**ANL252 \_ECA**

**Tan Jie Yi**

**M2272470**

**Question 1**

I ensure my excel file is saved in the same file as how where I save my code in and used read\_excel() function to read my dataset from Pandas data frame. Which read the Excel file name ‘ECA\_data.xlsx’ and denotes blanks, ‘Unkn' and ‘???’ are expressions in the dataset with the default NAN values in Pandas which sigify missing values.

By using isnull() and sum() function together to count the missing values in each column as shown in my output.

**Output for Question 1**

Claim\_ID 5

Policy\_No 0

Name 0

Planned 0

Actual 1677

Created 0

Amount 0

Paid 0

Category 0

Terms 7

Region 0

Type 0

dtype: int64

**Question 2**

There are many ways to treat missing data, depending on the nature of the dataset and the question given. Here are some of the methods to handle missing values which include:

* Removing missing data:

By removing missing data, it removes any rows or columns that contain missing data. However, this method may result in loss of valuable information and can introduce biasness in the end analysis. We can use this if the proportion of missing values is small e.g. less than 5 – 10 % of the total observation) we could simply use the .dropna() method.

* Imputing missing data:

By replacing the missing values with estimated values based on the statistical methods or using other data in the dataset The choice of imputation method depends on the type of ata and research question

* Keep the missing values:

If the missing values are meaningful or if we o not want to lose information associated with them, we could keep the missing values as they are and can provide a unbiased data accordingly

**Output for Question 2 , using dropna() method**

Claim\_ID 0

Policy\_No 0

Name 0

Planned 0

Actual 0

Created 0

Amount 0

Paid 0

Category 0

Terms 0

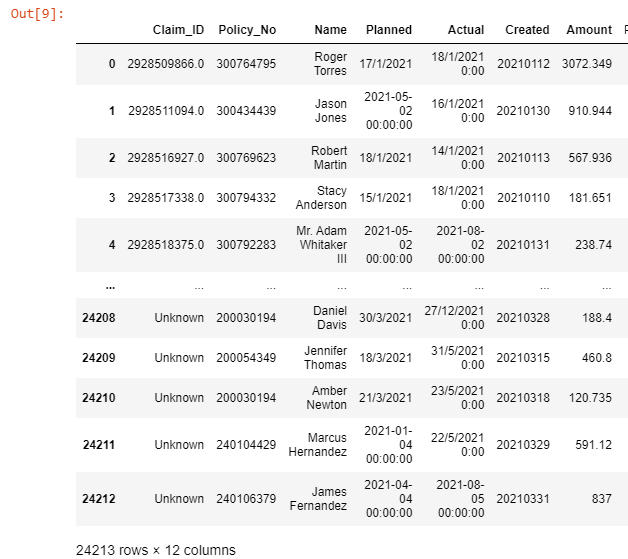
Region 0

Type 0

dtype: int64

This output was using the dropna() method as , the code removes all the rows with missing values and modify the data frame in place.

**Output for Question 2 using .fillna() method**



This output shows the fillna() method , as shown in the output some missing values from the ‘Claim\_ID’ column have change to indicate ‘Unknown’, in which I have code.

**Question 3**

Three other data preparation tasks could be performed for further analysis of the data

1. Convert date variables to datetime format

Firstly, depending on the dataset, some columns may be stored as the incorrect data type. For example, the ‘Created ‘column is stored as a string instead of a datetime object. By converting all the date variables like ‘Planned’, ‘Actual’ and ‘Created’ to datetime format, to make it easier to extract useful information such as day of the week, month or year. This can be done using the pandas to \_datetime () function as shown below from the output. Code is written in Appendix.

**Output for Question 3 [ Convert data variables to datetime format]:**

Planned

0 2021-01-17

1 2021-05-02

2 2021-01-18

3 2021-01-15

4 2021-05-02

...

24208 2021-03-30

24209 2021-03-18

24210 2021-03-21

24211 2021-01-04

24212 2021-04-04

Name: Planned, Length: 24213, dtype: datetime64[ns]

Actual

0 2021-01-18

1 2021-01-16

2 2021-01-14

3 2021-01-18

4 2021-08-02

...

24208 2021-12-27

24209 2021-05-31

24210 2021-05-23

24211 2021-05-22

24212 2021-08-05

Name: Actual, Length: 24213, dtype: datetime64[ns]

Created

0 2021-01-12

1 2021-01-30

2 2021-01-13

3 2021-01-10

4 2021-01-31

...

