BUS

**ANL252**

**PYTHON FOR DATA ANALYTICS**

# **End-of-Course Assessment**

**July Semester 2022**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **TAN BOON MAY (CHEN WENMEI)** | **W2210814** |

**Submission Date: 05/09/2022**

**Question 1 List the categorial and numeric variables in this dataset.**

|  |  |
| --- | --- |
| In | #Import data analysis modules  import pandas as pd  import numpy as np  #Read the csv file & displaying the top 5 rows  data = pd.read\_csv(r'C:\Users\user\OneDrive - SUSS University\SUSS\ANL252\ECA\_data (1).csv')  data.head() |
| Out |  |
| In | #Displaying the bottom 5 rows  data.tail() |
| Out |  |
| In | #Basic info summary  data.info()  # As shown below:- total no of entries = 18769  # Columns "Education" & "Marital" have lesser entries |
| Out |  |
| In | ##Additional step to reassure the null values and to obtain the shortfall for columns "Education" & "Marital"  data.isnull().sum() |
|  |  |
| In | ##As shown in the info summary, total entries = 18769  #But when printed out the bottom 5 entries, the ID reference does not match the total number of entries  # Possible existence of duplicated ID reference nos  print(data['ID'].value\_counts())  # ID reference "132", "420" & "378" have duplicated.  #ID reference supposed to be unique to individual and only the concerned company can rectify this bug.  # Moreover, only 3 IDs affected and the ID references are not significant info to the analysis process |
| Out |  |
|  | **Conlusions for Question 1**  During the importation & basic evaluation of data process, it was observed that :-   1. Columns "Education" & "Marital" have lesser than the full 18769 entries. The differences are 13 nos (0.07%) & 38 nos (0.3%) respectively. Due to the low percentage (both less than 1 percent), these variables should not cause any significant impacts to the analysis. Have opted to replace the variables with the computed median values. The values are lying at the midpoint of a frequency distribution of observed values, such that there is an equal probability of falling above or below 2. ID reference "132", "420" & "378" are duplicated. ID is supposed to be unique to individual hence,it is likely due to technical glitch. The affected IDs are only 3 out of 18769, thus, this error also should not cause any significant impacts to the analysis. |

