

**ANL252**

**Python for Data Analytics**

# **End-of-Course Assessment**

**July 2021 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **CHUNG WEI TAT** | **Y2070638** |

**Tutorial Group: T 09**

**Submission Date: 12/09/2021**

**Question 1**

**#(a)**

**(i)**

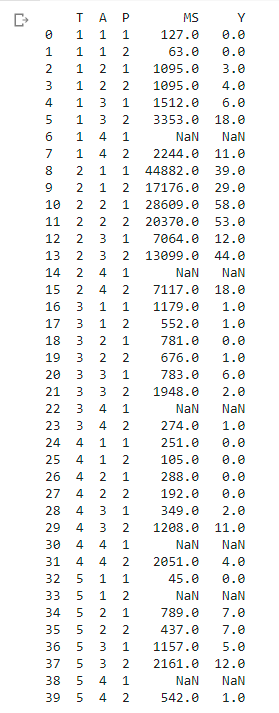
import numpy as np  # useful for many scientific computing in Python

import pandas as pd # primary data structure library

# detect "." as na\_values and na\_filter = true convert to Nan

ship = pd.read\_csv("ship.csv", na\_values = ".", na\_filter = True)

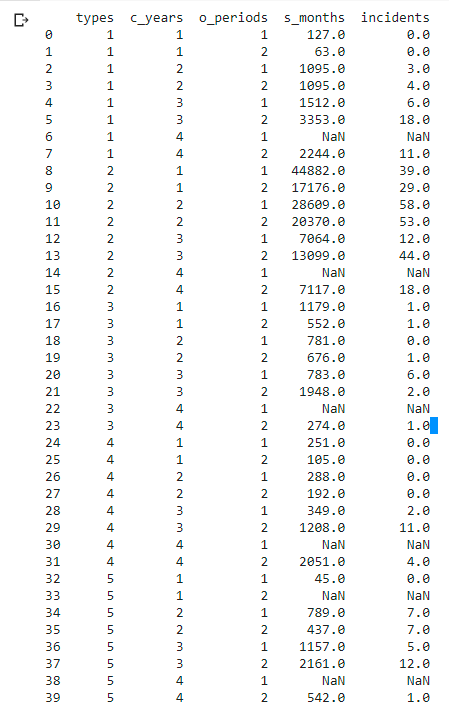
print(ship)



**#(ii)**

ship.rename(columns={'T':'types', 'A':'c\_years', 'P':'o\_periods', 'MS':'s\_months', 'Y':'incidents'}, inplace=True)

print(ship)



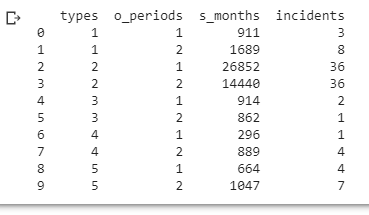
**#(iii)**

means\_months = ship.groupby(['types', 'o\_periods'])['s\_months'].mean().reset\_index().round().astype('int') # get avg service months + round off nearest int

means\_incidents = ship.groupby(['types', 'o\_periods'])['incidents'].mean().reset\_index().round().astype('int') # get avg operation periods + round off nearest int

shipgroup = means\_months.merge(means\_incidents) # merge two diff df together

print (shipgroup)



**#(iv)**

empty\_list = ship.isnull().any(axis = 1)

empty\_list\_row = empty\_list[empty\_list == True].index

print (empty\_list\_row)

for i in empty\_list\_row:

    row = ship.loc[i]

