EDA_Student___Data

September 28, 2022

1 EDA

1.0.1 Importing Libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

1.0.2 Importing Dataset

[2]:
```
data = pd.read_csv('/content/drive/MyDrive/FSDS_Job_Gurantee/Live Class/
↪25_SEP_2022_EDA/student_performance/student_performance/data/student.csv')
```

Top 5 rows

[3]:
```
data.head()
```

[3]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

Below 5 rows

[4]:
```
data.tail()
```

[4]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
995	female	group E	master's degree	standard	completed	88	99	95
996	male	group C	high school	free/reduced	none	62	55	55
997	female	group C	high school	free/reduced	completed	59	71	65
998	female	group D	some college	standard	completed	68	78	77
999	female	group D	some college	free/reduced	none	77	86	86

1.0.3 Seprating Categorical and Numerical Columns

Categorical: * Nominal * gender * race/ethnicity * lunch * test preparation course * Odinal * parental level of education

Numerical: * Continuous * math score * reading score * writing score * Discrete * No column

Shape of the data

[5]:
```
data.shape
```

[5]:
```
(1000, 8)
```

Basic information about the data

[6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64
 6   reading score                1000 non-null   int64
 7   writing score                1000 non-null   int64
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

checking data types of gender

[7]:
```
data['gender'].dtypes
```

[7]:
```
dtype('O')
```

[8]:
```
data['gender'].dtypes == 'O'
```

[8]:
```
True
```

Columns List

[9]:
```
data.columns
```

[9]:
```
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'test preparation course', 'math score', 'reading score',
       'writing score'],
      dtype='object')
```

1.0.4 Segregting Categorical Columns

[10]:
```
categorical_columns = [feature for feature in data.columns if data[feature].
↪dtypes == 'O']
```

1.0.5 Segregting Numerical Columns

[11]:
```
numerical_columns = [feature for feature in data.columns if data[feature].
↪dtypes != 'O']
```

[12]:
```
data[categorical_columns]
```

[12]:
```
     gender race/ethnicity parental level of education         lunch  \
0    female        group B           bachelor's degree      standard
1    female        group C                some college      standard
2    female        group B            master's degree      standard
3      male        group A          associate's degree  free/reduced
4      male        group C                some college      standard
..      …            …                          …            …
995  female        group E            master's degree      standard
996    male        group C                 high school  free/reduced
997  female        group C                 high school  free/reduced
998  female        group D                some college      standard
999  female        group D                some college  free/reduced

    test preparation course
```

```
0                       none
1                  completed
2                       none
3                       none
4                       none
..                       …
995                completed
996                     none
997                completed
998                completed
999                     none

[1000 rows x 5 columns]
```

[13]:
```
data[numerical_columns]
```

[13]:
```
     math score  reading score  writing score
0            72             72             74
1            69             90             88
2            90             95             93
3            47             57             44
4            76             78             75
..          …              …              …
995          88             99             95
996          62             55             55
997          59             71             65
998          68             78             77
999          77             86             86

[1000 rows x 3 columns]
```

1.0.6 Memory Usage

[14]:
```
data.memory_usage()
```

[14]:
```
Index                          128
gender                        8000
race/ethnicity                8000
parental level of education   8000
lunch                         8000
test preparation course       8000
math score                    8000
reading score                 8000
writing score                 8000
dtype: int64
```

1.0.7 Missing Value

[15]:
```
data.isnull().sum()
```

[15]:
```
gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
dtype: int64
```

[16]:
```
data.isnull().sum().sum()
```

[16]:
```
0
```

[17]:
```
data.duplicated().sum()
```

[17]:
```
0
```

1.0.8 Unique value in each column

[18]:
```
data.nunique()
```

[18]:
```
gender                         2
race/ethnicity                 5
parental level of education    6
lunch                          2
test preparation course        2
math score                    81
reading score                 72
writing score                 77
dtype: int64
```

Unique value in a column

[19]:
```
data['gender'].unique()
```

[19]:
```
array(['female', 'male'], dtype=object)
```

1.0.9 Statistical Analysis

T is for Transpose and describe will show all the statistical measures

[20]:
```
data.describe().T
```

[20]:

	count	mean	std	min	25%	50%	75%	max
math score	1000.0	66.089	15.163080	0.0	57.00	66.0	77.0	100.0
reading score	1000.0	69.169	14.600192	17.0	59.00	70.0	79.0	100.0
writing score	1000.0	68.054	15.195657	10.0	57.75	69.0	79.0	100.0

Correlartion

[21]:
```
data.corr()
```

[21]:

	math score	reading score	writing score
math score	1.000000	0.817580	0.802642
reading score	0.817580	1.000000	0.954598
writing score	0.802642	0.954598	1.000000

Conclusion:

- High '+ve' correlatiob between math and reading score. It means if you are a good reader then you are good in math as well.

Covariance

[22]:
```
data.cov()
```

[22]:

	math score	reading score	writing score
math score	229.918998	180.998958	184.939133
reading score	180.998958	213.165605	211.786661
writing score	184.939133	211.786661	230.907992

Skewness

[23]:
```
data.skew()
```

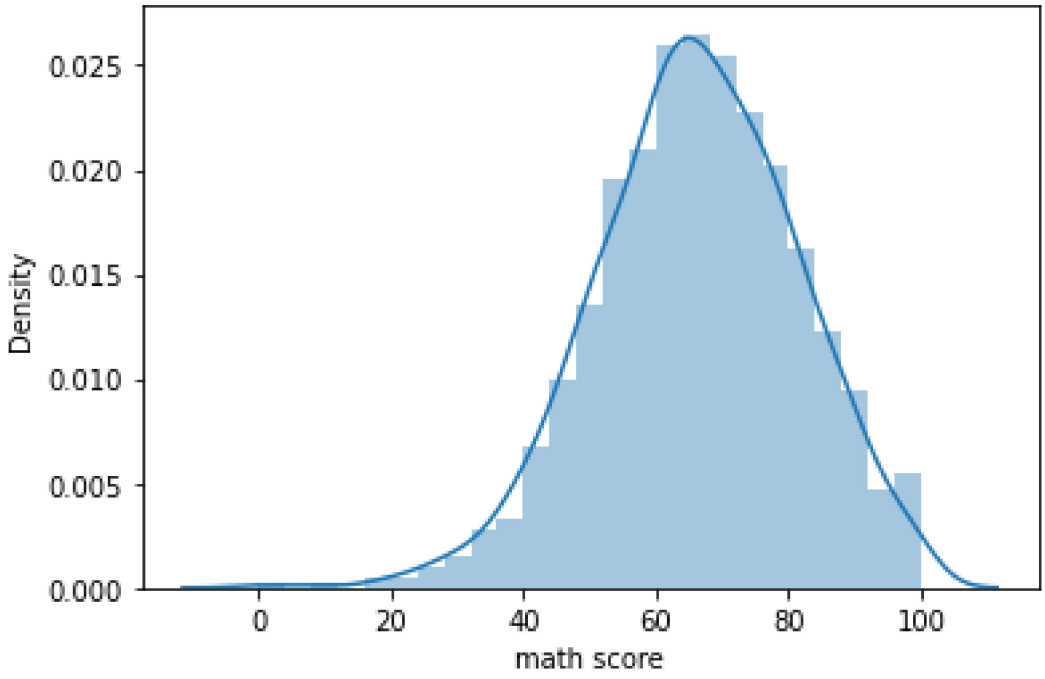
[23]:
```
math score      -0.278935
reading score   -0.259105
writing score   -0.289444
dtype: float64
```

