

**ANL252**

**Python for Data Analytics**

# **End Course Assignment**

**July 2021 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **Brandon Tan Xuan Ming** | **Y1970855** |

**Instructor’s Name: Mr. KUMAR MUNISH**

**Submission Date: 13/09/2021**

1ai)

#Qn1a pt i to indicate that they are missing values, declare this character as missing values in your program accordingly.

import pandas as pd

import numpy as np

ship = pd.read\_csv ('/Users/brandontan/Desktop/ANL 252/ECA/ship.csv')

ship = ship.replace({ ".": np.nan })

print(ship)

A screenshot of a computer

Description automatically generated with low confidenceA screenshot of a computer

Description automatically generated with low confidence

1aii)

#Qn1a pt ii rename the ship types to "types", construction years to "c\_years", operation periods to "o\_periods", the aggregated months of service to "s\_months", and the number of incidents to "incidents".

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

display(ship)

Table

Description automatically generatedTable

Description automatically generated

1aiii)

#Qn1aiii) average service months and the average number of incidents for the cross-products of every category in types and operation periods. The averages should be rounded to the nearest integers. Store the resulting table to an object named "shipgroup".

#to first look into their initial dtypes

ship.dtypes

#converting dtypes for s\_months and incidents into numeric float64

ship[["s\_months", "incidents"]] = ship[["s\_months", "incidents"]].apply(pd.to\_numeric)

ship.dtypes

#grouping the types and o\_periods into the avg of s\_months and incidents

shipgroup1 = ship.groupby(['types', 'o\_periods']).mean().round(0)

#renaming and excluding c\_years from the final print

shipgroup = shipgroup1.loc[:, ~shipgroup1.columns.isin(['c\_years'])]

display(shipgroup)

Table

Description automatically generated

1aiv)

#Qn1a pt iv to fill all NaN value with respective means of the other ships that share the same type AND the same operation period.

ship[['s\_months', 'incidents']] = ship.groupby(['types', 'o\_periods'])['s\_months', 'incidents']\

.transform(lambda x: x.fillna(x.mean())).round(0)

display(ship)

Table

Description automatically generatedTable

Description automatically generated

1av)

#Qn1a pt v target variable "incidents" in a pandas DataFrame named "Y".

dict = {'Incidents' : ship.loc[:,'incidents']}

Y = pd.DataFrame(dict)

display(Y)

Table

Description automatically generatedTable

Description automatically generated

1bi)

#Qn1bi) data type conversion for "types", "c\_years", and "o\_periods" into norminal data / categorical variables

ship.dtypes

ship[["types", "c\_years", "o\_periods"]] = ship[["types", "c\_years", "o\_periods"]].astype('category')

ship[["types", "c\_years", "o\_periods"]].dtypes

Text

Description automatically generated

1bii)

#Qn1bii) convert all categorical variables to dummy variables and save the result as "X".

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

display(X)

Table

Description automatically generated

Table

Description automatically generated

1biii)

#1biii) Perform a log-transformation for months of service and name it "log\_s\_months". Attach it to "X" and "ship"

X["log\_s\_months"] = np.log(ship["s\_months"])

ship["log\_s\_months"]= np.log(ship["s\_months"])

display(X)

display(ship)

“X”

Table

Description automatically generatedTable

Description automatically generated

“ship”

Table

Description automatically generated

Table

Description automatically generated with medium confidence

1c) As there are only 39 data sets here, splitting them into training and testing will be inefficient and might cause underfitting or overfitting as the number of samples are too low and will cause the predictive model to be inaccurate as there will be not enough data to have an effective mapping of inputs to outputs.

1d)

#Qn1d) Write 2 programs to export ship.dataframe into csv and sql.

ship.to\_csv("ship\_prepared.csv")

import sqlite3

conn = sqlite3.connect('test\_database')

c = conn.cursor()

conn.commit()

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

Application, table

Description automatically generated

Table

Description automatically generated