

**ANL252**

**Python for Data Analytics**

# **End-of-Course Assignment (ECA)**

**July 2022 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **Chan Zhi Hao** | **N2070913** |

**Tutorial Group: ­­­­­­­­­­­­ T 9**

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

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

**Total Words count: 1526**

Table of Contents

**Question 1 …………………………..….………………………………………….…….** 3 - 4

**Question 2 ……………………...……………….………………….…………………....** 4 - 8

**Question 3 ……………………………......………….…………....………………….....** 9 - 14

**Question 4 …………………………...……………….………………….………….....** 15 - 20

**Question 5 ..…………………………......………….…………....……………………...** 20-23

**Reference ..…………………………......………….…………....…………………….......** 23

**Question 1**

Categorical variable is gender, education, marital, rating and age. Numeric variables are limit, balance, income, S(n), B(n) and R(n). I will convert all the categorial variables from numbers to their actual name.

**Python Code for Question 1**

|  |  |
| --- | --- |
| In[1]: | ### Import the library  import pandas as pd |
| In[2]: | ### Import and read the excel file  df=pd.read\_csv("ECA\_data.csv")  df.head() |
| Out[2]: |  |
| In[3]: | ### Replacing column values to string  df['GENDER'].replace(0, 'Male',inplace=True)  df['GENDER'].replace(1, 'Female',inplace=True)  df['EDUCATION'].replace(0, 'Others',inplace=True)  df['EDUCATION'].replace(1, 'Postgraduate',inplace=True)  df['EDUCATION'].replace(2, 'Tertiary',inplace=True)  df['EDUCATION'].replace(3, 'High School',inplace=True)  df['MARITAL'].replace(0, 'Others',inplace=True)  df['MARITAL'].replace(1, 'Single',inplace=True)  df['MARITAL'].replace(2, 'Married',inplace=True)  df['RATING'].replace(0, 'Good',inplace=True)  df['RATING'].replace(1, 'Bad',inplace=True)  df |
| Out[3]: |  |
| In[4]: | ### 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+']  df['Agegroup'] = pd.cut(df["AGE"], bins, labels = labels)  df.head() |
| Out[4]: |  |
| In[5]: | df.info() |
| Out[5]: |  |

**Question 2**

There are 4 steps that need to take into consideration so to make sure the data is successfully pre-processed. They are data quality assessment, data cleaning, data transformation and data reduction.

Data quality assessment is the followings are various information abnormalities and inherent problems to pay special attention to in almost any data set. An example is missing data, after having a good look at the dataset I found out there is 49 blanks space in education and marital.

Data cleaning will address each of the conflicting information you uncovered in your information quality evaluation. It is a process of adding missing information and correcting, repairing or eliminating mistaken or irrelevant information from an informational index. An example is data cleaning is to remove the age is -1 & 199 and there are 10 rows in the dataset. After removing all the 49 blanks space and 10 rows with an age -1 and 199, I will look at the data transformation.

Data transformation will start with the most common way of transforming the information into the appropriate format(s) you'll require for analysis. An example is there is different age from 21 to 80, I will group them into age groups.

Data reduction can make the analysis easier and more accurate but cut down on data storage. An example is finding the average for S(n), B(n), and R(n) in order to reduce the dataset.

