

數據分析文件-學歷與職場發展之關聯

目錄

目錄.....	1
一. 前言.....	2
資料來源 (Kaggle)	2
匯入之套件.....	2
二. 數據簡介.....	2
三. 實作步驟.....	3
1.數據檢視與清理.....	3
2.欄位分析與可視化圖表.....	4
(1) 【晉升機會(Years_to_Promotion)之分析】(許瑞宏).....	4
(2) 【起薪(Starting Salary)之分析】(黃晨宇).	7
四. 其他【綜合分析(Combination)】(孫耀庭).....	10
五. 總結	12

一.前言

本技術文件詳細介紹了對 Kaggle 數據集之 "Education Career Success" 數據分析過程。我們將使用 Pandas, Numpy 等套件進行數據處理，並透過 Plotly, Matplotlib, Seaborn 等套件進行視覺化分析，以探索學歷、技能與職場表現（如起薪、工作滿意度）之間的關係。

(Kaggle 數據資料來源:<https://www.kaggle.com/datasets/adilshamim8/education-and-career-success>)

二.數據集簡介

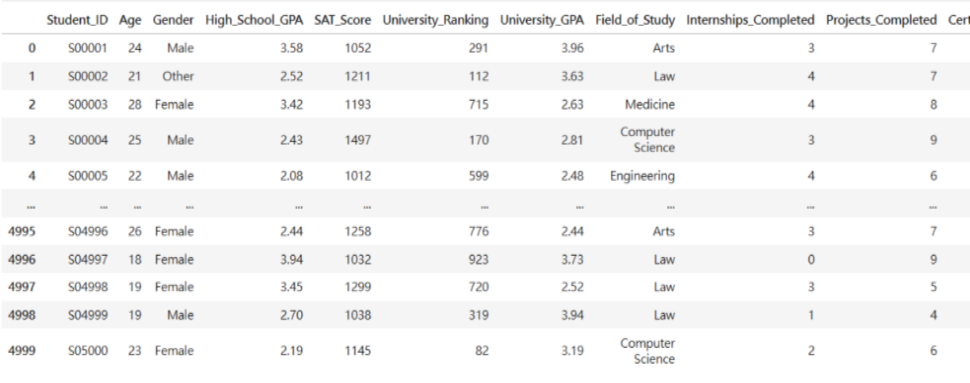
此份 Kaggle 中 "Education Career Success" 的數據集包含 5000 筆數據(X 軸)和 20 個特徵(Y 軸)，以下是數據集的主要欄位：

類別	欄位名稱	說明
基本資訊	Student_ID	學生編號
基本資訊	Age	年齡
基本資訊	Gender	性別 (Male / Female / Other)
學術背景	High_School_GPA	高中 GPA
學術背景	SAT_Score	SAT 考試分數
學術背景	University_Ranking	大學排名 (數值越小代表排名越高)
學術背景	University_GPA	大學 GPA
學術背景	Field_of_Study	學習領域 (如 Computer Science、Law 等)
技能與經歷	Internships_Completed	完成的實習數量
技能與經歷	Projects_Completed	完成的專案數量
技能與經歷	Certifications	取得的證書數量
技能與經歷	Soft_Skills_Score	軟技能評分 (1-10)
技能與經歷	Networking_Score	人脈網絡評分 (1-10)
職業發展	Job_Offers	獲得的工作機會數
職業發展	Starting_Salary	起薪 (美元)
職業發展	Career_Satisfaction	職業滿意度 (1-10)
職業發展	Years_to_Promotion	晉升所需年數
職業發展	Current_Job_Level	當前職位層級 (Entry, Mid, Senior)
職業發展	Work_Life_Balance	工作與生活平衡 (1-10)
職業發展	Entrepreneurship	是否創業 (Yes/No)

三.實作步驟

1. 數據檢視與清理

(1)

程式碼實作	程式碼解析
<pre>import numpy as np import matplotlib.pyplot as plt import pandas as pd import plotly.express as px import plotly.graph_objects as go # 讀取 CSV 檔案 df = pd.read_csv("education_career_success.csv") df</pre>  <p>5000 rows x 20 columns</p>	<ul style="list-style-type: none"> 匯入套件並進行數據預覽 (education_career_success.csv) 使用 pandas.read_csv() 讀取 CSV 檔案後的結果。

(2)

程式碼實作	程式碼解析
<pre>df.info() <class 'pandas.core.frame.DataFrame'> RangeIndex: 5000 entries, 0 to 4999 Data columns (total 20 columns): # Column Non-Null Count Dtype --- --- 0 Student_ID 5000 non-null object 1 Age 5000 non-null int64 2 Gender 5000 non-null object 3 High_School_GPA 5000 non-null float64 4 SAT_Score 5000 non-null int64 5 University_Ranking 5000 non-null int64 6 University_GPA 5000 non-null float64 7 Field_of_Study 5000 non-null object 8 Internships_Completed 5000 non-null int64 9 Projects_Completed 5000 non-null int64 10 Certifications 5000 non-null int64 11 Soft_Skills_Score 5000 non-null int64 12 Networking_Score 5000 non-null int64 13 Job_Offers 5000 non-null int64 14 Starting_Salary 5000 non-null float64 15 Career_Satisfaction 5000 non-null int64 16 Years_to_Promotion 5000 non-null int64 17 Current_Job_Level 5000 non-null object 18 Work_Life_Balance 5000 non-null int64 19 Entrepreneurship 5000 non-null object dtypes: float64(3), int64(12), object(5) memory usage: 781.4+ KB</pre> <pre>df.isnull().sum() Student_ID 0 Age 0 Gender 0 High_School_GPA 0 SAT_Score 0 University_Ranking 0 University_GPA 0 Field_of_Study 0 Internships_Completed 0 Projects_Completed 0 Certifications 0 Soft_Skills_Score 0 Networking_Score 0 Job_Offers 0 Starting_Salary 0 Career_Satisfaction 0 Years_to_Promotion 0 Current_Job_Level 0 Work_Life_Balance 0 Entrepreneurship 0 dtype: int64</pre>	<p>Method 1:</p> <ul style="list-style-type: none"> 使用 df.info() 檢視個特徵類別的筆數與類型。 從 Non-Null 得知並無缺失值 <p>Method 2:</p> <ul style="list-style-type: none"> 使用 df.isnull().sum() 亦顯示無缺失值

(3)