    find\_type = row['types'] # type of row contain empty

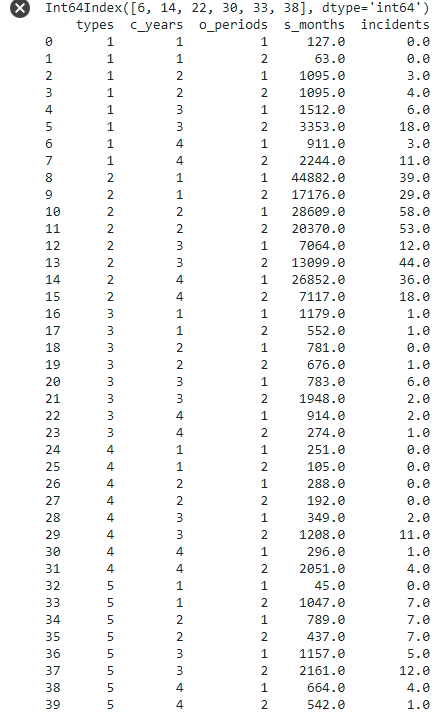
    find\_period = row['o\_periods'] # period of row contain empty

    hold = shipgroup.loc[(shipgroup['types'] == find\_type) & (shipgroup['o\_periods'] == find\_period)] # get data where types + o\_periods match

    ship.loc[[i], ['s\_months']] = hold['s\_months'].item() # get value only from dataframe and set

    ship.loc[[i], ['incidents']] = hold['incidents'].item() # get value only from dataframe and set

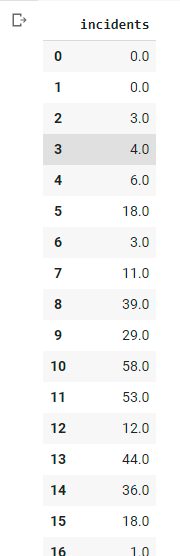
print (ship)



**#(v)**

Y = pd.DataFrame(ship['incidents']) #save incidents in dataframe

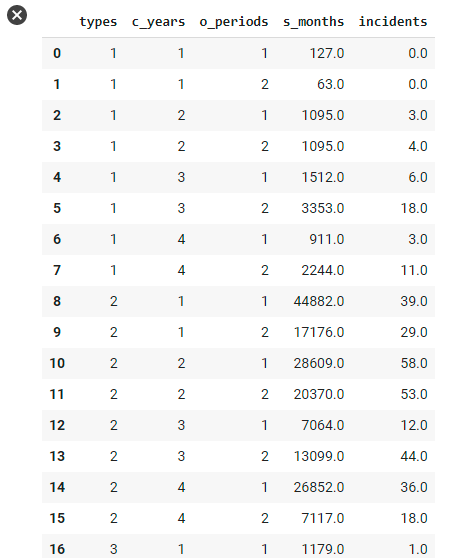
Y



**#(b)**

**#(i)**

ship.astype({'types': 'category', 'c\_years': 'category', 'o\_periods': 'category', 's\_months': 'str', 'incidents': 'str'}) # data conversion to category type

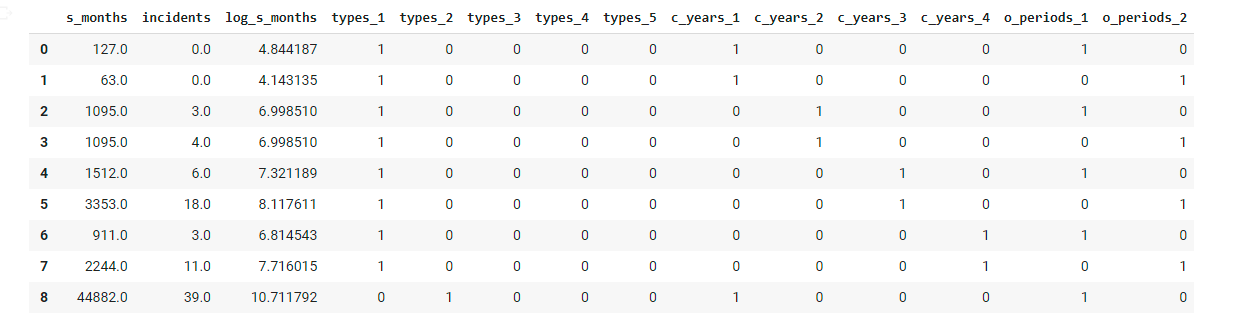


**#(ii)**

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

# convert to dummy variables

X



**#(iii)**

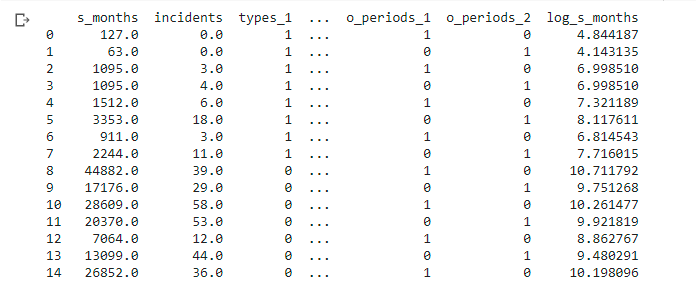
X['log\_s\_months'] = np.log(X['s\_months']) # log transformation and attach to X

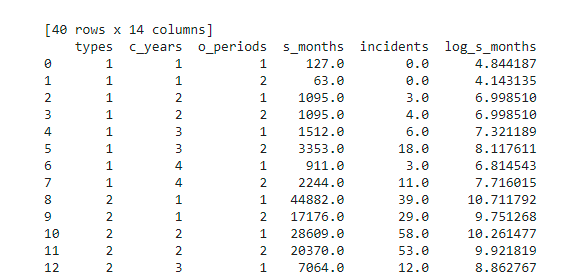
print(X)

ship['log\_s\_months'] = np.log(ship['s\_months'])  # log transformation and attach to ship

print(ship)

")





**#(c)**

The objective of this data is to find whether the number of ship damage incidents is related to the various individual variables. In this case, missing any one of the data will render the other data useless. For example, having just the ship type and the number of occurence for incidents is not beneficial as other factors such as duration of it in operation could have relevance to it. Unless our goal is to determine, for example, the quality of the ships that a company produce then it is ideal to split. Using the entire dataset can show use the correlation between the ship type, construction year and operation period.

**#(d)**

ship.to\_csv('ship\_prepared.csv', index = False) # export to csv

import sqlite3

conn = sqlite3.connect('ship.db')

cur = conn.cursor()

ship.to\_sql("ship", conn, if\_exists = "replace", index = False) # export to db

cur.execute("Select \* FROM ship;")

print(cur.fetchone()) # check if success export



**Question 2**

**#(a)**

Poisson regression is a type of generalised linear models. The module uses the vector of the as coefficient and the first coefficients as intercept.

The estimator determines how we want to segregate our data depending on what we want to achieve, such as predicting a quantity or category, followed by the amount of data available to build machine learning. In short, choose the best data model to learn based on reality or training data.

Fit is used with estimator. The fit function takes in parameters such as data which for unsupervised learning can be one array or for supervised learning, it can be two arrays. It also takes in labels for the output data. It is used to detect whether the estimator works but including additional parameters such as training data to show how well the algorithms fit the model.

Predict function takes in the data that needs to be processed by fit function. It runs for a number of iterations, depending on the number entered through the parameter. It is used to see if any of the fitted data gives back values that it out of the algorithm.

**#(b)**

!pip install scikit-learn # install scikit

from sklearn import preprocessing

from sklearn.linear\_model import PoissonRegressor

# X = list of data

# Y = list of incidents

X = X.to\_numpy() # convert to array form

Y = list(Y['incidents']) # convert to list

X = preprocessing.normalize(X) # normalize to prevent overflow in expression

clf = PoissonRegressor()

clf.fit(X, Y) # fit

coef = clf.coef\_

coef\_df = pd.DataFrame(coef) # convert array into dataframe

coef\_df.rename(columns={0:'Coefficients'}, inplace= True) # label column name

print (coef\_df)



**#(c)**