Word count: 233

**Python Code for Question 2**

|  |  |
| --- | --- |
| In[1]: | ### Import the library  import pandas as pd  from datetime import datetime  import matplotlib.pyplot as plt  import numpy as np  pd.options.mode.chained\_assignment = None |
| In[2]: | ### Import and read the excel file  df=pd.read\_csv("ECA\_data.csv")  df.head() |
| Out[2]: |  |
| In[3]: | ### Replacing column values to string  df['GENDER'].replace(0, 'Male',inplace=True)  df['GENDER'].replace(1, 'Female',inplace=True)  df['EDUCATION'].replace(0, 'Others',inplace=True)  df['EDUCATION'].replace(1, 'Postgraduate',inplace=True)  df['EDUCATION'].replace(2, 'Tertiary',inplace=True)  df['EDUCATION'].replace(3, 'High School',inplace=True)  df['MARITAL'].replace(0, 'Others',inplace=True)  df['MARITAL'].replace(1, 'Single',inplace=True)  df['MARITAL'].replace(2, 'Married',inplace=True)  df['RATING'].replace(0, 'Good',inplace=True)  df['RATING'].replace(1, 'Bad',inplace=True)  df |
| Out[3]: |  |
| In[4]: | ### Check the size of the dataset  df.shape |
| Out[4]: | ### Row is 18769 and Column is 24  (18769, 24) |
| In[5]: | ### Remove all rows with NULL values:  df.dropna(inplace = True)  df.shape |
| Out[5]: | ### Row is 18720 and Column is 24  (18720, 24) |
| In[6]: | ### Remove all the invalid age like -1 and 199  df = df[df.AGE != -1]  df = df[df.AGE != 199]  df.shape |
| Out[6]: | ### Row is 18710 and Column is 24  (18710, 24) |
| In[7]: | ### 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+']  df['Agegroup'] = pd.cut(df["AGE"], bins, labels = labels)  df.head() |
| Out[7]: |  |
| In[8]: | ### Check the dtypes  df.info() |
| Out[8]: |  |
| In[9]: | ### Remove the $ and commas with Pandas in R3 column  df['R3'] = df['R3'].str.replace(',', '')  df['R3'] = df['R3'].str.replace('$', '')  df['R3'] = df['R3'].astype(int)  df.info() |
| Out[9]: |  |
| In[10]: | ### Find the average for S(n), B(n) and R(n)  df['Average S'] = (df['S1'] + df['S2'] + df['S3'] + df['S4'] + df['S5']) / 5  df['Average B'] = (df['B1'] + df['B2'] + df['B3'] + df['B4'] + df['B5']) / 5  df['Average R'] = (df['R1'] + df['R2'] + df['R3'] + df['R4'] + df['R5']) / 5  print (df) |
| Out[10]: |  |
| In[11]: | ### Drop the column S1-S5, B1-B5 and R1-R5 to reduce the dataset  df1 = df.drop(columns=['S1', 'S2','S3', 'S4','S5', 'B1', 'B2','B3', 'B4','B5', 'R1', 'R2','R3', 'R4','R5'])  df1.head() |
| Out[11]: |  |

**Question 3**

**Python Code for Question 3**

|  |  |
| --- | --- |
| **Python code continued from Question 2** | |
| In[12]: | ### Let the index start from 1 instead of 0  df1.index = np.arange(1, len(df) + 1)  df1.head() |
| Out[12]: |  |
| In[13]: | ### Find the number of rating using count  df\_rating = df1.groupby(["RATING"])['RATING'].count()  df\_rating |
| Out[13]: |  |
| **Vertical Bar Chart** | |
| In[14]: | ### Group the gender by agegroup  df\_gender = df1.groupby(["Agegroup","GENDER"])["Agegroup"].count()  df\_gender.head() |
| Out[14]: |  |
| In[15]: | ### Blue colour is refer female and Red colour is refer male  df\_gender.plot(kind='bar', color=['blue', 'red'] ,stacked=False, title='Agegroup and gender bar chart ', xlabel="Agegroup", ylabel="Number of customer") |
| Out[15]: |  |
| **Pie Chart** | |
| In[16]: | ### Groupby the marital  df2 = df1.groupby(['MARITAL'])['MARITAL'].count()  df2 |
| Out[16]: |  |
| In[17]: | ### Use pie chart to show the marital percentage using autopct.  y = np.array([9822, 8699, 189])  mylabels = ["Married", "Single", "Others"]  ### Function of the pie chart  plt.pie(y, labels=mylabels, autopct='%1.1f%%', shadow=True, startangle=90)  plt.title ("Percentage marital pie chart")  plt.legend(labels = mylabels, loc ="lower left")  plt.show() |
| Out[17]: |  |
| **Grouped Bar Chart** | |
| In[18]: | ### Group the education by agegroup  df\_education = df1.groupby(["Agegroup",'EDUCATION'])['Agegroup'].count()  df\_education |
| Out[18]: |  |
| In[19]: | ### Key in the data and grouped bar chart by education  data = pd.DataFrame([['20-29', 655, 2621, 3407, 110], ['30-39', 909, 2494, 3217, 116], ['40-49', 1008, 946, 1745, 85], ['50-59', 467, 298, 448, 21],['60-69',63, 42, 46, 2],['70+', 5, 3, 2, 0]], columns=['X-Axis', 'High School', 'Postgraduate', 'Tertiary', 'Others'])  data.plot(x='X-Axis',kind='bar', stacked=False, title='Education grouped bar chart by agegroup', xlabel="Education", ylabel="Number of customer") |
| Out[19]: |  |
| **Line Chart** | |
| In[20]: | ### Use the df1 that have modify in question 2 earlier and groupby the agegroup  df2 = df1.groupby(['Agegroup']).mean()  df2 |
| Out[20]: |  |
| In[21]: | ### Use the line chart funtion to find the Income, limit and balance  df2["INCOME"].plot(color = "red", marker = "o", markerfacecolor = "black",markeredgecolor="black")  df2["LIMIT"].plot(color = "blue", marker = "x", markerfacecolor = "Red",markeredgecolor="Red")  df2["BALANCE"].plot(color = "green", marker = "s", markerfacecolor = "orange",markeredgecolor="orange")  plt.ylabel("Dollars ($) as per person")  plt.legend(["Income","Limit","Balance"],loc='upper left')  plt.title("Average per person Limit, Income and Balance line chart") |
| Out[21]: |  |
| In[22]: | ### Use the line chart funtion to find the average billable and repayment amount per person by groupage.  df2["Average B"].plot(color = "blue", marker = "x", markerfacecolor = "Red",markeredgecolor="Red")  df2["Average R"].plot(color = "green", marker = "s", markerfacecolor = "orange",markeredgecolor="orange")  plt.ylabel("Dollars ($)")  plt.legend(["Monthly average billable amount (per person)","Monthly average repayment amount (per person)"],loc='upper left')  plt.title("Monthly average for billable and repayment amount line chart") |
| Out[22]: |  |

**Summary:**

I have plotted 5 charts of data analysis for this dataset. The first 3 charts are based on the categorical variable to define the target customer and the last 2 are based on the numeric variable.

The first chart is a vertical bar chart used to find the number of customers in individual age groups and gender. The highest number of customers is females age group 20-29, and the lowest number of customers is males age group 70+. The reason is that females will tend to buy more branded clothes and bags, so they need to borrow money from the credit facility when their income is lower than other age groups.

The second chart is a percentage of the marital pie chart. The pie chart showed married 52.5%, single 46.5% and others 1%. The reason is that married young couples will need to pay for housing and car loans, so they tend to borrow money from the credit facility.

The third chart is grouped bar chart in which I have grouped the education by group age.  The highest number of customers is tertiary 20-29 years old. This chart tells us that more people have higher education compared to the age group 50-59 years old. Customers who have higher education will tend to borrow money from credit facilities to do investments and another reason is that customer who has higher education will have a higher salary but at the same time, they tend to spend more money.

After understanding the target customer, then the next step is knowing the income and how much money they borrow on average. The fourth and fifth chart is the line plot to analyse the numeric variable.  The fourth plot chart showed the average per person income, limit and balance. The limit is lowest than the income and this is because the credit facility has to ensure the customer has the money to repay them. The highest income earned in the age group is 70+ and the lowest income earned in the age group is 20-29 years. This plot chart has explained why the target customer is the age group 20-29 years because at this age group they start to work and earned lesser income than other age groups.

The fifth chart is also the plot chart that showed the monthly average billable amount per person and the monthly average repayment amount per person. The highest average billable amount per person is the agegroup 70+ and the lowest is the age group 20-29. This means the highest number of customers are age group 20-29, but they never borrow so much money from the credit facility, this may be due to the limit that set by the credit facility against their income. The monthly average repayment amount per person is lower than the billable amount per person which means more people are unable to pay the credit facility on time.

Overall, this five-chart plot has let me understand the target customer of the credit facility and how the credit facility operates in terms of setting the limit against the customer's income. They are some variables that were never included in the data analysis like customer repayment reflected status in nth month S(n) and rating, but both have been included in the python coding. For the rating, 14599 customers have chosen good, and  4111 customers have chosen bad. Then for the S(n), the average is 0 which is the minimum sum payment.

