

**ANL252**

**Python for Data Analytics**

**End-of-Course Assessment** **JANUARY 2023 Presentation**

**Presented by:**

**CHEN YAOGUANGHUA**

**E2279438**

**BSc Finance**

# End-of-Course Assessment (ECA)

## Question 1

**Input Code**

import pandas as pd

# Read ECA dataset, replace blanks, 'Unkn' and '???' under 'Nan' Values

df = pd.read\_csv('ECA.csv', na\_values=['', 'Unkn', '???'])

# Print out varaibles with boolen series to indicate whether it is under 'Nan'

missing = df.isna().any()

print(missing)

**Output**

Claim\_ID True

Policy\_No False

Name False

Planned False

Actual True

Created False

Amount False

Paid False

Category False

Terms True

Region False

Type False

dtype: bool

## Question 2

Before working with the missing data, it is very important for us to know why our data is missing and how it will affect our further analysis. Depending on the type of the missing data, different methods will be applied to it. The following will be my strategies when come in treating for missing data:

1. Delete missing data. Removing rows or columns that contain missing data is a good method only when the amount of data is small and the remaining data are not being manipulated completely. However, the problem is that, the missing data might be related to other variables in the dataset, such can cause information loss and agenda being very biased and unneutral.
2. Replace missing data. Another way to deal with the missing data is to replaces it with an estimate values of data. Replacement of data can be done in a few ways such as mean, median and mode etc. Although such method can help to preserve the sample size and smoothing the analytical process, it can be reflected in a way that it could be biased and uncertain. Not all data sets can be replaced.
3. Indicator variable. In some cases, it may be useful to generate an indicator variable that indicates no data. An indicator variable is a binary variable that equals 1 if the data is missing and 0 if the data is present. Adding indicator variables as additional variable to my analysis can help to identify missing data patterns and their impact on results.

## Question 3

There are a numerous of methods for data preparation, for myself I would prefer to use the following preparation methods:

1. Handling Missing Data:  
   It is very crucial to handle missing values first before any further analysis. One of the most useful method is to delete any rolls that contains any missing values. But this will not be a very effective method especially for those large datasets. Thus, another method to fill the missing values with a suitable value such as number “0”. The “fillna()” method from pandas library can be used to fill those missing values.  
     
   Example code:

import pandas as pd

# Load the dataset

df = pd.read\_csv('ECA.csv')

# Replace missing values in 'Terms' column with the value "0"

replaced\_value = 0

df['Terms'].fillna(replaced\_value, inplace=True)

1. Data Transformation:  
   Data can be transformed to make it more appropriate for further analysis. For example the date variables such as “Planned”, “Actual” and “Created” may need to be converted to datetime format using the “to\_datetime()” method from the pandas library. This transformation makes the data easier to manipulate and analyze.  
     
   Example code:

import pandas as pd

# Load the dataset

data = pd.read\_csv('ECA.csv')

# Convert the 'Planned', 'Actual', and 'Created' columns to datetime format

data['Planned'] = pd.to\_datetime(data['Planned'])

data['Actual'] = pd.to\_datetime(data['Actual'])

data['Created'] = pd.to\_datetime(data['Created'])

1. Data Aggregation:  
   Data aggregation involves grouping data based on certain variables and summarizing the data in a meaningful and insightful presentation. The particular technique can be useful to identify certain patterns and trends in the data. The “groupby()” method from the pandas library can be used to group in one or more variables. Functions such as “mean()”, “sum()” or “count()” can be used to summarize the data.  
     
   Example code:

import pandas as pd

# Load the dataset

data = pd.read\_csv('ECA.csv')

# Group the data by 'Type' and calculate the count of 'Amount'

type\_count = data.groupby('Type')['Amount'].count()

print(type\_count)

## Question 4

1. **Analysis of Claims Settlement Delay**:  
     
   The delaying in claim settlement is a crucial factor that affects the policy holder satisfaction and loyalty. We can do an analysis on the delaying in claims settlement by subtracting the planned settlement date from the actual settlement date.  
     
