EDA OF STUDENTS PERFORMANCE EXAM DATASET

August 3, 2023

DEEPAK YADAV(dy479958@gmail.com)

[1]:

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

[2]:

```
df=pd.read_csv("D:\stud.csv")
df.head()
```

[2]:

```
  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
```

[3]:

```
## Summary of the dataset
df.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
```

```
[4]: ## cheching Descriptive Statistic Summary of dataset
     df.describe()
```

```
[4]:        math_score  reading_score  writing_score
     count  1000.00000    1000.000000    1000.000000
     mean     66.08900      69.169000      68.054000
     std      15.16308      14.600192      15.195657
     min       0.00000      17.000000      10.000000
     25%      57.00000      59.000000      57.750000
     50%      66.00000      70.000000      69.000000
     75%      77.00000      79.000000      79.000000
     max     100.00000     100.000000     100.000000
```

Isights And Observation 1. From the above description of numerical data,all means values are very close to each other between 66 and 69 2. All the standard deviation close to each other between 14 and 15 3. While there is minimum 0 for maths some other having 10 and 17 value

```
[5]: ## List down all the dataset column names
     df.columns
```

```
[5]: Index(['gender', 'race_ethnicity', 'parental_level_of_education', 'lunch',
            'test_preparation_course', 'math_score', 'reading_score',
            'writing_score'],
           dtype='object')
```

```
[6]: ## Missing values in the dataset
     df.isnull().sum()
```

```
[6]: 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
```

There are no null or missing values

```
[7]: ## Duplicates Record
     df.duplicated()
```

```
[7]: 0      False
     1      False
```

```
2      False
3      False
4      False
       …
995    False
996    False
997    False
998    False
999    False
Length: 1000, dtype: bool
```

[ ]:

```
There are no duplicates values
```

[10]:

```python
## For sseing the duplicated records values
df[df.duplicated()]
```

[10]:

```
Empty DataFrame
Columns: [gender, race_ethnicity, parental_level_of_education, lunch,
test_preparation_course, math_score, reading_score, writing_score]
Index: []
```

[11]:

```python
## Remove the duplicates
df.drop_duplicates(inplace=True)
```

[12]:

```python
##  Checking the number of uniques values of each columns
df.nunique()
```

[12]:

```
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
```

[13]:

```python
[feature for feature in df.columns if df[feature].dtype=='O']
```

[13]:

```
['gender',
 'race_ethnicity',
 'parental_level_of_education',
 'lunch',
 'test_preparation_course']
```

[34]:

```python
# Seggregate numerical and categorical values
numerical_features=[feature for feature in df.columns if df[feature].dtype!='O']
```

```
categorical_feature=[feature for feature in df.columns if df[feature].
 ↪dtype=='O']
```

[35]:

```
numerical_features
```

[35]:

```
['math_score', 'reading_score', 'writing_score']
```

[36]:

```
categorical_feature
```

[36]:

```
['gender',
 'race_ethnicity',
 'parental_level_of_education',
 'lunch',
 'test_preparation_course']
```

[37]:

```
## Aggregate the total score with mean
df['total_score']=(df['math_score']+df['reading_score']+df['writing_score'])
df['average']=df['total_score']/3
df.head()
```

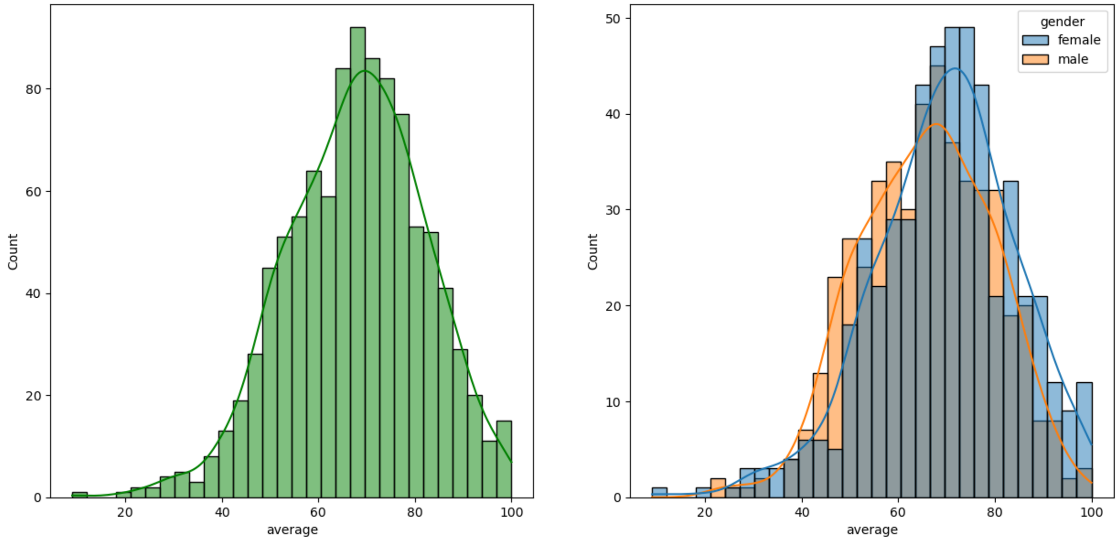
[37]:

```
   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

   total_score    average
0          218  72.666667
1          247  82.333333
2          278  92.666667
3          148  49.333333
4          229  76.333333
```

[38]:

```
##  Explore More Visualization
fig,axis=plt.subplots(1,2,figsize=(15,7))
plt.subplot(121)
sns.histplot(data=df,x='average',bins=30,kde=True,color='g')
plt.subplot(122)
sns.histplot(data=df,x='average',bins=30,kde=True,hue='gender')
```

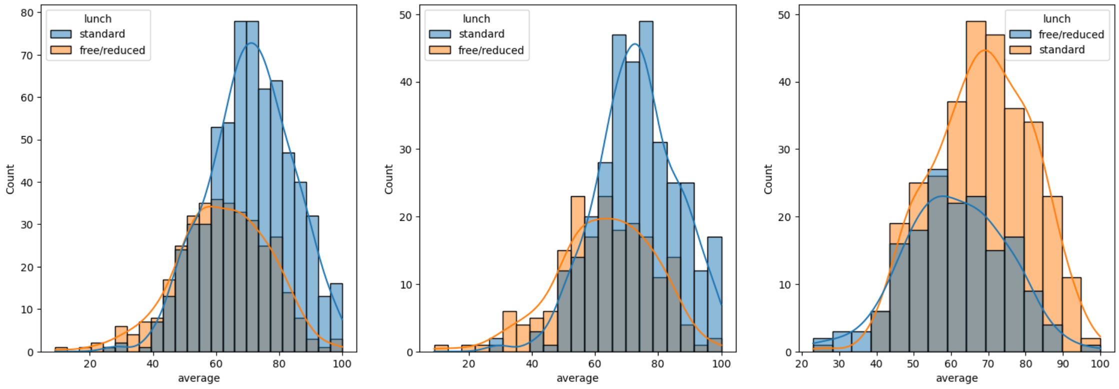
```
plt.show()
```



Insights 1. Female students tends to perform well than male students

[39]:

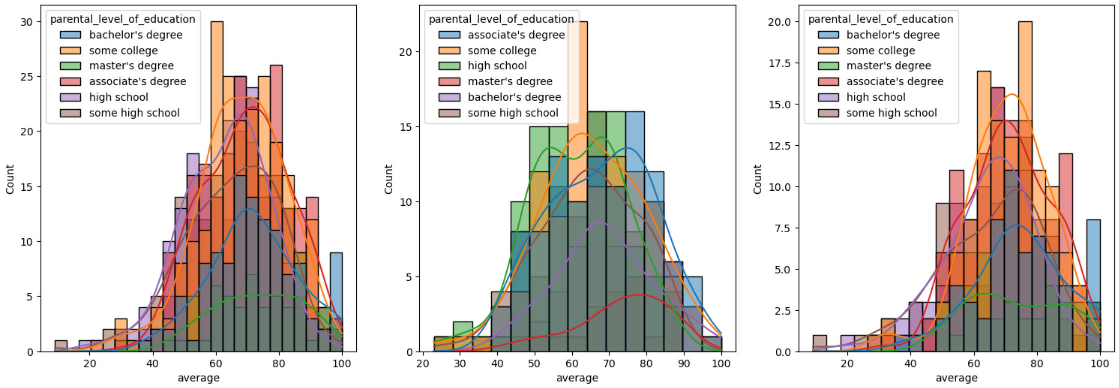
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
sns.histplot(data=df,x='average',kde=True,hue='lunch')
plt.subplot(142)
sns.histplot(data=df[df.gender=='female'],x='average',kde=True,hue='lunch')
plt.subplot(143)
sns.histplot(data=df[df.gender=='male'],x='average',kde=True,hue='lunch')
plt.show()
```



Insights 1. Standard Lunch helps students to perform well in exam 2. Standard Lunch helps perform well in exams be it a male of female

[40]:

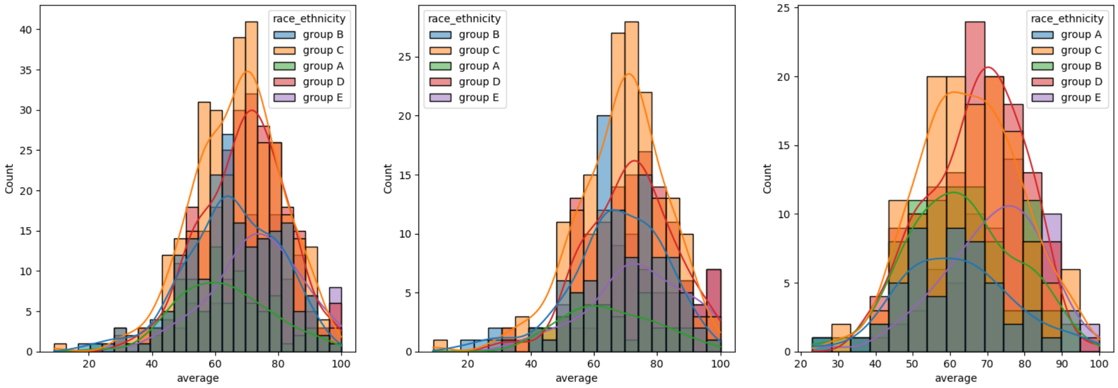
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
ax =sns.histplot(data=df,x='average',kde=True,hue='parental_level_of_education')
plt.subplot(142)
ax =sns.histplot(data=df[df.
  ↪gender=='male'],x='average',kde=True,hue='parental_level_of_education')
plt.subplot(143)
ax =sns.histplot(data=df[df.
  ↪gender=='female'],x='average',kde=True,hue='parental_level_of_education')
plt.show()
```



Insights 1. In general parents education don't help student to perform well in exam 2. Second plot shows that the parents whose education is of associate's degree or master's degree their male child perform well in exam 3. Third plot we can see there is no effect of parent's education on female students

[41]:

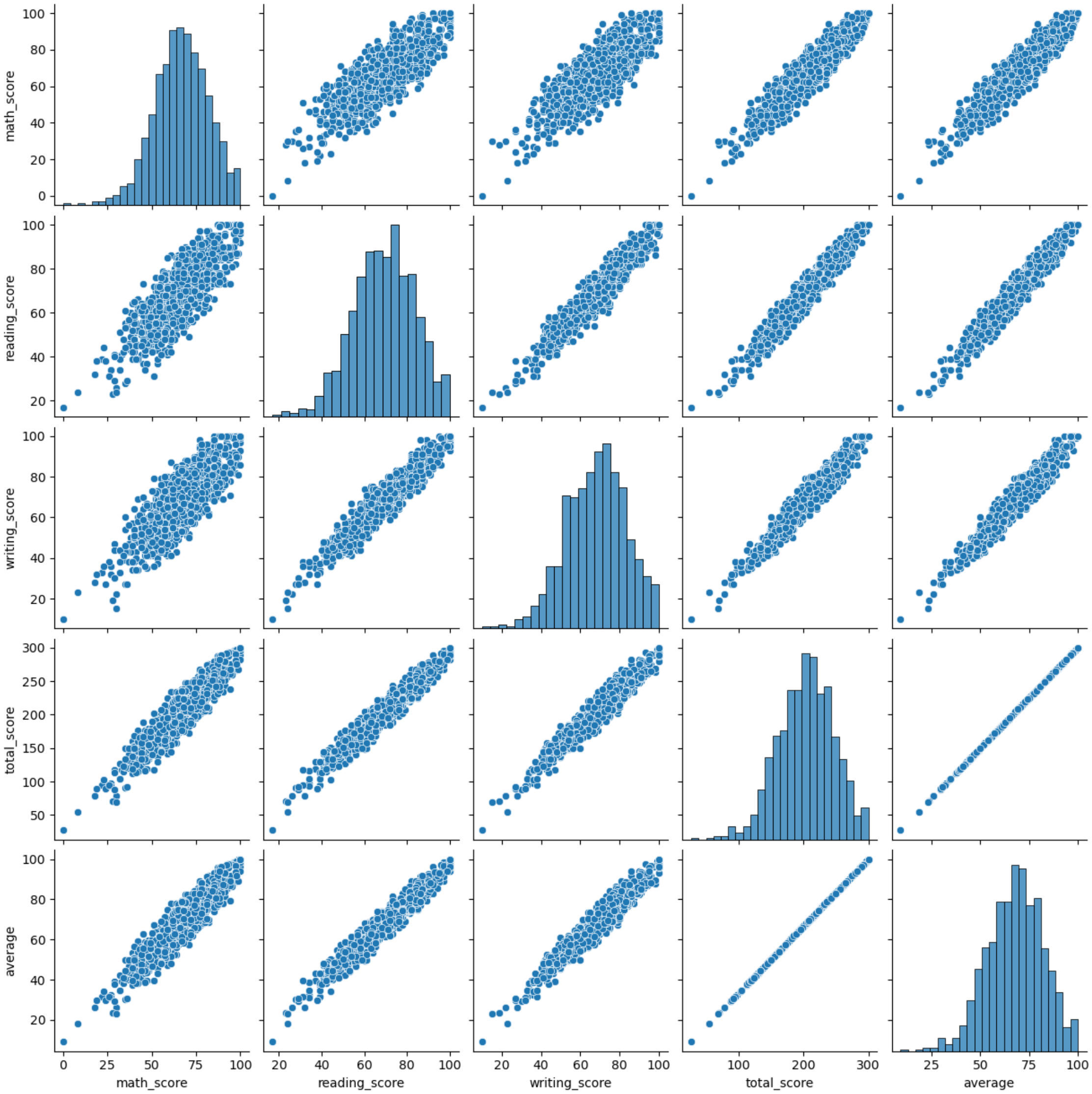
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
ax =sns.histplot(data=df,x='average',kde=True,hue='race_ethnicity')
plt.subplot(142)
ax =sns.histplot(data=df[df.
  ↪gender=='female'],x='average',kde=True,hue='race_ethnicity')
plt.subplot(143)
ax =sns.histplot(data=df[df.
  ↪gender=='male'],x='average',kde=True,hue='race_ethnicity')
plt.show()
```



Insights 1. Students of group A and group B tends to perform poorly in exam 2. Students of group A and group B tends to perform poorly in exam irrespective of whether they male or female