24208 2021-03-28

24209 2021-03-15

24210 2021-03-18

24211 2021-03-29

24212 2021-03-31

Name: Created, Length: 24213, dtype: datetime64[ns]

1. Removing duplicates

Secondly, removing duplicates can be implemented. The dataset contains some duplicated rows, to remove duplicates, we can use the pandas drop\_duplicates() function. I identify the duplicated rows first as shown below :

**Output for Question 3 [ Identifying the duplicate rows]:**

Claim\_ID Policy\_No Name Planned Actual \

13 2.928533e+09 300418007 Joan Elliott 2021-01-19 2021-01-22

16713 2.930189e+09 300769623 Robert Martin 2021-08-12 2021-04-12

18180 2.930371e+09 300803720 Zachary Gonzalez DDS 2022-11-01 2022-09-01

Created Amount Paid Category Terms Region Type

13 2021-01-14 183.71 Yes AT AD23 LOC L001

16713 2021-12-03 2913.655 Yes AT AB27 LOC L001

18180 2022-01-06 3631.118 Yes AT DA17 LOC L001

After identifying the duplicated rows, I use the pandas drop\_duplicated function to remove the duplicated rows. As shown below, all the duplicated rows are removed.

**Output for Question 3 [ After duplicated rows are dropped]:**

Empty DataFrame

Columns: [Claim\_ID, Policy\_No, Name, Planned, Actual, Created, Amount, Paid, Category, Terms, Region, Type]

Index: []

1. Rename all columns for clarity and have an additional column for number of delay days

Lastly, Column names that are not descriptive can make it difficult to understand data. By renaming the columns , we use the rename() function from the pandas Library as shown in the output. Creating a new variable for claim processing delay, having a new variable that shows the delay in days between the planned and actual claim settlement dates, which can be useful for further analysis and modelling.

**Output for question 3 [Additional column that shows Days delayed]**

Number of Days dalayed

0 1.0

1 -106.0

2 -4.0

3 3.0

4 92.0

...

24208 272.0

24209 74.0

24210 63.0

24211 138.0

24212 123.0

Name: Days Delayed, Length: 24210, dtype: float64

**Output for Question 3 {rename all columns to have better descriptive]**

**Shows all the renamed columns and the first row for all the variables**

Claim ID 2928509866.0

Policy Number 300764795

Name Roger Torres

Planned Date 2021-01-17 00:00:00

Actual Date 2021-01-18 00:00:00

Date Created 2021-01-12 00:00:00

Claim Amount 3072.349

Paid Amount Yes

Claim Category AT

Claim Terms AD23

Claim Region LOC

Claim Type L001

Days Delayed 1.0

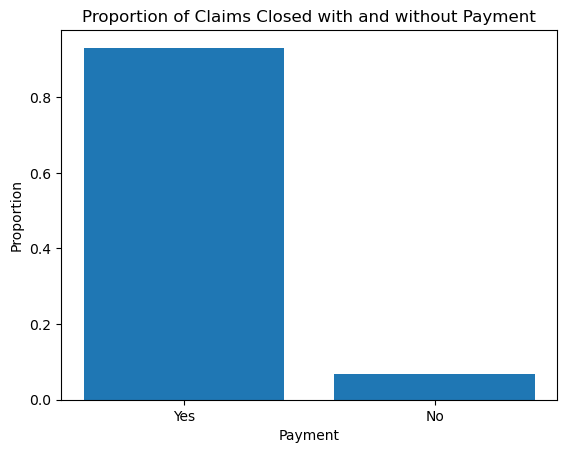
Name: 0, dtype: object

**Question 4**

**Insight 1 : Proportion of claims closed with and without payment**

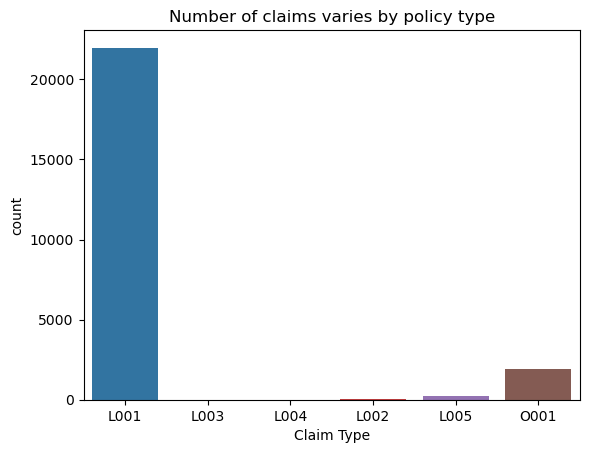
We can use a bar chart to visualize the proportion of claims that were paid and unpaid. The output shown as below, that there was close to 0.9 of claim has been paid but about 0.1 has not been paid. This insight is important for the company to identify potential areas for improvement in their claims processing workflow.