**Question 2\_Conduct four (4) data pre-processing tasks for the analysis of the data,**

**explaining results obtained**

|  |  |
| --- | --- |
| In | #As detected earlier, there are 13 'null' records under "EDUCATION" & 38 under "MARITAL"  #Hence have used the Python's computed median values to replace these null values.  data\_filled = data.fillna({"EDUCATION":data["EDUCATION"].median(),"MARITAL":data["MARITAL"].median()})  data\_filled.info() |
| Out |  |
| In | #To remove noncritical columns i.e "S1 to S5", "B1 to B5" & "R1 to R5"  #"S1 to S5" can be represented under "Rating" column, "B1 to B5" & R1 to R5" can be deduced from "LIMIT" and "BALANCE" columns.  data.drop(['S1','S2','S3','S4','S5','B1','B2','B3','B4','B5','R1','R2','R3','R4','R5'], axis =1, inplace = True)  data.head() |
| Out |  |
| In | #To find out the age range  data.sort\_values(by='AGE')  #Age ‘-1’ & Age ‘199’ will be treated as typo errors. |
| Out |  |
| In | #To extract all the illogical age i.e ‘-1’ & ‘199’  print(data['AGE'].value\_counts())  It was found that a total 10 records with typo errors |
| Out |  |
| In | #Age '29' has been used to replace the illogical age at ''-1' and ''-199', as 29 has accorded the highest frequency in this dataset.  data\_cleanedup\_age = data.replace (to\_replace=[-1, 199], value=29)  data\_cleanedup\_age.head() |
| Out |  |
| In | #To check if the replacements were done successfully  data\_cleanedup\_age.sort\_values(by='AGE') |
| Out |  |
| In | #To compute the amount of credit used  data\_cleanedup\_age['CREDIT USED'] = data\_cleanedup\_age['LIMIT'] - data\_cleanedup\_age['BALANCE']  data\_cleanedup\_age |
| Out |  |
| In | #To compute the percentage overdrawn based on income  data\_cleanedup\_age['CREDIT USED / INCOME ''(%)'] = data\_cleanedup\_age['CREDIT USED'] / data\_cleanedup\_age['INCOME'] \* 100  data\_cleanedup\_age |
| Out |  |
| In | ### Use discretisation to categorise the age group  bins = [20, 30, 40, 50, 60, 70, 120]  labels = ['20-29', '30-39', '40-49', '50-59', '60-69', '70+']  data\_cleanedup\_age['AGE'] = pd.cut(data\_cleanedup\_age["AGE"], bins, labels = labels)  data\_cleanedup\_age.head() |
| Out |  |
| In | #Preparation for Graph 1, to compute the counts for the desinated age group range  data\_graph1= data\_cleanedup\_age['AGE']  print(data\_graph1.value\_counts()) |
| Out |  |
| In | #To find out the percentage range under "CREDIT USED / INCOME"  data\_cleanedup\_age.sort\_values(by='CREDIT USED / INCOME ''(%)') |
| Out |  |
| In | #To check the Dtypes  data\_cleanedup\_age.sort\_values(by='CREDIT USED / INCOME ''(%)')  data\_cleanedup\_age.info() |
| Out |  |
| In | # converting “BALANCE’ & 'CREDIT USED / INCOME' from float to int  data\_cleanedup\_age['CREDIT USED / INCOME ''(%)'] = data\_cleanedup\_age['CREDIT USED / INCOME ''(%)'].apply(np.int64)  data\_cleanedup\_age['CREDIT USED'] = data\_cleanedup\_age['CREDIT USED'].apply(np.int64)  data\_cleanedup\_age['BALANCE'] = data\_cleanedup\_age['BALANCE'].apply(np.int64)  display (data\_cleanedup\_age.dtypes) |
| Out |  |
| In | #The results after the conversion from float to int.  data\_cleanedup\_age |
| Out |  |
| In | #Preparation for Graph 2, to extract “RATING’ & ‘CREDIT USED / INCOME’ for data analysis  data\_graph2 = data\_cleanedup\_age.iloc[:, [4,10]]  data\_graph2 |
| Out |  |
| In | #To sort the ‘RATING’ data in accending order  data\_graph2.sort\_values(by='RATING') |
| Out |  |
| In | #To establish the counts in ‘RATING’  data\_graph2\_rating = data\_graph2['RATING'].value\_counts()  print(data\_graph2\_rating) |
| Out |  |
| In | #Preparation for Graph 3, to establish the counts of “CREDIT USED / INCOME”  data\_graph3= data\_cleanedup\_age['CREDIT USED / INCOME (%)'].value\_counts()  print(data\_graph3) |
| Out |  |
| In | #Preparation for Graph 4, to establish age group range based on customers' INCOME  data\_graph4 = data\_cleanedup\_age.iloc[:, [3,8]]  data\_graph4 |
| Out |  |
| In | #To specifically pull out the dayta of cusomters @ Age ‘70+’  data\_graph4.loc[data\_graph4['AGE'] == '70+']  data\_G4\_70 = data\_graph4.loc[data\_graph4['AGE'] == '70+']  data\_G4\_70  #These 10 records are kind of dubious. Customers at agr 70+ most likely are retired but these 10 customers are still earning very substantial incomes every month. Based on my own assumptions, there are a few possibilities.:-   1. These customers have been engaging this company for decades. The profiles have not been updated since their joined date. 2. The age was input wrongly |
| Out |  |
