**ANL252**

**Python for Data Analytics**

**TMA**

**Submitted by:**

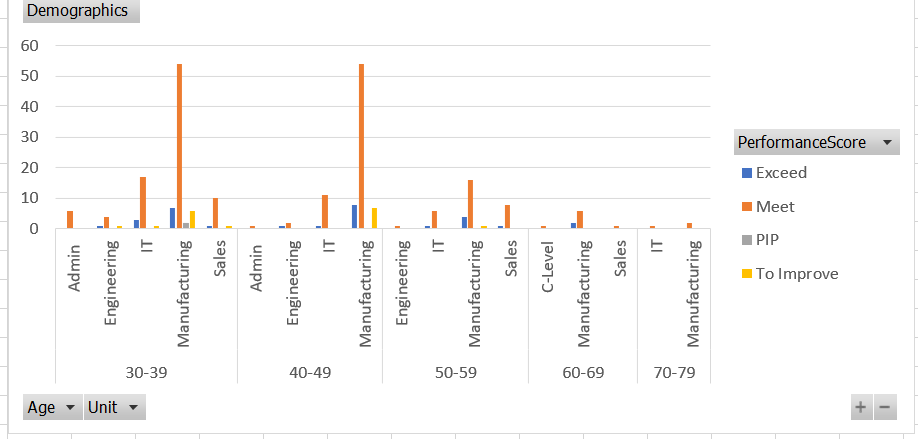
|  |  |
| --- | --- |
| **Name** | **PI No.** |
| Gan Ke Yi | B1972108 |

**Tutorial Group: ­­­­­­­­­­­­** T09

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

**Submission Date:** 14/08/2022

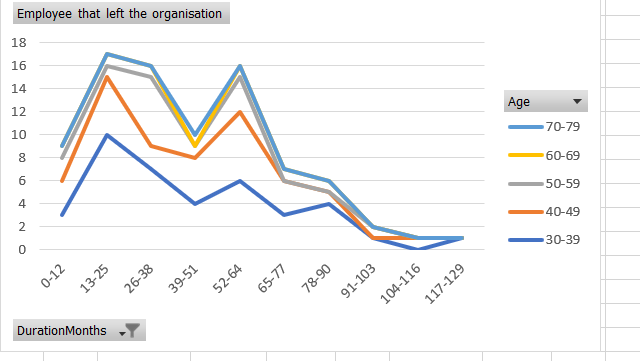
1. Formatted birth year, joindate and left date as a some data are formatted as mm/dd/yyyy. To make all date data consistent I have formatted it to dd/mm/yyyy.



In the first chart I would be giving insights on the existing employees that are still in the firm. Analysing on their age by unit by performance score. From the chart above we can observe the following:

1. Among the 5 units, manufacturing has the most number of people
2. Among the 5 groups of age group, the firm have the greatest number of employees in the age range 30 to 39. They also have the greatest number of people exceeding their performance score
3. Manufacturing units have the highest number of people expecting their performance as compared to the other units

In our second chart, I will be giving insights of the people whom left the organisation, this chart is as important to the firm as it allows the firm to understand what kind of trend / attributes do they have to look at to possibly find out on ways to retain people.



Analysing on their age by duration they were in the organisation. From the chart above we can observe the following:

1. Downward trend that employee that stayed in the organisation after 78-90 months will not leave the firm.
2. Upward trend of employee that are in the organisation in 13-25 months for all age group however for people that leave the organisation at 52-64 months there is a sharp increase and the majority seem to be people age 40 and above.
3. Overall, the most number people who left the organisation are mainly in age range 70-79. However, it should not be ignored that people aged 50 – 69 is following the trend very closely to people in 70-79. Hence it can be concluded that majority of the people that are leaving are from 50-79.

b) #importing the required libraries to handle the data

import pandas as pd

#reading the excel file

df=pd.read\_excel('/content/tma.xlsx')

df=df.dropna()

df['DurationMonths'] = df['DurationMonths'].apply(lambda x: int(x))

#function to sort values to group later

dfs = df.sort\_values('DurationMonths')

dfs = dfs[['DurationMonths','Age']]

dfs

#group duration\_months in to 12 each

duration\_months = []

cnt = 0

while(cnt<=117):

duration\_months.append(f'{cnt}-{cnt+12}')

cnt=cnt+13

duration\_months

df\_ = []

for duration\_month in duration\_months:

l , u = duration\_month.split('-')

l , u = int(l) , int(u)

df\_.append(dfs[(dfs['DurationMonths'] >= l) & (dfs['DurationMonths']<=u)])

age\_ = []

for d in df\_:

age = []

age.append(len(d[(d['Age']>=30) & (d['Age']<=39)]))

age.append(len(d[(d['Age']>=40) & (d['Age']<=49)]))

age.append(len(d[(d['Age']>=50) & (d['Age']<=59)]))

age.append(len(d[(d['Age']>=60) & (d['Age']<=69)]))

age.append(len(d[(d['Age']>=70) & (d['Age']<=79)]))

age\_.append(age)

df\_n = pd.DataFrame(columns = ['DMR','AR','AR\_CNT'])

