Import libraries, set directory and read the file

In [223…
```
import pandas as pd
import os
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
from scipy.stats import chi2_contingency
```

In [224…
```
#Lets check if we are in correct directory
os.getcwd()
```

Out[224]:
```
'C:\\Users\\tural\\OneDrive\\Desktop\\Study Materials\\Datasets'
```

In [225…
```
#As our file is in different directory lets change the directory to access the file
os.chdir("C:\\Users\\tural\\OneDrive\\Desktop\\Study Materials\\Datasets")
```

In [226…
```
#Now we can read our dataset to start our job
df = pd.read_csv("StudentsPerformance.csv")
```

Explore and Clean the Data

In [227…
```
df.head()
```

Out[227]:

	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 [228…
```
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 [229…

```
#Checking wether we have some NA values or not
df.isna().sum()
```

Out[229]:

```
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 [230…

```
df.columns
```

Out[230]:

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

In [231…

```
#Renaming columns to be able to work easier
df.rename(columns = {'race/ethnicity':'race'}, inplace = True)
df.rename(columns = {'parental level of education':'parents_education'}, inplace = Tru
df.rename(columns = {'test preparation course':'prep_course'}, inplace = True)
df.columns = df.columns.str.replace(' ', '_')
```

In [232…

```
#parents education seems to be a bit messy so lets check the unique values and see if
pd.unique(df['parents_education'])
```

Out[232]:

```
array(["bachelor's degree", 'some college', "master's degree",
       "associate's degree", 'high school', 'some high school'],
      dtype=object)
```

In [233…

```
#Renaming some strings inside of data frame to look more organized
df = df.replace('some ','', regex=True)
df = df.replace(' degree', '', regex=True)
```

Explaratory Data Analysis

In [234…

```
# Before starting the descriptive statistics lets create a column which will be the av
df['avg_score'] = np.mean(df, axis = 1)
df = df.round(1)
```

```
C:\Users\tural\anaconda3\lib\site-packages\numpy\core\fromnumeric.py:3438: FutureWarn
ing: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None')
is deprecated; in a future version this will raise TypeError.  Select only valid colu
mns before calling the reduction.
  return mean(axis=axis, dtype=dtype, out=out, **kwargs)
```

In [235…

```
df.head()
```

Out[235]:

	gender	race	parents_education	lunch	prep_course	math_score	reading_score	writing_s
0	female	group B	bachelor's	standard	none	72	72	
1	female	group C	college	standard	completed	69	90	
2	female	group B	master's	standard	none	90	95	
3	male	group A	associate's	free/reduced	none	47	57	
4	male	group C	college	standard	none	76	78	

In [236…

```
df.describe()
```

Out[236]:

	math_score	reading_score	writing_score	avg_score
count	1000.00000	1000.000000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000	67.769800
std	15.16308	14.600192	15.195657	14.257197
min	0.00000	17.000000	10.000000	9.000000
25%	57.00000	59.000000	57.750000	58.300000
50%	66.00000	70.000000	69.000000	68.300000
75%	77.00000	79.000000	79.000000	77.700000
max	100.00000	100.000000	100.000000	100.000000

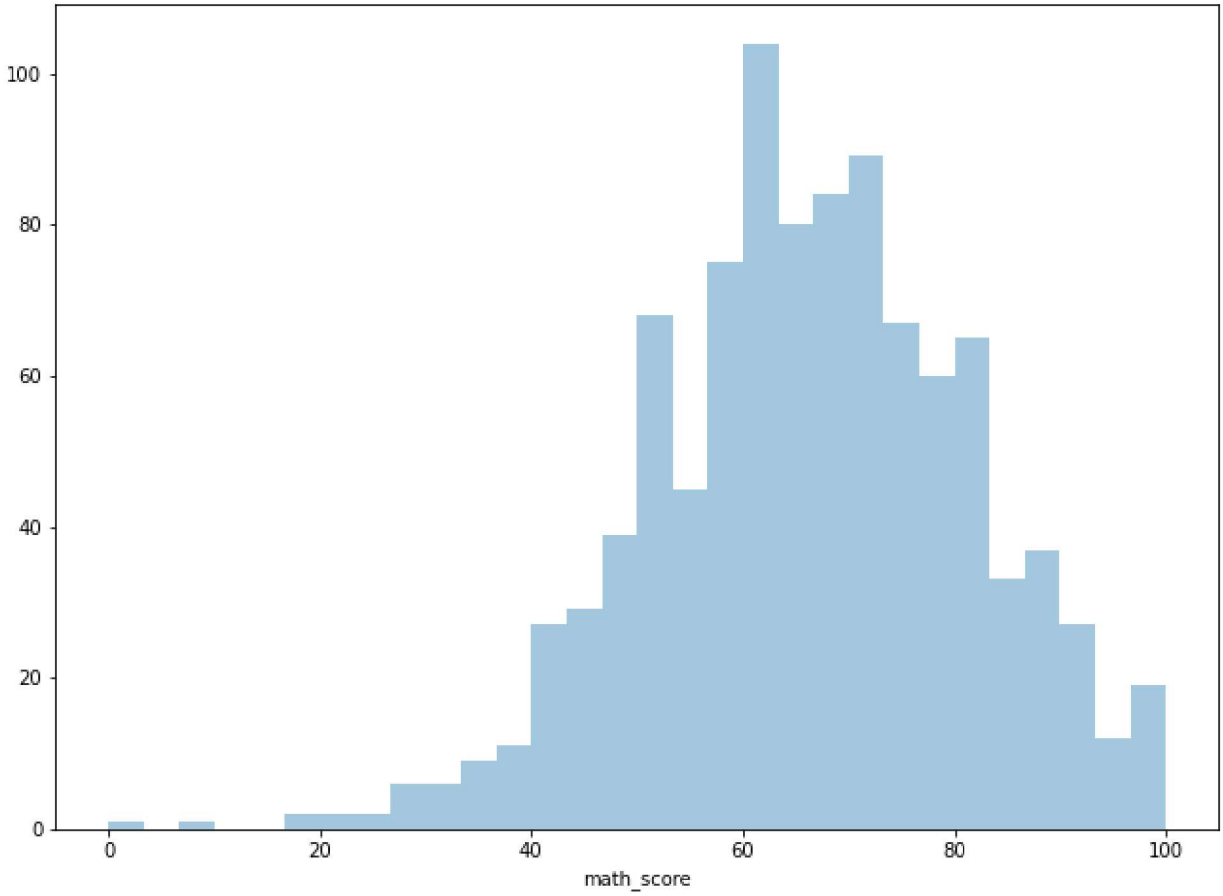
In [237…

```
fig, ax = plt.subplots(figsize=(11, 8))
sns.distplot(df['math_score'], hist=True, bins = 30, kde=False, )
```

```
C:\Users\tural\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[237]:

```
<AxesSubplot:xlabel='math_score'>
```

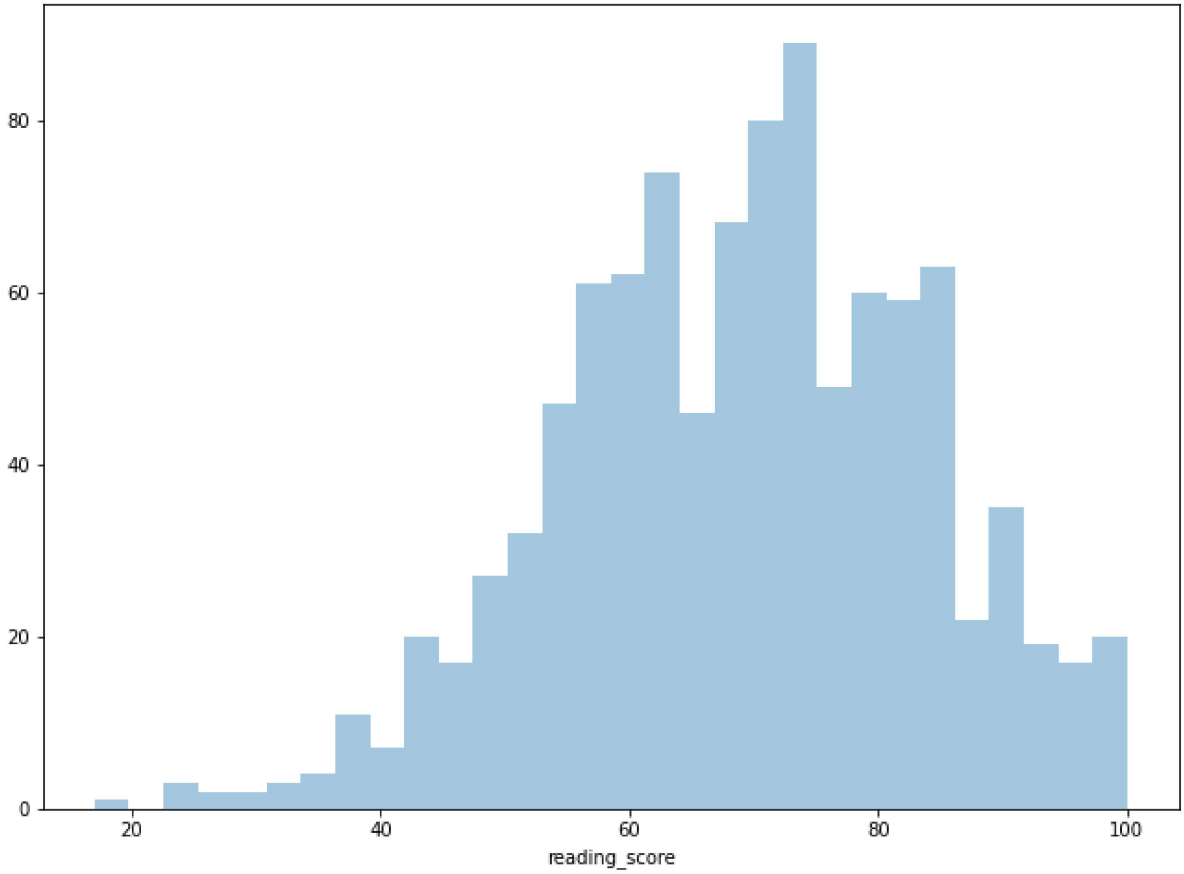


In [238…

```
fig, ax = plt.subplots(figsize=(11, 8))
sns.distplot(df['reading_score'], hist=True, bins = 30, kde=False, )
```

Out[238]:

```
<AxesSubplot:xlabel='reading_score'>
```

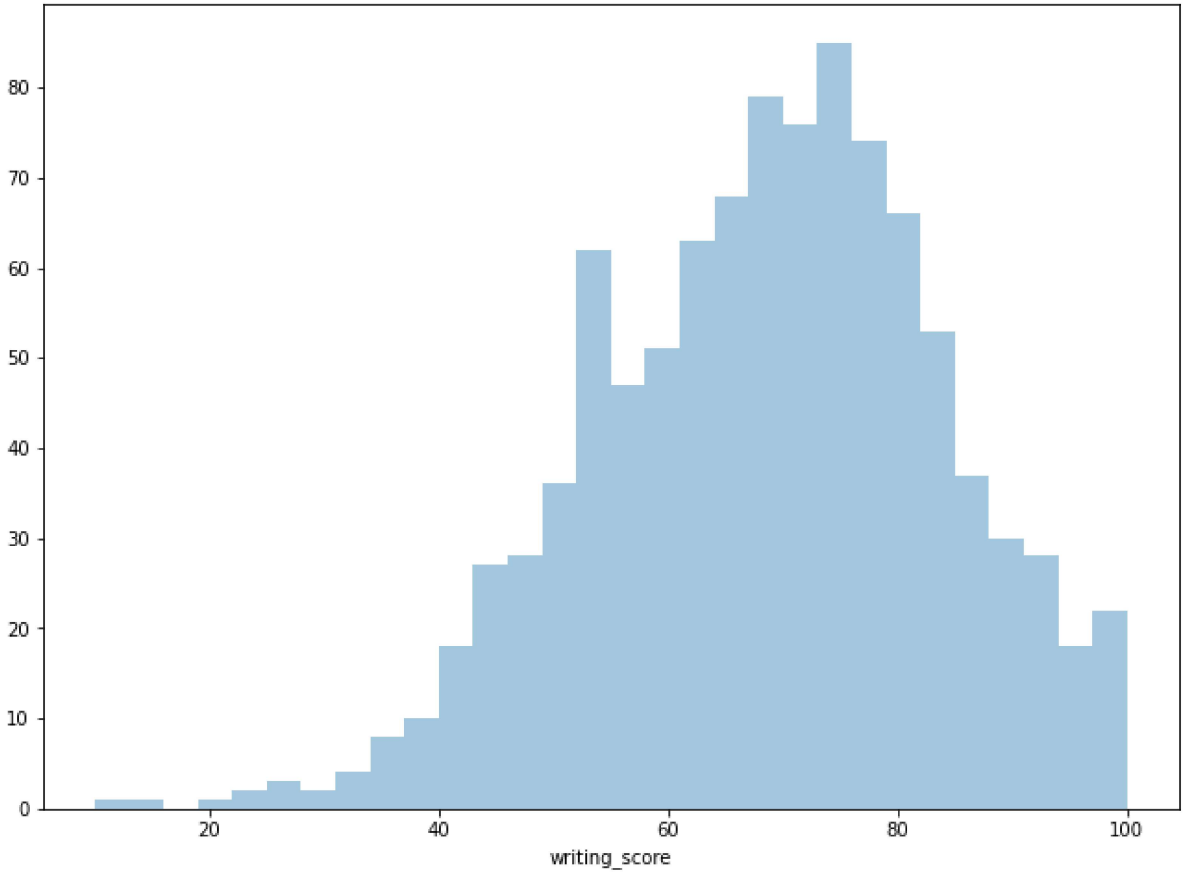


In [239…

```
fig, ax = plt.subplots(figsize=(11, 8))
sns.distplot(df['writing_score'], hist=True, bins = 30, kde=False, )
```

Out[239]:

```
<AxesSubplot:xlabel='writing_score'>
```

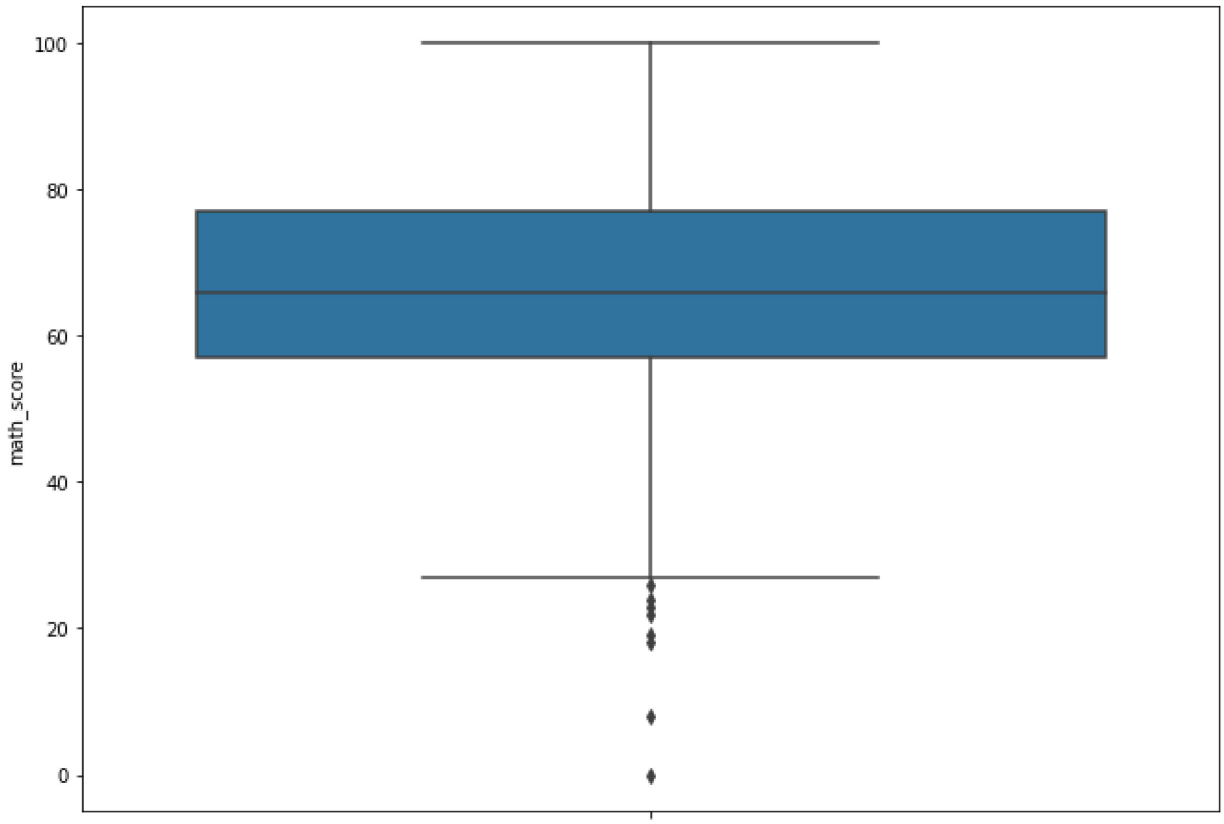


In [240…

```
fig, ax = plt.subplots(figsize=(11, 8))
sns.boxplot(y = 'math_score', data = df)
```

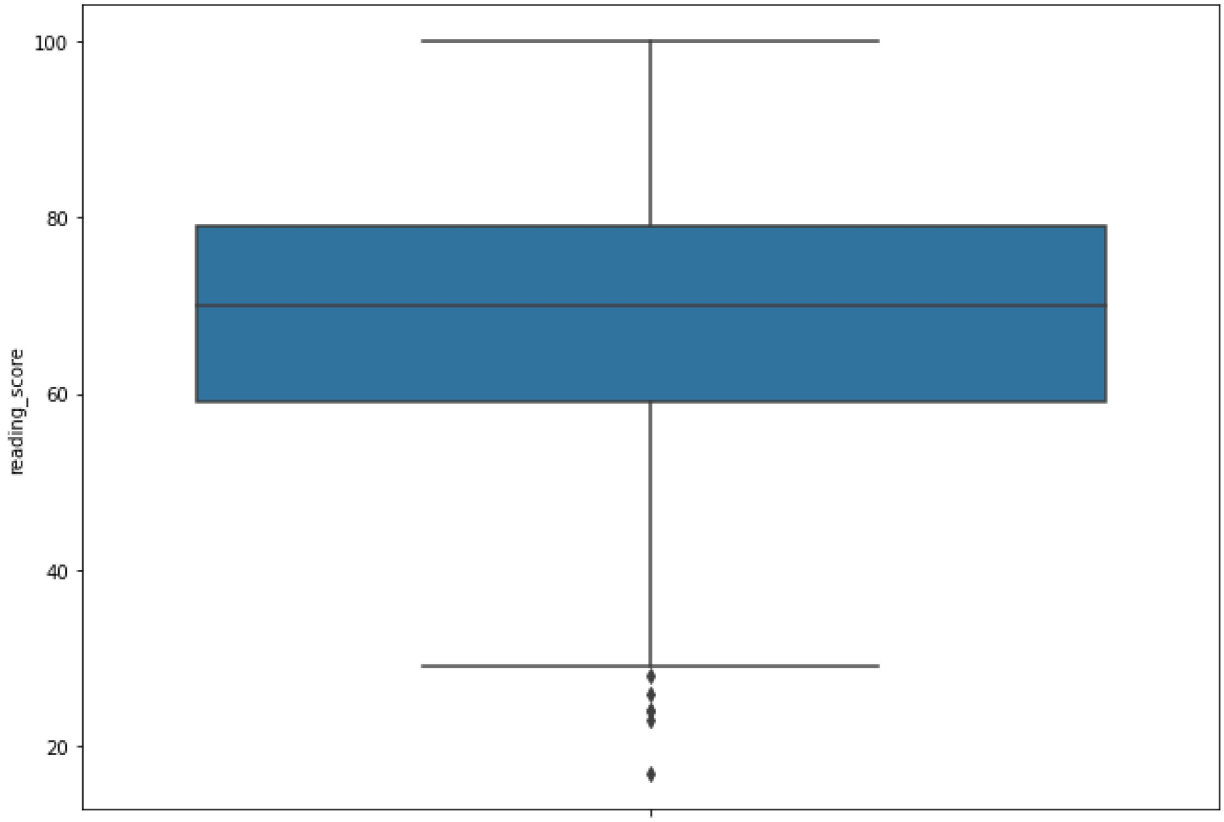
Out[240]:

```
<AxesSubplot:ylabel='math_score'>
```



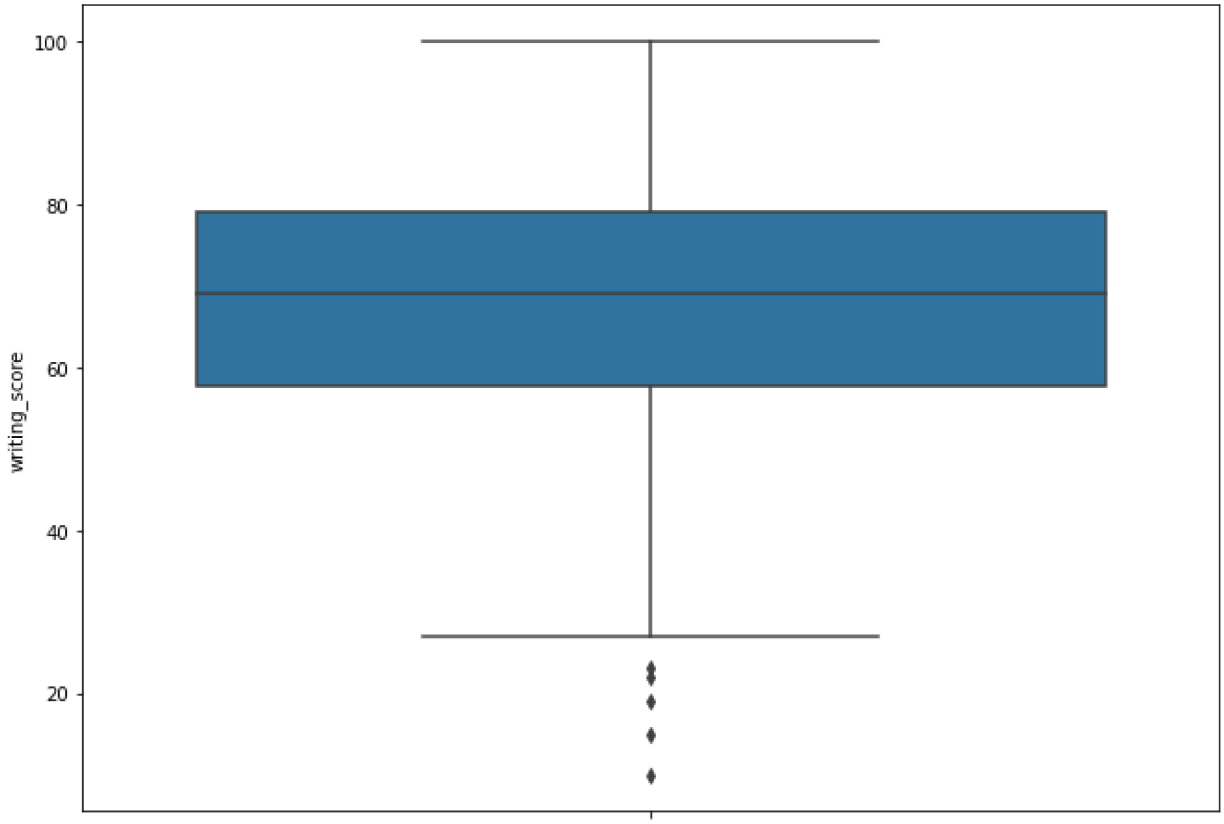
In [241…
```
fig, ax = plt.subplots(figsize=(11, 8))
sns.boxplot(y = 'reading_score', data = df)
```

Out[241]:
```
<AxesSubplot:ylabel='reading_score'>
```



In [242…
```
fig, ax = plt.subplots(figsize=(11, 8))
sns.boxplot(y = 'writing_score', data = df)
```

Out[242]:
```
<AxesSubplot:ylabel='writing_score'>
```



In [243…
```
df.columns
```

Out[243]:
```
Index(['gender', 'race', 'parents_education', 'lunch', 'prep_course',
       'math_score', 'reading_score', 'writing_score', 'avg_score'],
      dtype='object')
```

In [244…
```
df['parents_education'].value_counts(normalize = True) * 100
```

Out[244]:
```
high school    37.5
college        22.6
associate's    22.2
bachelor's     11.8
master's        5.9
Name: parents_education, dtype: float64
```

In [245…
```
pd.crosstab(columns = df['race'], index = df['gender'], margins = True, normalize = Tr
```

Out[245]:

race	group A	group B	group C	group D	group E	All
gender						
female	3.6	10.4	18.0	12.9	6.9	51.8
male	5.3	8.6	13.9	13.3	7.1	48.2
All	8.9	19.0	31.9	26.2	14.0	100.0

In [246…
```
pd.crosstab(index = df['parents_education'], columns = df['race'], margins = True, nor
```

Out[246]:

race	group A	group B	group C	group D	group E	All
parents_education						
associate's	1.4	4.1	7.8	5.0	3.9	22.2
bachelor's	1.2	2.0	4.0	2.8	1.8	11.8
college	1.8	3.7	6.9	6.7	3.5	22.6
high school	4.2	8.6	11.3	9.4	4.0	37.5
master's	0.3	0.6	1.9	2.3	0.8	5.9
All	8.9	19.0	31.9	26.2	14.0	100.0

In [247…

```
pd.crosstab(index = df['prep_course'], columns = df['parents_education'], margins = Tr
```

Out[247]:

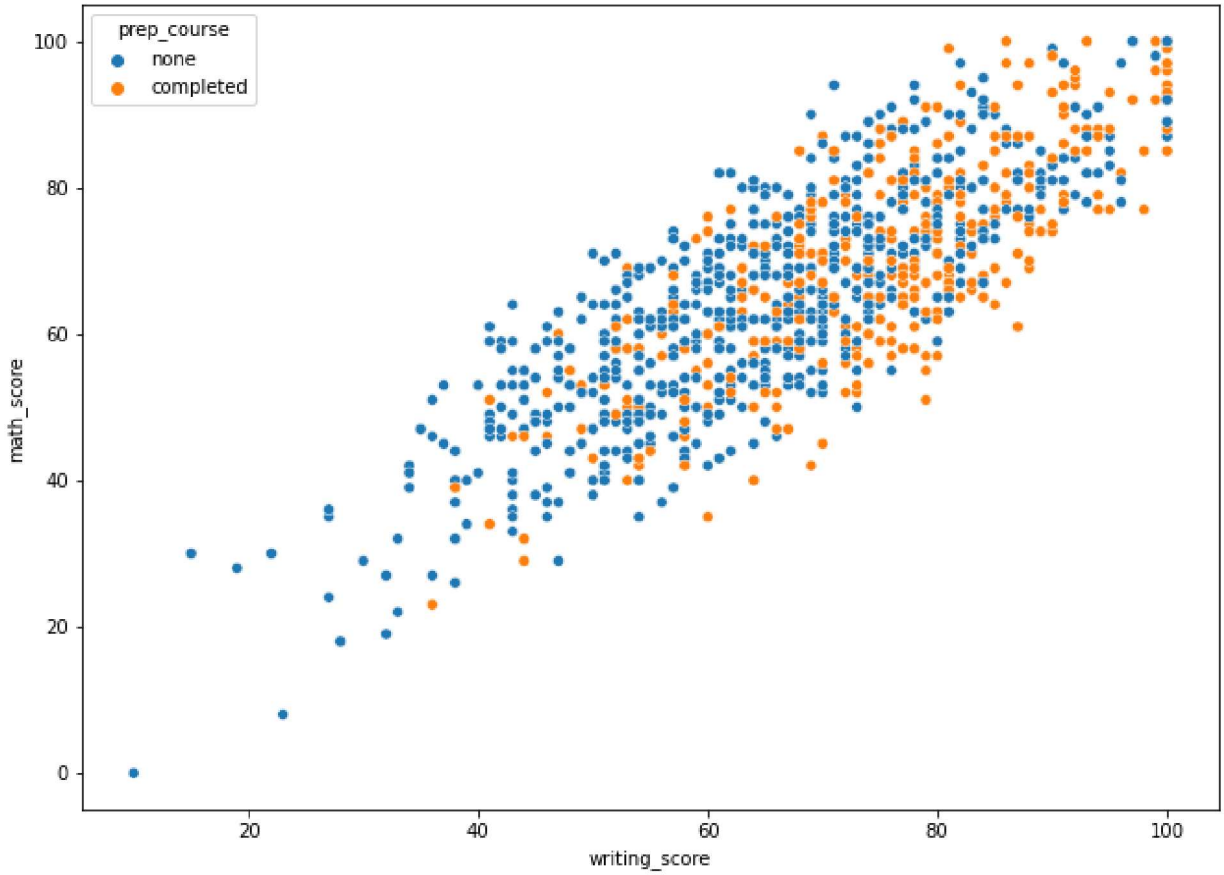
parents_education	associate's	bachelor's	college	high school	master's	All
prep_course						
completed	8.2	4.6	7.7	13.3	2.0	35.8
none	14.0	7.2	14.9	24.2	3.9	64.2
All	22.2	11.8	22.6	37.5	5.9	100.0

In [248…

```
fig, ax = plt.subplots(figsize=(11, 8))
sns.scatterplot(x = df['writing_score'], y = df['math_score'], hue = df['prep_course']
```

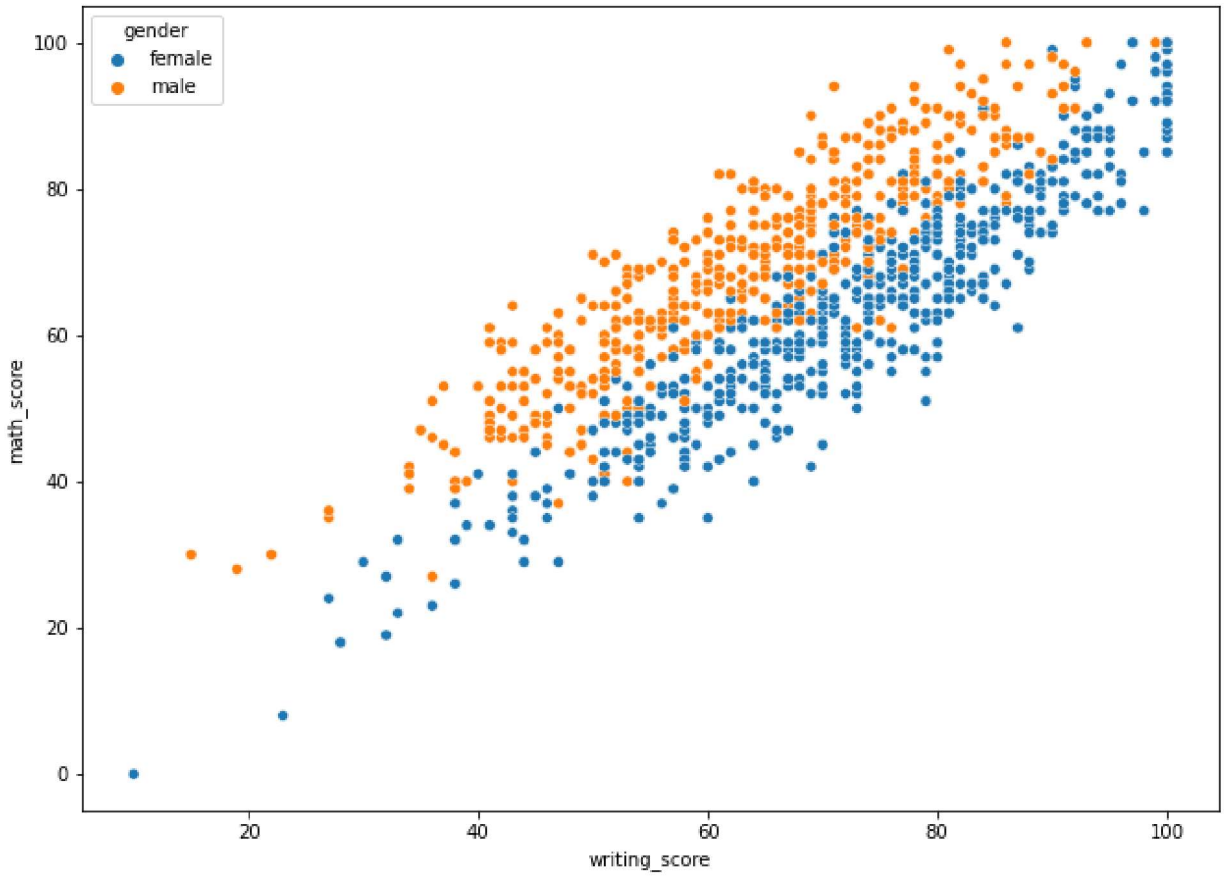
Out[248]:

```
<AxesSubplot:xlabel='writing_score', ylabel='math_score'>
```



In [249…
```
fig, ax = plt.subplots(figsize=(11, 8))
sns.scatterplot(x = df['writing_score'], y = df['math_score'], hue = df['gender'])
```

Out[249]:
```
<AxesSubplot:xlabel='writing_score', ylabel='math_score'>
```



In [250…
```
#Lets find out the outliers and remove them from our dataset for math score first
Q1 = df.math_score.quantile(0.25)
Q3 = df.math_score.quantile(0.75)
IQR = Q3-Q1
IQR
```

Out[250]:
```
20.0
```

In [251…
```
df = df[~((df.math_score <(Q1 - 1.5 * IQR)) | (df.math_score > (Q3 + 1.5 * IQR)))]
```

In [ ]:
```
# Outliers for reading
```

In [252…
```
Q1 = df.reading_score.quantile(0.25)
Q3 = df.reading_score.quantile(0.75)
IQR = Q3-Q1
IQR
```

Out[252]:
```
20.0
```

In [253…
```
df = df[~((df.reading_score <(Q1 - 1.5 * IQR)) | (df.reading_score > (Q3 + 1.5 * IQR))
```

In [ ]:
```
# Outliers for Writing score
```

In [254…
```
Q1 = df.writing_score.quantile(0.25)
```

```
Q3 = df.writing_score.quantile(0.75)
IQR = Q3-Q1
IQR
```

Out[254]:
```
21.0
```

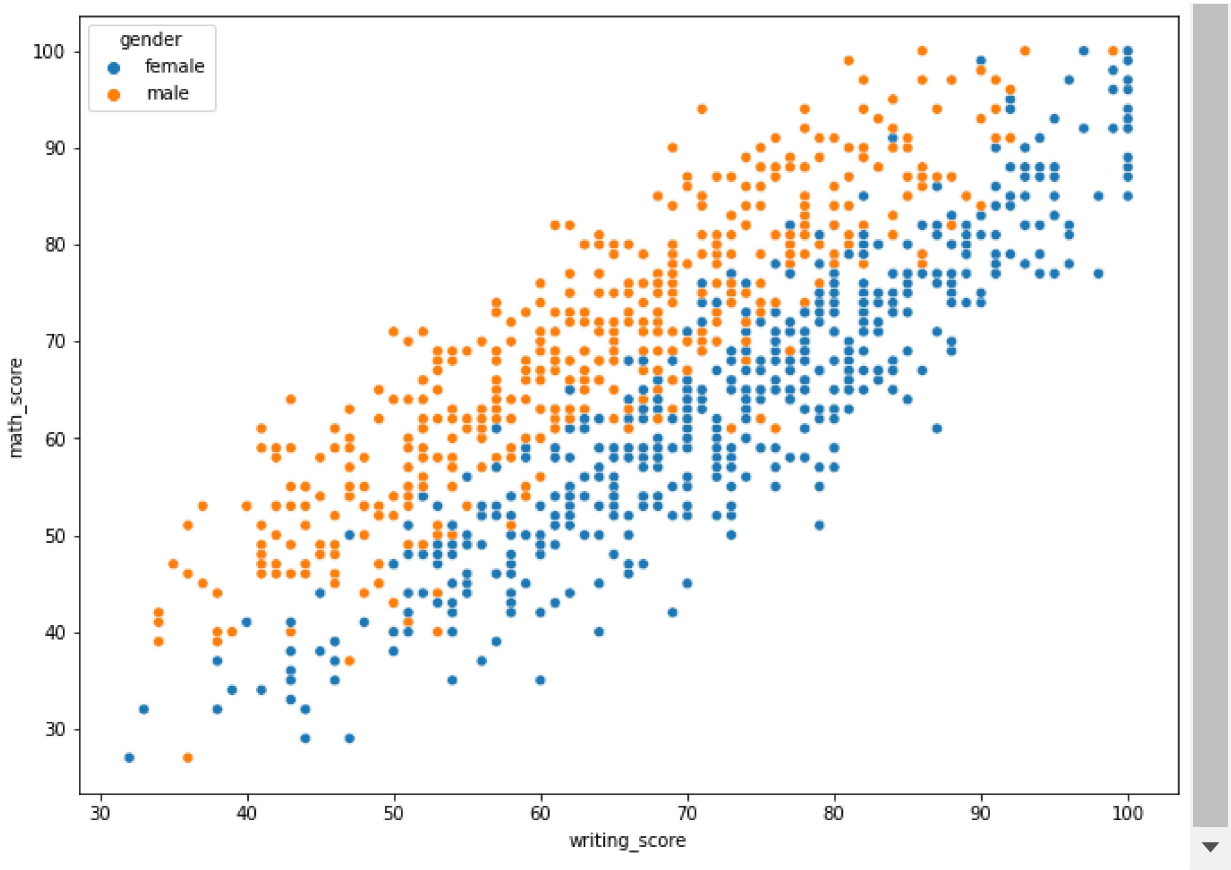
In [255…
```
df = df[~((df.writing_score <(Q1 - 1.5 * IQR)) | (df.writing_score > (Q3 + 1.5 * IQR))
```

In [280…
```
# Now lets check our data again with below visualizations as well
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 986 entries, 0 to 999
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   gender             986 non-null    object 
 1   race               986 non-null    object 
 2   parents_education  986 non-null    object 
 3   lunch              986 non-null    object 
 4   prep_course        986 non-null    object 
 5   math_score         986 non-null    int64  
 6   reading_score      986 non-null    int64  
 7   writing_score      986 non-null    int64  
 8   avg_score          986 non-null    float64
 9   label              986 non-null    object 
dtypes: float64(1), int64(3), object(6)
memory usage: 84.7+ KB
```

In [257…
```
fig, ax = plt.subplots(figsize=(11, 8))
sns.scatterplot(x = df['writing_score'], y = df['math_score'], hue = df['gender'])
```

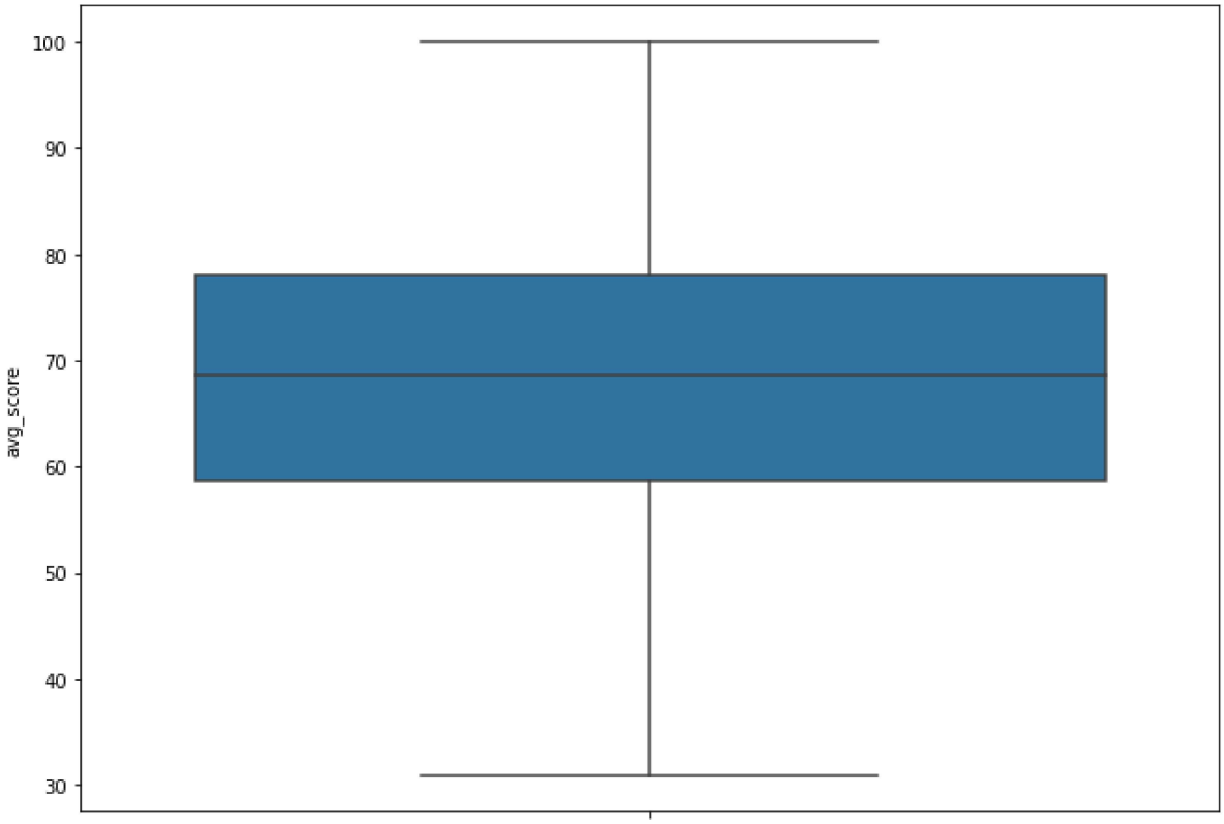
Out[257]:
```
<AxesSubplot:xlabel='writing_score', ylabel='math_score'>
```



In [258…

```
fig, ax = plt.subplots(figsize=(11, 8))
sns.boxplot(y = 'avg_score', data = df, )
```

Out[258]: <AxesSubplot:ylabel='avg_score'>



Lets try to prove if there is some relationship between parents education and students score. In

order to do this we will divide average score into 2 pieces where we will put over median as 1(for high results) 0(low results).

In [267…
```
print(df.avg_score.median())
print(df.avg_score.mean())
```

```
68.7
68.35446247464508
```

In [260…
```
df['label'] = pd.cut(x = df['avg_score'], bins=[0, 68.7, 100], labels=['0', '1'])
```

In [261…
```
df.label = df.label.astype('str')
```

In [262…
```
table = pd.crosstab(index = df['parents_education'], columns = df['label'])
```

In [263…
```
table
```

Out[263]:

label	0	1
parents_education		
associate's	103	118
bachelor's	48	70
college	110	112
high school	217	149
master's	22	37

In [264…
```
chi2_contingency(table)
```

Out[264]:
```
(21.390781120565926,
 0.0002648900389868005,
 4,
 array([[112.06896552, 108.93103448],
        [ 59.83772819,  58.16227181],
        [112.57606491, 109.42393509],
        [185.59837728, 180.40162272],
        [ 29.9188641 ,  29.0811359 ]]))
```

The first value shows the chi square value of 21.39. The chance of such a value or even more extrem, in a sample if there is no association in the population is 0.00026 which is less than 0.05 and we reject Null hyptothtesis. So there is something between parents education and students avg scores that makes it not by chance

Furthermore, 4 degree of freedom have the critical value of 9.49. As we have 21 which is exceeding our critical value, we reject Null hyptothesis.

The same way we will do it for the prep course

In [268…
```
table1 = pd.crosstab(index = df['prep_course'], columns = df['label'])
```

In [269…
```
table1
```

Out[269]:

label	0	1
prep_course		
completed	129	228
none	371	258

In [270…

```
chi2_contingency(table1)
```

Out[270]:

```
(46.65535009083159,
 8.46355967503398e-12,
 1,
 array([[181.03448276, 175.96551724],
        [318.96551724, 310.03448276]]))
```

Here we also have p value which is very close to 0, which means there is some relationship between students' average scores and preparation course. We reject the Null Hyptothesis

Furthermore, 1 degree of freedom have the critical value of 3.84. As we have 46 which is exceeding our critical value, we reject Null hyptothesis.

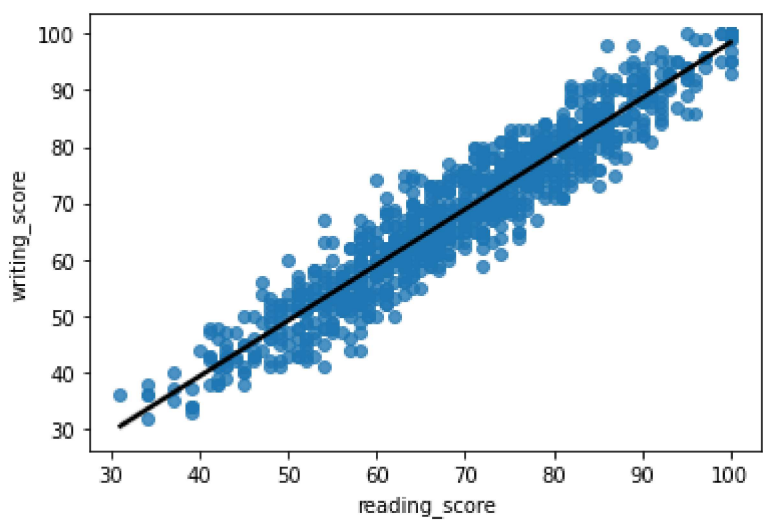
Linear Model Development to predict writing score exams after reading exams

In [278…

```
#Lets create a scatter plot with regression line to get sense if writing and reading i
sns.regplot(x = df['reading_score'], y = df['writing_score'], data = df, line_kws={"co
```

Out[278]:

```
<AxesSubplot:xlabel='reading_score', ylabel='writing_score'>
```



In [281…

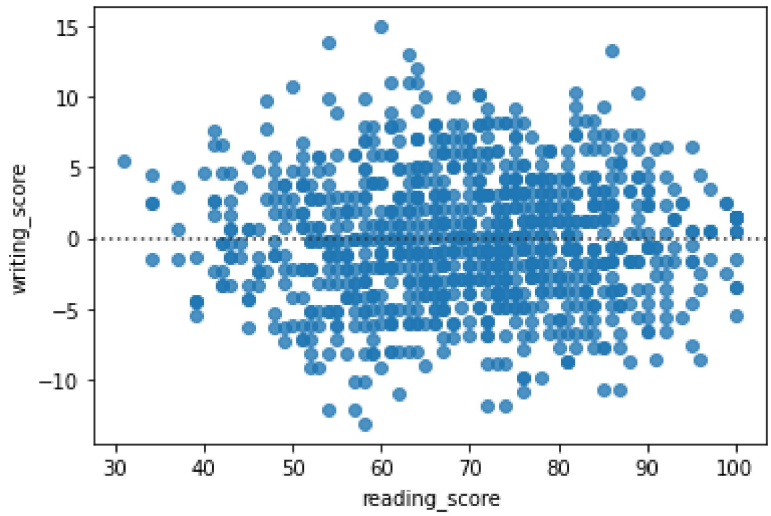
```
#Lets see the peaarson coefficient and p statistic value to see if the correlation is
pearson_coef, p_value = stats.pearsonr(df['writing_score'], df['reading_score'])
print(pearson_coef, p_value)
```

```
0.9498438915235968 0.0
```

In [284…

```
#As the residuals are evenly distributed along the mean axes we can conclude that line
sns.residplot(df['reading_score'], df['writing_score'])
```

Out[284]: <AxesSubplot:xlabel='reading_score', ylabel='writing_score'>



In [285…

```
#Lets develop our model and then show our equation for predicting the writing score
X = df[['reading_score']]
Y = df['writing_score']
lm.fit(X, Y)
Yhat = lm.predict(X)
print ("writing_score = ", "reading_score", "*", lm.coef_, "+", lm.intercept_)
```

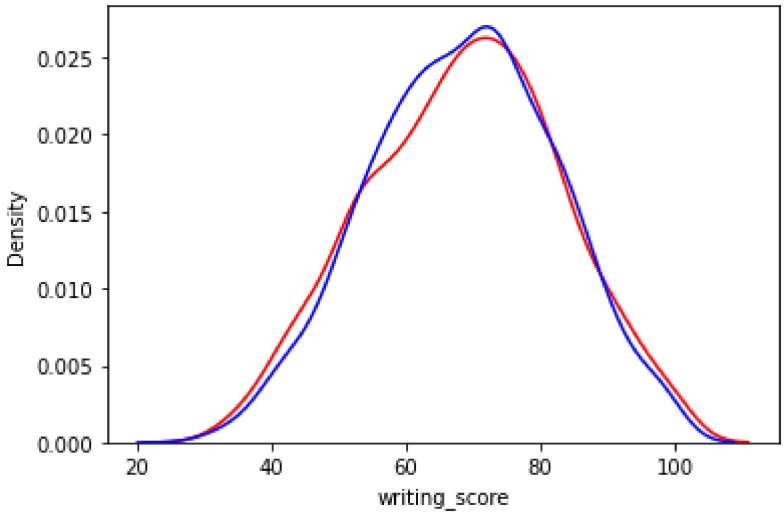
```
writing_score =  reading_score * [0.98510279] + -0.0363707002418181
```

In [289…

```
# Finally lets draw a distribution plot in order to compare the actual writing score w
ax1 = sns.distplot(df['writing_score'], hist = False, color = 'r', label = "Actual Wri
sns.distplot(Yhat, hist= False, color = 'b', label = "Fitted Value", ax = ax1)
```

```
C:\Users\tural\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `kdeplot` (an axes-level function for kernel density plots).
  warnings.warn(msg, FutureWarning)
C:\Users\tural\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `kdeplot` (an axes-level function for kernel density plots).
  warnings.warn(msg, FutureWarning)
```

Out[289]: <AxesSubplot:xlabel='writing_score', ylabel='Density'>



As we can see our model is good enough to take it into consideration to predict the writing score