| In | #Preparation for Graph 5, to establish the counts for credit balance records  data\_graph5= data\_cleanedup\_age['BALANCE']  print(data\_graph5.value\_counts()) |
| Out |  |
|  | **Question 3\_Articulate five (5) relevant insights of the data, with supporting visualization for each insight.** |
| In | data\_graph1= data\_cleanedup\_age['AGE']  print(data\_graph1.value\_counts()) |
| Out |  |
| In | import matplotlib as mpl  import matplotlib.pyplot as plt  print ('Matplotlib version: ', mpl.\_\_version\_\_) # >= 2.0.0  # creating the dataset to demonstrate customers age profile  data\_G1 = {'20-20':6813, '30-39':6756, '40-49':3800,'50-59':1237,'60-69':153,'70+': 10}  AGE = list(data\_G1.keys())  CUSTOMERS = list(data\_G1.values())  fig = plt.figure(figsize = (10, 10))    # creating the bar plot  plt.bar(AGE, CUSTOMERS, color ='lightblue',  width = 0.4)    plt.xlabel("AGE")  plt.ylabel("CUSTOMERS")  plt.title("CUSTOMERS AGE PROFILE")  plt.show()  #graph1 has demontrated the top age range of existing customers are ranging from age 20 to 29 years old. Infact range 20 to 29 is the youngest tier followed by the 2nd  Tier, age ranging 30 to 39. |
| Out |  |
| In | #Graph 2  data\_graph2\_rating = data\_graph2['RATING'].value\_counts()  print(data\_graph2\_rating) |
| Out |  |
| In | # creating the dataset to demonstrate customers’ ratings  data\_G2 = {'0=GOOD':14654, '1=BAD':4115}  RATING = list(data\_G2.keys())  CUSTOMERS = list(data\_G2.values())  fig = plt.figure(figsize = (10, 10))    # creating the bar plot  plt.bar(RATING, CUSTOMERS, color ='lightgreen',  width = 0.4)    plt.xlabel("RATING")  plt.ylabel("CUSTOMERS")  plt.title("CUSTOMERS'RATINGS")  plt.show()  #graph2 has releaved that the credit ratings of the customers are still acceptable @ 28% with bad ratings |
| Out |  |
| In | #Graph 3 demonstrate the Top 10 percentage of customers credit drawn vs income  data\_graph3= data\_cleanedup\_age['CREDIT USED / INCOME (%)'].value\_counts()  data\_graph3.head(10)  #graph 3 is reporting that majority of the customers have used up 83% to 94% of credit using their income as benchmark. |
| Out |  |
| In | # creating the dataset to demonstrate the percentage overdrawn take reference from the income  data\_G3 = {'86':761, '84':757, '85': 705,'83':703,'87':679,'88': 669, '90':614, '89':602, '91': 587,'94':552,}  CREDIT = list(data\_G3.keys())  CUSTOMERS = list(data\_G3.values())  fig = plt.figure(figsize = (10, 10))    # creating the bar plot  plt.bar(CREDIT, CUSTOMERS, color ='lightblue',  width = 0.4)    plt.xlabel("CREDIT USED vs INCOME")  plt.ylabel("NO OF CUSTOMERS")  plt.title("TOP 10 CREDIT USED vs INCOME RECORDS")  plt.show() |
| Out |  |
| In | #Graph 4  data\_graph4.loc[data\_graph4['AGE'] == '70+']  data\_G4\_70 = data\_graph4.loc[data\_graph4['AGE'] == '70+']  data\_G4\_70.sort\_values(by='INCOME')  data\_G4\_sorted = data\_G4\_70.sort\_values(by='INCOME')  data\_G4\_sorted  #Age 70+ was specially extracted as the 10 counts of 70+ customers still fetching substantial income. Using logical anlysis  #there are 2 possibilities. 1) the age was recorded wrongly 2) these customers have estoblished more than a decade long  #relationship with the company. Their retirement status has yet to be updated in the company's system. |
| Out |  |
| In | data\_G4\_70.shape |
| Out |  |
| In | data\_G4\_sorted .plot.scatter(x='AGE', y='INCOME', title='INCOME RANGE OF 70+ CUSTOMERS'); |
| Out |  |
| In | #Graph 5  data\_graph5= data\_cleanedup\_age['BALANCE'].value\_counts()  data\_graph5.head(5) |
| Out |  |
| In | # creating the dataset  data\_G5 = {'0':1578, '68':166, '57': 50,'437':46,'136':42,}  BALANCECREDIT = list(data\_G5.keys())  CUSTOMERS = list(data\_G5.values())  fig = plt.figure(figsize = (10, 10))    # creating the bar plot  plt.bar(BALANCECREDIT, CUSTOMERS, color ='lightpink',  width = 0.4)    plt.xlabel("CUSTOMERS BALANCE CREDIT")  plt.ylabel("NO OF CUSTOMERS")  plt.title("TOP 5 BALANCE CREDIT AMT")  plt.show()  #graph 5 is plotted to determine the credit balance of majority customers. Even though $0 credit balance has accorded the highest record, but the actual figure is only @ 1578. It is really not that significant as compared to the entire customers base of 18769 (i.e 8.4%) |
| Out |  |
|  | **Conclusions:-**   * **The 5 plotted graphs are focusing more on evluation of existing customer profile and to detect any irregularities** * **Generally the existing customer pool is considered healhty as :-**  1. **Big majority is young** 2. **Big majority with good credit rating** 3. **Credit used by big majority did not exceed their income** 4. **Only 10 cases of considered dubious profile (i.e 70+ customers )** 5. **Percentage of customers with $0 credit is not high** |
