallel coordinate plot

分析這張圖可以讓我們看到:

1. 營養變異性:不同製造商的產品在各營養成分的分佈有顯著差異，如某些品牌的穀片在鈉或糖的含量上可能比其他品牌高。
2. 群體差異:比如，一些品牌可能傾向於生產高蛋白或低糖的產品，這可以通過不同顏色的線條的分布模式來識別。
3. 異常值偵測:一些極端的營養指標值(如非常高的糖分或非常低的脂肪)可以迅速被視覺識別出來，有助於分析產品營養成分的平衡性。

S3. 結論

表一. 三個廠商每個營養物的Kruskal-Wallis表

	DF	卡方值 (Chi-squared)	Pr(>F)	結論
Calories	2	1.9978	0.3683	無顯著差異
Protein	2	1.1663	0.5581	無顯著差異
Fat	2	7.794	0.0203	顯著差異
Sodium	2	3.3146	0.1907	無顯著差異
Fiber	2	2.2876	0.3186	無顯著差異
Carbohydrates	2	5.9288	0.05159	無顯著差異
Sugar	2	2.5811	0.2751	無顯著差異
Postassium	2	1.5003	0.4723	無顯著差異

結合上面S1、S2加上 Kruskal-Wallis表，統整出以下是我推測三個廠商對每個添加物的結論。

A. 從箱型圖還有Kruskal-Wallis表可以發現，脂肪含量是最顯著的差異點。K 製造商的產品平均脂肪含量較低，推測可能是一個市場優勢。
B. 碳水化合物含量的差異也值得注意。Q 製造商的產品碳水化合物含量較低，推測可能吸引一些特定的消費群體。
C. 雖然在統計上不顯著，但 Q 製造商的產品在卡路里、鈉和糖的含量上平均值較低，推測這可能是一個潛在的市場定位點。
D. K 製造商的產品在纖維和鉀含量上平均值較高，推測可能對健康導向的消費者有吸引力。
E. G 製造商的產品在大多數營養成分上都處於中等水平，推測可能代表了一種平衡的方法。

附錄

程式碼(R)

```
#載入相關套件
install.packages("ggplot2")#繪圖
install.packages("plotly")#互動式
install.packages("dplyr")
install.packages("psych")#散佈矩陣圖
install.packages("corrplot")#相關係數散佈矩陣圖
install.packages("aplpack")
install.packages("tidyverse")
install.packages("car")
install.packages("rstatix")
library(tidyverse)
library(car)
library(rstatix)
library(aplpack)
library(corrplot)
library(ggplot2)
library(psych)
library(plotly)
library(dplyr)
library(htmlwidgets)
library(htmltools)

##資料匯入、整理
Cereal <- read.csv(file = "C:/Users/Trident/Downloads/Cereal.csv")
variables <- c("Calories", "Protein", "Fat", "Sodium", "Fiber", "Carbohydrates", "Sugar", "Potassium")

colors <- c("#E69F00", "#0072B2", "#F0E442", "#56B4E9")

# 創建一个向量來儲存所有生成的HTML文件的名稱
html_files <- c()

## 繪製BoxPlot
# 使用迴圈生成每個變數的箱型圖
for (var in variables) {
  summary_stats <- Cereal %>%
    group_by(Manufacturer) %>%
    summarize(
```

```
    Q1 = quantile(get(var), 0.25),
    Median = median(get(var)),
    Q3 = quantile(get(var), 0.75),
    .groups = 'drop'
  )

 P1 <- ggplot(Cereal, aes_string(x = "Manufacturer", y = var, fill = "Manufacturer")) +
  stat_boxplot(geom = "errorbar", width = 0.15, aes(color = "black")) +
  geom_boxplot(size = 0.5, outlier.fill = "white", outlier.color = "white") +
  geom_jitter(aes(fill = "Manufacturer"), width = 0.2, shape = 21, size = 2.5) +
  scale_fill_manual(values = colors) +
  scale_color_manual(values = c("black")) +
  ggtitle(paste(var, "Distribution by Manufacturer")) +
  theme_bw() +
  theme(legend.position = "none",
        axis.text.x = element_text(colour = "black", family = "Times", size = 14),
        axis.text.y = element_text(family = "Times", size = 14, face = "plain"),
        axis.title.y = element_text(family = "Times", size = 14, face = "plain"),
        axis.title.x = element_text(family = "Times", size = 14, face = "plain"),
        plot.title = element_text(family = "Times", size = 15, face = "bold", hjust = 0.5),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank()) +
  ylab(var) + xlab("Manufacturer")

 plotly_p <- ggplotly(P1)

 # 生成HTML文件名
 html_filename <- paste0(var, "_boxplot.html")

 # 保存交互式圖表為HTML文件
 saveWidget(plotly_p, file = html_filename, selfcontained = TRUE)

 #將文件名添加到向量中
 html_files <- c(html_files, html_filename)

 print(plotly_p)
}

# 創建一个包含所有連結的HTML文件
links_html <- tags$ul(
 lapply(html_files, function(file) {
  tags$li(
```

```
      tags$a(href = file, target = "_blank", file)
    )
  })
)

html_content <- tags$html(
  tags$head(
    tags$title("BoxPlot Links")
  ),
  tags$body(
    tags$h1("Links to BoxPlot HTML Files"),
    links_html
  )
)

# 保存HTML文件
save_html(html_content, file = "boxplot_links.html")

##製作Scatterplot Matrix
# 計算相關矩陣
cor_matrix <- cor(Cereal[variables], use = "complete.obs", method = "pearson")

# 使用 psych 套件的 pairs.panels 函數生成相關矩陣的散佈圖矩陣
pairs.panels(Cereal[variables],
             method = "pearson",  # 使用皮爾森相關係數
             hist.col = "#0072B2",  # 直方圖顏色
             density = TRUE,  # 顯示密度圖
             ellipses = FALSE)  # 顯示相關係數橢圓

# 繪製相關矩陣圖
corrplot(cor_matrix,
         method = "color",  # 使用顏色來表示相關性
         type = "upper",  # 只顯示矩陣的上半部
         tl.col = "black",  # 標籤顏色
         tl.srt = 45,  # 旋轉標籤
         addCoef.col = "black",  # 在圖上顯示相關係數
         col = colorRampPalette(c("red", "white", "blue"))(200))  # 設定顏色漸層

##繪製Chernoff Faces
# 提取需要的變數
variables <- c("Calories", "Protein", "Fat", "Sodium", "Fiber", "Carbohydrates", "Sugar", "Potassium")
subset_data <- Cereal[, variables]
```

```
# 確保數據中沒有 NA 值，如果有 NA 值，可以將它們去除或填補
subset_data <- na.omit(subset_data)

# 繪製 Chernoff faces
faces(subset_data, face.type = 1, main = "Chernoff Faces for Cereal Dataset")

##ANOVA分析
# 選擇所有數值變數
numeric_vars <- Cereal %>%
  select(where(is.numeric)) %>%
  names()

print(numeric_vars)

# 創建一個函數來執行分析
analyze_variable <- function(var) {
  formula <- as.formula(paste(var, "~ Manufacturer"))

  # Welch's ANOVA
  welch_result <- oneway.test(formula, data = Cereal, var.equal = FALSE)

  # Kruskal-Wallis test
  kruskal_result <- kruskal.test(formula, data = Cereal)

  # 描述性統計
  desc_stats <- Cereal %>%
    group_by(Manufacturer) %>%
    summarise(
      Mean = mean(!!sym(var)),
      SD = sd(!!sym(var)),
      Median = median(!!sym(var)),
      IQR = IQR(!!sym(var))
    )

  list(
    Variable = var,
    Welch_ANOVA = welch_result,
    Kruskal_Wallis = kruskal_result,
    Descriptive_Stats = desc_stats
  )
}
```

```
# 對每個數值變數執行分析
results <- map(numeric_vars, analyze_variable)

# 打印結果
for (r in results) {
  cat("\n\n===", r$Variable, "===\n")
  cat("\nWelch's ANOVA:\n")
  print(r$Welch_ANOVA)
  cat("\nKruskal-Wallis Test:\n")
  print(r$Kruskal_Wallis)
  cat("\nDescriptive Statistics:\n")
  print(r$Descriptive_Stats)
}

# 創建一個總結表格
summary_table <- map_dfr(results, function(r) {
  tibble(
    Variable = r$Variable,
    Welch_p_value = r$Welch_ANOVA$p.value,
    Kruskal_p_value = r$Kruskal_Wallis$p.value
  )
})

# 打印總結表格
print(summary_table)

# 可視化
library(ggplot2)

for (var in numeric_vars) {
  p <- ggplot(Cereal, aes(x = Manufacturer, y = !!sym(var))) +
    geom_boxplot() +
    labs(title = paste("Distribution of", var, "by Manufacturer"),
         y = var) +
    theme_minimal()

  print(p)
}
```