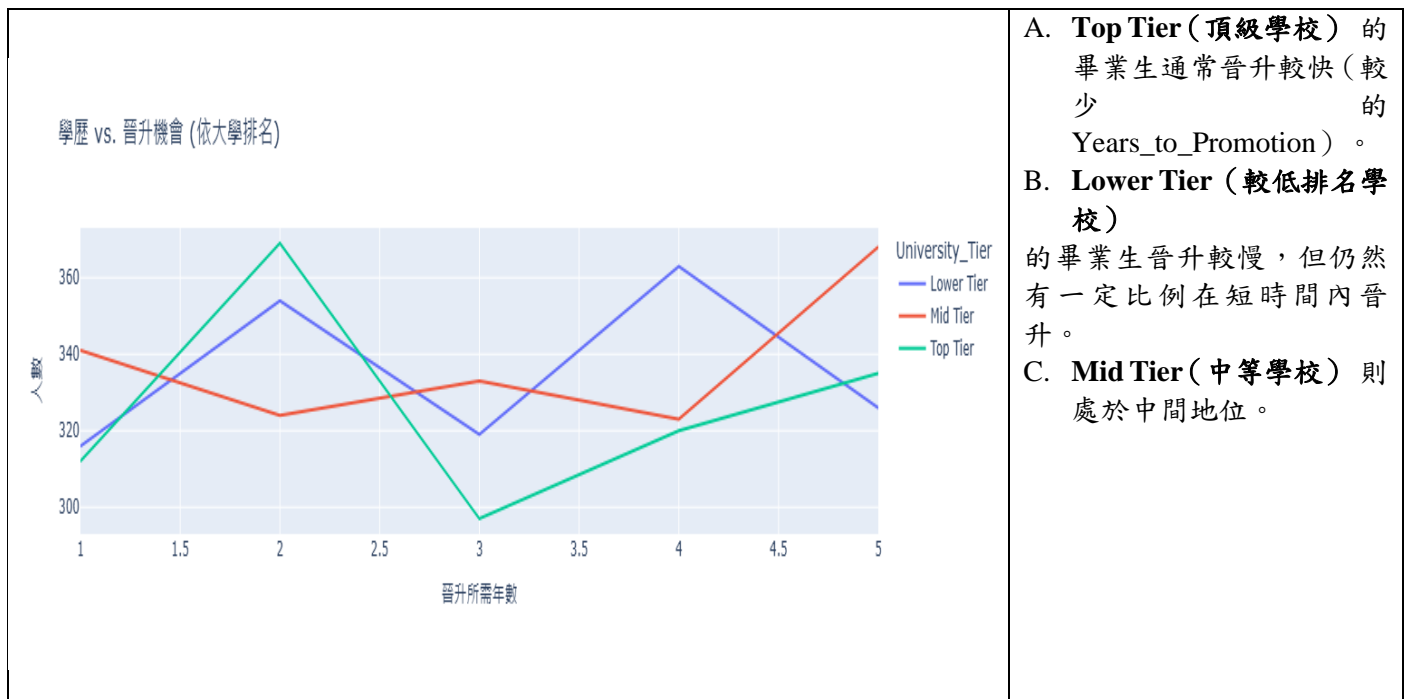
程式碼實作										程式碼解析
<code>df.describe()</code>										<ul style="list-style-type: none"> 使 <code>df.describe()</code> 得到統計資料
	Age	High_School_GPA	SAT_Score	University_Ranking	University_GPA	Internships_Completed	Projects_Completed	Certifications	Soft_Skills_Score	
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	
mean	23.442200	2.996978	1253.832000	504.335600	3.020028	1.982200	4.562800	2.512200	5.546000	
std	3.473712	0.575673	203.228954	291.060011	0.576047	1.408219	2.872927	1.703183	2.851159	
min	18.000000	2.000000	900.000000	1.000000	2.000000	0.000000	0.000000	0.000000	1.000000	
25%	20.000000	2.500000	1076.000000	256.000000	2.520000	1.000000	2.000000	1.000000	3.000000	
50%	23.000000	2.990000	1257.000000	501.500000	3.030000	2.000000	5.000000	3.000000	6.000000	
75%	26.000000	3.500000	1432.000000	759.000000	3.510000	3.000000	7.000000	4.000000	8.000000	
max	29.000000	4.000000	1600.000000	1000.000000	4.000000	4.000000	9.000000	5.000000	10.000000	
	Networking_Score	Job_Offers	Starting_Salary	Career_Satisfaction	Years_to_Promotion	Work_Life_Balance				
	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000				
	5.538000	2.488800	50563.540000	5.578000	3.015800	5.482400				
	2.850084	1.711859	14494.958207	2.871997	1.417446	2.883427				
	1.000000	0.000000	25000.000000	1.000000	1.000000	1.000000				
	3.000000	1.000000	40200.000000	3.000000	2.000000	3.000000				
	6.000000	2.000000	50300.000000	6.000000	3.000000	6.000000				
	8.000000	4.000000	60500.000000	8.000000	4.000000	8.000000				
	10.000000	5.000000	101000.000000	10.000000	5.000000	10.000000				