   The following code and histogram shows the distribution of the claim settlement delay:

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv('ECA.csv')

# calculate claim settlement delay

df['Settlement\_Delay'] = (pd.to\_datetime(df['Actual']) - pd.to\_datetime(df['Planned'])).dt.days

# plot histogram of claim settlement delay

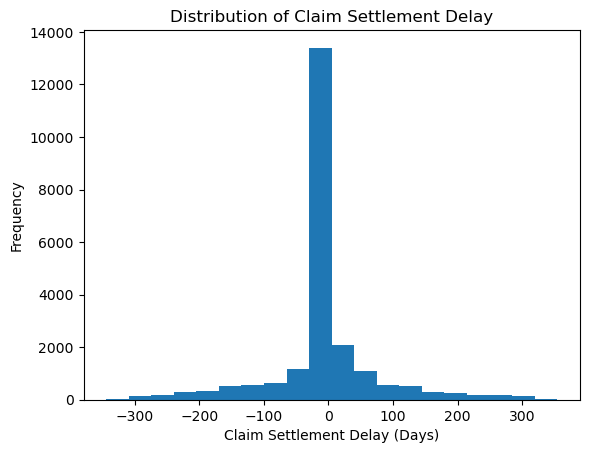
plt.hist(df['Settlement\_Delay'], bins=20)

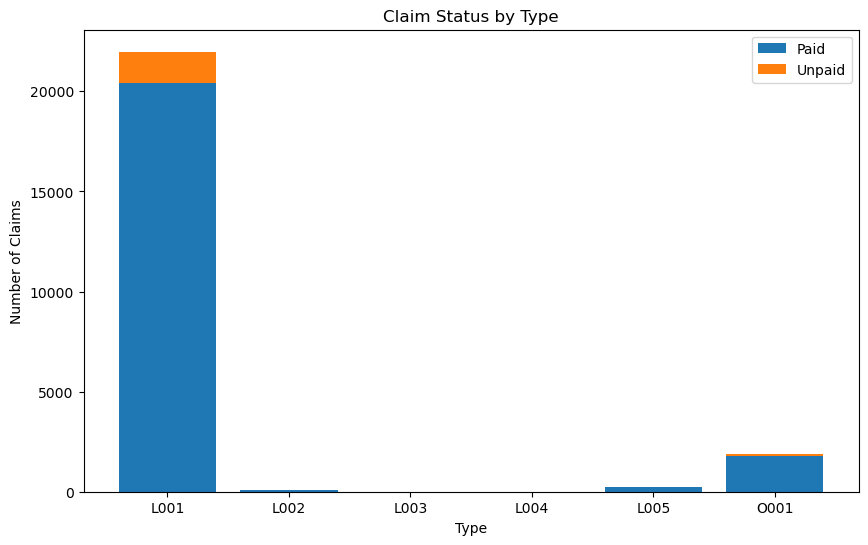
plt.xlabel('Claim Settlement Delay (Days)')

plt.ylabel('Frequency')

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

plt.show()

  
  
The histogram shows that the majority of the claims are settlement within 30 days of the planned settlement date. Some of the claims, may have been postponed for longer than 100 days. These are the claims that should be look into by the insurance provider, and for any necessary actions to be taken to shorten the settlement delay.

1. **Analysis of Payout Status**:  
     
   We can analysis status by types to see which types have the highest number of unpaid claims. To visualize this insight, we can create a stacked bar chart showing the number of paid and unpaid claims from each type.  
     
   The following code and stacked bar chart shows the distribution of the claim status:  
     
     
     
   The visualization shows that “L001” followed by “O001” has the highest amount of unpaid claims, this allows the insurance company to investigate any issues and improve their claims processing procedures (if any) pertaining to those type of claims.

import pandas as pd

import matplotlib.pyplot as plt

data = pd.read\_csv("ECA.csv")

status\_by\_type = data.groupby(["Type", "Paid"])["Claim\_ID"].count().reset\_index()

status\_by\_type = status\_by\_type.pivot(index="Type", columns="Paid", values="Claim\_ID").reset\_index()

status\_by\_type = status\_by\_type.fillna(0)

plt.figure(figsize=(10, 6))

plt.title("Claim Status by Type")

plt.xlabel("Type")

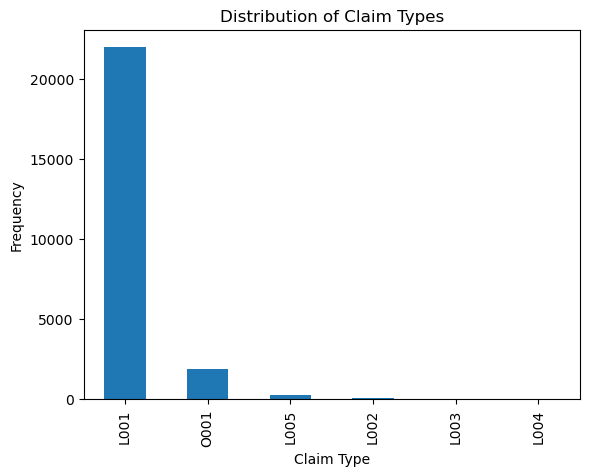
plt.ylabel("Number of Claims")

plt.bar(status\_by\_type["Type"], status\_by\_type["Yes"], label="Paid")

plt.bar(status\_by\_type["Type"], status\_by\_type["No"], bottom=status\_by\_type["Yes"], label="Unpaid")

plt.legend()

plt.show()

1. **Analysis of Claims Types**:  
     
   The claim types analysis will provide insights into the types of claims filed by the policy holders. We can further analyze the claim types by creating a bar plot of the category variable.   
     
   The following code and bar plot shows the distribution of the claim types:  
     
   

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv('ECA.csv')

# create bar plot of claim types

df['Type'].value\_counts().plot(kind='bar')

plt.xlabel('Claim Type')

plt.ylabel('Frequency')

plt.title('Distribution of Claim Types')

plt.show()

The bar plot shows that majority if the claims fall under the type of ‘L001’ followed by ‘O001’. The insurance company should really analyze and investigate the reasons for such high frequency of filings in these types of claims, which also can take actions to reduced such high numbers of claims in the future or do any automation so as to reduce manpower overhead for such matters, also to understand their risk exposure.

## Question 5

Linear Regression Model

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

import seaborn as sns

import warnings

warnings.filterwarnings('ignore')

%matplotlib inline

df = pd.read\_csv('ECA.csv')

df["Delay"] = pd.to\_datetime(df["Actual"]) - pd.to\_datetime(df["Planned"])

df["Delay"] = df["Delay"].dt.days

# check for missing values

display(df.isna().any())

# drop any missing values

df = df.dropna()

# check to make sure it is all removed

df.isna().any()

# define the x & y data

x = df['Delay']

y = df['Planned']

# replacement of dummies data

X\_encoded = pd.get\_dummies(X, columns=["Delay"])

# spliting the dataset into training and testing sets.

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_encoded, y, test\_size=0.3, random\_state=42)

# create linear regression model

lr = LinearRegression()

lr.fit(X\_train, y\_train)

# using trained model tp predict on the test model

y\_pred = lr.predict(X\_test)

# evaluate the performance of the model using the mean squared error (MSE)

# and the R-squared score.

from sklearn.metrics import mean\_squared\_error, r2\_score

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error:", mse)

print("R-squared:", r2)

## Question 6

Unfortunately, I was not able to able to obtain the desired result for the above testing of the results.

However, based on speculation, we can predict that the higher the MSE value is higher in value, this means that the relevant data points are scattered widely around its mean line, compare to the lower value MSE, the data points will be drawn near to the mean line.

For R-squared value, normally ranges from 0 to 1, typically the higher the value, indicates that how well the data fits into the model, whereas the lower value means that the values are spread widely from regression line.

Overall, if the linear regression model is able to predict the delay between the planned and actual settlement date, there will be two possibilities:

Scenario 1 – Low R square value:

There might be varies factors that is affecting the delay that are not captured by the model. Further analysis and adjustments may be needed to improve the performance of the model.

Scenario 2 – High R square value:

This may mean that there is relevance between the planned and actual settlement date, which means the insurance company should focus on these matters and find out what is the issue between such occurrences.

**References:**

Frost, J. (2022) *How to interpret R-squared in regression analysis*, *Statistics By Jim*. Available at: https://statisticsbyjim.com/regression/interpret-r-squared-regression/ (Accessed: March 6, 2023).

Frost, J. (2021) *Mean squared error (MSE)*, *Statistics By Jim*. Available at: https://statisticsbyjim.com/regression/mean-squared-error-mse/#:~:text=Mean%20squared%20error%20(MSE)%20measures,the%20observed%20and%20predicted%20values. (Accessed: March 6, 2023).

*What is data preparation? processes and example* (no date) *Talend*. Available at: https://www.talend.com/resources/what-is-data-preparation/ (Accessed: March 6, 2023).

*SUSS (2023) Study Unit 4, Data Management By SUSS ANL252:*https://canvas.suss.edu.sg/courses/55725/files/7707488?module\_item\_id=633470

*SUSS (2023) Study Unit 5, Data Analytics in Python By SUSS ANL252:*https://canvas.suss.edu.sg/courses/55725/files/7707489?module\_item\_id=633471

**Appendix:**

# APPENDIX 1 – DATA DICTIONARY

|  |  |
| --- | --- |
| **Variable** | **Description** |
| Claim\_ID | Unique identifier of claim |
| Policy\_No | Unique identifier of corporate policy tied to an organization |
| Name | Name of claimant |
| Planned | Planned date of claim settlement |
| Actual | Actual date of claim settlement |
| Created | Claim settlement record creation date |
| Amount | Payout amount |
| Paid | Status of payment (Yes or No) |
| Category | Internal categorization code |
| Terms | Internal terms and conditions code |
| Region | Internal region classification code |
| Type | Internal type classification code |

Note: The meaning of each value for the internal codes of the organization is unknown.