df\_n['DMR'] = ['0-12',

'0-12',

'0-12',

'0-12',

'0-12',

'13-25',

'13-25',

'13-25',

'13-25',

'13-25',

'26-38',

'26-38',

'26-38',

'26-38',

'26-38',

'39-51',

'39-51',

'39-51',

'39-51',

'39-51',

'52-64',

'52-64',

'52-64',

'52-64',

'52-64',

'65-77',

'65-77',

'65-77',

'65-77',

'65-77',

'78-90',

'78-90',

'78-90',

'78-90',

'78-90',

'91-103',

'91-103',

'91-103',

'91-103',

'91-103',

'104-116',

'104-116',

'104-116',

'104-116',

'104-116',

'117-129',

'117-129',

'117-129',

'117-129',

'117-129',]

df\_n['AR'] = ['30-39','40-49','50-59','60-69','70-79']\*len(duration\_months)

df\_n['AR\_CNT'] = sum(age\_, [])

import seaborn as sns sns.lineplot(x='DMR',y='AR\_CNT',hue='AR',ci=None,data=df\_n)

##reading the excel file

df1=pd.read\_excel('/content/tma.xlsx')

#table headers

df1 = df1[['Unit','Age','PerformanceScore']]

df1['Age'] = df1['Age'].mask(lambda x : (x>=30) & (x<=39),0)

df1['Age'] = df1['Age'].mask(lambda x : (x>=40) & (x<=49),1)

df1['Age'] = df1['Age'].mask(lambda x : (x>=50) & (x<=59),2)

df1['Age'] = df1['Age'].mask(lambda x : (x>=60) & (x<=69),3)

df1['Age'] = df1['Age'].mask(lambda x : (x>=70) & (x<=79),4)

df1=df1.sort\_values(by = ['Age','Unit','PerformanceScore']).reset\_index()

#function to allocate the age into the "Age" containers

s = pd.DataFrame(df1[['Age','Unit','PerformanceScore']].value\_counts()).sort\_values('Age').reset\_index()

for index,age in enumerate(s['Age']):

if age == 0:

s['Age'][index] = '30-39'

elif age == 1:

s['Age'][index] = '40-49'

elif age == 2:

s['Age'][index] = '50-59'

elif age == 3:

s['Age'][index] = '60-69'

elif age == 4:

s['Age'][index] = '70-79'

s

#importing the required libraries to handle the data

import seaborn as sns

#plot graph

sns.set(rc = {'figure.figsize':(15,8)})

sns.histplot(data=s, x="Age", hue="PerformanceScore", y = 0,multiple="stack",binwidth=1)

c) # importing the required libraries to handle the data

import pandas as pd

from datetime import date

# function to get the difference between two date objects in days

def getDiffInDays(date1, date2):

return (date2-date1).days

# function to search whether the given username exist in the passed dataframe or not

def searchStaff(df, name):

# iterating over complete staff record to search the person in record

for name in df["Staff"]:

# if found then return True

if staffName.lower() == name.lower():

return True

# if the whole dataframe is iterated and no match of the given name found then return False

return False

# reading the csv file into the dataframe

df = pd.read\_csv("Data.csv")

# filling the empty leftDate with the given default date of 1s May 2022

df["LeftDate"] = df["LeftDate"].fillna("05/01/2022")

# variables to store the minimum, maximum and average service length in days and also the count of the staff in the file

minDays = 0

maxDays = 0

averageDays = 0

numStaffCount = df.shape[0]

# iterating over each record of the dataframe

for i in range(numStaffCount):

# extracting the joindate and leftdate in the string list format

date1 = df["JoinDate"][i].split('/')

date2 = df["LeftDate"][i].split('/')

# creating date objects from the extracted data details

date1 = date(int(date1[2]), int(date1[0]), int(date1[1]))

date2 = date(int(date2[2]), int(date2[0]), int(date2[1]))

# calculating the difference between join and left date using the getDiffInDays function

diff = getDiffInDays(date1, date2)

# if minDays is more than current difference or its zero then assign minDays with current difference

if minDays == 0 or minDays > diff:

minDays = diff

# if maxDays is less than current difference then assign maxDays with current difference

if maxDays < diff:

maxDays = diff

# adding the difference to the averageDays variable for first calculating the sum of length of services

averageDays += diff

# calculating the average number of days of service in the organization

averageDays /= numStaffCount

# converting the min, max and average days to year by rounding it to the 1 decimal place

maxLenOfService = round(maxDays/365, 1)

minLenOfService = round(minDays/365, 1)

avgLenOfService = round(averageDays/365, 1)

# printing the max, min and average lengths of service as required

print("Minimum length of service:",minLenOfService)

print("Maximum length of service:",maxLenOfService)

print("Average length of service:",avgLenOfService)

print()

d)

staffName = ""

# using while loop to iterate until user chooses to end querying

while True:

# asking the user to enter the name of the user or to enter the exit command

staffName = input("Enter the full name of person to search it in organization's record or 'exit' to end querrying: ")

# checking if the user has entered the exit command or not, if it is entered then breaking the loop and ending the code

if staffName.lower() == "exit":

break

# if exit command is not entered then searching for the entered name and if its found then printing the required message that it can be found

# and if not found then also printing the message to indicate it cannot be found

if searchStaff(df, staffName) == True:

print(staffName+" was/is a staff of the organization. So it can be found in organization's records.")

else:

print(staffName+" was/is not a staff of the organization. So it cannot be found in organization's records.")

print()

# printing the thank you message at the end of the program

print("Thank you for using the system!!")