2. 欄位分析與可視化圖表

晉升機會(Years to Promotion) 分析

(1) 【學歷】與【晉升機會】分析

程式碼實作與可視化圖表	程式碼解析
<pre> # 分類大學排名 def university_tier(rank): if rank <= 333: return "Top Tier" elif rank <= 667: return "Mid Tier" else: return "Lower Tier" df["University_Tier"] = df["University_Ranking"].apply(university_tier) # 計算不同排名組別的晉升情況 promotion_trend = df.groupby(["Years_to_Promotion", "University_Tier"]).size().reset_index(name="Count") # **繪製堆疊柱狀圖** fig = px.line(promotion_trend, x="Years_to_Promotion", y="Count", color="University_Tier", title="學歷 vs. 晉升機會 (依大學排名)", labels={"Years_to_Promotion": "晉升所需年數", "Count": "人數"}) # 設定為堆疊模式，讓不同學歷的晉升情況直接顯示 fig.show()</pre>	<ul style="list-style-type: none"> 轉換分類大學排名(間距排名 333(含)以下，333-667(含)，以及 667-1000(含)) University_Tier 欄位把學校排名轉換為類別。(Top Tier, Mid Tier, and Lower Tier) 計算 不同學校層級的畢業生，在職場晉升所需年數。

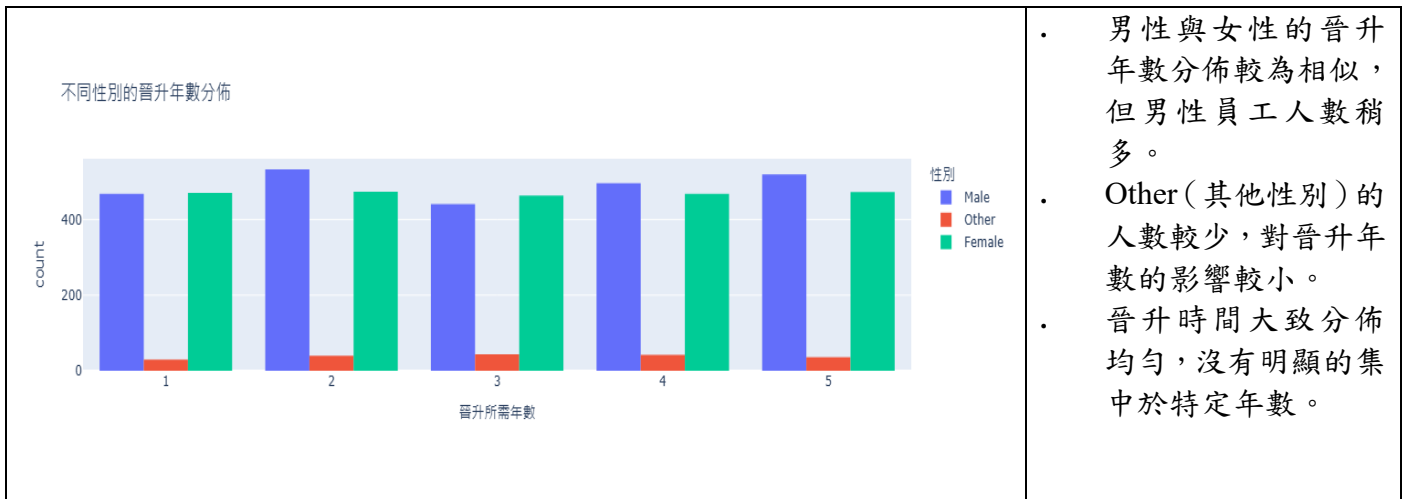


小結

- 學歷影響職場晉升速度，頂級學校的畢業生通常能夠更快獲得晉升機會。
- 中等學校畢業生晉升節奏較為平均，沒有明顯的優勢或劣勢，屬於職場的中堅份子。
- 較低排名學校的畢業生晉升較慢，但仍有一部分能夠在短時間內晉升，這可能取決於個人能力、職業選擇或產業需求。

(2) 【性別】與【晉升機會】分析

程式碼實作與可視化圖表	程式碼解析
<pre># 繪製不同性別的晉升年數直方圖 (不堆疊) fig = px.histogram(df, x="Years_to_Promotion", color="Gender", # 依照性別分類不同顏色 barmode="group", # 分組顯示，不堆疊 histfunc="count", # 讓 Y 軸顯示性別總數 title="不同性別的晉升年數分佈", labels={ "Years_to_Promotion": "晉升所需年數", "count": "性別總數", # 修正 Y 軸標籤 "Gender": "性別" }) # 顯示圖表 fig.show()</pre>	<ul style="list-style-type: none"> • 此程式碼使用 Plotly 來繪製 不同性別在各晉升年數上的人數分佈，以直方圖顯示，並區分性別。 • histfunc="count" 讓 Y 軸顯示該晉升年數的性別總數。 • barmode="group"：讓不同性別的數據分開顯示 (不堆疊)，方便比較。



小結

- 男性在前期晉升可能因人數原因上升，並到中間年數有男性較趨向於轉換跑道或離職，並在往後繼續招募男性新鮮人比例較高。
- 可能要再計算不同性別的平均晉升年數 來確定是否存在性別差異。
- 並進一步分析各性別在不同職位層級的分佈，以確保晉升公平性。

(3) 【社交網路分數】與【晉升機會】分析

程式碼實作與可視化圖表	程式碼解析
<pre>import plotly.express as px fig = px.violin(df, x="Years_to_Promotion", y="Networking_Score", box=True, # 顯示箱型數據 points="all", # 顯示所有數據點 title="不同晉升狀態下的 Networking Score 分佈", labels={"Years_to_Promotion": "第幾年獲得晉升", "Networking_Score": "社交網絡分數"}, color="Years_to_Promotion") fig.show()</pre>	<ul style="list-style-type: none"> 數據範圍較寬:說明某組的 Networking_Score 變異較大。 形狀較厚的部分:代表該區間人數較多 (更常見)。
<p>不同晉升狀態下的 Networking Score 分佈</p>	<ul style="list-style-type: none"> 第 1、2、4 年晉升者: Networking Score 分佈較廣，部分低分者仍能晉升，高分者可能更快獲得晉升。 第 3 年晉升者: 分佈較集中，顯示晉升標準較一致。 第 5 年晉升者: 變異較小，影響力較低，可能由資歷或績效決定晉升時機。

小結

- Networking Score 與晉升有一定關聯，但並不是唯一因素。
- 高 Networking Score 的員工，可能更早獲得晉升 (1~2 年晉升者中，有部分高分者)。
- 晉升時間較長 (5 年) 的員工，Networking Score 影響力較小，可能由其他因素決定晉升時機。

起薪 (Starting Salary) 分析

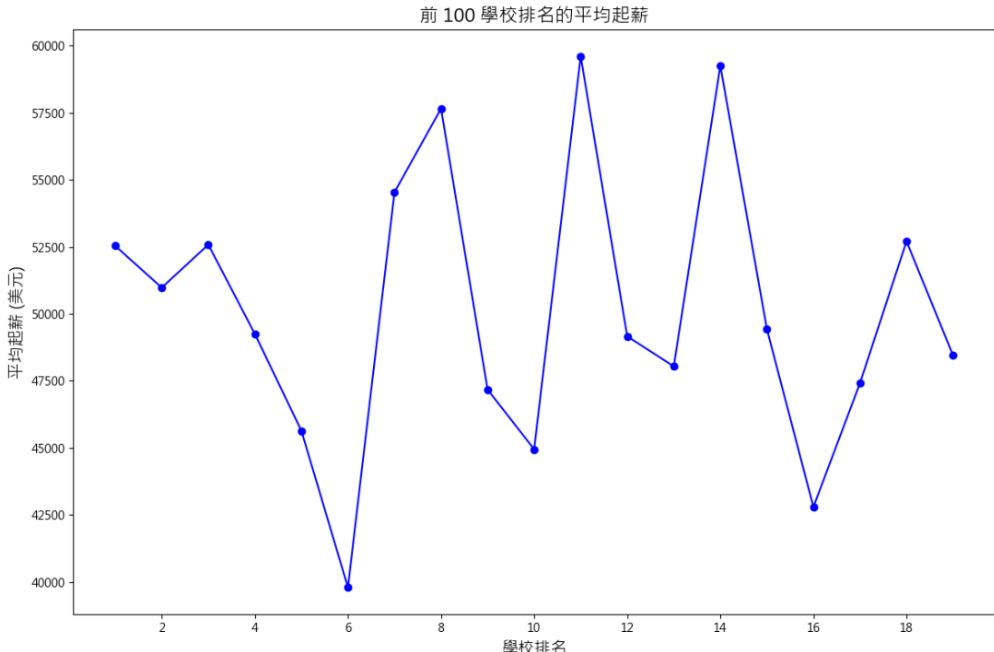
(1) 平均起薪性別比

程式碼實作與可視化圖表	程式碼解析								
<pre>import pandas as pd import matplotlib.pyplot as plt import numpy as np import seaborn as sns # -----# # 平均起薪性別比 # -----# df = pd.read_csv("education_career_success.csv") # -----# # 正規化資料將英文轉成中文 df['Gender'] = df['Gender'].replace({ 'Female': '女性', 'Male': '男性', 'Other': '其他' }) # -----# # 計算性別平均薪資 gender_avg_salary = df.groupby("Gender")["Starting_Salary"].mean().reset_index() plt.figure(figsize=(8, 6)) sns.barplot(data=gender_avg_salary, x="Gender", y="Starting_Salary", hue="Gender", legend=False, palette="Set2") # -----# # 加標題 & 標籤 plt.rcParams['font.family'] = 'Microsoft JhengHei' # 確保中文可以被輸出 plt.title("平均起薪性別比", fontsize=20) plt.xlabel("性別", fontsize=16) plt.ylabel("平均起薪", fontsize=16) # -----# # 顯示數值標籤 for i, v in enumerate(gender_avg_salary["Starting_Salary"]): plt.text(i, v + 500, f'{int(v):,}', ha='center', fontsize=12) plt.show()</pre>	<p>計算並繪製不同性別的平均起薪條形圖，並顯示數值標籤</p>								
 <table border="1"><thead><tr><th>性別</th><th>平均起薪</th></tr></thead><tbody><tr><td>其他</td><td>49,791</td></tr><tr><td>女性</td><td>50,536</td></tr><tr><td>男性</td><td>50,649</td></tr></tbody></table>	性別	平均起薪	其他	49,791	女性	50,536	男性	50,649	<p>A. 男性的最高起薪為 50649 年薪 (美元) B. 女性最高起薪為 50536 年薪(美元) C. 其他的平均年薪為 49791 年薪(美元)</p>
性別	平均起薪								
其他	49,791								
女性	50,536								
男性	50,649								

(2)各領域最高與最低起薪(年薪)

程式碼實作與可視化圖表	程式碼解析																								
<pre># 針對 Field_of_Study 找到每個領域的最高起薪 field_max_salary = df.groupby("Field_of_Study")["Starting_Salary"].max().reset_index() print(field_max_salary) # 針對 Field_of_Study 找到每個領域的最低起薪 field_min_salary = df.groupby("Field_of_Study")["Starting_Salary"].min().reset_index() print(field_min_salary) # ----- # 合併最高與最低起薪資料 merged = pd.merge(field_max_salary, field_min_salary, on="Field_of_Study", suffixes=("_max", "_min")) # ----- # 設定圖表大小 plt.figure(figsize=(14, 8)) # 畫出最高起薪與最低起薪的條形圖 bars_max = plt.barh(merged["Field_of_Study"], merged["Starting_Salary_max"], label="最高起薪", color='b', alpha=0.6) bars_min = plt.barh(merged["Field_of_Study"], merged["Starting_Salary_min"], label="最低起薪", color='r', alpha=0.6) # ----- # 在條形圖上顯示數值 for bar in bars_max: plt.text(bar.get_width(), bar.get_y() + bar.get_height()/2, f'{bar.get_width():.2f}', va='center', ha='left', color='blue', fontsize=12) for bar in bars_min: plt.text(bar.get_width(), bar.get_y() + bar.get_height()/2, f'{bar.get_width():.2f}', va='center', ha='left', color='white', fontsize=12) # 添加標題與標籤 plt.title("各領域的最高與最低起薪(年薪)", fontsize=20) plt.xlabel("起薪 (美元)", fontsize=16) plt.ylabel("領域", fontsize=16)</pre>	<ul style="list-style-type: none">計算並繪製各領域的最高與最低起薪(年薪)																								
<table><thead><tr><th>領域</th><th>最低起薪</th><th>最高起薪</th></tr></thead><tbody><tr><td>Medicine</td><td>25,000.00</td><td>90,400.00</td></tr><tr><td>Mathematics</td><td>25,000.00</td><td>89,900.00</td></tr><tr><td>Law</td><td>25,000.00</td><td>100,600.00</td></tr><tr><td>Engineering</td><td>25,000.00</td><td>98,200.00</td></tr><tr><td>Computer Science</td><td>25,000.00</td><td>90,800.00</td></tr><tr><td>Business</td><td>25,000.00</td><td>92,400.00</td></tr><tr><td>Arts</td><td>25,000.00</td><td>101,000.00</td></tr></tbody></table>	領域	最低起薪	最高起薪	Medicine	25,000.00	90,400.00	Mathematics	25,000.00	89,900.00	Law	25,000.00	100,600.00	Engineering	25,000.00	98,200.00	Computer Science	25,000.00	90,800.00	Business	25,000.00	92,400.00	Arts	25,000.00	101,000.00	<ul style="list-style-type: none">比較包含了製藥業，數學，法律，工程，電腦科學，商業與藝術等，不同領域的最高與最低起薪。最高的起薪是法律，其次則是藝術類領域。
領域	最低起薪	最高起薪																							
Medicine	25,000.00	90,400.00																							
Mathematics	25,000.00	89,900.00																							
Law	25,000.00	100,600.00																							
Engineering	25,000.00	98,200.00																							
Computer Science	25,000.00	90,800.00																							
Business	25,000.00	92,400.00																							
Arts	25,000.00	101,000.00																							

(3)校排名前 100 平均畢業起薪

程式碼實作與可視化圖表	程式碼解析																																								
<pre>df = pd.read_csv("education_career_success.csv") # 排序並取前100名學校 df_sorted = df.sort_values(by="University_Ranking") df_top_100 = df_sorted.head(100) # 計算每個排名的平均起薪 avg_salary_by_ranking = df_top_100.groupby('University_Ranking')['Starting_Salary'].mean().reset_index() # 設定圖表大小 plt.figure(figsize=(12, 8)) # 繪製線性圖，顯示每個學校排名的平均起薪 plt.plot(avg_salary_by_ranking['University_Ranking'], avg_salary_by_ranking['Starting_Salary'], marker='o', color='blue') # 添加標題與標籤 plt.title("前 100 學校排名的平均起薪", fontsize=16) plt.xlabel("學校排名", fontsize=14) plt.ylabel("平均起薪 (美元)", fontsize=14) # 設置 x 軸刻度為整數 plt.gca().xaxis.set_major_locator(MaxNLocator(integer=True)) # 顯示圖表 plt.tight_layout() plt.show()</pre>	<p>計算並繪製 前 100 校排 名的平均畢 業起薪（年 薪）</p>																																								
 <p>前 100 學校排名的平均起薪</p> <table border="1"> <thead> <tr> <th>學校排名</th> <th>平均起薪 (美元)</th> </tr> </thead> <tbody> <tr><td>1</td><td>52500</td></tr> <tr><td>2</td><td>51000</td></tr> <tr><td>3</td><td>52500</td></tr> <tr><td>4</td><td>49500</td></tr> <tr><td>5</td><td>45500</td></tr> <tr><td>6</td><td>40000</td></tr> <tr><td>7</td><td>54500</td></tr> <tr><td>8</td><td>57500</td></tr> <tr><td>9</td><td>47000</td></tr> <tr><td>10</td><td>45000</td></tr> <tr><td>11</td><td>59500</td></tr> <tr><td>12</td><td>49000</td></tr> <tr><td>13</td><td>48000</td></tr> <tr><td>14</td><td>59000</td></tr> <tr><td>15</td><td>49500</td></tr> <tr><td>16</td><td>43000</td></tr> <tr><td>17</td><td>47000</td></tr> <tr><td>18</td><td>52500</td></tr> <tr><td>19</td><td>48500</td></tr> </tbody> </table>	學校排名	平均起薪 (美元)	1	52500	2	51000	3	52500	4	49500	5	45500	6	40000	7	54500	8	57500	9	47000	10	45000	11	59500	12	49000	13	48000	14	59000	15	49500	16	43000	17	47000	18	52500	19	48500	<ul style="list-style-type: none"> • 可以看見校排名越前面，其畢業起薪不見得越高。 • 畢業起薪最高的是校排名約略 10-15。 • 畢業起薪最低則是校排名第 6。
學校排名	平均起薪 (美元)																																								
1	52500																																								
2	51000																																								
3	52500																																								
4	49500																																								
5	45500																																								
6	40000																																								
7	54500																																								
8	57500																																								
9	47000																																								
10	45000																																								
11	59500																																								
12	49000																																								
13	48000																																								
14	59000																																								
15	49500																																								
16	43000																																								
17	47000																																								
18	52500																																								
19	48500																																								

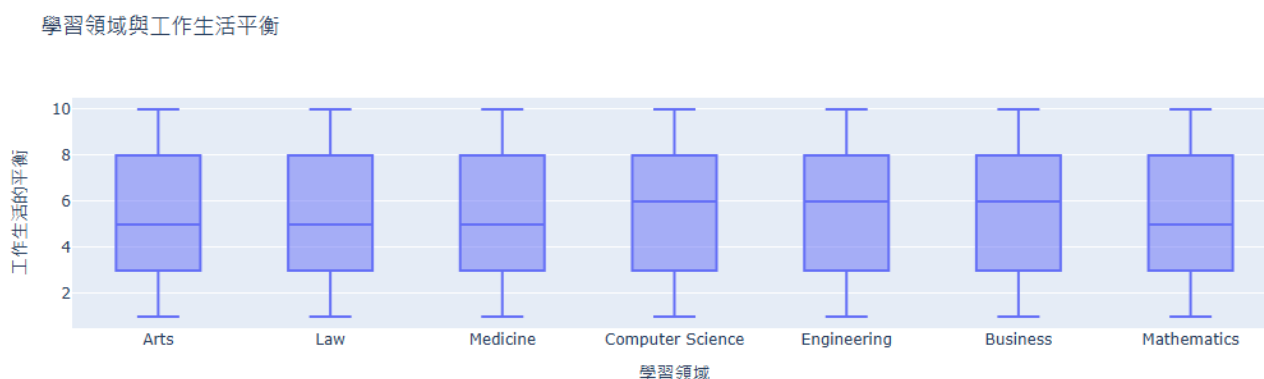
四.其他綜合分析

(1) 【Field_of_Study & Work_Life_Balance】

(探索學習領域給予未來的工作生活平衡的關聯性)

程式碼:

```
# 1. Field_of_Study & Work_Life_Balance (學習領域與工作生活平衡的關聯性)
fig1 = px.box(df, x='Field_of_Study', y='Work_Life_Balance', title='學習領域與工作生活平衡',
              labels={'Field_of_Study': '學習領域', 'Work_Life_Balance': '工作生活的平衡'})
fig1.show()
```



小結:

在中位數 1、中位數 3 基本都相差無異，但在中位數 2 中，電腦科學、工程、商業，有相較工作生活平衡上更好一些。

→學科領域選擇【影響】未來工作與生活平衡，具有未來影響性

(2) 【University_GPA & Job_Offers】

(成績與獲得的工作機會數關係)

程式碼:

```
# 2. University_GPA & Job_Offers (成績與獲得的工作機會數關係)
# 定義分類方式
def categorize_gpa(gpa):
    if gpa == 4.0:
        return '極好'
    elif 3.0 <= gpa < 4.0:
        return '一般'
    else:
        return '稍差'
# 對 GPA 進行分組
df['GPA_Category'] = df['University_GPA'].apply(categorize_gpa)
# 計算各 GPA 分組的人數
gpa_category_counts = df['GPA_Category'].value_counts()
print("各 GPA 分組的人數:")
print(gpa_category_counts)
# 計算 GPA 三組的工作機會數平均值
gpa_job_offers_avg = df.groupby('GPA_Category')['Job_Offers'].mean().reset_index()
fig2 = px.bar(gpa_job_offers_avg, x='GPA_Category', y='Job_Offers', title='不同 GPA 分組的平均工作機會數',
              labels={'GPA_Category': 'GPA 分類', 'Job_Offers': '平均工作機會數'}, category_orders={'GPA_Category': ['稍差', '一般', '極好']})
fig2.show()
```



小結:

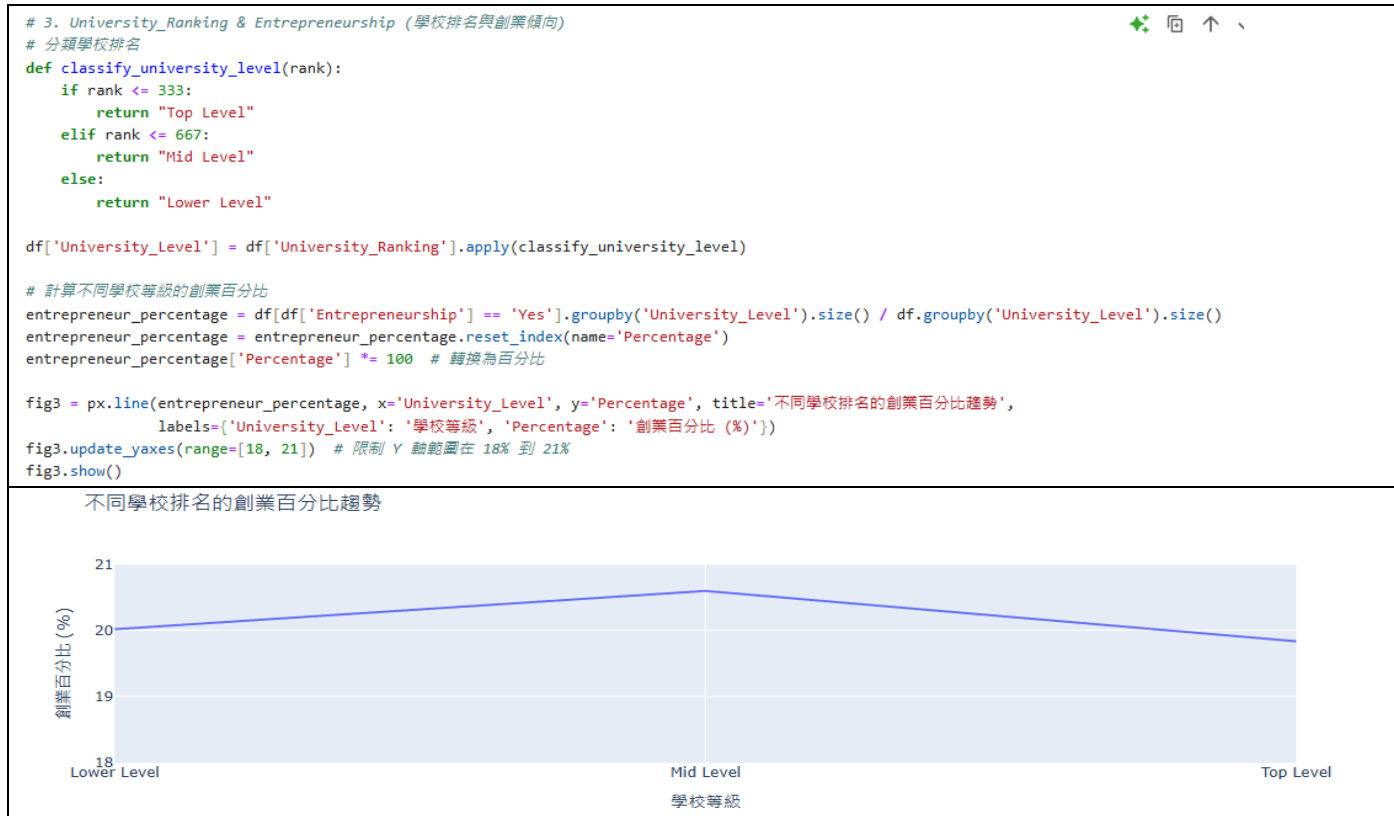
稍差(2.0-未滿 3.0)、一般(3.0-未滿 4.0)、極好(4.0)，一般(2.44)與稍差(2.53)並沒有太大的不一樣，甚至稍差大於一般的平均工作機會，但極好(2.85)有更多的工作機會是肯定的。

→GPA 成績【影響】獲得工作機會的數量，GPA 分數能提高面試官給予工作機會的機率

(3) 【University_Ranking & Entrepreneurship】

(是否位於更高優質學校的人，會更勇於創業)

程式碼:



小結:

在不同人數的學校等級中，透過百分比可以看到基本差異不大，其中 Mid Level 的學校為最高(20.6)，可見其影響不大。→不同學校等級【不影響】未來創業的數量，可見在不同等級學校，都可以有一個想創業的想法並實際行動

五.總結

本次我們透過數據分析，全面探討了學歷與職場發展的關聯，包括晉升機會、起薪差異及綜合表現。結果顯示，不同學歷背景在職場中的起點、成長速度與長期發展確實存在顯著差異，反映出學歷在職涯初期扮演著重要角色。然而，數據也揭示了學歷並非唯一的成功因素，個人能力、行業選擇及機遇同樣左右著最終的職場成就。

最後用一句話總結本次的心得「數字會說話」——它讓我們更理性地看待職涯發展，提供未來規劃與企業策略的重要參考。