EDA Student Performance Indicator

1. Problem statement This project understands how the student's performance (test scores) is affected by other variables such as Gender, Ethnicity, Parental level of education, Lunch and Test preparation course.
2. Data Collection Dataset Source - https://www.kaggle.com/datasets/spscientist (https://www.kaggle.com/datasets/spscientist) students-performance-in-exams?datasetId=74977 The data consists of 8 column and 1000 rows.

Dataset Information

gender : sex of students -> (Male/female)

race/ethnicity : ethnicity of students -> (Group A, B,C, D,E)

parental level of education : parents' final education ->(bachelor's degree,some college,master's degree,associate's degree,high school)

lunch : having lunch before test (standard or free/reduced)

test preparation course : complete or not complete before test

math score

reading score

writing score

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

In [ ]:
```
#reading the Dataset

df=pd.read_csv("/content/sample_data/StudentsPerformance.csv")
```

In [ ]:
```
df.head()
```

Out[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

In [ ]:
```
df.shape
```

Out[4]:
```
(1000, 8)
```

In [ ]:
```
df.size
```

Out[5]:
```
8000
```

In [ ]:
```
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
```

In [ ]:
```
df.describe()
```

Out[8]:

	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

In [ ]:
```
# Some of the major EDA steps to perform

"""Checking Missing Values
Check Duplicates
Check the datatype
Check the number of unique values of each columns
check the statistics of dataset
check various categories present in the different categorical column"""
```

...

In [ ]:
```
#check missing values

df.isnull().sum()
```

Out[10]:
```
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
```

In [ ]:
```
df.isna().sum()
```

Out[11]:
```
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
```

In [ ]:
```
#checking duplicates

df.duplicated().sum()
```

Out[12]:
```
0
```

In [ ]:
```
## check datatypes
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
```

In [ ]:
```
##Checking the number of uniques values of each columns


df.nunique()
```

Out[14]:
```
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
```

In [ ]:
```
## Check the statistics of the dataset


df.describe()
```

Out[15]:

	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

In [ ]:
```
"""Insights or Observation

From the above description of numerical data,all means are very close to eac

All the standard deviation are also close- between 14.6- 15.19

While there is a minimum of 0 for maths,other are having 17 and 10 value"""
```

...

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

Out[17]:
```
['gender',
 'race/ethnicity',
 'parental level of education',
 'lunch',
 'test preparation course']
```

In [ ]:
```
#segrregate numerical and categorical features


numerical_features=[feature for feature in df.columns if df[feature].dtype!=
categorical_feature=[feature for feature in df.columns if df[feature].dtype=
```

In [ ]:
```
numerical_features
```

Out[25]:
```
['math score', 'reading score', 'writing score']
```

In [ ]:
```
categorical_feature
```

Out[26]:
```
['gender',
 'race/ethnicity',
 'parental level of education',
 'lunch',
 'test preparation course']
```

In [ ]:
```
#finding the total score and average score of the data

df['total_score']=(df['math score']+df['reading score']+df['writing score']
```

In [ ]:
```
df["total_score"]   #got total score of every student
```

Out[30]:
```
0      218
1      247
2      278
3      148
4      229
      ...
995    282
996    172
997    195
998    223
999    249
Name: total_score, Length: 1000, dtype: int64
```

In [ ]:
```
df["average"]=df["total_score"]/3
```

In [ ]:
```
df["average"] #got the average score of every student
```

Out[32]:
```
0      72.666667
1      82.333333
2      92.666667
3      49.333333
4      76.333333
         ...
995    94.000000
996    57.333333
997    65.000000
998    74.333333
999    83.000000
Name: average, Length: 1000, dtype: float64
```

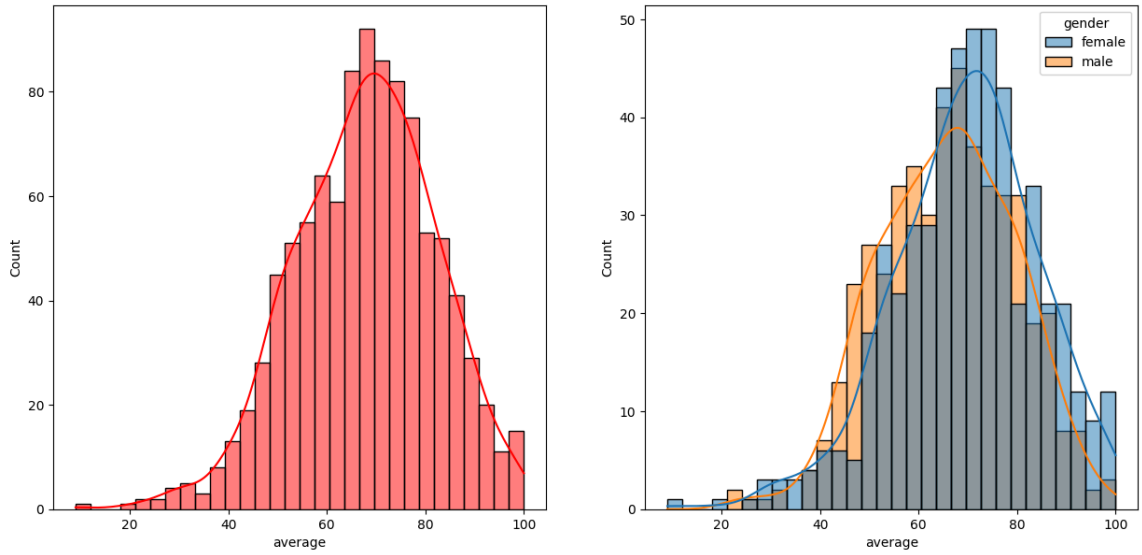
In [ ]:
```
df.head()
```

Out[33]:

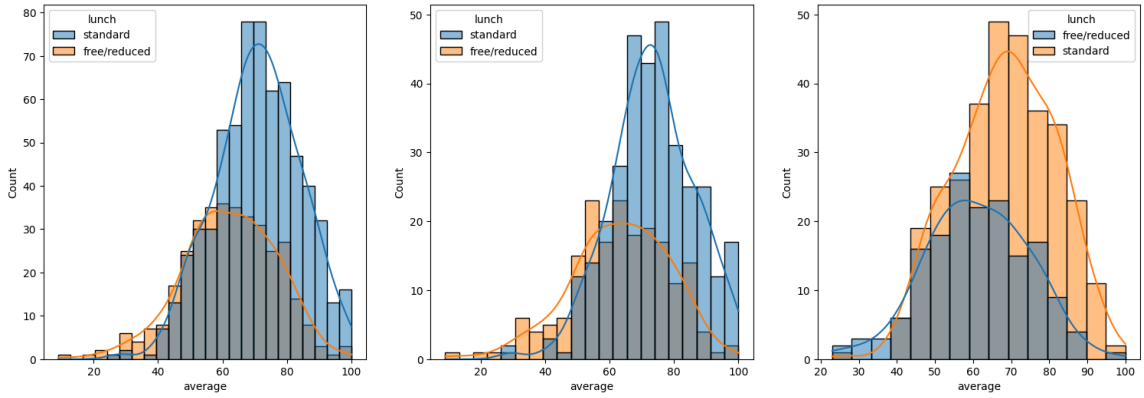
	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score	total_s
0	female	group B	bachelor's degree	standard	none	72	72	74	
1	female	group C	some college	standard	completed	69	90	88	
2	female	group B	master's degree	standard	none	90	95	93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

In [ ]:
```
# visualization

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

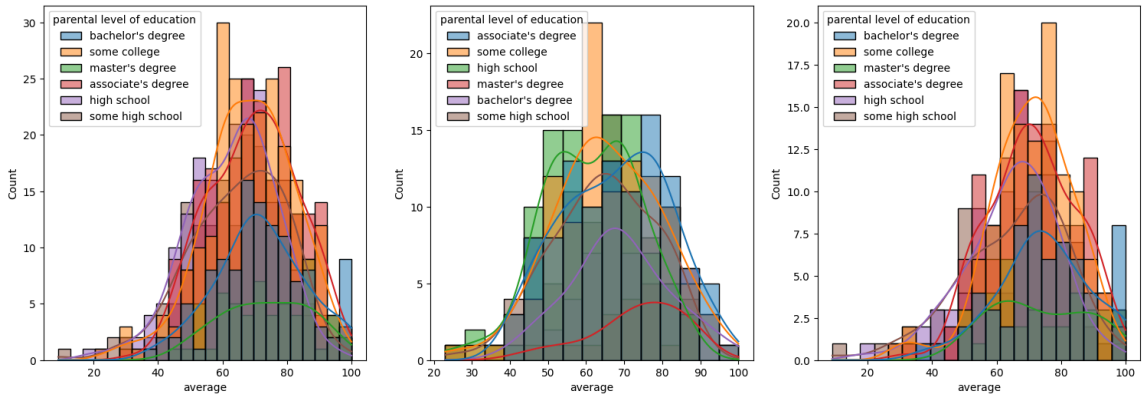


In [ ]:
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
sns.histplot(data=df,x='average',kde=True,hue='lunch')
plt.subplot(142)
sns.histplot(data=df[df.gender=='female'],x='average',kde=True,hue='lunch')
plt.subplot(143)
sns.histplot(data=df[df.gender=='male'],x='average',kde=True,hue='lunch')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



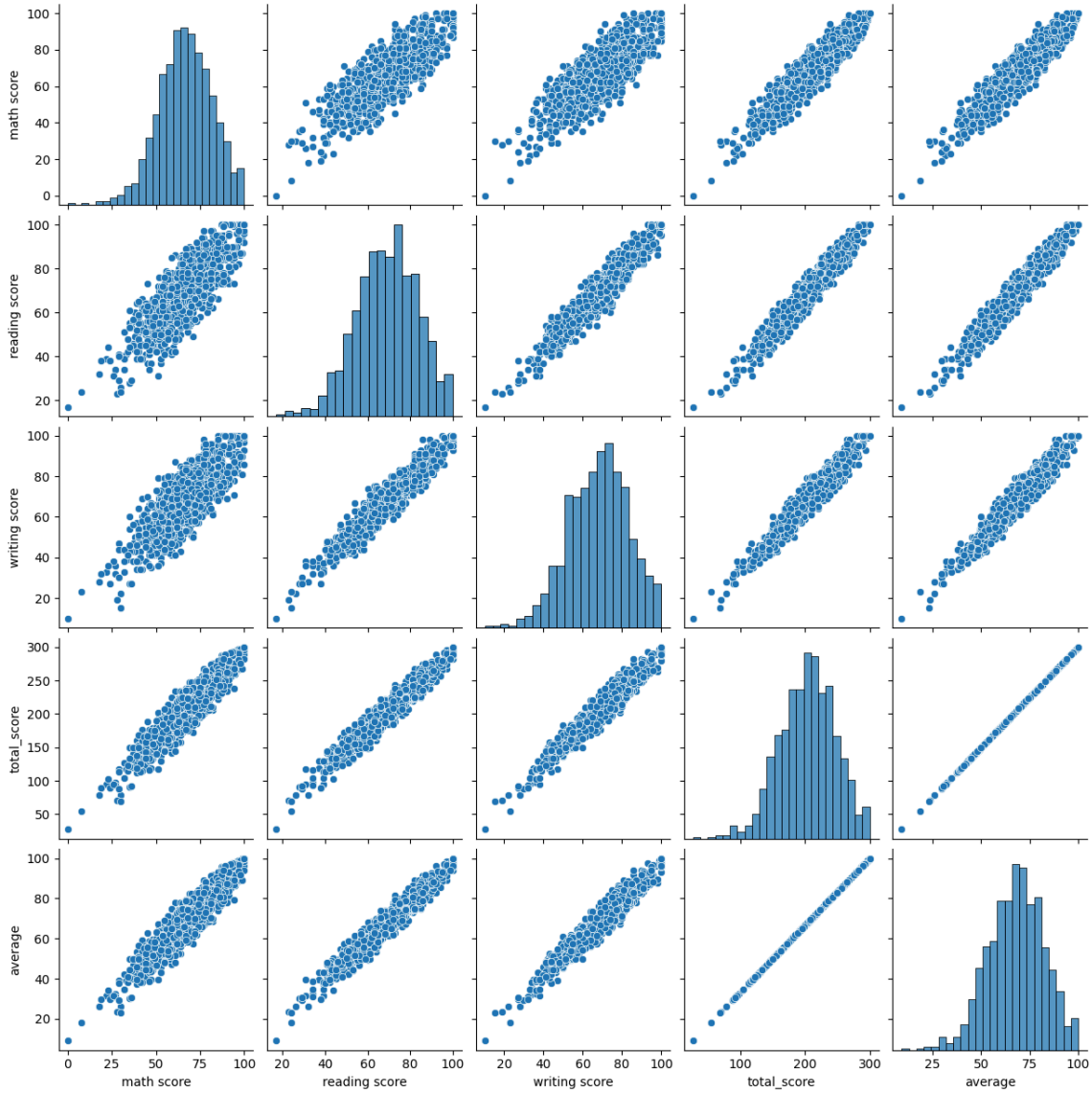
In [ ]:
```
"""Standard Lunch help students perform well in exams
Standard lunch helps perform well in exams be it a male of female"""
```

In [ ]:
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
ax=sns.histplot(data=df,x='average',kde=True,hue='parental level of educati
plt.subplot(142)
ax=sns.histplot(data=df[df.gender=='male'],x='average',kde=True,hue='parent
plt.subplot(143)
ax=sns.histplot(data=df[df.gender=='female'],x='average',kde=True,hue='pare
plt.show()
```



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

Out[40]: <seaborn.axisgrid.PairGrid at 0x7c336f31a3e0>



In [ ]: