

**ANL252**

**End-of-Course Assessment - January Semester 2023**

# Python for Data Analytics

**End-of-Course Assessment**

**Jazlyn J Binte Abdullah**

**(Z1882515)**

## Please refer to APPENDIX 2 for all lines of code.

## Question 1

Firstly, we will have to import necessary libraries to read the dataset as a Pandas dataframe. By using the df.head() function, it returns the first N rows. This simple test makes sure we have the right type of data in the Dataset. Figure 1 below shows the N rows of the ECA Dataset. The output shows and assures us that we can work with the Dataset.



Figure 1

Since the terms ‘Unkn' and ‘???’ denotes missing values in the set, we need to find the variables that contain these values. The code would return us a series which contains the amount of missing values in every column in the Dataset. As shown in Figure 2 below, the columns “Claim\_ID” and “Actual” contains 5 and 1677 missing values respectively.

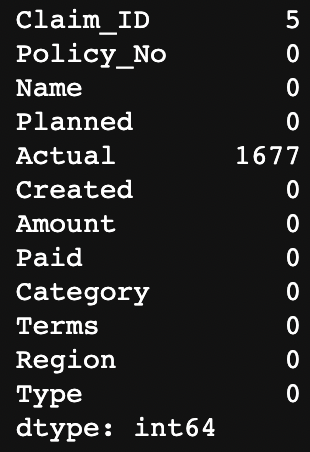
**

Figure 2

Even though we have found that “Claim\_ID” and “Actual” are the columns that contain either ‘Unkn’ or ‘???’, there are still empty or blank cells that we have not found. To find these blanks, we utilize the numpy library. By replacing the missing values with “NaN”, we can easily group them together to count the total values. “NaN” stands for "Not a Number". It represents undefined or unrepresentable value, usually used to denote null values in a dataset.

I would also like to create a more organized way to find out which column the missing values are derived from. In Figure 3 below, the organisation makes it easy to see all the columns that contain missing values. We can observe that the variables, “Claim\_ID”, “Actual” and “Terms” contain 5, 1677 and 7 missing values respectively.

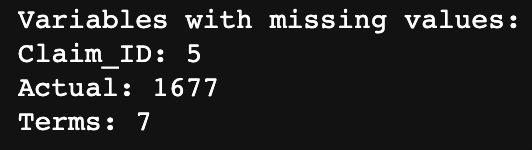


Figure 3

## Question 2

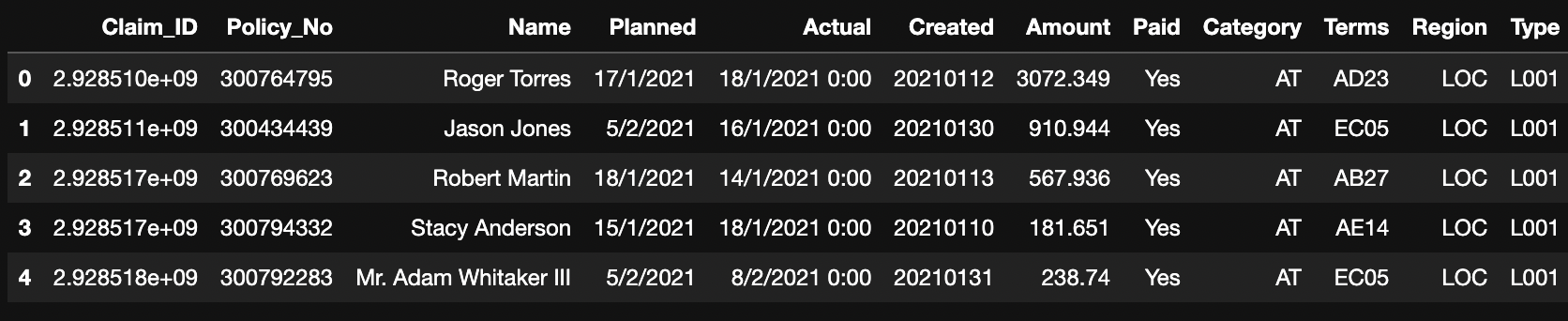


Figure 4

Using the forward-fill method (ffill), this code fills the missing values in the 'Actual' column with the last valid observation and updates the DataFrame accordingly.

Then, we filter the ‘Terms’ column by removing all cells that contain ‘Unkn’ and ‘???’. This creates a new DataFrame that consists of proper values for the ‘Terms’ column and removes all of the ‘Unkn’ and ‘???’ data.

Aside from ‘Unkn’ and ‘???’, we also have to remove values that are undefined, or NaN. By removing the rows that contain NaN values, this will properly remove all the missing values from the DataFrame.

After all the filtering, the end result of the DataFrame will only contain the rows that have no missing values, with an example shown in the Figure 4 above.

Nearly all real-world research encounters the issue of missing data, which can significantly affect the inferences and conclusions that can be drawn from the data. To conduct proper data analysis, it is important to remove null, missing or NaN values from a DataFrame in Python. There are several reasons why:

Missing data can cause errors and inconsistencies in an analysis. NaN values in a DataFrame might result in errors in machine learning models, visualisations, and calculations. Null data can skew the results and reduce their reliability. Aside from that, many machine learning algorithms do not support data with missing values. (Brownlee, 2020)

Missing data can also reduce the sample size. When null data is present in a DataFrame, it reduces the sample size for analysis, as there would be less usable data. This can make the conclusions drawn from the analysis to be less representative of the population. It may cause bias and a reduction in efficiency and accuracy (Madley-Dowd et al., 2019)

Many Python libraries for data analysis, visualization, and machine learning do not support nor recognise null data. Therefore, removing null datasets can make it easier to work with these libraries, so as to conduct a more efficient analysis.

## Question 3

To further analyse the data, we can implement three other data preparation techniques.

1. **Removing unnecessary variables**

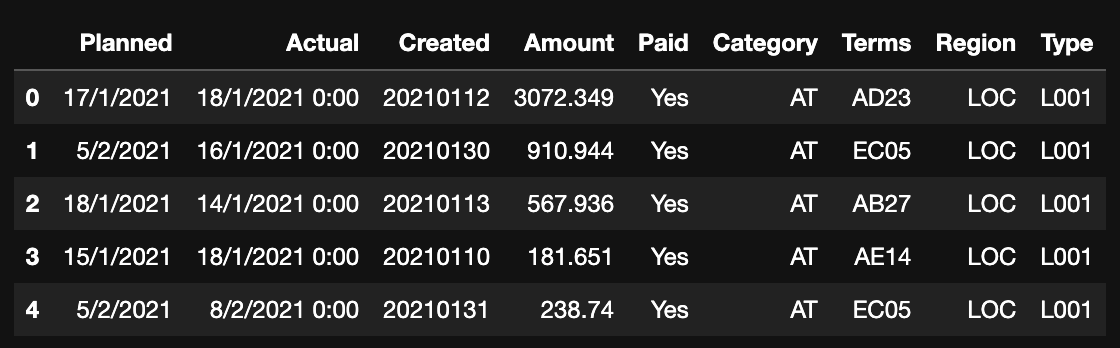


Figure 5

In a Dataset, not all data is actually usable as it may not be of interest for a project or analysis. Removing unneeded or unnecessary variables makes it easier to focus on the variables that we need to work with. (Banghart, 2019) I removed the “Claim\_ID”, “Policy\_No” and “Name” columns as they did not serve a specific purpose and will not aid the analysis.

1. **Converting date columns to datetime format**

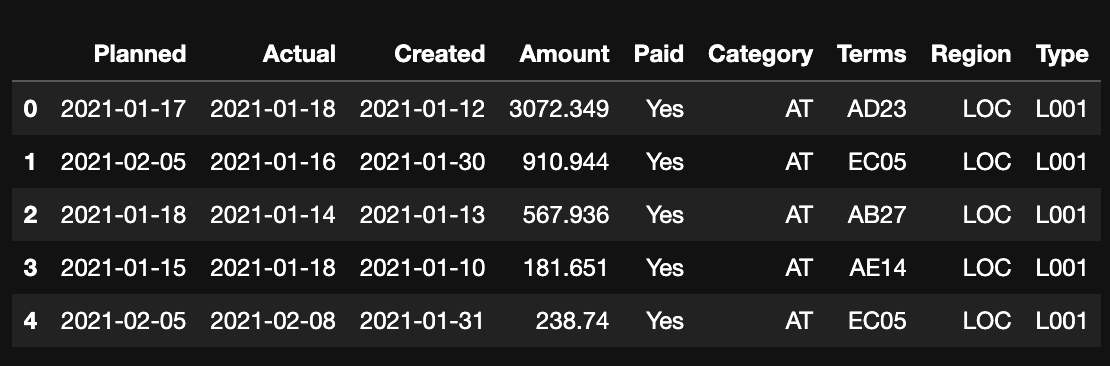


Figure 6

The dataset consists of three columns that uses dates: Planned, Actual, and Created. To perform any time-series analysis, it is essential to convert these columns to datetime format. We can use the pandas.to\_datetime method to convert these columns to datetime format.

After converting these columns to datetime format, we can observe that in Figure 5, the dates under ‘Planned’, ‘Actual’ and ‘Created’ are all following a standard format. Following a standard format not only makes it easier for Python libraries to work with, it also allows users of the worksheet to clearly see and define the data. With respect to making decisions, it allows users to quickly see all important information. (Org, n.d)

1. **Encoding Categorical Variables**

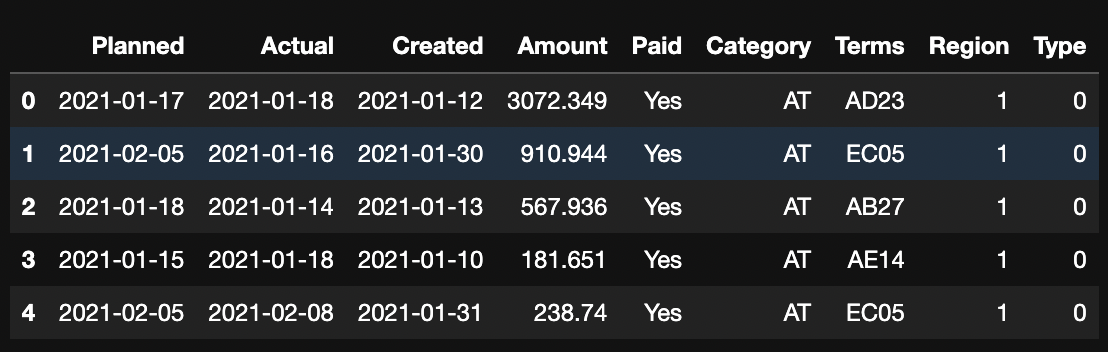


Figure 7

The encoded values are integers that start from 0 and go up to the number of unique values depending on how much data we have in a certain column. Since the “Region” and “Type” columns consist of non-numerical labels, they can be changed to numerical labels should I need to use them during data analysis. I chose to use LabelEncoder as there are only 2 Regions, making them 1 = LOC and 0 = FVS. For the “Type” column, since there are 6 different Types, it would make them 0 = L001, 1 = L002, 2 = L003, 3 = L004, 4 = L005, and 5 = O001.

By doing this we are able to better use these values for data analysis as they are now numerical labels.

## Question 4

**Insight 1: Most claims are paid within the planned timeframe**

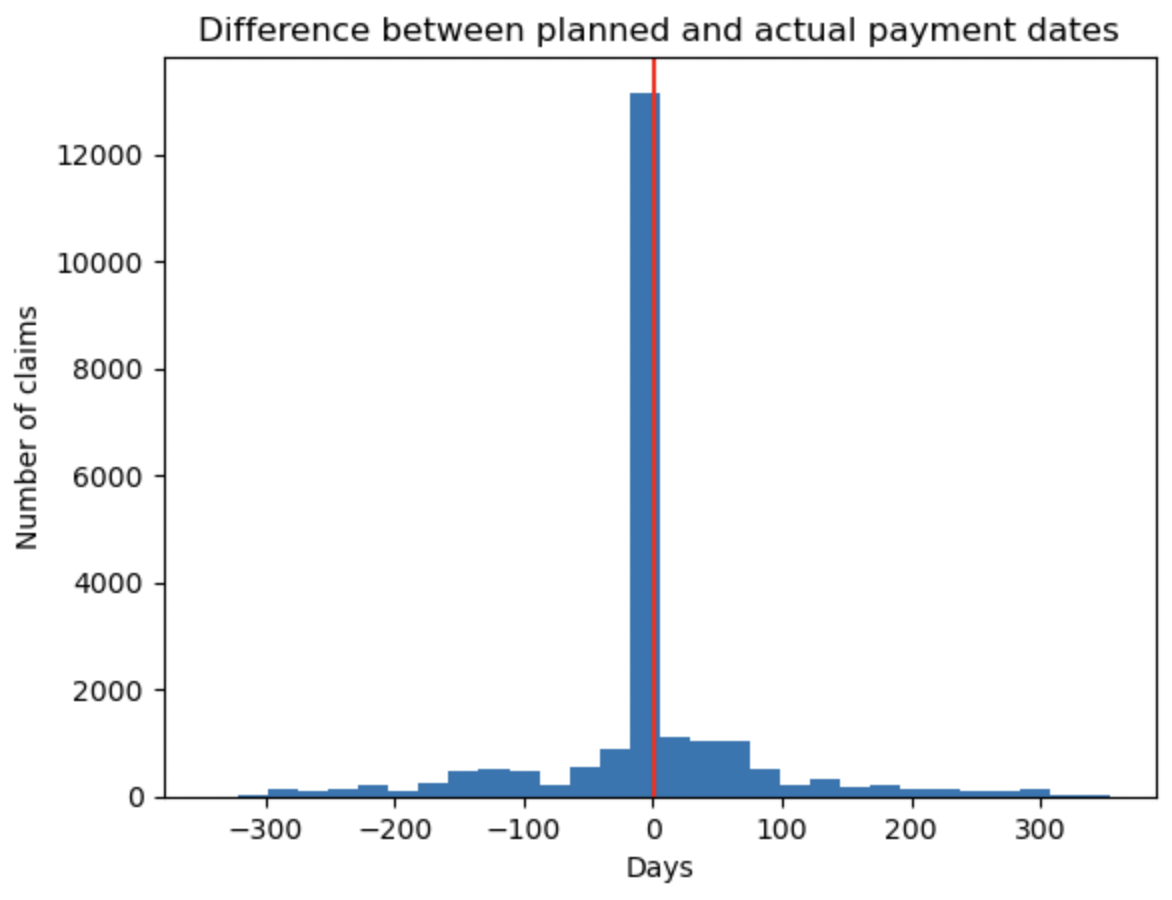
****

Figure 8

By plotting a histogram, it gives us a clear visualization of when claims are paid. We can see that in Figure 8, it is found that a significant number of claims are paid within the planned timeframe. Finding the difference between planned and actual payment dates will show us how many days the insurance company took to payout to their policyholders.

When most insurance claims are paid within the planned timeframe, it reflects positively on the insurance company's efficiency, reliability, and commitment to their policyholders. This suggests that the company has a very efficient system and process to ensure accurate handling of claims.

Timely settlement of claims help builds trust and confidence with policyholders. This can result in higher customer satisfaction and retention rates. It can also boost the company’s reputation to draw in new clients, which will increase business growth.

As there are still claims that are not processed on time, the company can look into the reasons and how to improve these numbers. It could be job satisfaction issue or a system issue. If their employees feel low job satisfaction, resulting in claims not being processed on time, the company can look into what can be done about it. If it is a system issue where claims take a while to be processed, the company can look into other systems or alternatives to speed up the process.

**Insight 2: The amount paid for claims are mostly less than $5000**

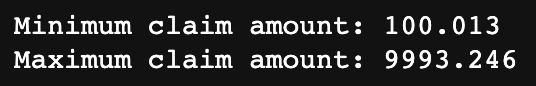
****

Figure 9

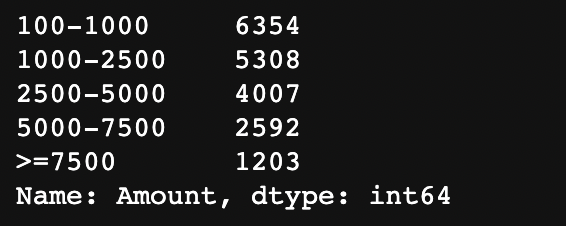


Figure 10

First we find the minimum and maximum claim amounts. In Figure 9 we can see that the minimum claim amount is $100.013 and the maximum claim amount is $9993.246. This is a difference of $9893.233. This is a rather significant difference in amounts, but this tell us that the insurance company handles many different claims from small to large.

Not only that, it shows us that the insurance company handles many different types of policies a well. This allows the company to offer customized policies that are tailored to the specific needs of their customers. By offering a wide range of policies, the insurance company can cater to the diverse needs of a large customer base, which is not just limited to individuals. This can include businesses and organisations as well.

As customers are more likely to choose an insurance company that can provide tailored coverage, by offering customised policies, this can increase customer satisfaction and loyalty, which in turns may result in higher customer retention rates. If customers are satisfied, they are less likely to switch to another insurance provider that may not offer the same level of customisation.

Handling different types of policies can provide the insurance company with a diversified portfolio as well. This can help mitigate risks and minimize losses. For instance, if the company suddenly experiences high claims for one type of policy, they can offset those losses with the profits from other policies. This can ensure the insurance company's financial stability and to allow them to continue offering high-quality services.

In Figure 10, it tells us that a significant number of their claims are below $5000. Only 1203 claims are above $7500. This is such a positive insight for the insurance company. Knowing that most claims are below $5000, this can reduce financial risk, lower administrative costs and attract more customers. Let me elaborate further.

**Lower Financial Risk**

Insurance companies operate by collecting premiums from policyholders and using that money to pay out claims whenever claims occur. If a majority of claims are below $5000, the financial risk for the insurance company is relatively low. This means that the company is less likely to experience significant financial losses.

**Lower Administrative Costs**

Processing insurance claims can be a time-consuming and expensive process for insurance companies. If most claims are below $5000, it means that the process is easier and less costly which allows the claims to be paid out quick. With faster processing times for claims, this could lead to a higher customer satisfaction.

**Attractive to Customers**

Customers are more enticed to purchase insurance policies if they believe that the claims process is straightforward. It would appeal to customers more if they believe that they are able to receive adequate compensation in the event of a claim as well.

**Insight 3: The LOC Region (1) made more payments than the FVS Region (0)**

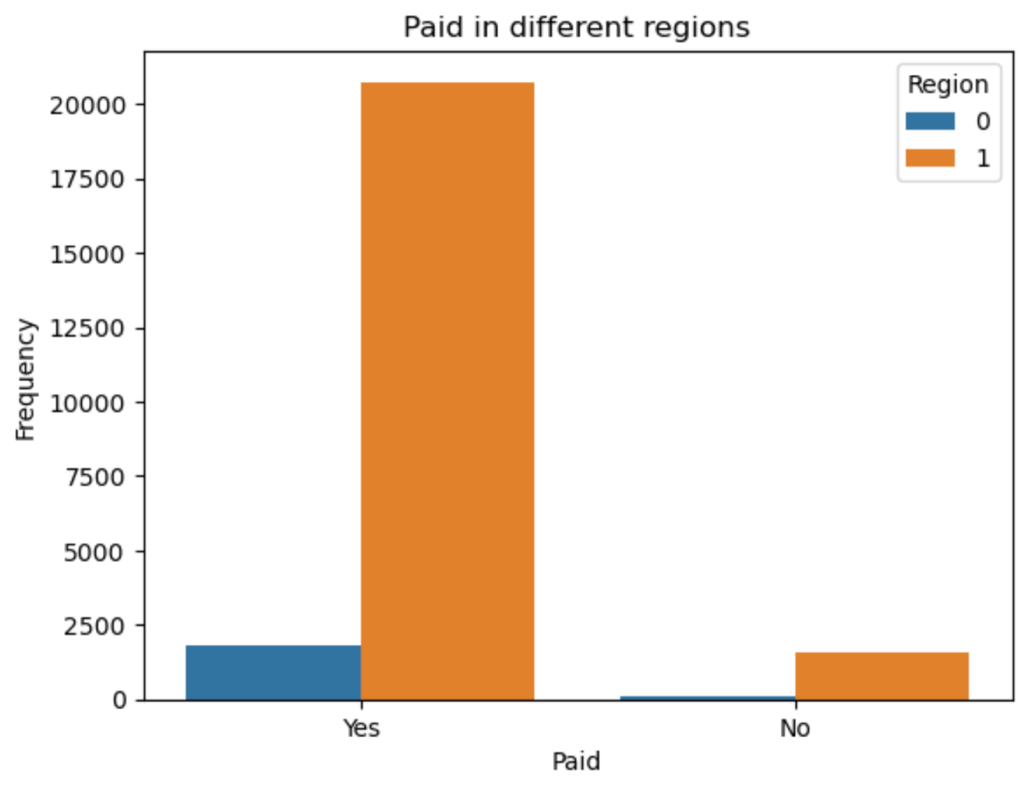


Figure 11

From Figure 11, we can see that Region 1, LOC, made the most payouts.

This suggests that there may be regional differences in the types of risks that policyholders in each region face.

LOC region may have a higher concentration of policyholders who work in high-risk industries, such as construction or manufacturing, which could result in a higher number of claims related to workplace injuries. As claims are mostly below $5000, we can assume that they are not property damage type claims as the amounts would be significantly higher.

Understanding this claims pattern can be valuable for the insurance company in several ways.

Firstly, understanding the pattern can allow the company to tailor their insurance policies and pricing to reflect the risks specific to each region. Higher-risk regions pay higher premiums. This can ensure that policyholders in each region are sufficiently covered, while minimizing the risk of financial losses for the company.

Secondly, the company is able to allocate resources such as claims adjusters, more effectively. Since LOC region is experiencing a higher volume of claims, the company can allocate more resources or manpower to that region to ensure that claims are processed more efficiently.

In general, understanding the differences in claims patterns for different regions can help the insurance company to make more informed decisions on managing risks and allocating resources better.

## Question 5

## 

Figure 12

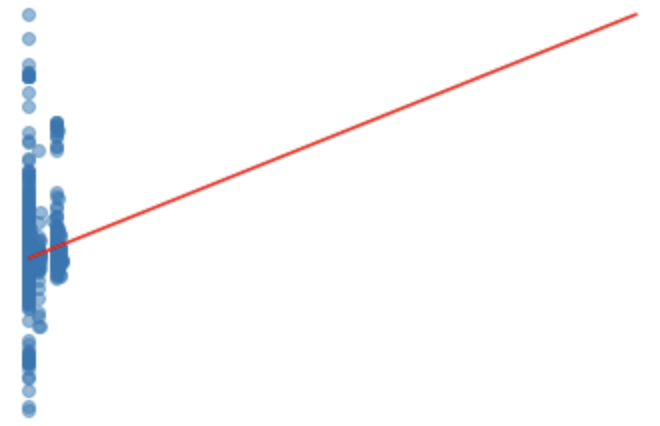


Figure 13

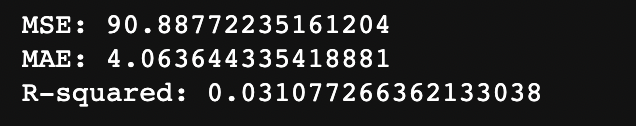


Figure 14

In Figure 12, we can see that the negative amounts refer to claims that have been processed earlier than planned. In Data 1, we can see that the Planned date is 05-02-2021 but it was processed 20 days earlier, on 16-01-2021.

Figure 13 is the scatter plot that shows the Delay in days processed.

In Figure 14 we can see that the MSE is 90.88772235, the MAE is 4.06364433and the R-squared is 0.03017726.

## Question 6

The results obtained from the linear regression modelling indicate that the model has a low level of accuracy in predicting the delay in processing claims. The mean squared error (MSE) of 90.88 and mean absolute error (MAE) of 4.06 suggest that the predicted delays are on average around 4 days off from the actual delays.

Additionally, the low R-squared value of 0.03 indicates that only a small portion of the variation in delay can be explained by the features used in the model.

The linear regression equation can be stated as:

Delay = -0.215(Amount) + 0.109(Region) + 0.528\*(Type) - 8.721

This equation suggests that as the amount of the claim decreases, the delay in processing the claim is expected to increase. Additionally, the Region and Type of claim also have a positive effect on the delay, with claims from certain regions and of certain types experiencing longer delays. However, it is important to note that the low accuracy of the model suggests that these relationships may not be reliable for making accurate predictions.

The general linear regression equation can be written as:

y = β0 + β1x1 + β2x2 + ... + βnxn + ε

where:

y is the dependent variable (in this case, the delay in days) β0 is the intercept β1, β2, ..., βn are the coefficients for the independent variables x1, x2, ..., xn ε is the error term

**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.

**APPENDIX 2 – LINES OF CODE**

**Figure 1**

#importing necessary library/libraries

import pandas as pd

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

#making sure data is usable

df.head()

**Figure 2**

#checking for missing values

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

print(missing\_values)

**Figure 3**

#importing necessary library/libraries

import numpy as np

#replace missing values with NaN

df.replace(['', 'Unkn', '???'], np.nan, inplace=True)

#checking for missing values

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

#make the missing values clear to any user

print("Variables with missing values:")

for column, value in missing\_values.items():

if value > 0:

print(f"{column}: {value}")

**Figure 4**

df['Actual'].fillna(method="ffill",inplace=True)

#trying to handle the missing values

df = df[df['Terms'] != 'Unkn']

df = df[df['Terms'] != '???']

#removing the NaN values

df.dropna(inplace=True)

**Figure 5**

df.drop(['Claim\_ID','Policy\_No','Name'],axis=1, inplace=True)

df.head()

**Figure 6**

df['Planned'] = pd.to\_datetime(df['Planned'], format='%d/%m/%Y')

df['Actual'] = pd.to\_datetime(df['Actual'], format='%d/%m/%Y %H:%M')

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

df.head()

**Figure 7**

from sklearn.preprocessing import LabelEncoder

#create a LabelEncoder object

le = LabelEncoder()

#encode the 'Region' column

df['Region'] = le.fit\_transform(df['Region'])

#encode the 'Type' column

df['Type'] = le.fit\_transform(df['Type'])

**Figure 8**

import matplotlib.pyplot as plt

df1=pd.DataFrame()

# Insight 1: Most claims are paid within the planned timeframe

df1['planned\_diff'] = (pd.to\_datetime(df['Actual']) - pd.to\_datetime(df['Planned'])).dt.days

plt.hist(df1['planned\_diff'], bins=30)

plt.axvline(x=1, color='red')

plt.title('Difference between planned and actual payment dates')

plt.xlabel('Days')

plt.ylabel('Number of claims')

plt.show()

**Figure 9**

#find minimum & maximum claim amounts

min\_amount = df['Amount'].min()

max\_amount = df['Amount'].max()

print('Minimum claim amount:', min\_amount)

print('Maximum claim amount:', max\_amount)

**Figure 10**

#define ranges for each slab

ranges = [100, 1000, 2500, 5000, 7500, 10000]

df['Amount'] = pd.to\_numeric(df['Amount'], errors='coerce')

# Create the slabs

slabs = pd.cut(df['Amount'], ranges, labels=[ '100-1000', '1000-2500', '2500-5000', '5000-7500', '>=7500'])

#count the number of claims in each slab

slab\_counts = slabs.value\_counts()

#print the slab details

print(slab\_counts)

**Figure 11**

#visualise the data paid by each region

sns.countplot(x=df["Paid"], hue = df["Region"])

plt.xlabel("Paid")

plt.ylabel("Frequency")

plt.title("Paid in different regions")

plt.show()

**Figure 12**

#calculate delay in days

df['Delay'] = (df['Actual'] - df['Planned']).dt.days

#checking 10 rows

df.head(10)

**Figure 13**

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

from sklearn.linear\_model import LinearRegression

# Replace missing values with column medians

df = df.fillna(df.median())

#df.dropna()

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(df.drop(['Claim\_ID', 'Policy\_No', 'Name', 'Planned', 'Actual', 'Created', 'Delay'], axis=1), df['Delay'], test\_size=0.2, random\_state=42)

# Build linear regression model

lr = LinearRegression()

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

# Predict delay for test set

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

import matplotlib.pyplot as plt

#create scatter plot of predicted vs actual delay

plt.scatter(y\_pred, y\_test, alpha=0.5)

# Add a line representing perfect predictions

plt.plot([0, max(y\_test)], [0, max(y\_test)], color='red')

# Add labels and title

plt.xlabel('Predicted Delay (days)')

plt.ylabel('Actual Delay (days)')

plt.title('Predicted vs Actual Delay')

# Show the plot

plt.show()

**REFERENCES**

Brownlee, J. (2020, August 27). *How to handle missing data with python*. MachineLearningMastery.com. Retrieved March 5, 2023, from https://machinelearningmastery.com/handle-missing-data-python/

Madley-Dowd, P., Hughes, R., Tilling, K., & Heron, J. (2019, June 1). *The proportion of missing data should not be used to guide decisions on multiple imputation*. Journal of clinical epidemiology. Retrieved March 5, 2023, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6547017/#:~:text=Missing%20data%20is%20a%20common,cause%20a%20reduction%20in%20efficiency.

Org, S. (n.d.). Formatting and data analysis. Retrieved March 5, 2023, from https://saylordotorg.github.io/text\_how-to-use-microsoft-excel-v1.1/s05-03-formatting-and-data-analysis.html

Banghart, M. (2019, November 8). *Data wrangling essentials*. 4.4 Dropping unneeded variables. Retrieved March 5, 2023, from https://sscc.wisc.edu/sscc/pubs/DWE/book/4-4-dropping-unneeded-variables.html

**----- END OF ECA PAPER -----**

of