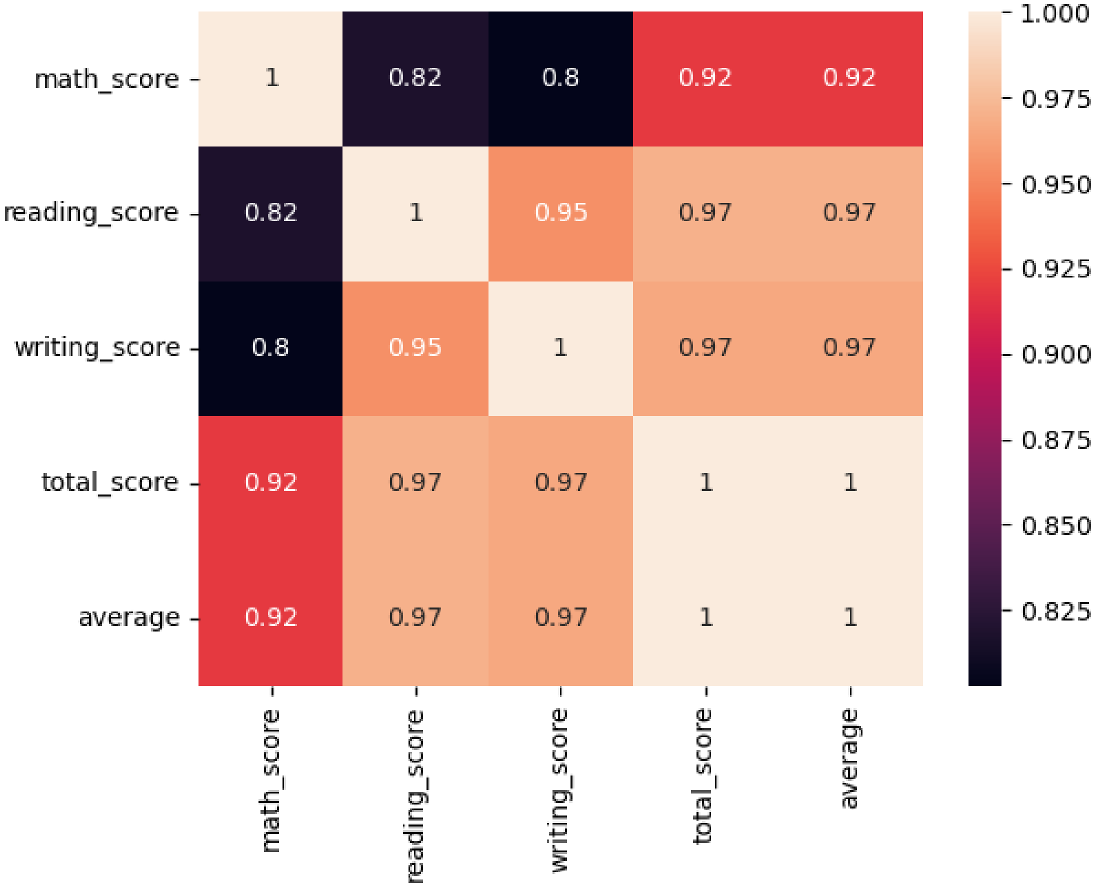
[43]:
```
sns.pairplot(df)
```

[43]:
```
<seaborn.axisgrid.PairGrid at 0x222601ef5b0>
```



```
[46]: sns.heatmap(df.corr(), annot=True)
```

```
[46]: <AxesSubplot:>
```


math_score
reading_score
writing_score
total_score
average
1
0.82
0.8
0.92
0.92
0.82
1
0.95
0.97
0.97
0.8
0.95
1
0.97
0.97
0.92
0.97
0.97
1
1
0.92
0.97
0.97
1
1
math_score
reading_score
writing_score
total_score
average
1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825

[ ]: