BDA 355 - Business Analytics with Python

Python assignment(3)

Note: please add your answer to each question in its own answer box.

```
# run the following lines - Do not change lines!
names=input("Write your full names! ")# write your full name and your team member full name. e.g., Mark Fuller and Eli Roger
print("names: ", names)
```

```
     Write your full names! David Galietti and Armaan Singh
     names:  David Galietti and Armaan Singh
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Part 1

- write your code wherever it is instructed
- Do not change print functions

```
# TASK 1
# upload and read "StudentsPerformance.csv" data. Call it df
from google.colab import files
uploaded=files.upload()
df=pd.read_csv("StudentsPerformance.csv")
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please re

```
Saving StudentsPerformance.csv to StudentsPerformance.csv
```

```
# TASK 2
# get an overall view about the data using head function, showing 10 rows

df.head(10)
```

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score
0	female	group B	bachelor's degree	standard	none	72.0	72.0
1	female	group C	some college	standard	completed	69.0	90.0
2	female	group B	master's degree	standard	none	90.0	95.0
3	male	group A	associate's degree	free/reduced	none	47.0	57.0
4	male	group C	some college	standard	none	76.0	78.0
5	female	group B	associate's degree	standard	none	71.0	83.0
6	female	group B	some college	standard	completed	88.0	95.0
7	male	group B	some college	free/reduced	none	40.0	43.0
8	male	group D	high school	free/reduced	completed	64.0	64.0
9	female	group B	high school	free/reduced	none	38.0	60.0

```
# TASK 3
# get the shape of df. Call it df_shape
df_shape=df.shape
print(df_shape)
```

```
(1000, 8)
```

```
# TASK 4
# run this cell
#sample size and number of variables
sample_size=df_shape[0]
variables=df_shape[1]
f"The sample size is {sample_size} and there are {variables} variables in this dataset"
```

```
'The sample size is 1000 and there are 8 variables in this dataset'
```

```
# TASK 5
#create a list of columns. Call it columns
columns=df.columns.to_list()
print(columns)
```

```
['gender', 'race/ethnicity', 'parental level of education', 'lunch', 'test preparation course', 'math score', 'reading
```

```
# TASK 6
#check the dtype of each variable
df.dtypes
```

```
gender                          object
race/ethnicity                  object
parental level of education     object
lunch                           object
test preparation course         object
math score                     float64
reading score                  float64
writing score                  float64
dtype: object
```

```
# TASK 7
#select object variables and put them in a list called it cat_var (the list should include names of 5 object variables)
cat_var=df.select_dtypes("object").columns.to_list()
print(cat_var)
```

```
['gender', 'race/ethnicity', 'parental level of education', 'lunch', 'test preparation course']
```

```
# TASK 8
# for each in cat_var find the information regarding categories using unique()
for each in cat_var:
  print(each, df[each].unique())
  print("=====================================") #DO NOT CHANGE THIS LINE
```

```
gender ['female' 'male']
=====================================
race/ethnicity ['group B' 'group C' 'group A' 'group D' 'group E']
=====================================
parental level of education ["bachelor's degree" 'some college' "master's degree" "associate's degree"
 'high school' nan 'some high school']
=====================================
lunch ['standard' 'free/reduced' nan]
=====================================
test preparation course ['none' 'completed' nan]
=====================================
```

```
# TASK 9
# for each in cat_var, find the frequency of each category using value_counts()
for each in cat_var:
  print(df[each].value_counts())
  print("=============================================") #DO NOT CHANGE THIS LINE
```

```
female    518
male      482
Name: gender, dtype: int64
=============================================
group C    319
group D    262
group B    190
group E    140
group A     89
Name: race/ethnicity, dtype: int64
=============================================
some college          219
associate's degree    214
high school           189
some high school      171
bachelor's degree     114
master's degree        57
Name: parental level of education, dtype: int64
=============================================
standard        644
free/reduced    354
Name: lunch, dtype: int64
=============================================
none         624
completed    347
Name: test preparation course, dtype: int64
=============================================
```

```
# TASK 10
#use describe() for finding statistics about numeric variables, save it into a variable called description
description=df.describe()
print(description)
```

```
       math score  reading score  writing score
count  978.000000     976.000000     963.000000
mean    66.118609      69.106557      68.271028
std     15.193742      14.689571      14.984963
min      0.000000      17.000000      15.000000
25%     57.000000      59.000000      58.000000
50%     66.000000      70.000000      69.000000
75%     77.000000      79.250000      79.000000
max    100.000000     100.000000     100.000000
```

Now you have all information required for describing data

- Read the data description example word file
- Practice on data description example jupyter notebook file
- use the template for submitting the first part of the assignment-3
- Ignore the Nan values in describing categorical variables

Part 2

```
# TASK 11
# what is the average of math, reading, and writing scores for each race/ethnicity group?
race_groups_scores=df.groupby('race/ethnicity')[['math score', 'reading score', 'writing score']].mean()
print(race_groups_scores)
```

```
                math score  reading score  writing score
race/ethnicity
group A          61.441860      64.779070      63.305882
group B          63.365591      67.245989      65.333333
group C          64.662379      69.073248      68.309211
group D          67.277344      69.936508      70.160156
group E          73.820144      72.912409      71.572464
```

```
# TASK 12
# which race/ethnicity group has the highest scores in all categories?
Answer1="Group E"
print(Answer1)
```

```
Group E
```

```
# TASK 13
# what is the average of math, reading, and writing scores for gender groups?
```

```
gender_group_scores=df.groupby('gender')[['math score', 'reading score', 'writing score']].mean()
print(gender_group_scores)
```

```
        math score  reading score  writing score
gender
female   63.608696      72.578740      72.705645
male     68.809322      65.337607      63.561028
```

```
# TASK 14
# which gender group has the highest score in math?
Answer2="male"
print(Answer2)
```

```
male
```

```
# TASK 15
# which gender group has the highest score in reading?
Answer3="female"
print(Answer3)
```

```
female
```

```
# TASK 16
# which gender group has the highest score in writing?
Answer4="female"
print(Answer4)
```

```
female
```

```
# TASK 17
# what is the average of math, reading, and writing scores for test preparation course groups?
preparation_group_scores=df.groupby('test preparation course')[['math score', 'reading score', 'writing score']].mean()
print(preparation_group_scores)
```

```
                         math score  reading score  writing score
test preparation course
```

```
    completed                 69.818991      73.985207      74.313609
    none                      64.075041      66.357377      64.891304
```

```
# TASK 18
# what is your conclusion about the effect of test preparation course on scores?
Answer5="Individuals who completed a test preperation course score higher in math, reading, and writing than people who do n
print(Answer5)
```

```
    Individuals who completed a test preperation course score higher in math, reading, and writing than people who do not
```

```
# TASK 19
# what is the average of math, reading, and writing scores for lunch groups?
lunch_group_scores=df.groupby('lunch')[['math score', 'reading score', 'writing score']].mean()
print(lunch_group_scores)
```

```
                  math score  reading score  writing score
    lunch
    free/reduced   58.979769      64.544928       63.40708
    standard       70.069841      71.627981       70.94061
```

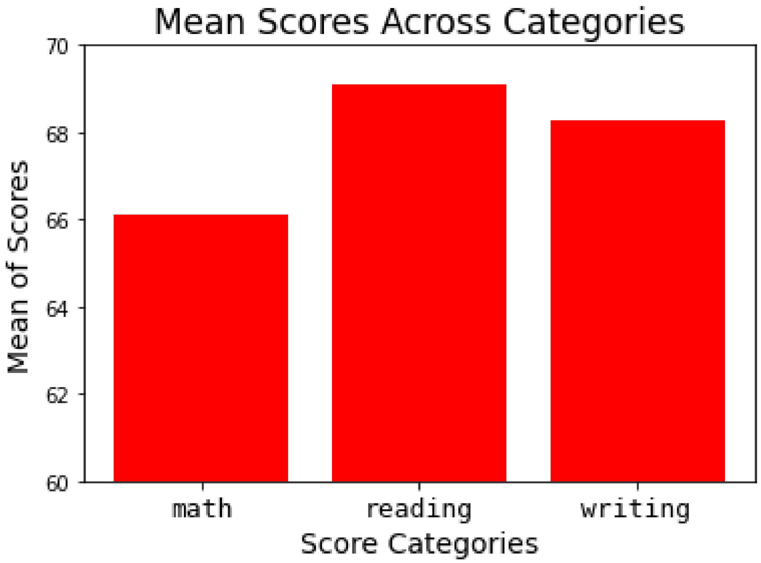
```
# TASK 20
# what is your conclusion about the effect of lunch on scores?
Answer6="Individuals who receive the standard lunch score higher in math, reading, and writing than people who receive free/r
print(Answer6)
```

```
    Individuals who receive the standard lunch score higher in math, reading, and writing than people who receive free/redu
```

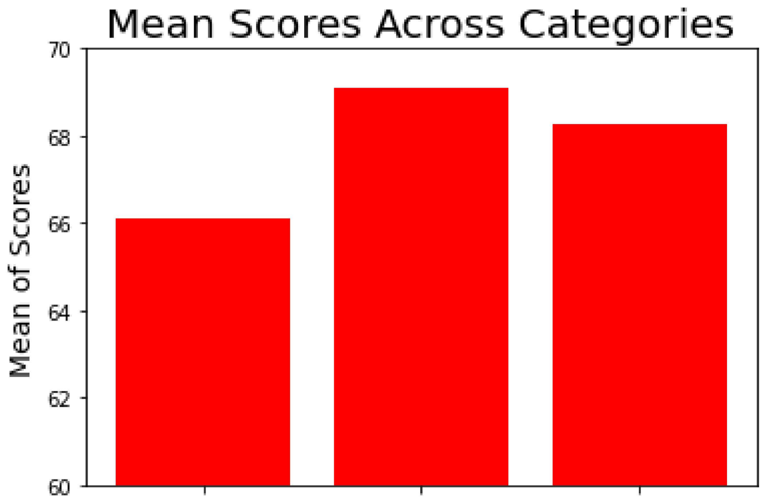
Part 3: Visualization

```
# TASK 21
# plot a bar chart for average of reading score, writing score and math score
```

The final chart should be like this:



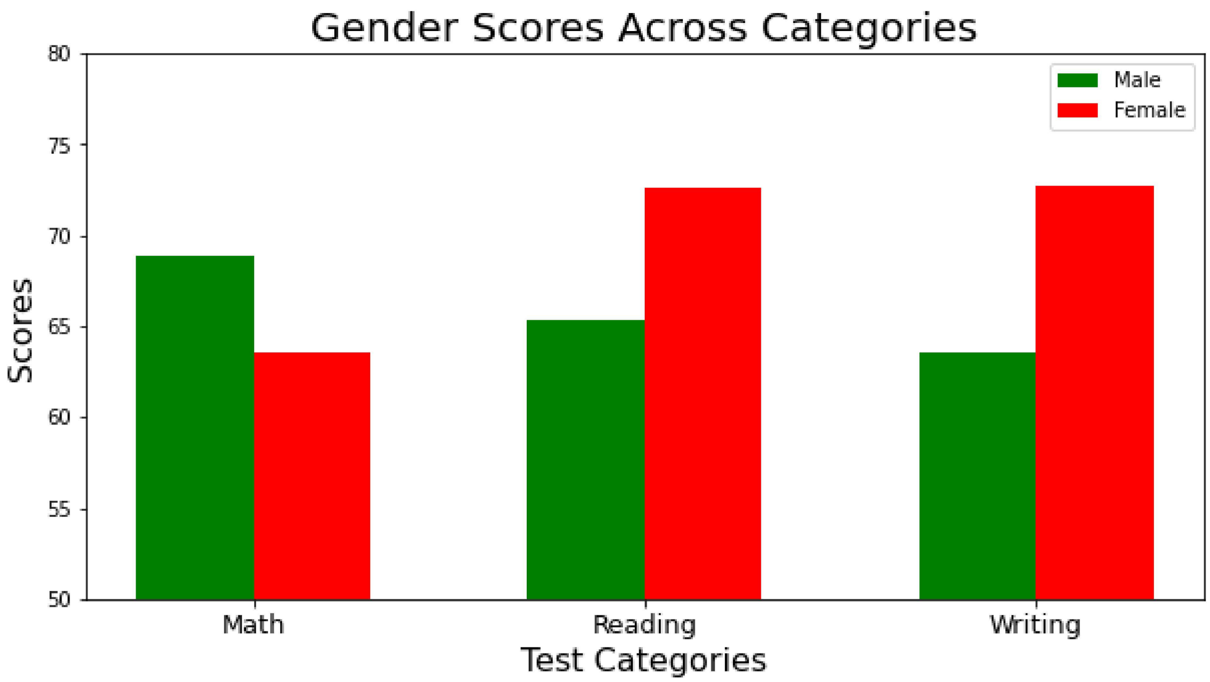
```
#write your code here
Score_Categories=["math", "reading", "writing"]
Mean_of_scores=df[['math score', 'reading score', 'writing score']].mean()
plt.figure(figsize=(6,4))
plt.ylim([60,70])
plt.bar(Score_Categories, Mean_of_scores,color='red')
plt.title("Mean Scores Across Categories", fontsize=20)
plt.xlabel("Score Categories", fontsize=14)
plt.ylabel("Mean of Scores", fontsize=14)
plt.xticks(fontsize=13)

plt.show()
```



```
# LEAVE THIS CELL BLANK
```

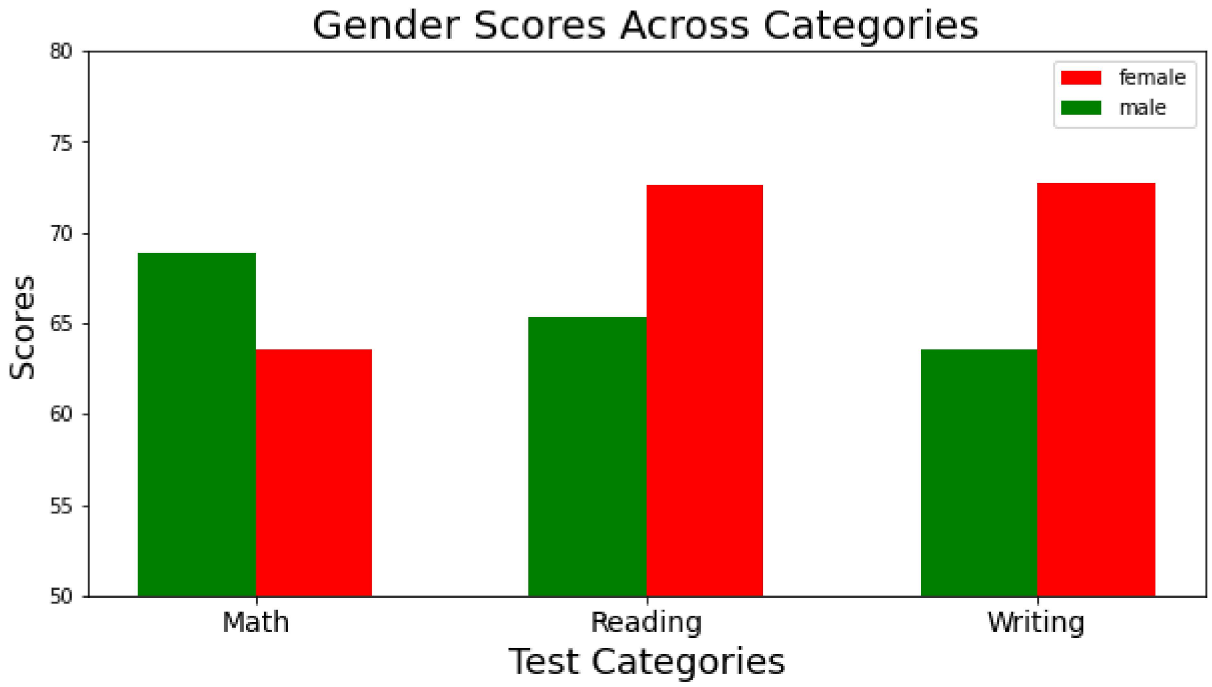
```
# # TASK 22
# plot a bar chart that shows the mean of math, reading, and writing across gender groups
```

The final chart should be like this:



```
# write your code here

barwidth=.3

Score_Categories=["math", "reading", "writing"]
avg_male=[68.809322, 65.337607, 63.561028]
avg_female=[63.608696, 72.578740, 72.705645]

br1=np.arange(len(avg_male))
br2=[x+ barwidth for x in br1]

plt.figure(figsize=(10,5))
plt.bar(br2, avg_female,width=barwidth,color='red',label='female')
plt.bar(br1, avg_male,width=barwidth,color='green',label='male')

n_groups=3
index=np.arange(n_groups)
plt.xticks(index + barwidth/2, ('Math', 'Reading', 'Writing'), fontsize=14)
```

```
ax=plt.axis()
plt.title("Gender Scores Across Categories", fontsize=20)
plt.xlabel("Test Categories", fontsize=18)
plt.ylabel("Scores", fontsize=16)
plt.legend(loc="best")
plt.ylim([50,80])
plt.show()
```



✓ 0s completed at 1:50 PM