**Output for Question 4 insight 1**



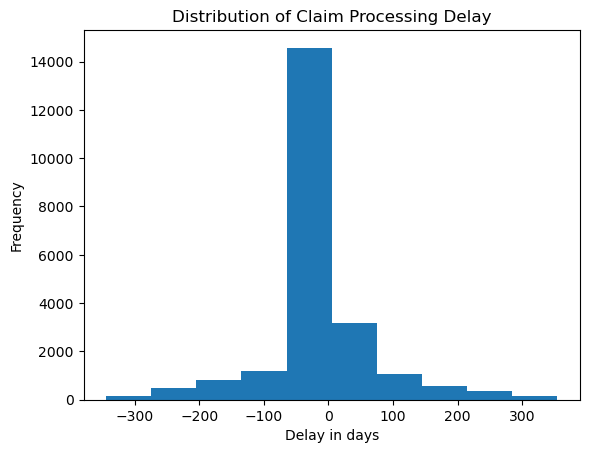
**Insight 2: Number of claims varies by policy type**To understand the distribution of claims by policy type, we can create a count plot using the Seaborn library. The count plot shows that most claims are of type L001, which indicates that the insurance company receives mostly from L001 claims. However, there are also a significant number of claims of other types like L003, L004, L002, L005 and O001 which may be important for the company to consider in their claims processing strategies.

**Output for Question 4 Insight 2**

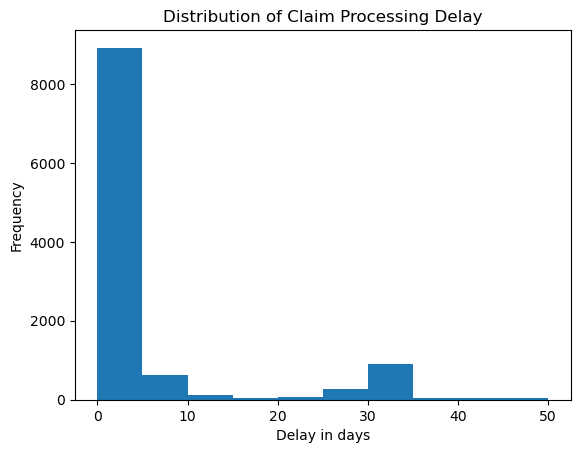


**Insight 3 : Planned and actual dates of claim differ significantly**To understand the difference between planned and actual dates of claims, we can create a histogram of the time difference between these two dates.

**Output insight 3.1 [Histogram – Full data ]**



**Output insight 3.2 [ Histogram – Range (0-50 days)]**



The histogram shows that the time difference between planned and actual dates of claims is mostly concentrated between 0 and 10 days as seen in Histogram 3.2 more clearly, indicating that most claims are processed within the planned time frame. However, there are also some claims that are processed after the planned date, with the maximum time difference being over 40 days.

**Question 5**

Linear regression is a statistical modeling approach used to examine the connection between a dependent variable and one or more independent variables. Here, we wish to estimate the number of days that will pass between the anticipated and actual dates for processing a claim. By gathering and pre-processing the data is the initial step in the linear regression modeling process. The data may need to be cleaned up and transformed, pertinent variables may need to be chosen, and outliers or missing numbers may need to be handled.

Once we have prepared the data, we may use an appropriate procedure to construct a linear regression model. The Ordinary Least Squares (OLS) approach will be used in this situation to reduce the sum of squared residuals between the expected and actual values. The model's success may then be assessed using a variety of metrics, such as the R-squared value, which quantifies the percentage of the dependent variable's variation that can be accounted for by the independent variables.

We may also take into account strategies like feature selection or regularization, which serve to lessen overfitting and enhance generalization, to increase the model's accuracy.

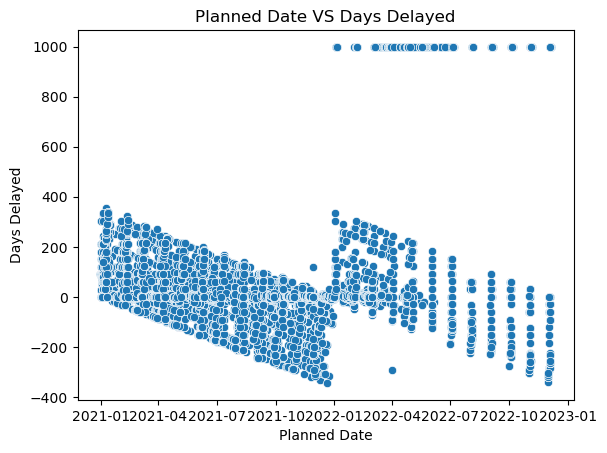
The following are the main actions in carrying out linear regression modeling to forecast claim processing delays:

* collection and pre-processing of data
* choosing pertinent independent variables
* With the OLS technique, creating a linear regression model
* Model performance is assessed using measures like R-squared.
* accuracy of models is increased by applying methods like feature selection or regularization.

