

**ANL252**

**Python for Data Analytics**

**Group-based Assignment**

**July 2022 Presentation**

**Shawn Tan Jin Chuan - J2172224**

**Ng Rui Feng Jonathan - Y2210848**

**Ng Wei Ling - B2210595**

**Ong Jia Ying - B1410552**

**Question 1a**

**In the dataset, ‘-’, ‘--’, and ‘?’ are considered as missing values, and the variable columns of the dataset are noted as in the data dictionary in order. As part of data preparation, read the dataset in as a Pandas dataframe, with the above considerations.**

**In Jupyter Notebook:**

#import pandas

import pandas as pd

#identify missing data, replace 'non blank' missing data with 'NaN'

df = pd.read\_csv('GBA\_data.csv', na\_values = ("-","--","?"), na\_filter = True)

df#added this to show missing data is filled with 'NaN'

**Plain text:**

‘-‘, ‘—’ and ‘?’ are considered missing data, but python will recognize them as values. We have to convert them to ‘NaN’ first.

**Question 1b**

**Identify the variable columns which have missing values. As part of data preparation, implement ways to treat them, and explain your rationale. State any interesting observation(s).**

**In Jupyter Notebook:**

#import pandas

import pandas as pd

#identify missing data, replace 'non blank' missing data with 'NaN'

df = pd.read\_csv('GBA\_data.csv', na\_values = ("-","--","?"), na\_filter = True)

#check and count dataframe for missing values

df.isnull().sum(axis = 0)

#drop rows with missing data and update DataFrame

df = df.dropna(axis = 0, how = 'any')

df.isnull().sum(axis = 0) #added this to show that its dropped

**Plain text:**

Both Origin and Destination columns have 504 missing values. Yob, Age and Type columns have 10 missing values and Gender column has 3 missing values.

There are a few ways to handle missing values in dataset in relation to data preparations and we would suggest imputation for categorical columns where missing values are replaced with most frequent category or new category. That way, it can help prevent data loss (deletion of rows or columns) and negate the loss of data.

Another method would be deleting rows with missing values where a robust model would be created upon removing all missing values. However, there would be a risk of losing information, hence, it might not be the best option.

**Question 1c**

**As part of data preparation, identify *three (3)* other data quality issues in the data. Similarly, suggest and implement ways to treat them, and explain your rationale.**

**(Issue 1) In Jupyter Notebook:**

from datetime import date

#Create new column 'age\_current', populated by commuter age as of today

today = date.today()

df[['age\_current']]=today.year -df[['yob']]

df#Display to show added column

**Data Issue 1: Age of commuters not accurate (Inaccurate Data)**

The “age” data does not reflect the actual age of commuter as of today. When the data are used at a later time, profiling of the commuters might not be as accurate.

We proposed to add a column of data that reflects the age of commuters as of current date, so that the data get updated even at a later time.

**(Issue 2) In Jupyter Notebook:**

#Generating Unique Trip ID

#Create new column 'trip\_id', populated using 'origin' + 'destination'

df['trip\_id'] = df['origin'].astype(str) + "-" + df['destination'].astype(str)

df#Display to show added column

#### Data Issue 2: ID Data not unique to commuters (Duplicated Data)

The same id numbers are used for multiple commuters, making the data not so useful. We proposed to create a "Trip ID" for commuter's trip by using their start and end destinations. This way, we are able to profile the commuters by location.

**(Issue 3) In Jupyter Notebook:**

#Drop column 'id' and 'age' as they are obsolete.

df.drop(['id','age'], axis=1, inplace=True)

df#Display to show removed column

**Data Issue 3: Data Overload**

There are some data provided that are not used/touched, making it more troublesome to prepare the needed data, sort the data out and organizing the data as compared to only having the data that we needed. In some cases, useless data would cloud our judgement in selecting the correct data to use. Hence, it is logical to identify and remove the useless data to save preparation time.

**Question 1d**

**Develop a user-defined function that will print the hour, expressed in the 12-hour clock format (e.g., 12am, 1pm), whereby the highest number of commuters start their journey.**

**In Jupyter Notebook:**

import pandas as pd

from datetime import datetime

import numpy as np

df = pd.read\_csv('GBA\_data.csv', na\_values = ("-","--","?"), na\_filter = True)

df = df.dropna(axis = 0, how = 'any')

#Create function to convert datetime format to 12hours format

def convert\_time(time\_string):

#define datetime format pattern for python

time\_object = datetime.strptime(time\_string, "%Y-%m-%d %H:%M:%S.%f")

#returns datetime in 12hours format

return time\_object.strftime("%I%p")

#Create function to identify the time with the highest number of commuters at the start of their journey

def highest\_commuters(df):

hour = df["start"].apply(convert\_time)

return hour.mode().to\_string(index=False)

#Print the time with the highest number of commuters at the start of their journey

print(highest\_commuters(df))

**Plain text:**

The highest number of commuters start their journey at 5pm.

**Question 1e**

**Write a Python code to create appropriate visualisations of the commuter data. Analyse the results and then discuss *three (3)* interesting insights.**

**(Insight 1) In Jupyter Notebook:**

import pandas as pd

from pandas import DataFrame, cut

from datetime import datetime

import numpy as np

df = pd.read\_csv('GBA\_data.csv', na\_values = ("-","--","?"), na\_filter = True)

#Assuming station number is in running sequence

#Create a new column "Distance Travelled", populated by the number of stations each commuter travelled.

df["Distance Travelled"] = abs(df["origin"] - df["destination"])

n = 10

#Identify the unique variables in column "type"

df['type'].value\_counts()[:n].index.tolist()

#Replace missing values with "Regular"

df['type'] = df['type'].replace("?", "Regular")

#Create pivot table between "type" and "Distance Travelled", return mean distance travelled

output = pd.pivot\_table(data=df,

index=['type'],

values=['Distance Travelled'],

aggfunc='mean')

#Show table

output

#Plot bar graph

output.plot(kind="bar", title = "Average number of stations travelled by each commuter type")

#Sum up total distance travelled by each type of commuters

output1 = pd.pivot\_table(data=df,

index=['type'],

values=['Distance Travelled'],

aggfunc='sum')

#Show table

output1

**Interesting Insight 1**

The above is the pivot chart of the distance travelled with respect to the customer profile type. We can see that on average 'Ad-Hoc' customer type travels the most. This could be because people who are 'Ad-Hoc' customers could be tourists or business owners who travels more often, hence this led to a higher amount of distance travelled as compared to the rest of the customer profile type.

The next kind of customer profile type is the 'Concession' type. This could highly likely be students who are going to school on weekdays and as we know, 'Concession' type passes allows a fixed number of rides a day and a fixed fee is paid to the commuting service provider. Students usually purchase concession passes if they travel a long way to school, for example, from Pasir Ris to SUSS, NUS or NTU. Hence, the distance travelled on average is very close to the 'Ad-Hoc' and 'Regular' customer profile type, but no doubt, the distance travelled on average is lesser. In addition, the total distance travelled by the 'Concession' customer profile type are the lowest. This could be because of the low number of 'Concession' customer profile type commuters.

The third kind of customer profile type is the 'Regular' type. These commuters could have been working adults. We can tell this from the total distance travelled, it is the highest and the average distance travelled is also in the middle. These commuters seems to be travelling to somewhere further than the 'Concession' customer profile type commuters.

**(Insight 2) In Jupyter Notebook:**

#Create pivot table for number of "subscriber" by "gender"

output2 = pd.pivot\_table(data=df,

index=['gender'],

values=['subscriber'],

aggfunc='count')

#Show table

output2

#Plot bar graph

output2.plot(kind="bar", title = "Number of Subscribers by Gender")

**Interesting Insight 2**

From the above pivot chart with respect to the gender and number of subscribers for each category. 'Male' has the highest number of subscribers. The second highest number of subscribers is 'Female' and followed by 'Other' with the lowest number of subscribers. We can tell from this that, it could be because 'Male' commuters are mostly the bread-winners of the family. Hence, they tend to travel more often and thus needs the subscription. However, 'Female' has a much lower number of subscriptions compared to 'Male'. This could be because most of the 'Female' commuters here could be housewives and does not need to travel out often. Hence, 'Female' commuters may not need the subscription as much as 'Male' commuters.

**(Insight 3) In Jupyter Notebook:**

df['startTime'] = pd.to\_datetime(df['start'])

df['endTime'] = pd.to\_datetime(df['end'])

#Create new column "timeTravelled", populated by the difference in start time and end time.

df['timeTravelled']=abs(df['endTime']-df['startTime'])

#Create new column "timeTravelledMinutes", populated by converting "timeTravelled" into minutes

df['timeTravelledMinutes'] = df['timeTravelled'].dt.total\_seconds().div(60).astype(int)

#display dataframe

display(df)

#Create pivot table for the number of commuters for each type

df['type1'] = df['type']

output4 = pd.pivot\_table(data=df,

index=['type'],

values=['type1'],

aggfunc='count')

#Show table

output4

#Create pivot table of total time travelled for each type of commuters

output3 = pd.pivot\_table(data=df,

index=['type'],

values=['timeTravelledMinutes'],

aggfunc='sum')

#Show table

output3

#Plot bar graph

output3.plot(kind="bar", title = "Type of commuters that spent the most time travelling")

**Interesting Insight 3**

From the above pivot chart, we can see that Customer Profile Type: 'Regular' travelled the most time in minutes. This goes further to prove our analysis above is correct that 'Regular' commuters could have been working adults because they commute to and fro from work almost everyday and 'Regular' commuters also form the largest number across all 3 types at count of 555227. 'Concession' customer profile type makes up the least number of commuters in the dataset, this could be because they are mainly students commuting to and fro from school. It is evidently shown in the pivot chart above, where the time travelled in minutes is the lowest. Lastly, 'Ad-Hoc' customer profile type customers are in the middle. In addition, as what we saw from above, they could be business owners needing ad-hoc transport arrangements. Hence, they make up the second largest amount of time travelled in minutes.