****

**ANL252**

**Python For Data Analytics**

**End-of-Course Assessment (ECA)**

**July 2021 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **LOO CHENKAI** | **Y2071132** |

**Submission Date:** 29 August 2021

**Question 1**

**Q1 A(i)**

# #Q1 A(i)

# Load pandas

import pandas as pd

import numpy as np

# Read CSV file into DataFrame df

ship = pd.read\_csv(r'ship.csv', index\_col=0)

#replace . with "no value"

ship = ship.replace(".", 'no value').reset\_index()

# Show dataframe

ship

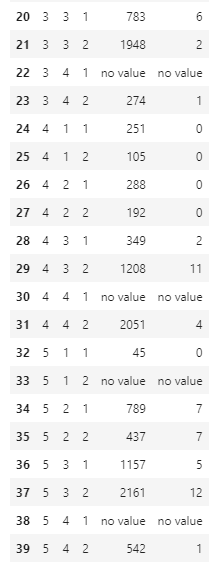
**Screenshot for Q1Ai:**

**Graphical user interface

Description automatically generated with medium confidence**

**Table

Description automatically generated with medium confidence**

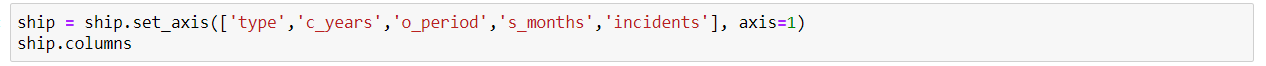


**Q1A(ii)**

ship = ship.set\_axis(['type','c\_years','o\_period','s\_months','incidents'], axis=1)

ship.columns

**Screenshot for Q1A(ii):**

****

****

**Q1 A(iii)**

#Makes table of average of cross product of categories

table = list()

for i in range(5):

cur1 = pd.DataFrame(ship.loc[(ship['type'] == i+1)]['incidents'])

cur1\_ = pd.DataFrame(ship.loc[(ship['type'] == i+1)]['o\_period'])

for j in range(2):

table.append([])

cur2 = ship.loc[(ship['o\_period'] == j+1)]['incidents']

cur2\_ = ship.loc[(ship['o\_period'] == j+1)]['o\_period']

mean = np.mean(np.mean(cur1.merge(cur2, how='cross').astype(int)))

mean\_ = np.mean(np.mean(cur1\_.merge(cur2\_, how='cross').astype(int)))

table[-1].append(i)

table[-1].append(j)

table[-1].append(mean)

table[-1].append(mean\_)

shipgroup = pd.DataFrame(table)

shipgroup.columns = ['type','o\_period','incidents','o\_period']

shipgroup

**Screenshot for Q1A(iii):**

**Text

Description automatically generated**

**Table

Description automatically generated**

**Q1 A(iv)**

#iterates through ship to find where s\_months and icidents is "no\_value"

missing\_df = ship.loc[(ship['s\_months'] == "no value")]

for i in range(len(missing\_df)):

#Find rows where data is missing

missing = (int(missing\_df['type'][i:i+1]),int(missing\_df['o\_period'][i:i+1]))

a = ship.loc[ship['type'] == missing[0]]

a = a.loc[a['o\_period'] == missing[1]]

a = a.loc[a['s\_months'] != "no value"]

#gets average of datapoints with same o\_period and type values and updates dataframe

av = sum(list(map(int, list(a['incidents']))))/len(a)

ship['incidents'] = ship.mask((ship['o\_period'] == missing[1]) & (ship['type'] == missing[0]) & (ship['incidents'] == "no value"), av)['incidents']

#does same update to s\_months

av = sum(list(map(int, list(a['s\_months']))))/len(a)

ship['s\_months'] = ship.mask((ship['o\_period'] == missing[1]) & (ship['type'] == missing[0]) & (ship['s\_months'] == "no value"), av)['s\_months']

ship

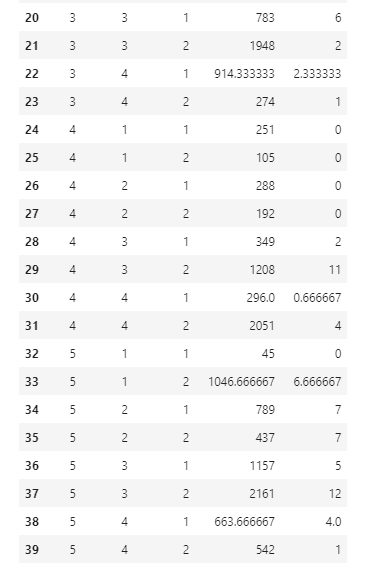
**Screenshot for Q1A(iv):**

**Text

Description automatically generated with low confidence**

**A picture containing table

Description automatically generated**



**Q1 A(v)**

Y = ship['incidents'].astype("float")

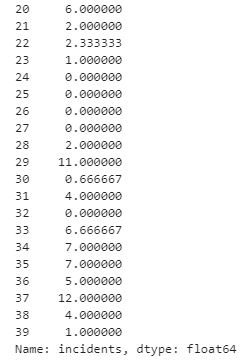
Y

**Screenshot for Q1 A(v):**

****

**Table

Description automatically generated with medium confidence**



**Q1 B(i)**

I

#turn column types into categories

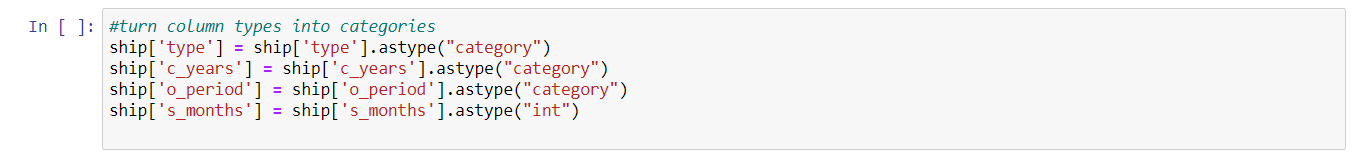
ship['type'] = ship['type'].astype("category")

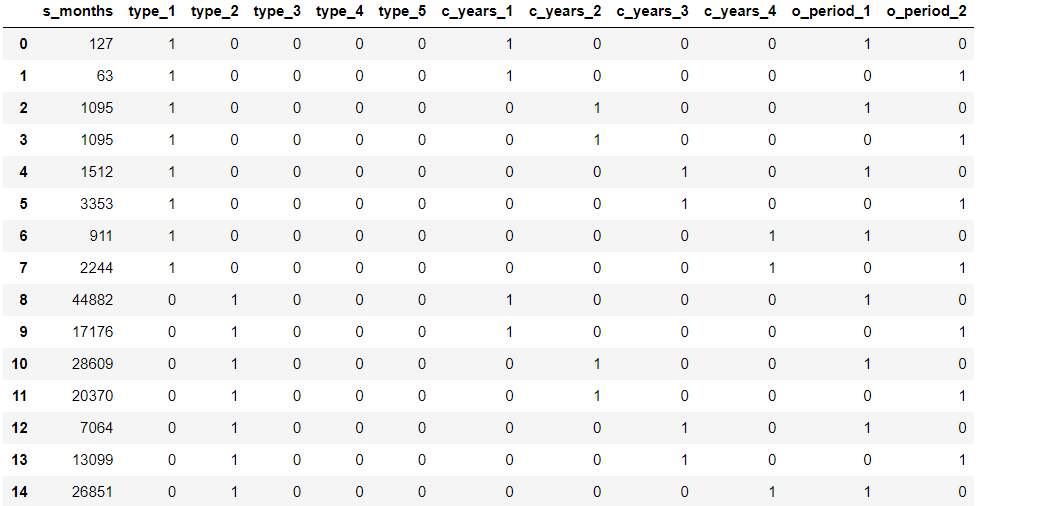
ship['c\_years'] = ship['c\_years'].astype("category")

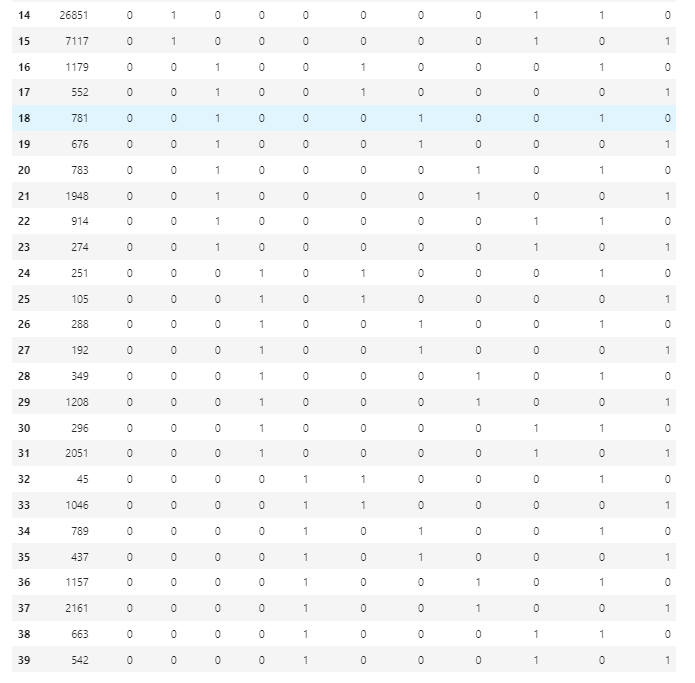
ship['o\_period'] = ship['o\_period'].astype("category")

ship['s\_months'] = ship['s\_months'].astype("int")

**Screenshot for Q1B(i):**

****

****



**Q1 B(ii)**

#get dummy variables

X = ship.copy()

X = pd.get\_dummies(ship, columns=['type','c\_years','o\_period'])

del X['incidents']

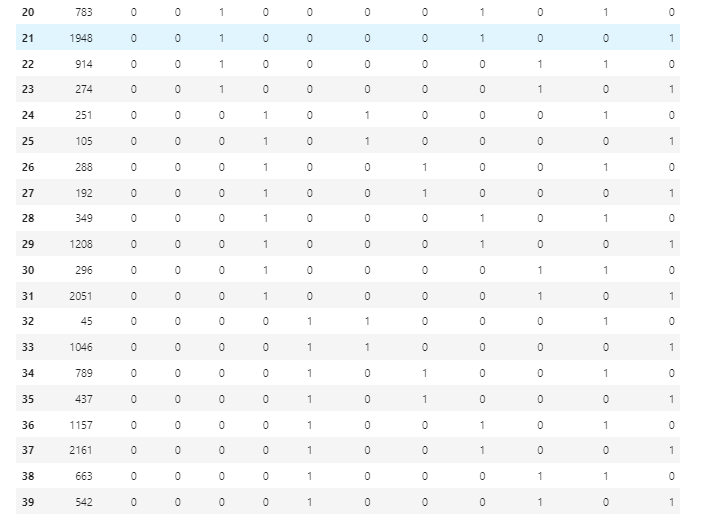
X

**Screenshot for Q1B(ii):**

****

**Application

Description automatically generated with low confidence**



**Q1 B(iii)**

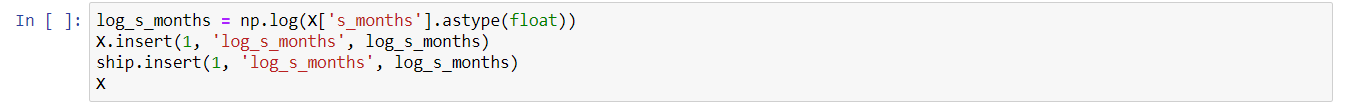
log\_s\_months = np.log(X['s\_months'].astype(float))

X.insert(1, 'log\_s\_months', log\_s\_months)

ship.insert(1, 'log\_s\_months', log\_s\_months)

X

**Screenshot:**

****

**Application

Description automatically generated with low confidence**



**Q1(C)**

This dataset is unusually small. It contains only 39 datapoints. One downside to this is that there is a chance that this dataset is biased. Splitting this into a training and testing set will only make this issue worse by further shrinking it. Doing a traditional 80-20 split will make the dataset only 32 points and make the data even more reliable. And when testing a model, it is important for the testing set to be unbiased as well, but the testing set will only contain around 7 points, drastically reducing the reliability of it.

**Q1(D)**

from sqlalchemy import create\_engine

engine = create\_engine('sqlite://', echo=False)

#Save dataframe as csv

ship.to\_csv('shipdf.csv')

#Save dataframe as db file

ship.to\_sql('shipdb.db', con=engine)

**Screenshot for Q1D:**

**Text, letter

Description automatically generated**

**Question 2**

**Q2 A)**

The Sklearn.linear\_model.PoissonRegressor class is one which can create a poisson regression model. In order to train this model on any data, the fit() function must be used. The input and output is passed to this function, and the model calculates its coefficients given the data. Predict() is used once the model is fitted onto the dataset. It takes in data and returns an estimation of the correct output.

**Q2 B)**

from sklearn import linear\_model

from sklearn import preprocessing

#Normalize Data

X\_ = (X-X.mean())/X.std()

#Fit model to data

reg = linear\_model.PoissonRegressor(max\_iter = 1000000)

reg.fit(X\_, Y)

reg.coef\_

**Screenshot for Q2B:**

**A picture containing chart

Description automatically generated**

**Text

Description automatically generated**

**Q2 C)**

#create function to calculate deviance

def dev(y, y\_hat):

s = y \* np.log(y/y\_hat) - (y - y\_hat)

return np.sum(s) \* 2

dev(Y,reg.predict(X\_))

**Screenshot for Q2C:**

**Text, whiteboard

Description automatically generated**



**Embed Jupyter file:**

****

**Screenshot of all codes:**

**Graphical user interface, text, application

Description automatically generatedText

Description automatically generated with low confidenceText

Description automatically generatedA picture containing text

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

**Graphical user interface, text, application, email

Description automatically generated**