程式碼(Python)

```
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
from pathlib import Path

# 讀取CSV檔案
csv_file = "C:/Users/Trident/Downloads/Cereal.csv"
data = pd.read_csv(csv_file)

# 定義要分析的變數
variables = ["Calories", "Protein", "Fat", "Sodium", "Fiber", "Carbohydrates", "Sugar", "Potassium"]

# 創建HTML文件的樣式
html_template = """
<html>
<head>
    <title>穀物營養成分分析</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
            background-color: #f0f2f5;
        }
        h1 {
            color: #1a237e;
            text-align: center;
        }
        .chart-container {
            background-color: white;
            border-radius: 10px;
            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
            margin-bottom: 20px;
            padding: 15px;
        }
        .description {
            color: #37474f;
            margin-bottom: 10px;
        }
    </style>
```

```
</head>
<body>
    <h1>穀物營養成分分析報告</h1>
"""

# 開始添加內容
html_content = html_template

# 1. 創建營養成分箱型圖矩陣
fig_matrix = make_subplots(rows=4, cols=2, subplot_titles=variables)
for i, var in enumerate(variables, 1):
    row = (i-1) // 2 + 1
    col = (i-1) % 2 + 1

    box_trace = go.Box(x=data['Manufacturer'], y=data[var], name=var)
    fig_matrix.add_trace(box_trace, row=row, col=col)

fig_matrix.update_layout(height=1200, showlegend=False, title_text="營養成分分佈概覽")
html_content += f"""
<div class="chart-container">
    <div class="description">
        <h2>各製造商穀物營養成分分佈</h2>
        <p>這個圖表展示了不同製造商的穀物在各營養成分上的分佈情況。使用箱型圖可以直觀地看出每個營養成分的中位數、四分位數範圍和異常值。</p>
    </div>
    {fig_matrix.to_html(full_html=False)}
</div>
"""

# 2. 添加相關性熱圖
corr_matrix = data[variables].corr()
fig_heatmap = go.Figure(data=go.Heatmap(
    z=corr_matrix,
    x=variables,
    y=variables,
    colorscale='RdBu',
    zmin=-1, zmax=1
))
fig_heatmap.update_layout(
    title='營養成分相關性熱圖',
    height=600,
)
```

```
html_content += f"""
<div class="chart-container">
    <div class="description">
        <h2>營養成分相關性分析</h2>
        <p>此熱圖展示了各營養成分之間的相關性。紅色表示正相關，藍色表示負相關，顏色越深表示相關性越強。</p>
    </div>
    {fig_heatmap.to_html(full_html=False)}
</div>
"""

# 3. 添加散點圖矩陣
fig_scatter = px.scatter_matrix(
    data,
    dimensions=variables,
    color="Manufacturer"
)
fig_scatter.update_layout(height=1000, title="營養成分散點圖矩陣")
html_content += f"""
<div class="chart-container">
    <div class="description">
        <h2>營養成分關係散點圖</h2>
        <p>這個散點圖矩陣展示了所有營養成分之間的兩兩關係，不同顏色代表不同製造商。您可以通過此圖發現營養成分之間的潛在關係。</p>
    </div>
    {fig_scatter.to_html(full_html=False)}
</div>
"""

# 4. 添加雷達圖
manufacturers = data['Manufacturer'].unique()
fig_radar = go.Figure()

for manufacturer in manufacturers:
    manufacturer_data = data[data['Manufacturer'] == manufacturer]
    mean_values = manufacturer_data[variables].mean()

    fig_radar.add_trace(go.Scatterpolar(
        r=mean_values,
        theta=variables,
        fill='toself',
        name=manufacturer
```

```
    ))

fig_radar.update_layout(
    polar=dict(radialaxis=dict(visible=True, range=[0, data[variables].max().max()])),
    showlegend=True,
    title="製造商營養成分平均值比較"
)

html_content += f"""
<div class="chart-container">
    <div class="description">
        <h2>製造商營養特徵雷達圖</h2>
        <p>這個雷達圖比較了不同製造商的穀物在各營養成分上的平均水平。通過這個圖，您可以快速識別每個製造商的營養特徵。</p>
    </div>
    {fig_radar.to_html(full_html=False)}
</div>
"""

# 完成HTML內容
html_content += "</body></html>"

# 將HTML內容寫入檔案
output_path = Path("cereal_analysis_enhanced.html")
output_path.write_text(html_content)

print(f"增強的分析報告已保存到 {output_path.absolute()}")
```