|  | **Question 4\_Perform linear regression modelling to predict the variable, B1, explaining the approach taken, including any further data pre-processing.** |
| In | #Import data analysis modules  import pandas as pd  data = pd.read\_csv(r'C:\Users\user\OneDrive - SUSS University\SUSS\ANL252\ECA\_data (1).csv')  data.head() |
| Out |  |
| In | #Checing the shape of the data  data.shape |
| Out |  |
| In | #Extract 'BALANCE' & 'B1' for further analysis  data\_B1\_BALANCE = data.iloc[:, [2,14]]  data\_B1\_BALANCE  #Trying to establish some spending patterns or habits of customers |
| Out |  |
| In | #Checing the shape of the data  Data\_B1\_BALANCE.shape |
| Out |  |
| In | #Trying to detect linear relationship through Scatterplots  data\_B1\_BALANCE.plot.scatter(x='BALANCE', y='B1', title='Scatterplot of BALANCE VS B1');  #The graph has demonstrated that it is ikely to have good correlation between the as there is an obvious stright line. |
| Out |  |
| In | #Using 'correlation' analysis module to establish the level of correlationship  print(data\_B1\_BALANCE.corr())  #as per result, the percentage of correlatship is high at 95%.. it is a positive linear correlation between ‘B1’ & ‘BALANCE’variables with results more than 80%. |
| Out |  |
| In | #Using 'describe' analysis module to obtain a summery for analysis (eg, mean, maximum, minimum values of each columns  print(data\_B1\_BALANCE.describe()) |
| Out |  |
| In | #To establish Linear Regression with Sklearn  y = data\_B1\_BALANCE['B1'].values.reshape(-1, 1)  X = data\_B1\_BALANCE['BALANCE'].values.reshape(-1, 1)  print(data\_B1\_BALANCE['BALANCE'].values)  print(data\_B1\_BALANCE['BALANCE'].values.shape) |
| Out |  |
| In | #To divide the data 2 arrays, to predict the B1 result depending on the BALANCE value selected.  From sklearn.model\_selection import train\_test\_split  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2)  SEED = 0  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = SEED)  print(X\_train)  print(y\_train) |
| Out |  |
| In | from sklearn.linear\_model import LinearRegression  regressor = LinearRegression()  regressor.fit(X\_train, y\_train) |
| Out |  |
| In | print(regressor.intercept\_) |
| Out |  |
| In | print(regressor.coef\_) |
| Out |  |
| In | #B1 = 5.26352176\*BALANCE + 2003.94307465  #Assumeing "BALANCE" = 0  #B1 = 5.26352176\* 0 + 2185.36465466  #B1 = 2185.36465466  def calc(slope, intercept,’ BALANCE’):  return slope\*’ BALANCE’+intercept  B1 = calc(regressor.coef\_, regressor.intercept\_, 9.5)  print(B1) |
| Out |  |
| In | B1 = regressor.predict([[9.5]])  print(B1) |
| Out |  |
| In | y\_pred = regressor.predict(X\_test)  data\_B1\_BALANCE\_preds = pd.DataFrame({'Actual': y\_test.squeeze(), 'Predicted': y\_pred.squeeze()})  print(data\_B1\_BALANCE\_preds) |
| Out |  |
|  | **Question 5**  State the linear regression equation and explain key insights from the results obtained in Question 4. |
|  | Linear Regression equation = y = ax + b  \*\* y = B1\_Billable amt on the 1st month  a = slope of the line  x = Balance credit available  b = Y-axis intercept  Thus,  #B1 = 5.26352176\*BALANCE + 2185.36465466  #Assumeing "BALANCE" = 0  #B1 = 5.26352176\* 0 + 2185.36465466  #B1 = 2185.36465466  After obtaining the resuts of regressor.coef + regressor.intercept separately, when using manual calcuation (2185.36465466), the results and the computed (2235.18154715) there is still a slight diff of 2.2% |
|  | key insights from the results obtained in Question 4.   * The scatterplot graph plotted has provided an obvious defined straight line demonstration high level of correlationship between Balance Credit & B1 (Billable amount for the 1st month) * The ‘Correlation’ analysis module has confirmed with scoring @ 95%      * As highlighted in yellow, there are some points are out of the normal range where majorities have settled. These are ether outliners that don’t follow the natural direction of the data, or extremely unique situations, * SKlearn helped to determine the linear by digesting all the data provided. The conventional way of obtaining through x=ax + b with thousands of data seems impossible. * Using logical analysis, the correlation between Balance credit vs Repayment indeed has high correlation as the once repayment is done, Balance credit will naturally increase. It is a direct impact. |