**Question 6**

In order to examine the outcomes after doing the linear regression, we must first look at the model's coefficients. When all predictor variables are equal to zero, the response variable should have its predicted value, which is known as the intercept term. The response variable's predicted change for a one-unit increases in the predictor variable, leaving all other variables fixed, is represented by the predictor variable's coefficient.

**Output of Scatter Plot**



The graph shows a scatter plot with the Delay in Days (between Planned and Actual Date) on the y-axis and the Initial Planned date on the x-axis. Each point represents a single claim in the dataset.

The points are scattered all over the plot, which indicates that there is a wide range of delays for claims of payment from the planned date. The scatter plot shows a fairly large number of claims being paid on time or before the planned date which is the negative side of the y axis and also a large number of claims paid late after the planned date, which is the positive side of the Y axis. There are a few outliers, which are claims with very high delay times compared to the initial planned date. This could be due to various reasons such as complexities in the claim, lack of resources, or other external factors.

Overall, the scatter plot provides insight into the relationship between the delay in claims processing and the initial planned date of payment. However, it does not provide any clear indication of the strength or direction of the relationship, and more statistical analysis is required to understand this relationship in more detail.

The linear regression equation is given by **y = mx + b**, where **y** is the amount paid, **x** is the planned date of service, **m** is the slope of the line, and **b** is the y-intercept. We can obtain the values of **m** and **b** from the **coef\_** and **intercept\_** attributes of the fitted model, respectively.

**Output for linear regression equation**

y = 0.00x + -11880.43

According to this equation, neither the money paid nor the number of days the service is provided later than expected. The y-intercept of -11880.43 indicates that the number of days delayed to pay would be negative and would not

make sense in the context of the problem if the scheduled date of service were 0 (which is not conceivable because the dates are in 2021). There was no regression line as a result.

**APPENDIX**

**Code for Question 1**

import pandas as pd

df = pd.read\_excel('ECA\_data.xlsx', na\_values = ['', 'Unkn', '???'])

missing\_values = df.isnull().sum()

print(missing\_values)

**Code for Question2 [ dropna() method]**

**Using dropna() method**

import pandas as pd

df = pd.read\_excel('ECA\_data.xlsx', na\_values = ['', 'Unkn', '???'])

missing\_values = df.isnull().sum()

df.dropna(inplace=True)

#To check if missing values are all handled and no missing values are left behind

missing\_values = df.isnull().sum()

print(missing\_values)

**Code for Question 2 [ fillna() method]**

import pandas as pd

df = pd.read\_excel('ECA\_data.xlsx', na\_values = ['', 'Unkn', '???'])

missing\_values = df.isnull().sum()

#To replace blanks with NaN values in the relevant variables

df.replace([" ",'Unkn','???'],pd.NA,inplace=True)

#impute missing values with appropriate values

df['Claim\_ID'].fillna('Unknown',inplace=True)

df['Actual'].fillna('Unknown',inplace=True)

df['Terms'].fillna('Unknown',inplace=True)

**Code for Question 3 [Convert date variables to datetime format]**

import pandas as pd

import numpy as np

#Read the dataset

df = pd.read\_excel('ECA\_data.xlsx', na\_values = ['', 'Unkn', '???'])

#convert date variables to datetimeformate

df['Planned'] = pd.to\_datetime(df['Planned'], dayfirst=True, errors='coerce')

df['Actual'] = pd.to\_datetime(df['Actual'], dayfirst=True, errors ='coerce')

df['Created'] = pd.to\_datetime(df['Created'], format='%Y%m%d', errors = 'coerce')

print("Planned")

print(df['Planned'])

print("Actual")

print(df['Actual'])

print("Created")

print(df['Created'])

**Code for Question 3 [ Remove duplicate rows]**

#identify duplicate rows

duplicates = df[df.duplicated()]

#removing duplicate rows

df = df.drop\_duplicates()

print(duplicates)

**Code for Question 3 [ Rename all columns for clarity and have an additional column for number of delay days ]**

#Rename columns for clarity

df = df.rename(columns={'Claim\_ID':'Claim ID','Policy\_No':'Policy Number', 'Planned':'Planned Date', 'Actual': 'Actual Date','Created': 'Date Created', 'Amount': 'Claim Amount', 'Paid': 'Paid Amount','Category':'Claim Category', 'Terms': 'Claim Terms', 'Region': 'Claim Region', 'Type': 'Claim Type'})

#create a new variable for claim procesing delay

df['Days Delayed'] = (df['Actual Date'] - df['Planned Date']).dt.days

print("Number of Days dalayed")

print (df['Days Delayed'])

print(df.loc[0])

**Code for Question 4 Insight 1 [Bar Chart - Proportion of claims closed with and without payment]**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

#Read the dataset

df = pd.read\_excel('ECA\_data.xlsx', na\_values = ['', 'Unkn', '???'])

#convert date variables to datetimeformate

df['Planned'] = pd.to\_datetime(df['Planned'], dayfirst=True, errors='coerce')

df['Actual'] = pd.to\_datetime(df['Actual'], dayfirst=True, errors ='coerce')

df['Created'] = pd.to\_datetime(df['Created'], format='%Y%m%d', errors = 'coerce')

#Rename columns for clarity

df = df.rename(columns={'Claim\_ID':'Claim ID','Policy\_No':'Policy Number', 'Planned':'Planned Date', 'Actual': 'Actual Date','Created': 'Date Created', 'Amount': 'Claim Amount', 'Paid': 'Paid Amount','Category':'Claim Category', 'Terms': 'Claim Terms', 'Region': 'Claim Region', 'Type': 'Claim Type'})

#create a new variable for claim procesing delay

df['Days Delayed'] = (df['Actual Date'] - df['Planned Date']).dt.days

print("Number of Days dalayed")

# Calculate the proportion of claims that were closed with and without payment

payment\_counts = df['Paid Amount'].value\_counts(normalize=True)

# Create a bar chart to visualize the proportion of claims closed with and without payment

plt.bar(payment\_counts.index, payment\_counts.values)

plt.xlabel('Payment')

plt.ylabel('Proportion')

plt.title('Proportion of Claims Closed with and without Payment')

plt.show()

**Code for Question 4 Insight 2 [Count Plot - Number of claims varies by policy type]**

import seaborn as sns

import matplotlib.pyplot as plt

sns.countplot(x="Claim Type", data=df)

plt.title('Number of claims varies by policy type')

plt.show()

**Code for Question 4 Insight 3 [ Histogram – for the distribution of claims processing delay]**

#Create a histogram of claim processing delay

plt.hist(df['Days Delayed'], bins=10, range=(0,50))

plt.xlabel('Delay in days')

plt.ylabel('Frequency')

plt.title('Distribution of Claim Processing Delay')

plt.show()

**Code for Question 5**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

#Read the dataset

df = pd.read\_excel('ECA\_data.xlsx', na\_values = ['', 'Unkn', '???'])

#convert date variables to datetimeformate

df['Planned'] = pd.to\_datetime(df['Planned'], dayfirst=True, errors='coerce')

df['Actual'] = pd.to\_datetime(df['Actual'], dayfirst=True, errors ='coerce')

df['Created'] = pd.to\_datetime(df['Created'], format='%Y%m%d', errors = 'coerce')

#Rename columns for clarity

df = df.rename(columns={'Claim\_ID':'Claim ID','Policy\_No':'Policy Number', 'Planned':'Planned Date', 'Actual': 'Actual Date','Created': 'Date Created', 'Amount': 'Claim Amount', 'Paid': 'Paid Amount','Category':'Claim Category', 'Terms': 'Claim Terms', 'Region': 'Claim Region', 'Type': 'Claim Type'})

#create a new variable for claim procesing delay

df['Days Delayed'] = (df['Actual Date'] - df['Planned Date']).dt.days

#Create scatterplot

sns.scatterplot(data=df, x='Planned Date', y='Days Delayed')

plt.title("Planned Date VS Paid Amount")

plt.show()

#To fit a linear regression model, we can use the sklearn library.

from sklearn.linear\_model import LinearRegression

df.replace([np.inf, -np.inf], np.nan, inplace=True)

df.fillna(0, inplace=True)

X = df['Planned Date'].values.reshape(-1, 1)

y = df['Days Delayed'].values.reshape(-1, 1)

# Create and fit linear regression model

model = LinearRegression()

model.fit(X, y)

#Linear equation equation

m = model.coef\_[0][0]

b = model.intercept\_[0]

print(f'y = {m:.2f}x + {b:.2f}')

#form linear equation line in scatter plot

sns.scatterplot(data=df, x='Planned Date', y='Days Delayed')

sns.lineplot(data=df, X='Planned Date', y=model.predict(X).ravel(), color='red')

plt.show()