

**ANL252**

**Python for Data Analytics**

**Tutor-Marked Assignment**

**July 2022 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **Lim Shu Han** | **B2082598** |

**Tutorial Group: ­­­­­­­­­­­­ T 09**

**Instructor’s Name: Dr. Kumar Munish**

**Submission Date: 16/08/2022**

**Declaration Statement:**

I declare that this assignment is my own work, unless otherwise acknowledged or credited by appropriate referencing. I have read and abide by the SUSS Honour Code and I am aware of the penalties associated with plagiarism and collusion listed in the SUSS Student Handbook.

**Question 1(a)**

**Chart 1** is a clustered bar chart depicted in Figure 1 below, showing employee count for the organisation by unit and further clustered by gender. The corresponding summarised table is in Table 1 below.

**Figure 1**

*Clustered bar chart showing employee count by unit/gender, created with Excel*



**Table 1**

*Summarised table for Figure 1*

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Gender** | |  |
| **Unit** | **F** | **M** | **Grand Total** |
| Admin | 4 | 3 | 7 |
| C-Level | 1 | 0 | 1 |
| Engineering | 5 | 5 | 10 |
| IT | 18 | 23 | 41 |
| Manufacturing | 105 | 64 | 169 |
| Sales | 11 | 11 | 22 |
| **Grand Total** | **144** | **106** | **250** |

Chart 1 above allows for comparison of gender ratios by unit and highlights a large gender disparity in the Manufacturing unit in favour of females. This contrasts with the general lack of women in the manufacturing industry (Kofsky & Lieberman, 2020), and could be attributable to the type of work performed or the category of products manufactured (fashion or feminine products.). The other units are almost equally gender diverse, with slightly more males than females in the IT unit. While it is possible to focus on hiring more male employees in Manufacturing and females in IT, the organisation can continue its generally diverse hiring practices across units.

There is also a stark difference in unit size across units, with the Admin, Engineering and sales units seemingly understaffed.

**Chart 2** is a 100% stacked bar chart depicted in Figure 2 below, showing the percentage proportion of each category of performance score by employees’ satisfaction scores. The corresponding summarised table is in Table 2 below.

**Figure 2**

*100% Stacked bar chart showing proportion of performance score by satisfaction score*



**Table 2**

*Summarised table for Figure 2*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Performance Score** | | | |  |
| **Satisfaction** | **Exceed** | **Meet** | **PIP** | **To Improve** | **Grand Total** |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 2 | 0 | 1 | 1 | 2 | 4 |
| 3 | 9 | 69 | 0 | 6 | 84 |
| 4 | 9 | 66 | 0 | 5 | 80 |
| 5 | 12 | 65 | 0 | 4 | 81 |
| **Grand Total** | **30** | **201** | **2** | **17** | **250** |

Chart 2 above indicates some positive correlation between employees’ satisfaction scores and performance scores. Those who exceeded performance expectations tend to have satisfaction scores above 3 and those with low satisfaction levels at 2 out of 5 scored poorly in performance. The overwhelming majority of employees have met performance expectations, indicating satisfactory organisation-wide performance.

**Question 1(b)**

**Chart 1** was re-created using Jupyterlab Notebook, resulting in Figure 3 below.

**Figure 3**

*Recreation of Chart 1 using Python*

Chart, bar chart

Description automatically generated

**Code used to create Figure 3:**

#General Libraries

import seaborn as sns

import matplotlib.pyplot as plt

import matplotlib.patches as mpatches

import pandas as pd

import numpy as np

import datetime as dt

#Indicate successful importing of libraries

print("Libraries imported.")