Words count: 574

**Question 4**

**Python Code for Question 4**

|  |  |
| --- | --- |
| In [1]: | ### Import the library  import pandas as pd  import numpy as np  import matplotlib.pyplot as plt  import seaborn as seabornInstance  from sklearn.model\_selection import train\_test\_split  from sklearn.linear\_model import LinearRegression  from sklearn import metrics  from sklearn.metrics import r2\_score  %matplotlib inline |
| In [2]: | ### Import and read the excel file  df=pd.read\_csv("ECA\_data.csv")  df.head() |
| Out [2]: |  |
| In [3]: | ### Reset the index starting from 1 instead of 0  df.index = np.arange(1, len(df) + 1)  df.head() |
| Out [3]: |  |
| In [4]: | ### Check the size of the dataset  df.shape |
| Out [4]: | (18769, 24) |
| In [5]: | ### Check the statistical details of the dataset  df.describe() |
| Out [5]: |  |
| In [6]: | ### Use the balance and B1 to do the analysis  df.plot(x='BALANCE', y='B1', style='o')  plt.title('Actual data')  plt.xlabel('BALANCE')  plt.ylabel('B1')  plt.show() |
| Out [6]: |  |
| In [7]: | ### Find the average B1 and it shows between nearly 25000 to 50000  plt.figure(figsize=(25,10))  plt.tight\_layout()  seabornInstance.distplot(df['B1']) |
| Out [7]: |  |
| In [8]: | ### Predict the B1 depending upon the balance using label(y-axis) dependent variable and attribute(x-axis) independent variable  X = df['BALANCE'].values.reshape(-1,1)  y = df['B1'].values.reshape(-1,1) |
| In [9]: | ### Use the test set 20% and training set 80% of the data  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0) |
| In [10]: | ### Import linearRegression class to train our algorithm.  regressor = LinearRegression()  regressor.fit(X\_train, y\_train) |
| Out [10]: | LinearRegression() |
| In [11]: | ### To retrieve the intercept:  print(regressor.intercept\_)  ### For retrieving the slope:  print(regressor.coef\_) |
| Out [11]: | [2185.36465466]  [[5.24388342]] |
| In [12]: | ### Make some predictions using our test data  y\_pred = regressor.predict(X\_test) |
| In [13]: | ### Compare the predicted value with actual output values for X\_test  df = pd.DataFrame({'Actual': y\_test.flatten(), 'Predicted': y\_pred.flatten()})  df.index = np.arange(1, len(df) + 1)  df |
| Out [13]: |  |
| In [14]: | ### Bar chart showing the comparison of predicted and actual values.  df1 = df.head(25)  df1.plot(kind='bar',figsize=(16,10))  plt.grid(which='major', linestyle='-', linewidth='0.5', color='green')  plt.grid(which='minor', linestyle=':', linewidth='0.5', color='black')  plt.title('Bar graph showing the comparison of Actual and Predicted values')  plt.ylabel('Values')  plt.xlabel('Index')  plt.show() |
| Out [14]: |  |
| In [15]: | ### Plot the straight line with test data  plt.scatter(X\_test, y\_test, color='green')  plt.plot(X\_test, y\_pred, color='red', linewidth=2)  plt.xlabel('BALANCE')  plt.ylabel('B1')  plt.title('Test set data')  plt.show() |
| Out[15]: |  |

**Summary:**

I have to use single linear regression instead of multiple linear regression because they are no other variables that will really affect the billable amount only balance. The approach taken in this linear regression has 5 parts. The first part is to import all the library/files, read the heading and understand the shape/description of the dataset which we usually will do at the beginning. (Coding 1 to 5)

The second part is to use plot our data using a balance (x-axis) and B1 (y-axis). The reason for using balance vs B1 to plot our data because balance is shows the snapshot in time which tell us whether the customer will have the ability to pay the billable amount. To confirm the B1 average we will plot another graph, so it confirms that the average B1 is between nearly 25000 to 50000. This question is asking us to predict the variable B1 which was depending upon the balance recorded. We will divide the data into “attributes” and “labels”. Therefore, attributes (x-axis) will be the independent and labels (y-axis) will be the dependent. (Coding 6 to 8)

The third part uses the train-test evaluation technique for evaluating the performance of a machine learning algorithm so we will be using 20 per cent of the data for to test set and 80 per cent of the data for the training set. We use this technique when there is a sufficiently large dataset available. For the linear regression formulae, I will be talking more in question 5. (Coding 9 to 11).

The fourth step is to compare the actual and the predictions to see how accurately our algorithms predict the percentage score using our data. I have visualized the comparison results using a bar chart as the number of records is huge, I will use only 25 records for the representation purpose. The bar chart showed our model is not very precise due to the error but predicted are close to the actual percentage. Then we plot the straight-line graph with the test data. (Coding 12 to 15) The last step is to evaluate the performance of the algorithm. (Coding 16)

Words count: 357

**Question 5**

**Python Code for Question 5**

|  |  |
| --- | --- |
| **Python code continued from Question 4** | |
| In[16]: | ### Final step is to evaluate the testing performance set of the algorithm using MAE, MSE and RMSE  print('Mean Absolute Error:', (metrics.mean\_absolute\_error(y\_test, y\_pred)))  print('Mean Squared Error:', (metrics.mean\_squared\_error(y\_test, y\_pred)))  ### r2 refer to square root 2 and round up to 2 decimal place  r2 = round(r2\_score(y\_test, y\_pred),2)  print('The model performance for testing set of RMSE is', (round(np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred)), 2)), 'and R2 score is {}'.format(r2)) |
| Out[16]: | Mean Absolute Error: 9305.433115039266  Mean Squared Error: 610655272.1379098  The model performance for testing set of RMSE is 24711.44 and R2 score is 0.88 |

**Summary**

The equation for linear regression is . Yi refers to the dependent variable, Xi refers to the independent variable, β0 refers to constant/ intercept and β1 refers to slope/coefficient. For question 4, we have used Yi as b1 variables and Xi as balance variables. The result showed the y-intercept is 2185.36 (2 decimal points). The coefficient is 5.24 (2 decimal points) and this means that for every unit of change in balance, the change in the b1 is about 5.24. (Refer to Question 4 python coding) This coefficient is used to measure the slope of a line steepness also known as the gradient.

The root means absolute error showed the value is 24711 and R2 score is 0.88, which is less than 10% of the mean value percentage of the B1. [(24,711/ 29,433)\*100 = 18%] For the balance mean is 9119 and B1 mean is 49985 so the working is [(49985-9119)/2] = 20,433 and plus 9119 equal to 29433. This showed that our algorithm was not very accurate but can make acceptable good predictions. There is need to proceed to do the data pre-processing like the data cleaning method to remove the negative value or zero value in the B1.

We will use the actual and the predicted graph to explain the key insights from the results.

First insight, the model is assessed on the test set and the model's performance while making predictions on new information is a mean absolute error (MAE) of about 9305 and root mean squared error (RMSE) of about 24711. MAE is used to measure the average magnitude of the errors in a set of predictions. RMSE is a quadratic scoring rule that also calculates the average extent of the error. (Please refer to below figure 1.1 and 1.2)

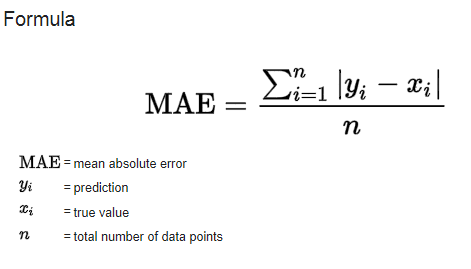


Figure 1.1: MAE formula

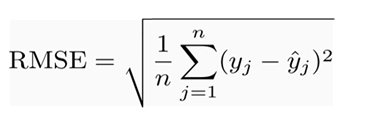


Figure 1.1: RMAE formula

The second insight, in the actual data plot graph more of the data point in between 0-200000 for the y-axis and 0-30000 for the x-axis. In this case, it will affect the prediction of the straight line. We want to see the data point closer to the straight line which can give us a better prediction. (Please refer to below figure 1.3)

The third insight, there is some data points are far away from the straight line. This is called the outlier if the data point doesn’t fit the pattern. Outliers should remove from the data because they can affect measurement errors or poor sampling. (Please refer to below figure 1.3 and 1.4)

Overall, the actual set and test set plot graph of the MAE is different by 18 per cent which is higher than 10 per cent so it tells us that our algorithm was accurate but can still make reasonably good predictions.

Words count: 330

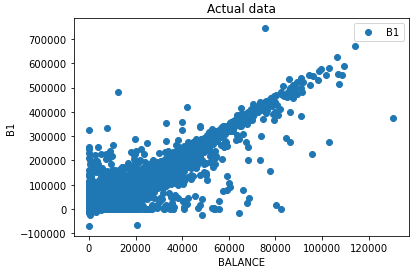


Figure 1.3: Actual plot graph

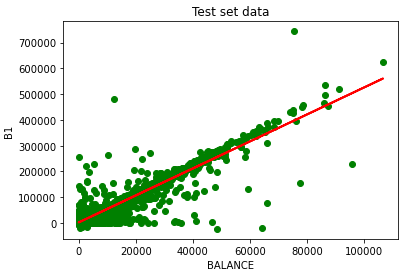


Figure 1.4: Test set plot graph

**Reference**

Kotz, S.; et al., eds. (2006), Encyclopedia of Statistical Sciences, Wiley (figure 1.1 & 1.2)

Retrieve from: <https://www.statisticshowto.com/absolute-error/>

**The End**