Graded Questions (Logistic Regression):

1. Multi-Variate Logistic Regression: Model Building and Model Evaluation

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

1. Coding Questions:

**Fibonacci Series**

**Description**

Compute and display Fibonacci series upto n terms where n is a positive integer entered by the user.

You can go [here](https://www.mathsisfun.com/numbers/fibonacci-sequence.html) to read about Fibonacci series.

**Sample Input**:

5  
**Sample Output:**

0

1

1

2

3

Solution:

n=int(input())

#write your code here

# first two terms

n1, n2 = 0, 1

count = 0

if n == 1:

print(n1)

else:

while count < n:

print(n1)

nth = n1 + n2

# update values

n1 = n2

n2 = nth

count += 1

**Prime Numbers**

**Description**

Determine whether a positive integer n is a prime number or not. Assume n>1.

Display “number entered is prime” if n is prime, otherwise display “number entered is not prime”.  
**Sample Input:**

7

**Sample Output:**

number entered is prime

Solution:

n=int(input())

#write your code here

if n > 1:

for i in range(2, n):

if (n % i) == 0:

print("number entered is not prime")

break

else:

print("number entered is prime")

else:

print("number entered is not prime")

**Armstrong number**

**Description**

Any number, say n is called an Armstrong number if it is equal to the sum of its digits, where each is raised to the power of number of digits in n.  
For example:  
153=13+53+33Write Python code to determine whether an entered three digit number is an Armstrong number or not.   
Assume that the number entered will strictly be a three digit number.  
Print "True" if it is an Armstrong number and print "False" if it is not.  
Sample Input:  
153  
Sample Output:  
True

Solution:

n=int(input())

# initialize sum

sum = 0

# find the sum of the cube of each digit

temp = n

while temp > 0:

digit = temp % 10

sum += digit \*\* 3

temp //= 10

# Populate the result

if n == sum:

print("True")

else:

print("False")

**Selecting data frame columns**

**Description**

Write a program to select all columns of a data frame except the ones specified.

The input will contain a list of columns that you should skip.  
You should print the first five rows of the dataframe as output where the columns are **alphabetically sorted.**  
  
Sample Input:  
['PassengerId', 'Pclass', 'Name', 'Sex','Embarked']  
Sample Output:

**Age** **Cabin** **Fare** **Parch** **SibSp** **Ticket**

0 34.5 **NaN** 7.8292 0 0 330911

1 47.0 **NaN** 7.0000 0 1 363272

2 62.0 **NaN** 9.6875 0 0 240276

3 27.0 **NaN** 8.6625 0 0 315154

4 22.0 **NaN** 12.2875 1 1 3101298

Solution:

import pandas as pd

import ast,sys

pd.set\_option('display.max\_columns', 500)

df=pd.read\_csv("https://media-doselect.s3.amazonaws.com/generic/X0kvr3wEYXRzONE5W37xWWYYA/test.csv")

input\_str = sys.stdin.read()

to\_omit = ast.literal\_eval(input\_str)

df=df[df.columns[~df.columns.isin(to\_omit)]]

print(df.loc[:, sorted(list(df.columns))].head())

**Two series**

**Description**

Given two pandas series, find the position of elements in series2 in series1.  
You can assume that all elements in series2 will be present in series1.  
The input will contain two lines with series1 and series2 respectively.  
The output should be a list of indexes indicating elements of series2 in series 1.  
Note: In the output list, the indexes should be in ascending order.  
Sample Input:  
[1,2,3,4,5,6,7]  
[1,3,7]  
Sample Output:  
[0,2,6]

Solution:

import ast,sys

import pandas as pd

input\_str = sys.stdin.read()

input\_list = ast.literal\_eval(input\_str)

series1=pd.Series(input\_list[0])

series2=pd.Series(input\_list[1])

out\_list=[]

for i in list(series2):

out\_list.append(list(series1).index(i))

print(list(map(int,out\_list)))#do not alter this step, list must be int type for evaluation purposes

**Cleaning columns**

**Description**

For the given dataframe, you have to clean the "Installs" column and print its correlation with other numeric columns of the dataframe.(print df.corr())  
You have to do the following:  
1. Remove characters like ',' from the number of installs.  
2. Delete rows where the Installs column has irrelevant strings like 'Free'  
3. Convert the column to int type  
You can access the dataframe using the following URL in your Jupyter notebook:  
https://media-doselect.s3.amazonaws.com/generic/8NMooe4G0ENEe8z9q5ZvaZA7/googleplaystore.csv  
**Note: You should try this problem on your own Jupyter notebook before submitting. Do not clean any column other than "Installs".  
Sample Output:**            Rating  Installs  
Rating    1.000000  0.051355  
Installs  0.051355  1.000000

Solution:

import pandas as pd

df=pd.read\_csv("https://media-doselect.s3.amazonaws.com/generic/8NMooe4G0ENEe8z9q5ZvaZA7/googleplaystore.csv")

#add your cleaning code here

df['Installs']=df['Installs'].apply(lambda x: x.replace(',',''))

df['Installs']=df['Installs'].apply(lambda x: x.replace('+',''))

df.drop(df[df['Installs'] == 'Free'].index,inplace=True)

df['Installs']=df['Installs'].astype(int)

print(df.corr())

import pandas as pd

df=pd.read\_csv("https://media-doselect.s3.amazonaws.com/generic/8NMooe4G0ENEe8z9q5ZvaZA7/googleplaystore.csv")

df.Installs=df.Installs.str.replace(',','')

df.Installs=df.Installs.str.replace('+','')

df=df[df.Installs!='Free']

df.Installs=df.Installs.astype(int)

print(df.corr())

**Description**

Given a sentence as a string, capitalise the first letter of every word. Output the final sentence after capitalising.  
  
Hint: Since strings are immutable, you need to output a new string.  
  
**Example:  
Input 1:**I am now a master of Logistic regression **Output 1:**I Am Now A Master Of Logistic Regression

Solution:

# Read the input string

input\_string = input()

# Write your code here

print(input\_string.title())

**Second Maximum Number in a List**

**Description**

Given a list of numbers, find the second largest number in the list.  
  
**Note:**There might be repeated numbers in the list. If there is only one number present in the list, return 'not present'.  
  
**Examples:  
Input 1:**[7, 2, 0, 9, -1, 8]  
**Output 1:**8  
  
**Input 2:**[3, 1, 4, 4, 5, 5, 5, 0, 2, 2] **Output 2:**4  
  
**Input 2:**[6, 6, 6, 6, 6] **Output 2:**not present

Solution:

# Read the input list

import ast,sys

input\_str = sys.stdin.read()

input\_list = ast.literal\_eval(input\_str)

# Write your code here

l\_sorted=sorted(input\_list)

l\_set=sorted(list(set(input\_list)))

if len(l\_set) == len(l\_sorted):

print(l\_set[-2])

elif len(l\_set) == 1:

print("not present")

else:

print(l\_set[-2])

**Data Quality**

**Description**

You have a Wholesale Customers dataset. Here are the first few rows of the dataset:

﻿A screenshot of a cell phone

Description automatically generated﻿The last column 'Channel' contains information about the type of channel the items are supplied to. There are only three types of channels - 'Hotel', 'Restaurant', and 'Cafe'. But as you can notice, there are quality issues in the last data. A lot of the times Cafe as been written as 'C', Hotel has been written as 'Hot', etc. Inspect these incorrect values and replace them with their actual full names.

After this operation, the final column should only have 3 kinds of label - 'Hotel', 'Restaurant', and 'Cafe'.   
  
The dataframe has already been read to you in the variable 'wholesale'.  
  
The print statement to print the final dataframe has also been provided. You just need to perform the operations for correcting the names present in the last column.

Solution:

# Importing the Pandas package and reading the data

import pandas as pd

wholesale = pd.read\_csv('https://media-doselect.s3.amazonaws.com/generic/OkbnaOBqrBXZOpRQw1JGMgaM9/Wholesale\_Data.csv')

# Inspecting the first few rows

wholesale.head(10)

# Checking the value counts of the column 'Channel' to see what all values are present

wholesale['Channel'].value\_counts()

# Define a function to correct the names. Code self-explanatory

def correction(x):

if x in ['H', 'Hote', 'Hot']:

x = 'Hotel'

elif x in ['C', 'Caffe']:

x = 'Cafe'

elif x in ['R', 'Rest']:

x = 'Restaurant'

return(x)

# Apply the function written above to the column 'Channel'

wholesale['Channel'] = wholesale['Channel'].apply(lambda x: correction(x))

# Checking the final dataframe

wholesale.head()

# The final print statement. Please don't edit this part. Also don't write any other

# print statement otherwise your answer will not match even if it is correct

print(wholesale['Channel'].value\_counts())

**Worst Channel**

**Description**

You again have the wholesale dataset but this time it is cleaned. Here are the first few rows:﻿A screenshot of a cell phone

Description automatically generated﻿  
Each of the columns except for the column 'Channel' tells you the sales for each kind of product and the 'Channel' column tells you which 'Channel' you're selling to. Your task is to find out which of the three channels has the least total sales (for all the products combined). For example, if the total sales to Cafe, Hotel, and Restaurant turn out to be 1 lakh, 2 lakh, and 3 lakh rupees, you need to print 'Cafe' since it has the lowest sales.

Solution:

# Import Pandas and read the dataframe

import pandas as pd

wholesale = pd.read\_csv('https://media-doselect.s3.amazonaws.com/generic/OqwpypRKN09x5GYej2LvVrprn/Wholesale\_Data\_Cleaned.csv')

# Inspect the dataframe

wholesale.head()

wholesale.shape

# Create a new column 'Total' that contains the sum of all the columns

wholesale['Total'] = wholesale.sum(axis=1)

# Inspect the dataframe again

wholesale.head()

# Group the dataframe using 'Channel' as the index, 'Total' as the values, and

# 'sum' as the aggfunc

channels = wholesale.pivot\_table(index='Channel', values = 'Total', aggfunc = 'sum')

# Check the grouped dataframe

channels.head()

# Sort the grouped dataframe and print the smallest value

print(channels.sort\_values(by='Total').index[0])