#Import and read csv Data file

DataFrame\_name = pd.read\_csv("/Users/aralynn/Downloads/TMA\_Data.csv")

# Define labels for clustered bar chart

labels = ['Admin', 'C-Level', 'Engineering', 'IT', 'Manufacturing', 'Sales']

adminM = 0

clevelM = 0

engineeringM = 0

itM = 0

manufacturingM = 0

salesM = 0

adminF = 0

clevelF = 0

engineeringF = 0

itF = 0

manufacturingF = 0

salesF = 0

# Create a for-loop to count the number of employees in each unit by gender

for counter in range(0, len(DataFrame\_name)):

if DataFrame\_name.loc[counter,'Gender'] == "M ":

if DataFrame\_name.loc[counter,'Unit'] == "Admin ":

adminM = adminM + 1

elif DataFrame\_name.loc[counter,'Unit'] == "C-Level":

clevelM = clevelM + 1

elif DataFrame\_name.loc[counter,'Unit'] == "Engineering":

engineeringM = engineeringM + 1

elif DataFrame\_name.loc[counter,'Unit'] == "IT":

itM = itM + 1

elif DataFrame\_name.loc[counter,'Unit'] == "Manufacturing":

manufacturingM = manufacturingM + 1

elif DataFrame\_name.loc[counter,'Unit'] == "Sales":

salesM = salesM + 1

elif DataFrame\_name.loc[counter,'Gender'] == "F":

if DataFrame\_name.loc[counter,'Unit'] == "Admin ":

adminF = adminF + 1

elif DataFrame\_name.loc[counter,'Unit'] == "C-Level":

clevelF = clevelF + 1

elif DataFrame\_name.loc[counter,'Unit'] == "Engineering":

engineeringF = engineeringF + 1

elif DataFrame\_name.loc[counter,'Unit'] == "IT":

itF = itF + 1

elif DataFrame\_name.loc[counter,'Unit'] == "Manufacturing":

manufacturingF = manufacturingF + 1

elif DataFrame\_name.loc[counter,'Unit'] == "Sales":

salesF = salesF + 1

m\_count = [adminM, clevelM, engineeringM, itM, manufacturingM, salesM]

f\_count = [adminF, clevelF, engineeringF, itF, manufacturingF, salesF]

# Label locations

x = np.arange(len(labels))

# Set bar width

width = 0.35

fig, ax = plt.subplots()

rects1 = ax.bar(x - width/2, m\_count, width, label='M')

rects2 = ax.bar(x + width/2, f\_count, width, label='F')

# Define labels, title and other bar chart parameters

ax.set\_ylabel('Count')

ax.set\_xlabel(‘Unit’)

ax.set\_title('Employee Count by Unit and Gender')

ax.set\_xticks(x, labels)

ax.legend()

ax.bar\_label(rects1, padding=3)

ax.bar\_label(rects2, padding=3)

fig.tight\_layout()

plt.show()

**Chart 2** was also re-created using Jupyterlab Notebook, resulting in Figure 4 below. Its corresponding summary table is shown in Figure 5 below.

**Figure 4**

*Recreation of Chart 2 using Python*

Chart, bar chart

Description automatically generated

**Figure 5**

*Screenshot showing corresponding summary table for Figure 4*

Text

Description automatically generated with medium confidence

**Code used to create Figure 4:**

# Define X and Y variables

x\_var, y\_var = "Satisfaction", "PerformanceScore"

# Group data in Y by X, count entries and normalise entries

DataFrame\_name\_grouped = DataFrame\_name.groupby(x\_var)[y\_var].value\_counts(normalize=True).unstack(y\_var)

# Using panda function to plot data in a stacked bar chart

DataFrame\_name\_grouped.plot.barh(stacked=True)

# Plot legend box

plt.legend(

bbox\_to\_anchor=(0.5, 1.03),

loc="lower center",

borderaxespad=0,

frameon=True,

ncol=50,

)

# Create a for loop to print data text labels on chart

for ix, row in DataFrame\_name\_grouped.reset\_index(drop=True).iterrows():

print(ix, row)

cumulative = 0

for element in row:

if not pd.isna(element):

plt.text(

cumulative + element / 2,

ix,

f"{round(float(element \* 100),1)} %",

va="center",

ha="center",

)

cumulative += element

# Centralise the chart

plt.tight\_layout()

**Question 1(c)**

# Convert Object Datatype to Datetime datatype

DataFrame\_name['JoinDate'] = pd.to\_datetime(DataFrame\_name['JoinDate'])

DataFrame\_name['LeftDate'] = pd.to\_datetime(DataFrame\_name['LeftDate'])

# Create variable and assign the Datetime value of 1st May 2022

date1 = dt.datetime.strptime('2022/5/1', "%Y/%m/%d")

# Create variable of minimum, maximum and average length of service

minLength = 0

maxLength = 0

totalLength = 0

# For loop to go through every row in the dataframe

for counter in range(0, len(DataFrame\_name)):

# Create variables and assign the Staff name, join date and left date from the TMA\_Data.csv data.

name = DataFrame\_name.loc[counter,"Staff"]

joinDate = DataFrame\_name.loc[counter,"JoinDate"]

leftDate = DataFrame\_name.loc[counter,"LeftDate"]

# If-else condition to find which staff leftdate that is empty

if pd.isna(leftDate):

# subtract the staff joindate from provided leftdate (1st May 2022), to 1 decimal place

lengthOfService = round((date1 - joinDate).days / 365, 1)

else:

# subtract the joindate from leftdate, to 1 decimal place

lengthOfService = round((leftDate - joinDate).days / 365, 1)

# Assign the first staff length of service to minLength

if minLength == 0:

minLength = lengthOfService

# If the next staff length of service is shorter then the previous, assign it to minLength

elif minLength > lengthOfService:

minLength = lengthOfService

# If the next staff length of service is longer then the previous, assign it to maxLength

if maxLength < lengthOfService:

maxLength = lengthOfService

# Retrieve the total lengthOfService

totalLength = totalLength + lengthOfService

# Print the staff name and length of service

print(f"Name: {name} \nLength Of Service: {lengthOfService} Years\n")

# Divide the total length of service by total number of staff and round to the nearest 1 decimal place.

averageLength = round(totalLength/len(DataFrame\_name),1)

# Print the minimum, maximum and average length of service

print(f"Minimum length of service: {minLength} Years")

print(f"Maximum length of service: {maxLength} Years")

print(f"Average length of service: {averageLength} Years")

**Question 1(d)**

# Create a boolean 'True' variable

loop = True

# Create a User-defined function to search for the staff name in the "TMA\_Data.csv" data

def name\_search (staffName):

#For loop to go through every row in the "TMA\_Data.csv" data

for counter in range(0, len(DataFrame\_name)):

#If the staff name is found in the "TMA\_Data.csv" data, return true.

#Using .lower() to change the string to lower case due to case sensitivity.

if DataFrame\_name.loc[counter,'Staff'].lower() == staffName.lower():

return True

#If the staff name is not found in the "TMA\_Data.csv" data, return false

return False

# Create a while loop to allow user to continue another query. The loop terminates when the variable 'loop' is assigned with false.

while loop:

# Create a boolean 'True' variable

query = True

# User to input the staff name

name = input("Please enter the staff name: ")

# Using the User-defined function as the if-else condition

if name\_search(name) == True:

print(f"Yes. {name} is a staff of this organization.")

else:

print("Staff not found.")

# Using a while loop for user to choose either to continue or exit.

while query:

# User to input choice to continue or exit

a = input("Continue Y/N? ")

# Condition for continuing the loop

if a.lower() == "y":

query = False

# If user inputs 'N', the the variable 'query' and 'loop' will change to false thus exiting the while loop and the program ends.

elif a.lower() == "n":

query = False

loop = False

# If user input is other than 'Y' or 'N', the loop continues and prints "invalid selection".

else:

print("Invalid selection.")

**References**

Kofsky, A., & Lieberman, R. (8 December, 2020). *The Gender Gap: Women Fill Only 33% of Manufacturing Industry Jobs, 26% of Industry Leadership Positions*. Retrieved from Business Wire: https://www.businesswire.com/news/home/20201208005232/en/The-Gender-Gap-Women-Fill-Only-33-of-Manufacturing-Industry-Jobs-26-of-Industry-Leadership-Positions