1.0.10 Graphical Analysis

distplot : distribution plot

[24]:
```
sns.distplot(data['math score'])
```

[24]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff68e6936d0>
```



Conclusion: * Left skewed graph

1.0.11 Soms Business Analysis

[25]:
```
data.columns
```

[25]:
```
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'test preparation course', 'math score', 'reading score',
       'writing score'],
      dtype='object')
```

Average score for all subject

[26]:
```
data['Average Score']=(data['math score']+data['reading score']+data['writing␣
↪score'])/3
```

[27]:
```
data.head()
```

[27]:

	gender	race/ethnicity	parental level of education	lunch	\
0	female	group B	bachelor's degree	standard	
1	female	group C	some college	standard	
2	female	group B	master's degree	standard	
3	male	group A	associate's degree	free/reduced	
4	male	group C	some college	standard	

```
  test preparation course  math score  reading score  writing score  \
0                    none          72             72             74
1               completed          69             90             88
2                    none          90             95             93
3                    none          47             57             44
4                    none          76             78             75

   Average Score
0      72.666667
1      82.333333
2      92.666667
3      49.333333
4      76.333333
```

Gender wise mean value for numerical column

[28]:
```
data.groupby('gender').mean()
```

[28]:
```
        math score  reading score  writing score  Average Score
gender
female   63.633205      72.608108      72.467181      69.569498
male     68.728216      65.473029      63.311203      65.837483
```

Gender wise count for all the column

[29]:
```
data.groupby('gender').count()
```

[29]:
```
        race/ethnicity  parental level of education  lunch  \
gender
female             518                          518    518
male               482                          482    482

        test preparation course  math score  reading score  writing score  \
gender
female                      518         518            518            518
male                        482         482            482            482

        Average Score
gender
female            518
male              482
```

Que. Find out no of student whoever is having less than 30 marks in math?

[30]:
```
data[data['math score'] < 30].count()
```

[30]:
```
gender                         14
race/ethnicity                 14
parental level of education    14
lunch                          14
test preparation course        14
math score                     14
reading score                  14
writing score                  14
Average Score                  14
dtype: int64
```

1.0.12 Normailty check for numerical column

[31]:
```
data.columns
```

[31]:
```
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'test preparation course', 'math score', 'reading score',
       'writing score', 'Average Score'],
      dtype='object')
```

[32]:
```
numerical_data = data[numerical_columns]
```

[33]:
```
numerical_data.head()
```

[33]:

	math score	reading score	writing score
0	72	72	74
1	69	90	88
2	90	95	93
3	47	57	44
4	76	78	75

[34]:
```
from scipy.stats import normaltest
```

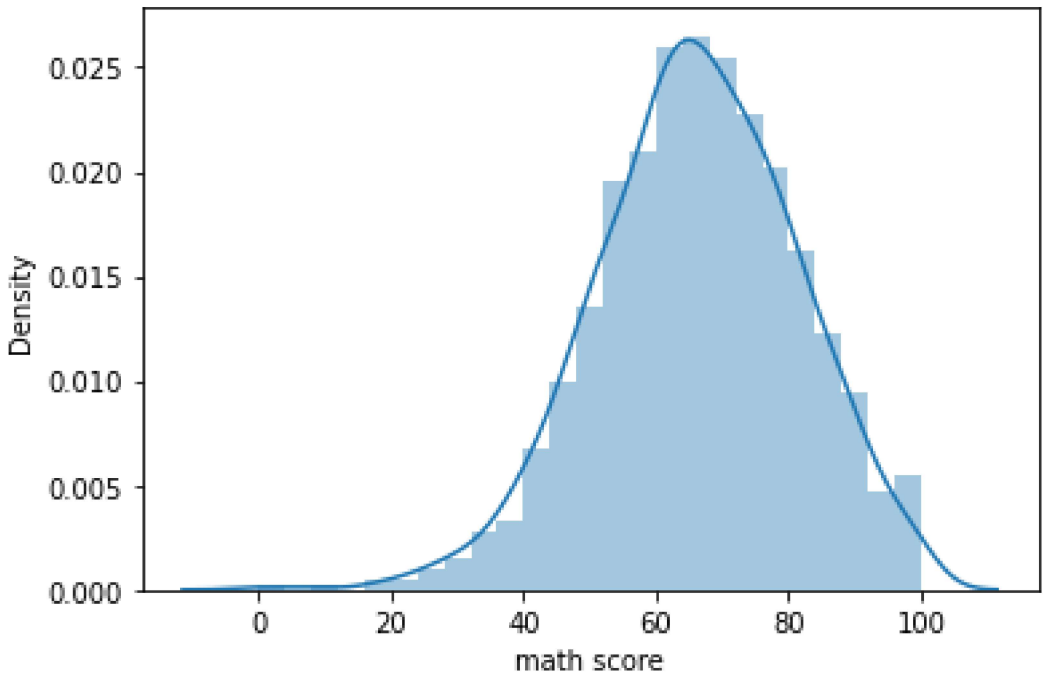
[35]:
```
p_value = normaltest(numerical_data['math score'])[1]*100
```

[36]:
```
if p_value > 0.05:
  print(" Data is normally distributed")
else:
  print("Data is not nomally distributed")
```

```
Data is not nomally distributed
```

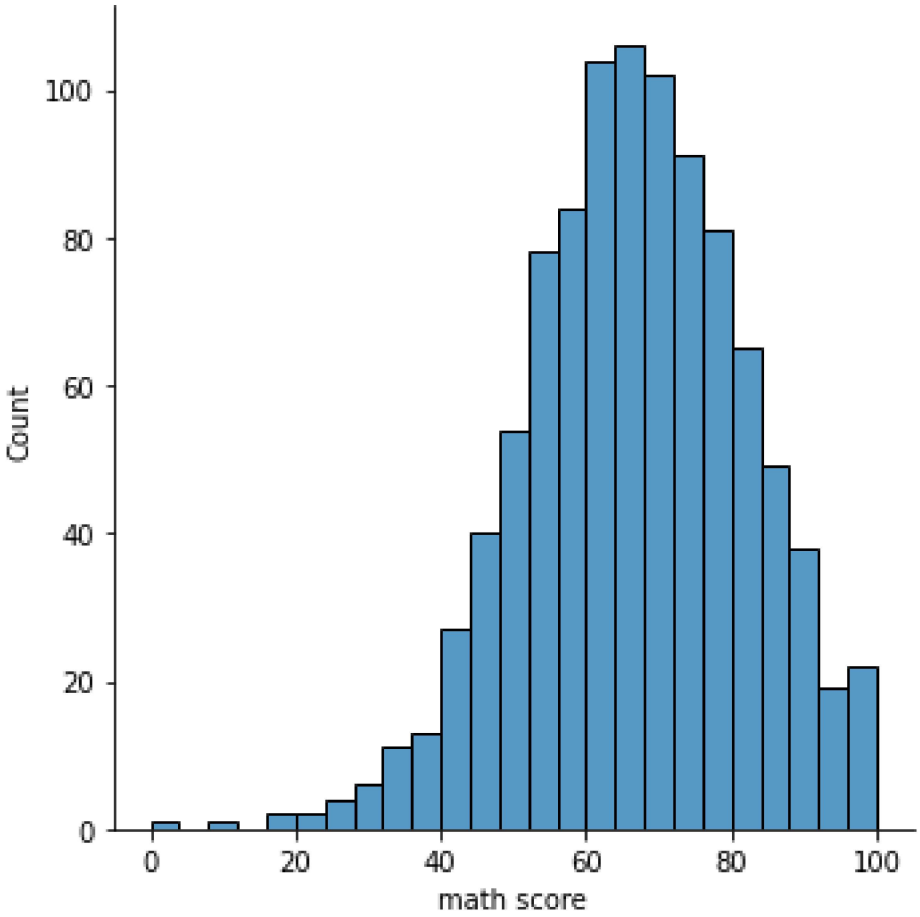
[37]:
```
sns.distplot(numerical_data['math score'])
```

[37]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff68e134950>
```



[38]:
```
sns.displot(numerical_data['math score'])
```

[38]: <seaborn.axisgrid.FacetGrid at 0x7ff68e070d90>



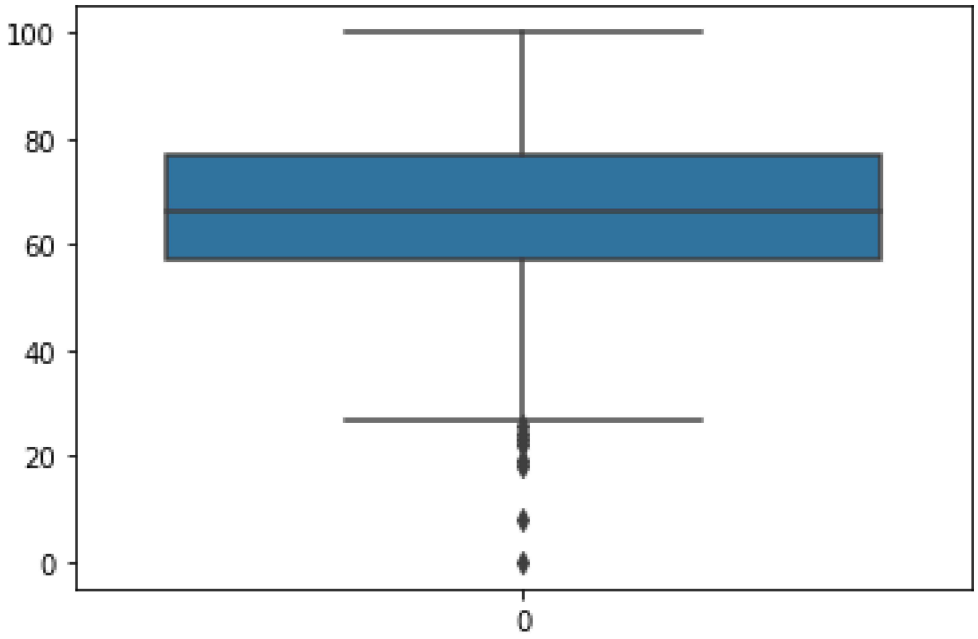
Conclusion: * Less Normally Distributed

1.0.13 Outliers

Graphical Methods Box Plot

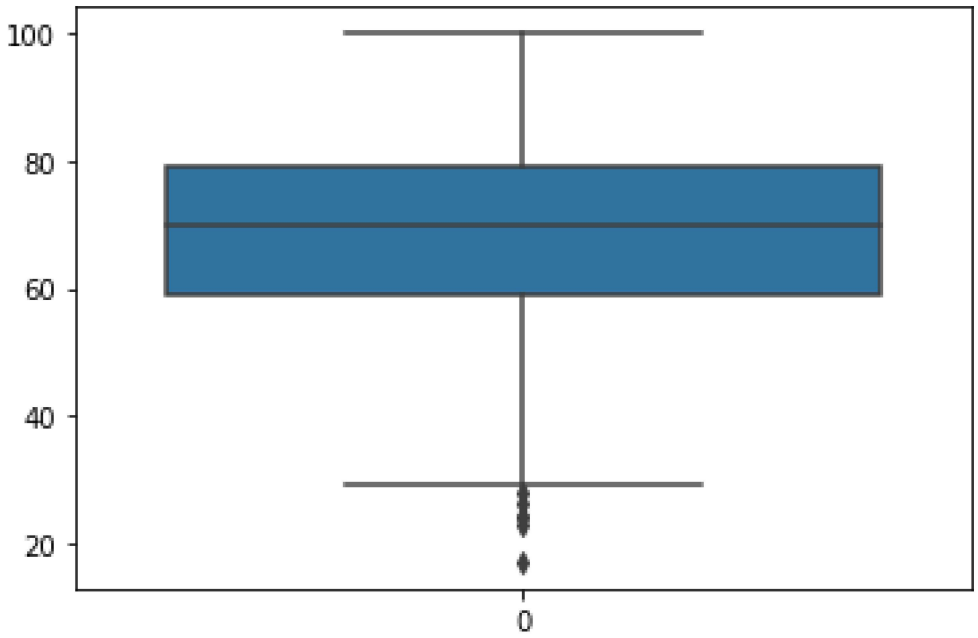
[39]:
```
sns.boxplot(data = data['math score'])
```

[39]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff68b7b7f90>
```



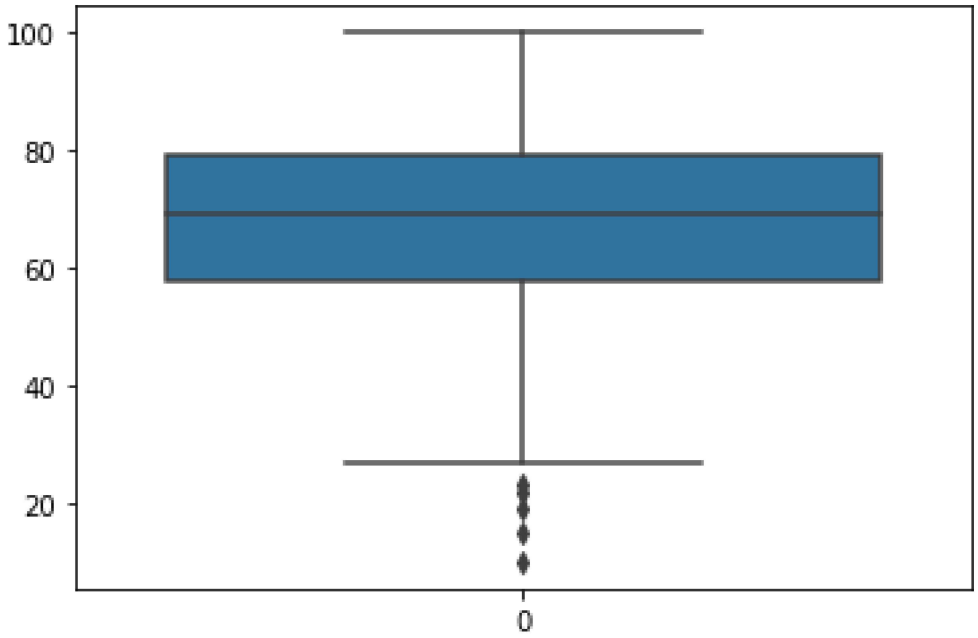
[40]:
```
sns.boxplot(data = data['reading score'])
```

[40]: <matplotlib.axes._subplots.AxesSubplot at 0x7ff68e6d1710>



[41]:
```
sns.boxplot(data = data['writing score'])
```

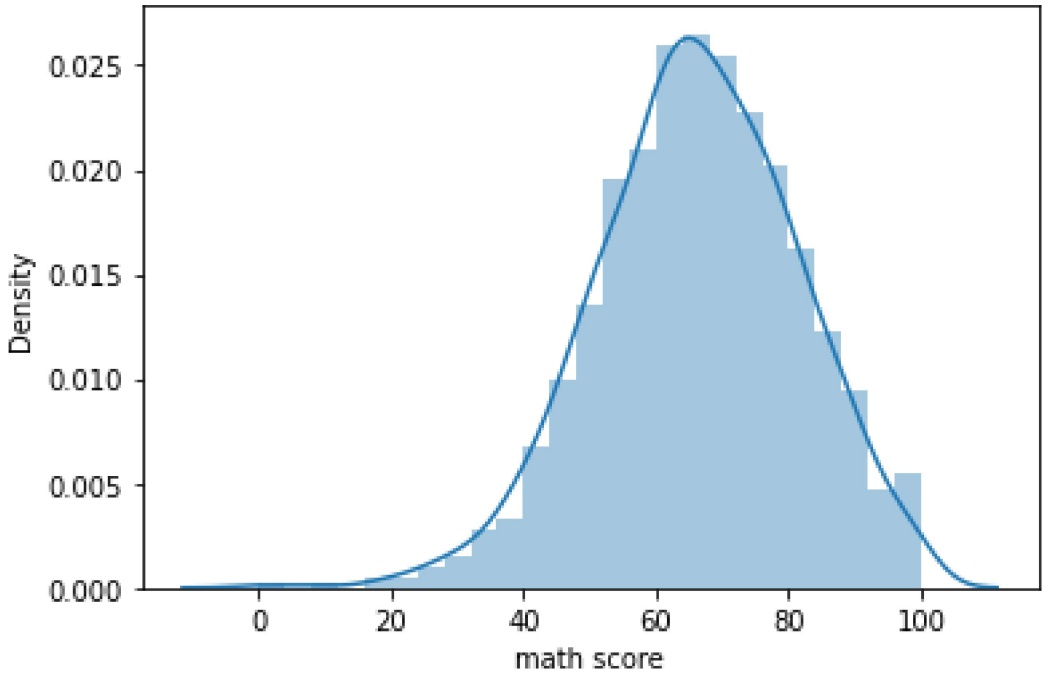
[41]: <matplotlib.axes._subplots.AxesSubplot at 0x7ff68b6eec90>



Distplot

[42]:
```
sns.distplot(data['math score'])
```

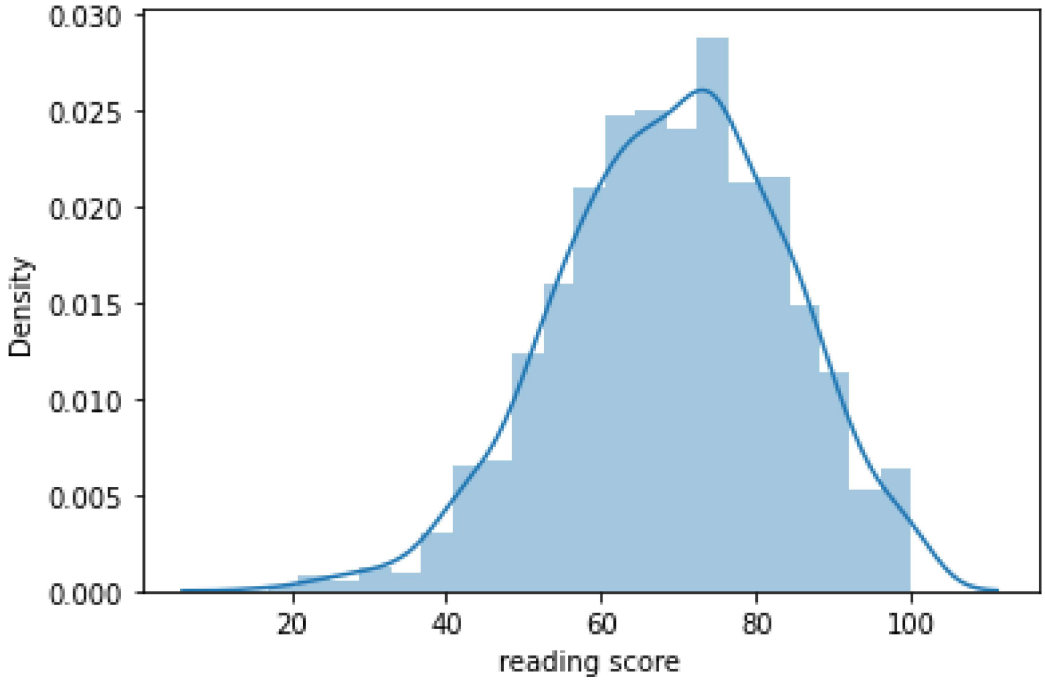
[42]: <matplotlib.axes._subplots.AxesSubplot at 0x7ff68b6f6750>



Conclusion: * Left Skewed Data

[43]:
```
sns.distplot(data['reading score'])
```

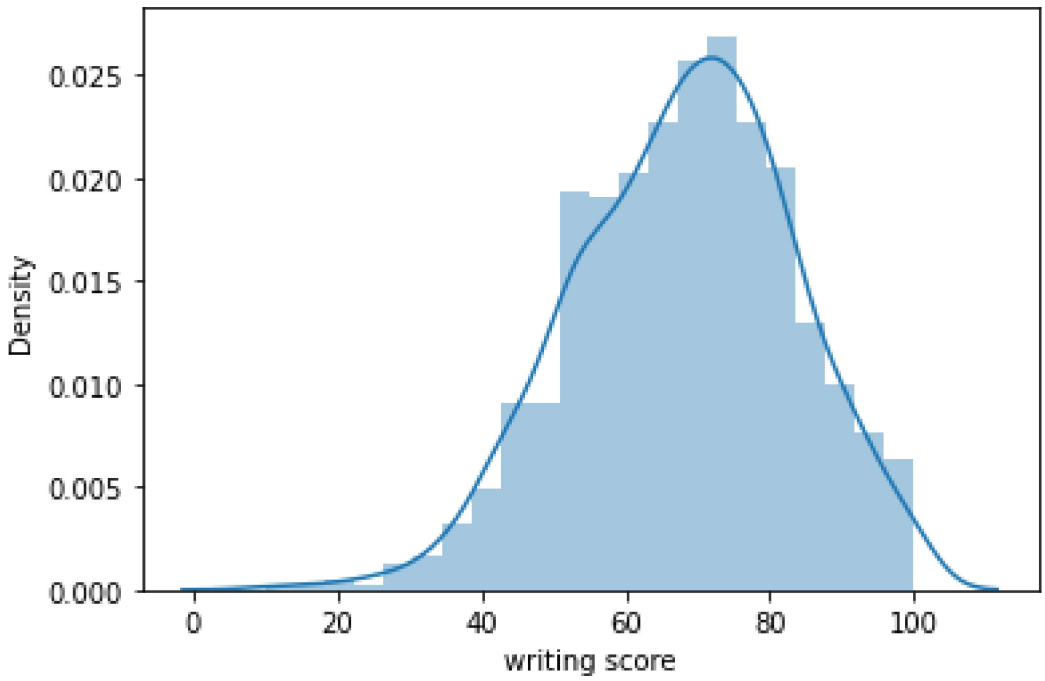
[43]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff68b6461d0>
```



Conclusion: * Left Skewed Data

[44]:
```
sns.distplot(data['writing score'])
```

[44]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff68b63f210>
```



Conclusion: * Left Skewed Data

Quantile Methods 10th percentile

[45]:
```
q1 = data['math score'].quantile(0.25)
```

90th percentile

[46]:
```
q3 = data['math score'].quantile(0.75)
```

100th percentile

[47]:
```
data['math score'].quantile(1.00)
```

[47]:
```
100.0
```

Minimum value in column ”math score”

[48]:
```
data['math score'].min()
```

[48]:
```
0
```

Maximum value in column "math score"

[49]:
```
data['math score'].max()
```

[49]:
```
100
```

[50]:
```
data['math score'].unique()
```

[50]:
```
array([ 72,  69,  90,  47,  76,  71,  88,  40,  64,  38,  58,  65,  78,
        50,  18,  46,  54,  66,  44,  74,  73,  67,  70,  62,  63,  56,
        97,  81,  75,  57,  55,  53,  59,  82,  77,  33,  52,   0,  79,
        39,  45,  60,  61,  41,  49,  30,  80,  42,  27,  43,  68,  85,
        98,  87,  51,  99,  84,  91,  83,  89,  22, 100,  96,  94,  48,
        35,  34,  86,  92,  37,  28,  24,  26,  95,  36,  29,  32,  93,
        19,  23,   8])
```

IQR : Inter-Quartile Range

[51]:
```
IQR = q3 - q1
IQR
```

[51]:
```
20.0
```

Upper Limit

[52]:
```
upper_limit = q3 + 1.5 * IQR
upper_limit
```

[52]:
```
107.0
```

Lower Limit

[53]:
```
lower_limit = q1 - 1.5 * IQR
lower_limit
```

[53]:
```
27.0
```

[54]:
```
data[data['math score'] < lower_limit]
```

[54]:

	gender	race/ethnicity	parental level of education	lunch	\
17	female	group B	some high school	free/reduced	
59	female	group C	some high school	free/reduced	
145	female	group C	some college	free/reduced	
338	female	group B	some high school	free/reduced	
466	female	group D	associate's degree	free/reduced	

```
787  female        group B                some college      standard
842  female        group B                 high school  free/reduced
980  female        group B                 high school  free/reduced

    test preparation course  math score  reading score  writing score  \
17                     none          18             32             28
59                     none           0             17             10
145                    none          22             39             33
338                    none          24             38             27
466                    none          26             31             38
787                    none          19             38             32
842               completed          23             44             36
980                    none           8             24             23

     Average Score
17       26.000000
59        9.000000
145      31.333333
338      29.666667
466      31.666667
787      29.666667
842      34.333333
980      18.333333
```

[55]:
```
data.drop(data[data['math score'] < lower_limit].index, inplace = True)
data.reset_index(inplace = True)
```

[56]:
```
data.head()
```

[56]:
```
   index  gender race/ethnicity parental level of education         lunch  \
0      0  female        group B           bachelor's degree      standard
1      1  female        group C                some college      standard
2      2  female        group B             master's degree      standard
3      3    male        group A          associate's degree  free/reduced
4      4    male        group C                some college      standard

  test preparation course  math score  reading score  writing score  \
0                    none          72             72             74
1               completed          69             90             88
2                    none          90             95             93
3                    none          47             57             44
4                    none          76             78             75

   Average Score
0      72.666667
1      82.333333
2      92.666667
```

```
3      49.333333
4      76.333333
```

[57]:
```
data[data['math score'] > upper_limit]
```

[57]:
```
Empty DataFrame
Columns: [index, gender, race/ethnicity, parental level of education, lunch,
test preparation course, math score, reading score, writing score, Average
Score]
Index: []
```

Creating function determine lower limit, upper limit and IQR

[58]:
```
def IQR(df, column_name, q1, q3):
  q1 = df[column_name].quantile(q1)
  q3 = df[column_name].quantile(q3)
  IQR = q3 - q1
  lower_range = q1 - 1.5 * IQR
  upper_range = q3 + 1.5 * IQR
  return lower_range, IQR, upper_range
```

[59]:
```
IQR(data, 'math score', 0.25, 0.75)
```

[59]:
```
(27.0, 20.0, 107.0)
```

[60]:
```
lr = []
iqr = []
ur  = []
for variable in numerical_data.columns:
  lr.append(IQR(numerical_data, variable, 0.25, 0.75)[0])
  iqr.append(IQR(numerical_data, variable, 0.25, 0.75)[1])
  ur.append(IQR(numerical_data, variable, 0.25, 0.75)[2])
```

[61]:
```
# def replace_with_threshold(data, numeric_column):
#   for variable in numeric_column:
#   lr.append(IQR(numerical_data, variable, 0.25, 0.75)[0])
#   iqr.append(IQR(numerical_data, variable, 0.25, 0.75)[1])
#   ur.append(IQR(numerical_data, variable, 0.25, 0.75)[2])
#   data.loc[data[variable] < lr] = lr
#   data.loc[data[variable] < lr] = ur
```

1.0.14 Graphical Analysis

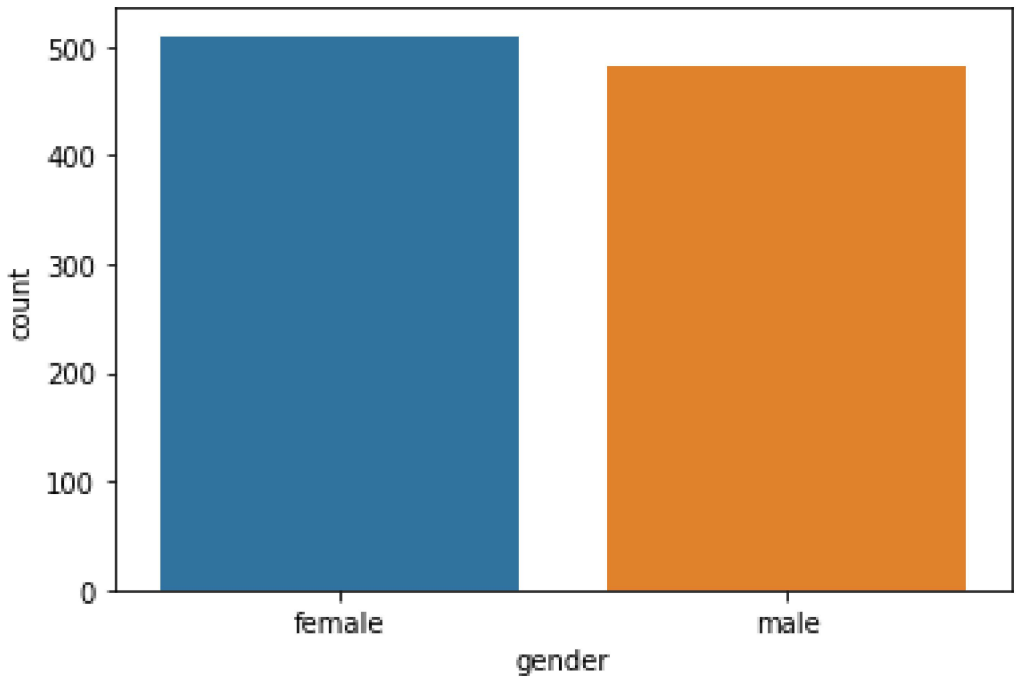
[62]:
```
data
```

[62]:
```
     index  gender race/ethnicity parental level of education         lunch  \
0        0  female        group B           bachelor's degree      standard
1        1  female        group C                some college      standard
2        2  female        group B            master's degree      standard
3        3    male        group A         associate's degree  free/reduced
4        4    male        group C                some college      standard
..     …     …            …                          …            …
987    995  female        group E            master's degree      standard
988    996    male        group C                 high school  free/reduced
989    997  female        group C                 high school  free/reduced
990    998  female        group D                some college      standard
991    999  female        group D                some college  free/reduced

    test preparation course  math score  reading score  writing score  \
0                      none          72             72             74
1                 completed          69             90             88
2                      none          90             95             93
3                      none          47             57             44
4                      none          76             78             75
..                      …         …              …              …
987               completed          88             99             95
988                    none          62             55             55
989               completed          59             71             65
990               completed          68             78             77
991                    none          77             86             86

     Average Score
0        72.666667
1        82.333333
2        92.666667
3        49.333333
4        76.333333
..             …
987      94.000000
988      57.333333
989      65.000000
990      74.333333
991      83.000000

[992 rows x 10 columns]
```

Countplot

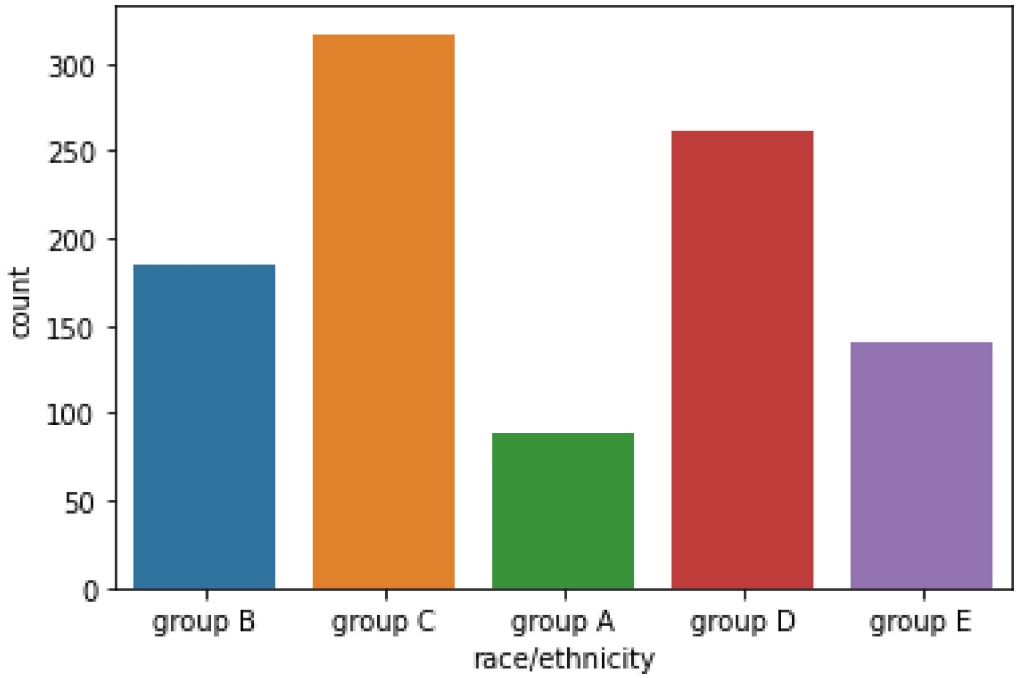
[63]:
```
sns.countplot(data['gender'])
```

[63]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff68e715150>
```



[64]:
```
sns.countplot(data['race/ethnicity'])
```

[64]: <matplotlib.axes._subplots.AxesSubplot at 0x7ff68b3e7b10>



Barplot

[65]:
```
df = data.groupby('gender').mean()
```

[66]:
```
df['Average Score']
```

[66]:
```
gender
female    70.249020
male      65.837483
Name: Average Score, dtype: float64
```

[67]:
```
df['Average Score'][0]
```

[67]:
```
70.24901960784314
```

[68]:
```
df['Average Score'][1]
```

[68]:
```
65.8374827109267
```

[69]:
```
df['math score']
```

[69]:
```
gender
female    64.356863
male      68.728216
Name: math score, dtype: float64
```

[70]:
```
df['math score'][0]
```

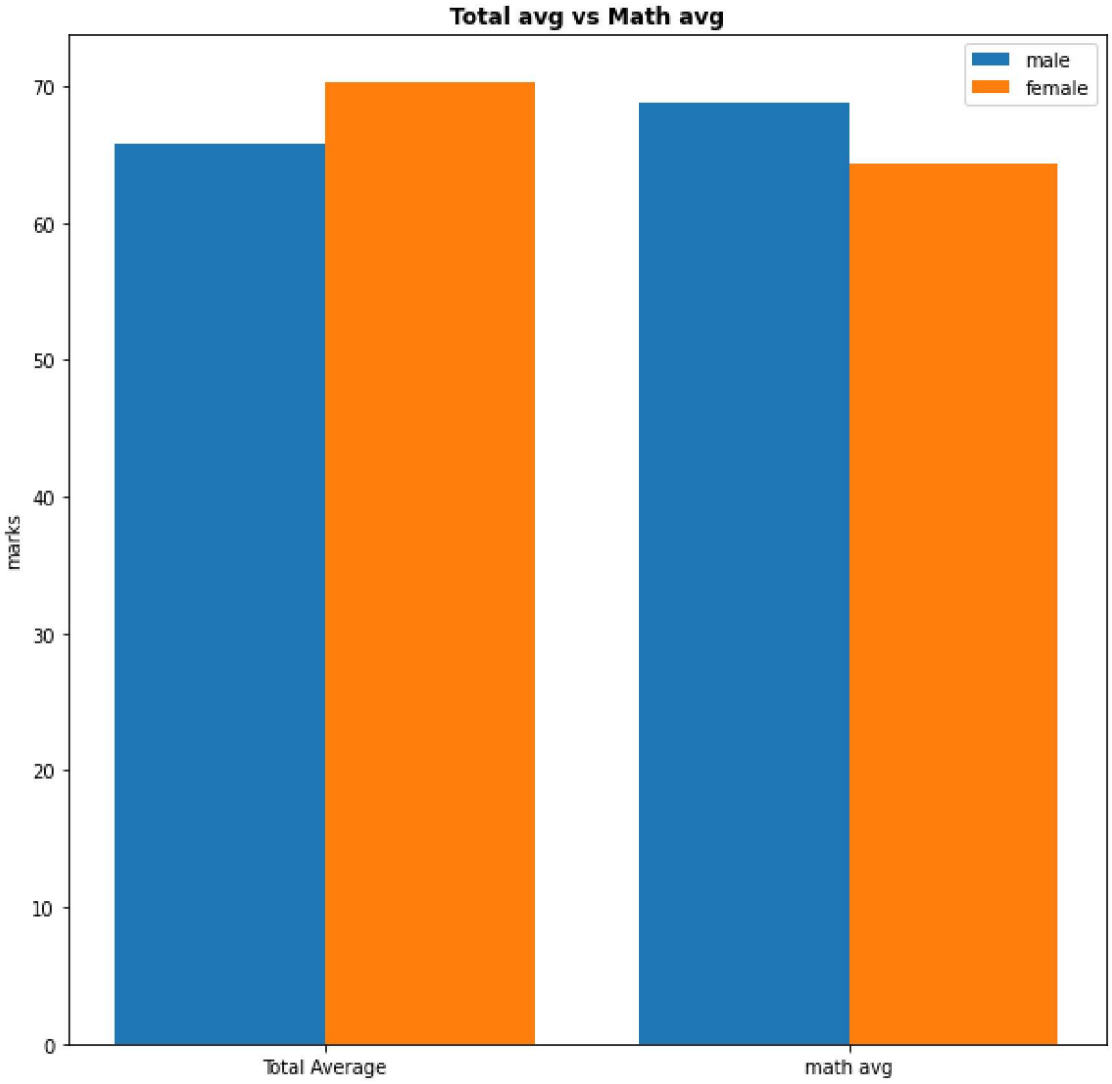
[70]:
```
64.35686274509804
```

[71]:
```
df['math score'][1]
```

[71]:
```
68.72821576763485
```

[72]:
```
plt.figure(figsize= (10, 10))
X = ['Total Average', 'math avg']
female_score = df['Average Score'][0],df['math score'][0]
male_score = df['Average Score'][1],df['math score'][1]
X_axis =  np.arange(len(X))
plt.bar(X_axis - 0.2, male_score, 0.4, label = 'male')
plt.bar(X_axis + 0.2, female_score, 0.4, label = 'female')

plt.xticks(X_axis, X)
plt.ylabel("marks")
plt.title("Total avg vs Math avg", fontweight = 'bold')
plt.legend()
```

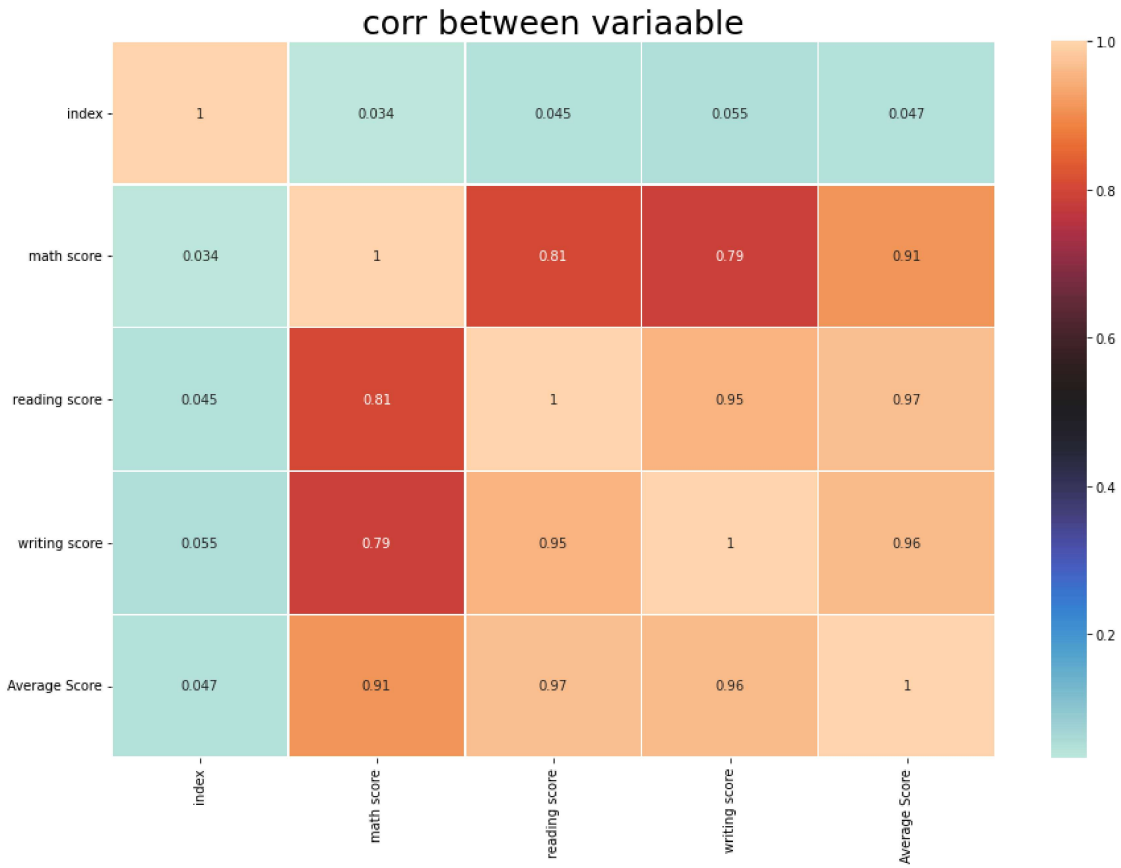
```
plt.show()
female_score
```



[72]: (70.24901960784314, 64.35686274509804)

Heatmap

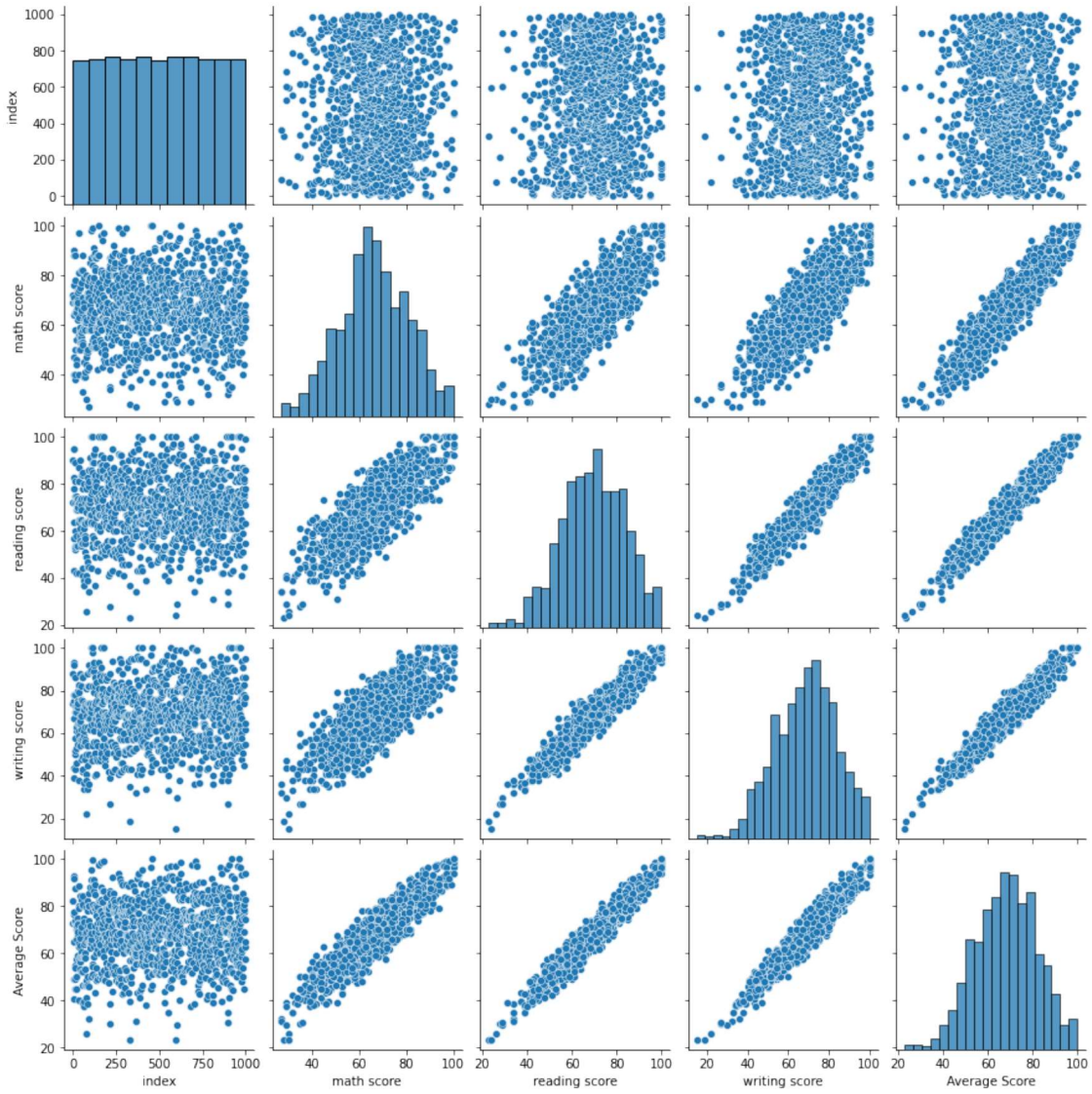
[73]:
```
sns.heatmap(data.corr(), annot = True, cmap='icefire', linewidths = 0.3)
fig = plt.gcf()
fig.set_size_inches(15, 10)
plt.title("corr between variaable", color = "black", size = 25)
plt.show()
```



Pairplot

[74]:
```
sns.pairplot(data)
```

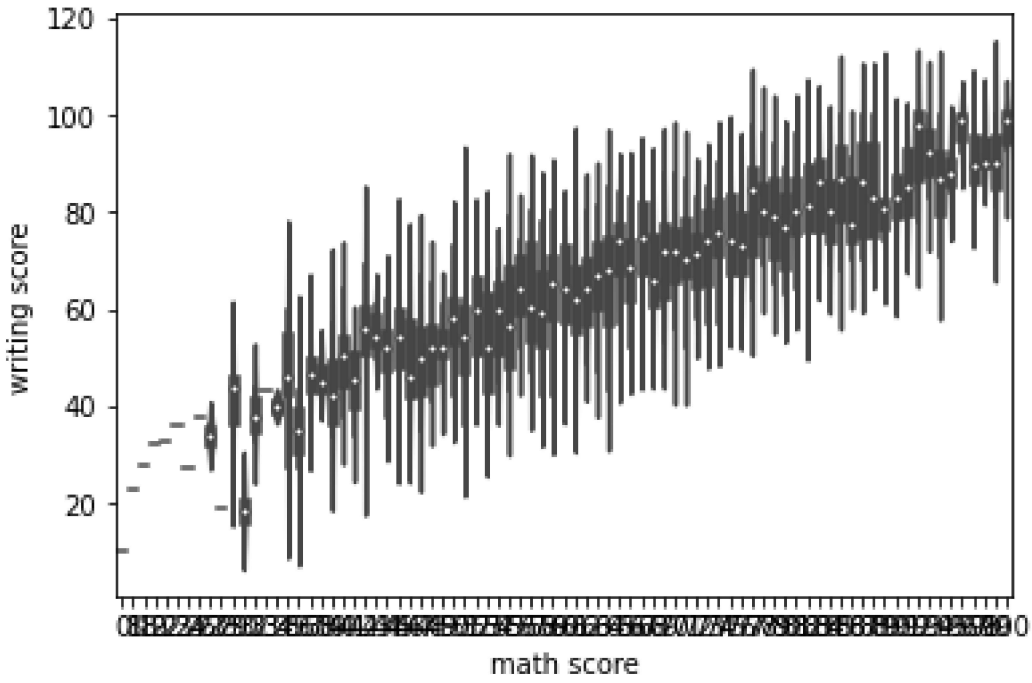
[74]:
```
<seaborn.axisgrid.PairGrid at 0x7ff68b218dd0>
```



ViolinpLot * It shows the data distribution

[75]:
```
sns.violinplot(data = numerical_data, x = numerical_data['math score'], y =␣
↪numerical_data['writing score'])
```

[75]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff688dfab50>
```



[76]:
```
sns.violinplot(data = numerical_data)
```

[76]: <matplotlib.axes._subplots.AxesSubplot at 0x7ff6889ad550>