Exploratory Data Analysis on Student Performance in Exams

Data Analysis with Python: Zero to Pandas - Course Project Guidelines

(remove this cell before submission)

Important links:

- Make submissions here: https://jovian.ml/learn/data-analysis-with-python-zero-to-pandas/assignment/course-project
- Ask questions here: https://jovian.ml/forum/t/course-project-on-exploratory-data-analysis-discuss-and-share-your-work/11684
- Find interesting datasets here: https://jovian.ml/forum/t/recommended-datasets-for-course-project/11711

This is the starter notebook for the course project for Data Analysis with Python: Zero to Pandas. You will pick a real-world dataset of your choice and apply the concepts learned in this course to perform exploratory data analysis. Use this starter notebook as an outline for your project . Focus on documentation and presentation - this Jupyter notebook will also serve as a project report, so make sure to include detailed explanations wherever possible using Markdown cells.

Evaluation Criteria

Your submission will be evaluated using the following criteria:

- Dataset must contain at least 3 columns and 150 rows of data
- You must ask and answer at least 4 questions about the dataset
- Your submission must include at least 4 visualizations (graphs)
- Your submission must include explanations using markdown cells, apart from the code.

- Your work must not be plagiarized i.e. copy-pasted for somewhere else.

Follow this step-by-step guide to work on your project.

Step 1: Select a real-world dataset

- Find an interesting dataset on this page: https://www.kaggle.com/datasets?fileType=csv
- The data should be in CSV format, and should contain at least 3 columns and 150 rows
- Download the dataset using the `opendatasets` Python library

Here's some sample code for downloading the US Elections Dataset:

```
import opendatasets as od
dataset_url = 'https://www.kaggle.com/tunguz/us-elections-dataset'
od.download('https://www.kaggle.com/tunguz/us-elections-dataset')
```

You can find a list of recommended datasets here: https://jovian.ml/forum/t/recommended-datasets-for-course-project/11711

Step 2: Perform data preparation & cleaning

- Load the dataset into a data frame using Pandas
- Explore the number of rows & columns, ranges of values etc.
- Handle missing, incorrect and invalid data
- Perform any additional steps (parsing dates, creating additional columns, merging multiple dataset etc.)

Step 3: Perform exploratory analysis & visualization

- Compute the mean, sum, range and other interesting statistics for numeric columns
- Explore distributions of numeric columns using histograms etc.
- Explore relationship between columns using scatter plots, bar charts etc.
- Make a note of interesting insights from the exploratory analysis

Step 4: Ask & answer questions about the data

- Ask at least 4 interesting questions about your dataset
- Answer the questions either by computing the results using Numpy/Pandas or by plotting graphs using Matplotlib/Seaborn
- Create new columns, merge multiple dataset and perform grouping/aggregation wherever necessary
- Wherever you're using a library function from Pandas/Numpy/Matplotlib etc. explain briefly what it does

Step 5: Summarize your inferences & write a conclusion

- Write a summary of what you've learned from the analysis
- Include interesting insights and graphs from previous sections
- Share ideas for future work on the same topic using other relevant datasets

- Share links to resources you found useful during your analysis

Step 6: Make a submission & share your work

- Upload your notebook to your Jovian.ml profile using `jovian.commit .`
- **Make a submission here**: https://jovian.ml/learn/data-analysis-with-python-zero-to-pandas/assignment/course-project
- Share your work on the forum: https://jovian.ml/forum/t/course-project-on-exploratory-data-analysis-discuss-and-share-your-work/11684
- Browse through projects shared by other participants and give feedback

(Optional) Step 7: Write a blog post

- A blog post is a great way to present and showcase your work.
- Sign up on Medium.com to write a blog post for your project.
- Copy over the explanations from your Jupyter notebook into your blog post, and embed code cells & outputs
- Check out the Jovian.ml Medium publication for inspiration: https://medium.com/jovianml

Example Projects

Refer to these projects for inspiration:

- Analyzing StackOverflow Developer Survey Results
- Analyzing Covid-19 data using Pandas
- Analyzing your browser history using Pandas & Seaborn by Kartik Godawat
- WhatsApp Chat Data Analysis by Prajwal Prashanth
- Understanding the Gender Divide in Data Science Roles by Aakanksha N S
- 2019 State of Javscript Survey Results
- 2020 Stack Overflow Developer Survey Results

NOTE: Remove this cell containing the instructions before making your submission. You can do using the "Edit > Delete Cells" menu option.

Project Title - change this

TODO - Write some introduction about your project here: describe the dataset, where you got it from, what you're trying to do with it, and which tools & techniques you're using. You can also mention about the course Data Analysis with Python: Zero to Pandas, and what you've learned from it.

How to run the code

This is an executable *Jupyter notebook* hosted on Jovian.ml, a platform for sharing data science projects. You can run and experiment with the code in a couple of ways: *using free online resources* (recommended) or *on your own computer*.

Option 1: Running using free online resources (1-click, recommended)

The easiest way to start executing this notebook is to click the "Run" button at the top of this page, and select "Run on Binder". This will run the notebook on mybinder.org, a free online service for running Jupyter notebooks. You can also select "Run on Colab" or "Run on Kaggle".

Option 2: Running on your computer locally

1. Install Conda by following these instructions. Add Conda binaries to your system `PATH`, so you can use the `conda` command on your terminal.

2. Create a Conda environment and install the required libraries by running these commands on the terminal:

```
conda create -n zerotopandas -y python=3.8
conda activate zerotopandas
pip install jovian jupyter numpy pandas matplotlib seaborn
opendatasets --upgrade
```

1. Press the "Clone" button above to copy the command for downloading the notebook, and run it on the terminal. This will create a new directory and download the notebook. The command will look something like this:

```
jovian clone notebook-owner/notebook-id
```

1. Enter the newly created directory using `cd directory-name` and start the Jupyter notebook.

```
jupyter notebook
```

You can now access Jupyter's web interface by clicking the link that shows up on the terminal or by visiting http://localhost:8888 on your browser. Click on the notebook file (it has a `.ipynb` extension) to open it.

Downloading the Dataset

I'll start by downloading the dataset. This dataset comes from an open-source library of datasets. I have chosen a dataset about students and their marks in tests.

In [1]:
```
!pip install jovian opendatasets --upgrade --quiet
```

Let's begin by downloading the data, and listing the files within the dataset.

In [2]:
```
import opendatasets as od
dataset_url = 'https://www.kaggle.com/spscientist/students-performance-in-exams'
od.download('https://www.kaggle.com/spscientist/students-performance-in-exams')
```

```
Skipping, found downloaded files in "./students-performance-in-exams" (use force=Tru
e to force download)
```

The dataset has been downloaded and extracted.

In [3]:
```
data_dir = './students-performance-in-exams'
```

In [4]:
```
import os
os.listdir(data_dir)
```

Out[4]:
```
['StudentsPerformance.csv']
```

Let us save and upload our work to Jovian before continuing.

In [5]:
```
project_name = "EDA:Students Performance in Exams"
```

In [6]:
```
!pip install jovian --upgrade -q
```

In [7]:
```
import jovian
```

In [8]:
```
jovian.commit(project=project_name)
```

```
[jovian] Updating notebook "sethufleck/eda-students-performance-in-exams" on http
s://jovian.com
[jovian] Committed successfully! https://jovian.com/sethufleck/eda-students-performa
nce-in-exams
```

Out[8]:
```
'https://jovian.com/sethufleck/eda-students-performance-in-exams'
```

Data Preparation and Cleaning

I've stored the data in a pandas array, then I've run a quick analysis and I added a column for the mean of the 3 scores performed by each student. I've returned the number of students who attended and completed a test preparation course. I also used the function "describe()" to return a little statistics information. Finally, I changed the name of the columns, so that it's going to be easier for me to use them without misunderstandings by Python.

In [9]:
```
import pandas as pd
exams_df=pd.read_csv(data_dir+'/StudentsPerformance.csv')
```

In [10]:
```
exams_df
```

Out[10]:

	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

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
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
...	...	...	...	...	...	...	...	...
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

1000 rows × 8 columns

In [11]:
```
#To check for missing values in a dataframe exams_df, we can use the isnull() functi
print(exams_df.isnull().sum())
```

```
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 [12]:
```
# The drop_duplicates() method in Pandas is used to remove duplicate rows from a dat
exams_df.drop_duplicates(inplace=True)
```

In [13]:
```
# summary statistics for each numerical column in a dataframe exams_df, we can use t
exams_df.describe()
```

Out[13]:

	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

I plan to modify the column names in order to eliminate any spaces that might be present.

In [14]:
```
exams_df.columns=['gender', 'race_ethnicity', 'parental_level_of_education', 'lunch'
```

In [15]:
```
exams_df
```

Out[15]:

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_
0	female	group B	bachelor's degree	standard	none	
1	female	group C	some college	standard	completed	
2	female	group B	master's degree	standard	none	
3	male	group A	associate's degree	free/reduced	none	
4	male	group C	some college	standard	none	
...	...	...	...	...	...	
995	female	group E	master's degree	standard	completed	
996	male	group C	high school	free/reduced	none	
997	female	group C	high school	free/reduced	completed	
998	female	group D	some college	standard	completed	
999	female	group D	some college	free/reduced	none	

1000 rows × 8 columns

In [16]:
```
exams_df['sum_of_score'] = exams_df[['math_score', 'reading_score', 'writing_score']
```

In [17]:
```
exams_df['mean_of_score'] = exams_df[['math_score', 'reading_score', 'writing_score'
```

In [18]:
```
exams_df
```

Out[18]:

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_
0	female	group B	bachelor's degree	standard	none	
1	female	group C	some college	standard	completed	
2	female	group B	master's degree	standard	none	
3	male	group A	associate's degree	free/reduced	none	
4	male	group C	some college	standard	none	
...	...	...	...	...	...	
995	female	group E	master's degree	standard	completed	
996	male	group C	high school	free/reduced	none	
997	female	group C	high school	free/reduced	completed	
998	female	group D	some college	standard	completed	

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_s
999	female	group D	some college	free/reduced	none	

1000 rows × 10 columns

At first let us find the gender difference and there mean of scores from DataFrame exams_df

In [19]:
```
# group data by gender and calculate the mean of scores for each group and count of
gender_mean = exams_df.groupby('gender')['mean_of_score'].mean()

# get the count of each gender
gender_count = exams_df['gender'].value_counts()

# concatenate mean scores and group counts
result = pd.concat([gender_mean, gender_count], axis=1)
result.columns = ['mean_of_score', 'count']

# print the result
print(result)
```

```
        mean_of_score  count
female      69.569498    518
male        65.837483    482
```

The students who have completed a test preparation course from DataFrame called exams_df.

The loop iterates through the test_preparation_course column of the exams_df dataframe, and checks if the value of each element is equal to 'completed'

In [20]:
```
prepared_students = {'male': 0, 'female': 0}
for index, row in exams_df.iterrows():
    if row['test_preparation_course'] == 'completed':
        gender = row['gender']
        prepared_students[gender] += 1

print('{} students have attended a preparation course: {} males, {} females'.format(

#  It assumes that the test_preparation_course column only contains two possible val
```

```
358 students have attended a preparation course: 174 males, 184 females
```

In [21]:
```
import jovian
```

In [22]:
```
jovian.commit()
```

```
[jovian] Updating notebook "sethufleck/eda-students-performance-in-exams" on http
s://jovian.com
[jovian] Committed successfully! https://jovian.com/sethufleck/eda-students-performa
nce-in-exams
```

Out[22]:
```
'https://jovian.com/sethufleck/eda-students-performance-in-exams'
```

Exploratory Analysis and Visualization

I used the matplotlib and seaborn libraries for plotting some graphs that will help us to better understand the correlation between different types of data. We can't be sure our conclusions are completely correct, but graphs and ulterior calculations can help us making hypothesis.

Let's begin by importing `matplotlib.pyplot` and `seaborn` .

In [23]:
```
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.cluster import KMeans
%matplotlib inline

sns.set_style('darkgrid')
sns.set_palette('bright')
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['figure.figsize'] = (9, 5)
matplotlib.rcParams['figure.facecolor'] = 'white'
```

In [24]:
```
exams_df
```

Out[24]:

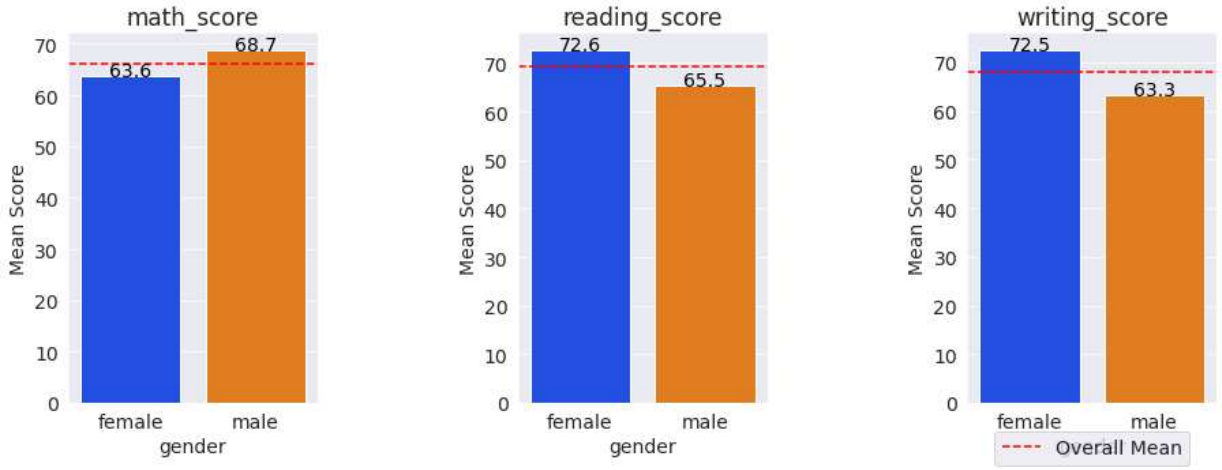
	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_s
0	female	group B	bachelor's degree	standard	none	
1	female	group C	some college	standard	completed	
2	female	group B	master's degree	standard	none	
3	male	group A	associate's degree	free/reduced	none	
4	male	group C	some college	standard	none	
...	...	...	...	...	...	
995	female	group E	master's degree	standard	completed	
996	male	group C	high school	free/reduced	none	
997	female	group C	high school	free/reduced	completed	
998	female	group D	some college	standard	completed	
999	female	group D	some college	free/reduced	none	

1000 rows × 10 columns

let us see the distribution of mean math_score, reading_score, and writing_score test results,is there any difference.

In [25]:
```
fig, ax = plt.subplots()
fig.subplots_adjust(hspace=0.8, wspace=0.8, left=0.2, right=1.5)

for idx in range(3):
    plt.subplot(1,3, idx+1)
    gender_df = exams_df.groupby("gender")[list(exams_df.columns[-5:])[idx]].describ
    ax = sns.barplot(x=gender_df.index, y=gender_df.loc[:,"mean"].values, data=gende
    plt.ylabel("Mean Score")
```

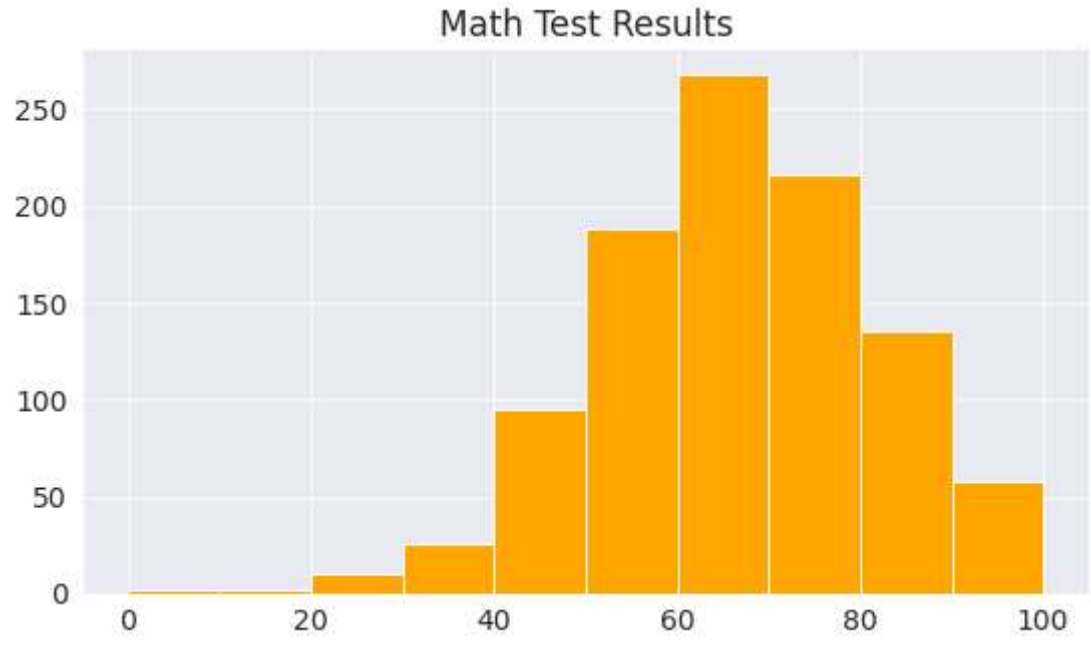
```
    plt.title(f"{list(exams_df.columns[-5:])[idx]}")

    # Add label values to the bars
    for i, v in enumerate(gender_df.loc[:,"mean"].values):
        ax.text(i, v, "{:.1f}".format(v), color='black', ha="center")

    # Add overall mean
    overall_mean = exams_df[list(exams_df.columns[-5:])[idx]].mean()
    plt.axhline(y=overall_mean, color='red', linestyle='--', label='Overall Mean')

ax.legend(loc='upper center', bbox_to_anchor=(0.5, -0.05), ncol=2)
plt.show()
```



It appears that males perform better in mathematics, but exhibit poorer performance in reading and writing when comparing with female students

Let's plot the histogram for math test results see there distribution bins make any significance

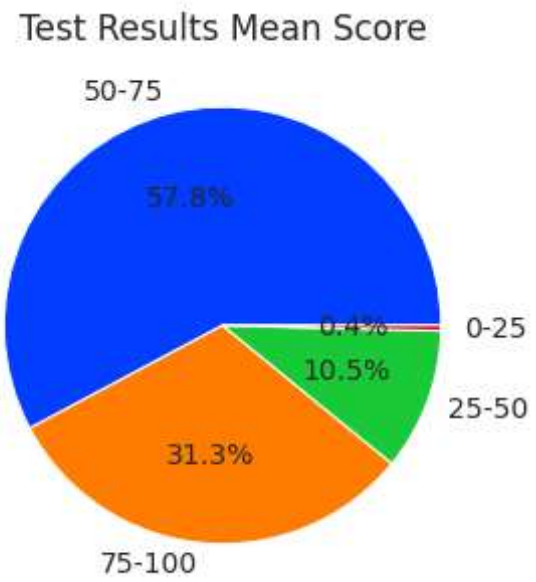
In [26]:
```
plt.title('Math Test Results')
plt.hist(exams_df.math_score, bins=10,color='orange');
```



The histograms seem to follow a normal distribution trend

Major score scored by students ?

In [27]:
```
#Pie Chart for mean_of_score results and group them as 4 bins
import matplotlib.pyplot as plt

# group scores into 4 bins
bins = [0, 25, 50, 75, 100]
labels = ['0-25', '25-50', '50-75', '75-100']
exams_df['score_range'] = pd.cut(exams_df['mean_of_score'], bins=bins, labels=labels

# count the number of scores in each range
scores_by_range = exams_df['score_range'].value_counts()

# create pie chart
plt.title('Test Results Mean Score')
plt.pie(scores_by_range, labels=scores_by_range.index, autopct='%1.1f%%')
plt.show()
```



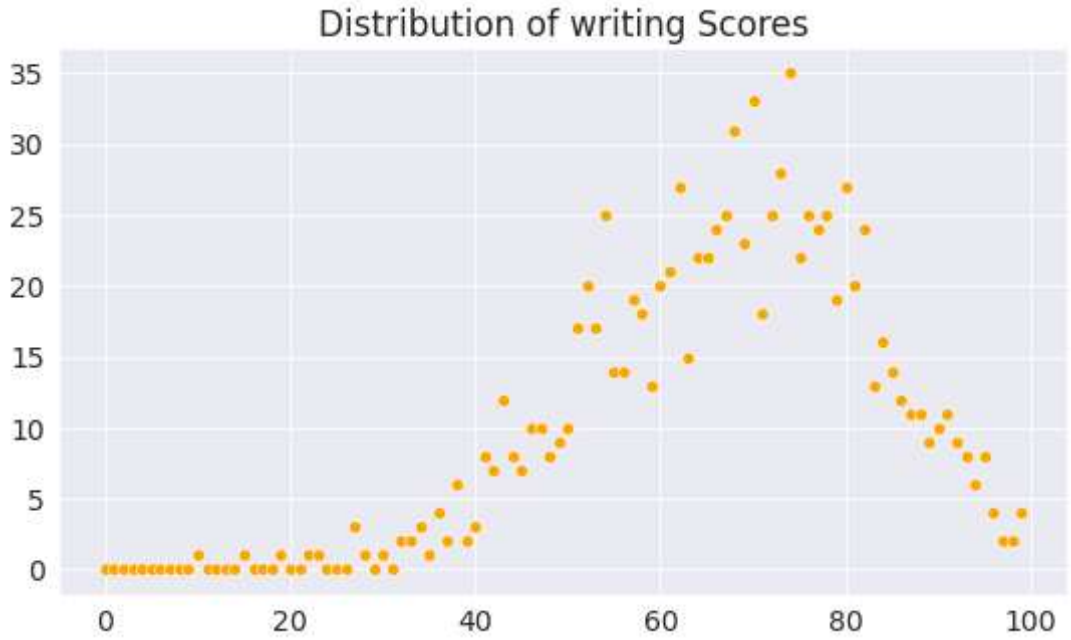
From graph it clear that the mean_score 50-75 bins shares 57.8%

lets checks the distribution of writing_score

Scatter plot to see exactly how the students performed in writing_score. I used a nested for and an if for storing the data in two arrays.

In [28]:
```
import numpy as np
import seaborn as sns

results_array_writing = np.zeros(100)
for i in range(0, 100):
    for j in range(0, 1000):
        if i == exams_df.writing_score[j]:
            results_array_writing[i] += 1

results_array_writing1 = np.arange(100)
sns.scatterplot(x=results_array_writing1, y=results_array_writing, color='orange')
plt.title('Distribution of writing Scores')
```

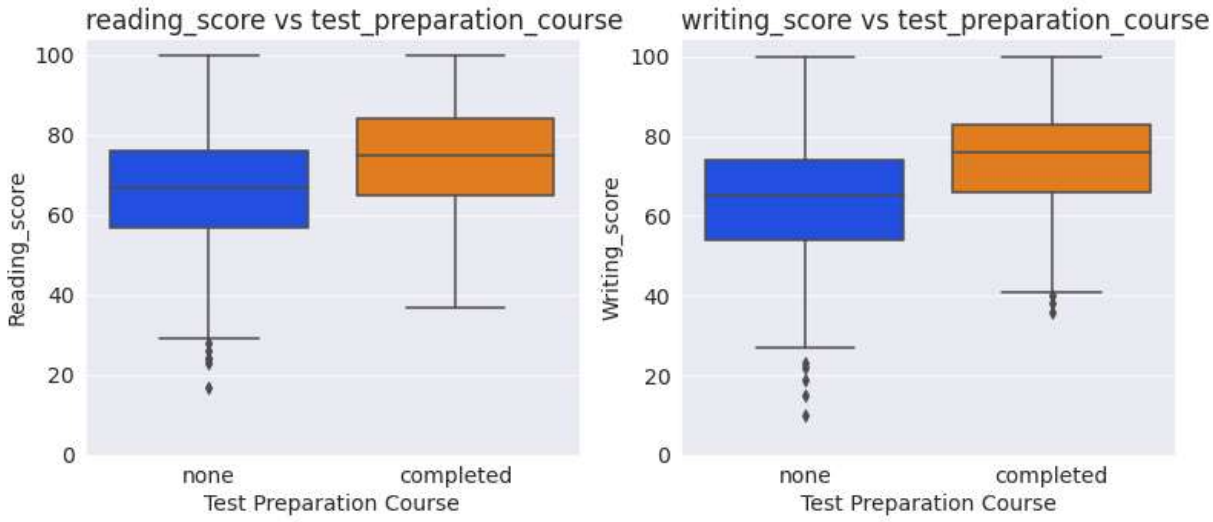
Out[28]:
```
Text(0.5, 1.0, 'Distribution of writing Scores')
```



Its seems to be normal distribution

Is the test preparation course helpful made any impact in the results

In [29]:
```
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(11, 5))

for i, item in enumerate(exams_df.columns[-5:-3]):
    ax = axes[i]
    sns.boxplot(x=exams_df["test_preparation_course"], y=exams_df[item], ax=ax)
    ax.set_title(item+" vs test_preparation_course", loc="left")
    ax.set_xlabel("Test Preparation Course")
    ax.set_ylabel(item.capitalize())
    ax.set_ylim(bottom=0)

plt.tight_layout()
plt.show()
```



If students complete the test preparation course before taking the exam, we observe a narrower distribution of scores and a higher average score.

Let us save and upload our work to Jovian before continuing

In [30]:
```
import jovian
```

In [31]:
```
jovian.commit()
```

```
[jovian] Updating notebook "sethufleck/eda-students-performance-in-exams" on http
s://jovian.com
[jovian] Committed successfully! https://jovian.com/sethufleck/eda-students-performa
nce-in-exams
```

Out[31]: 'https://jovian.com/sethufleck/eda-students-performance-in-exams'

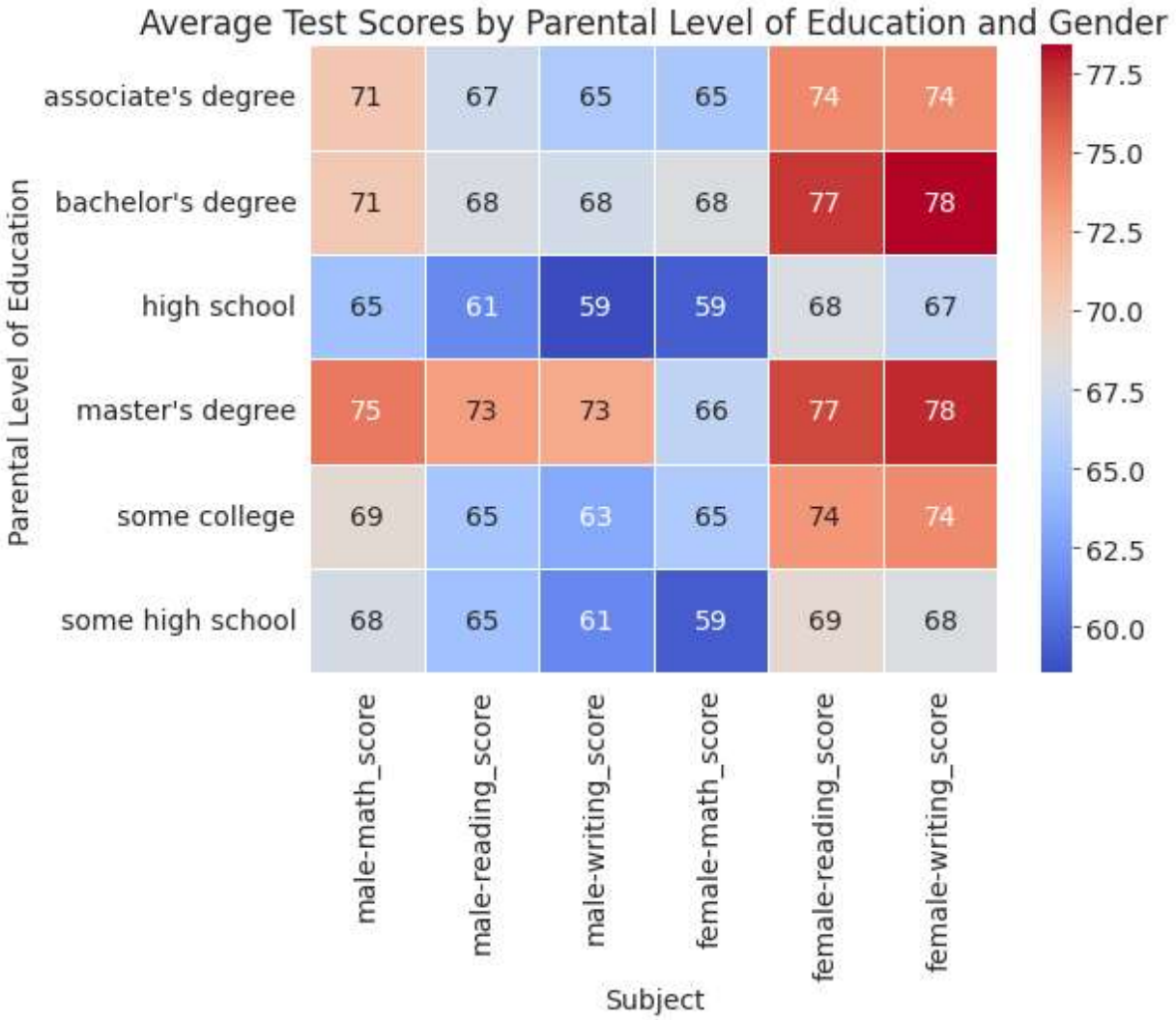
Asking and Answering Questions

I formulate five assumptions and endeavor to provide an appropriate response by analyzing the data through graphical representations

Q1: Does the parent level of education has any impact in the student score ?

Heat map that visualizes the average test scores for each subject by parental level of education and gender.

In [32]:
```
# Filter the dataset by gender
male_df = exams_df[exams_df["gender"] == "male"]
female_df = exams_df[exams_df["gender"] == "female"]

# Calculate the average test scores by parental level of education for male and fema
male_scores_by_parent_edu = male_df.groupby("parental_level_of_education")[["math_sc
female_scores_by_parent_edu = female_df.groupby("parental_level_of_education")[["mat

# Combine the scores for male and female students into a single dataframe
scores_by_parent_edu = pd.concat([male_scores_by_parent_edu, female_scores_by_parent
scores_by_parent_edu.columns = pd.MultiIndex.from_product([['male', 'female'], ['mat

# Create a heat map to visualize the results
fig, ax = plt.subplots(figsize=(8,6))
sns.heatmap(scores_by_parent_edu, annot=True, cmap='coolwarm', linewidths=.5, ax=ax)
plt.title("Average Test Scores by Parental Level of Education and Gender")
plt.xlabel("Subject")
plt.ylabel("Parental Level of Education")

# Display the plot
plt.show()
```

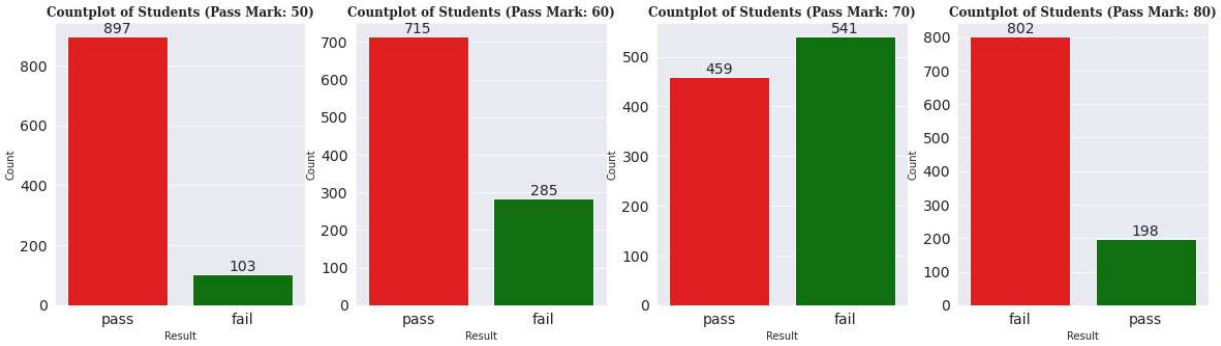


As the general trend remains almost consistent, it is possible to deduce a substantial correlation between gender and the parental level education where Degree parental student have scored higher marks.

Q2.What pass count of student if the pass mark ?

In [33]:

```
#pass_marks = [50, 60, 70, 80]
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

pass_marks = [50, 60, 70, 80]

fig, axs = plt.subplots(1, 4, figsize=(20, 5))

for i, mark in enumerate(pass_marks):
    pass_df = exams_df[(exams_df[['math_score', 'reading_score', 'writing_score']].m
    fail_df = exams_df[(exams_df[['math_score', 'reading_score', 'writing_score']].m
    ax = axs[i]
    sns.countplot(x=np.where(exams_df[['math_score', 'reading_score', 'writing_score
    ax.set_title(f"Countplot of Students (Pass Mark: {mark})", fontfamily='serif', f
    ax.set_xlabel("Result", fontsize=10)
    ax.set_ylabel("Count", fontsize=10)
    for p in ax.patches:
        ax.text(p.get_x() + p.get_width()/2, p.get_height(), '{:.0f}'.format(p.get_h

plt.show()
```



Q3: lets check the Correlation be established between the results obtained and the level of education of the parents?

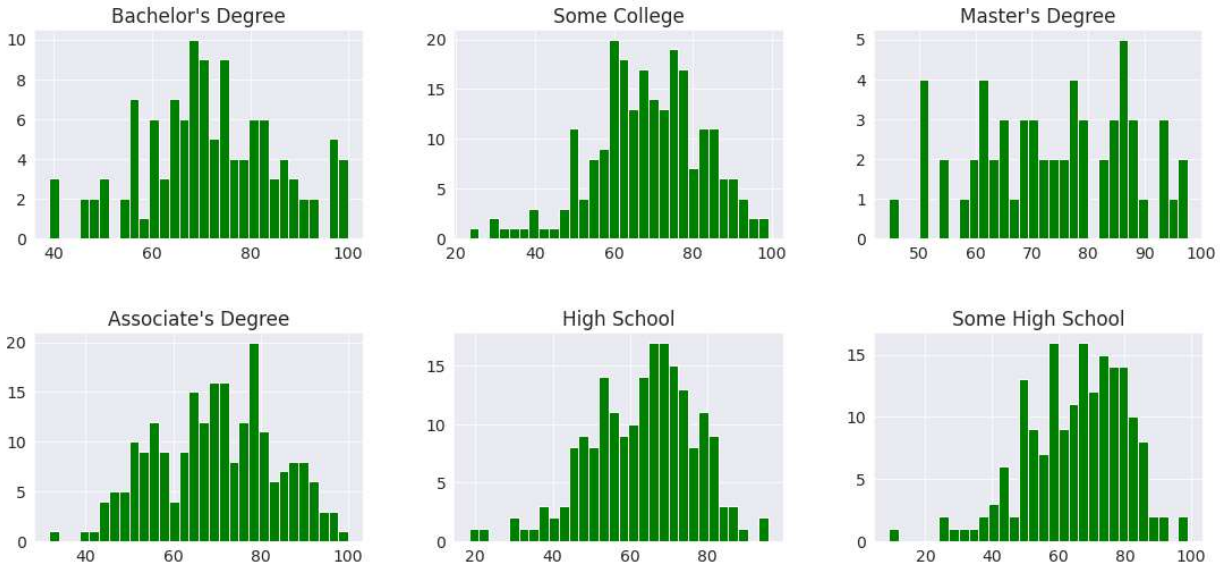
To compare the overall trend across different parental levels of education, I will create six graphs, one for each level, using the subplot function to reduce the space needed for the visuals.

In [34]:
```
exams_df.parental_level_of_education.unique()
```

Out[34]:
```
array(["bachelor's degree", 'some college', "master's degree",
       "associate's degree", 'high school', 'some high school'],
      dtype=object)
```

In [35]:
```
results_parent1=exams_df[exams_df.parental_level_of_education=="bachelor's degree"]
results_parent2=exams_df[exams_df.parental_level_of_education=="some college"]
results_parent3=exams_df[exams_df.parental_level_of_education=="master's degree"]
results_parent4=exams_df[exams_df.parental_level_of_education=="associate's degree"]
results_parent5=exams_df[exams_df.parental_level_of_education=="high school"]
results_parent6=exams_df[exams_df.parental_level_of_education=="some high school"]
```

In [36]:
```
fig, axes = plt.subplots(2, 3, figsize=(16, 8))
axes[0,0].set_title("Bachelor's Degree")
axes[0,0].hist(results_parent1.mean_of_score, bins=30,color='green');
axes[0,1].set_title("Some College")
axes[0,1].hist(results_parent2.mean_of_score, bins=30,color='green');
axes[0,2].set_title("Master's Degree")
axes[0,2].hist(results_parent3.mean_of_score, bins=30,color='green');
axes[1,0].set_title("Associate's Degree")
axes[1,0].hist(results_parent4.mean_of_score, bins=30,color='green');
axes[1,1].set_title("High School")
axes[1,1].hist(results_parent5.mean_of_score, bins=30,color='green');
axes[1,2].set_title("Some High School")
axes[1,2].hist(results_parent6.mean_of_score, bins=30,color='green');
plt.tight_layout(pad=3)
```



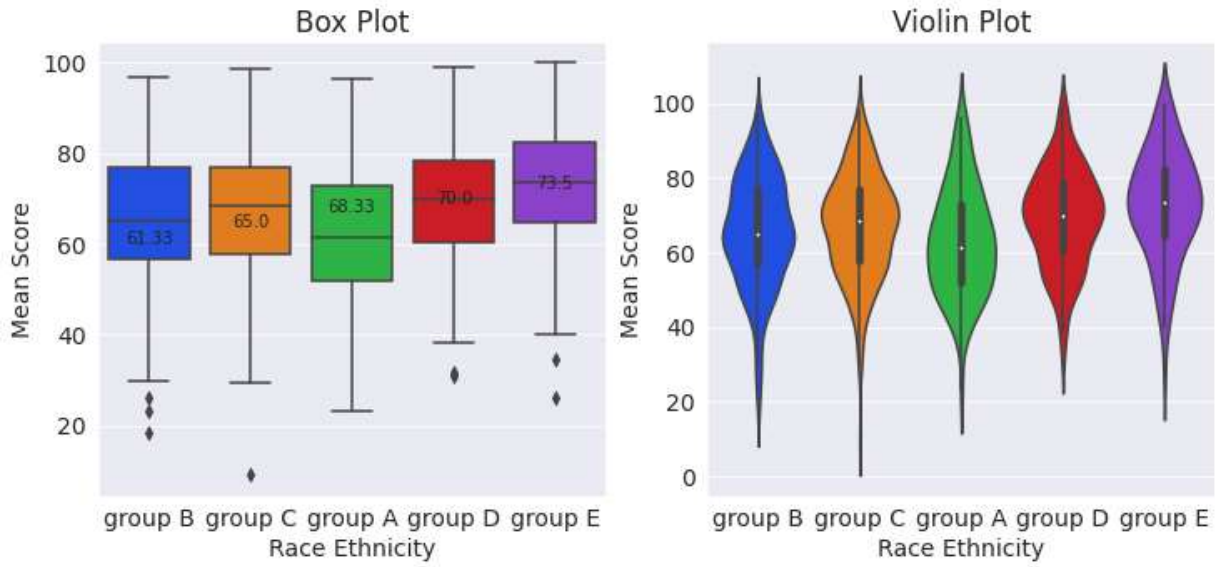
The disparity between these trends is not significant, as some of them exhibit minor dissimilarities, which are insufficient to infer a strong correlation between the data.

Q4: Is there any difference in scores between different race ethnicities of students and their results?

The trend across different race ethnicity of students and their results, I will create box plot and viol plot

In [37]:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Create a figure with two subplots
fig, axs = plt.subplots(ncols=2, figsize=(12,5))

# Create the box plot in the first subplot
sns.boxplot(x='race_ethnicity', y='mean_of_score', data=exams_df, ax=axs[0])
axs[0].set_title('Box Plot')
axs[0].set_xlabel('Race Ethnicity')
axs[0].set_ylabel('Mean Score')

# Add data labels to the box plot
for i, box in enumerate(axs[0].artists):
    # Get the median value of each box
    median = exams_df.groupby('race_ethnicity')['mean_of_score'].median().values[i]
    # Add the median value as a label
    axs[0].text(i, median, round(median,2), horizontalalignment='center', verticalal

# Create the violin plot in the second subplot
sns.violinplot(x='race_ethnicity', y='mean_of_score', data=exams_df, ax=axs[1])
axs[1].set_title('Violin Plot')
axs[1].set_xlabel('Race Ethnicity')
axs[1].set_ylabel('Mean Score')

# Add data labels to the violin plot
for i, violin in enumerate(axs[1].artists):
    # Get the median value of each violin
    median = exams_df.groupby('race_ethnicity')['mean_of_score'].median().values[i]
    # Add the median value as a label
    axs[1].text(i, median, round(median,2), horizontalalignment='center', verticalal

# Show the plot
plt.show()
```



it appears that Groups E, D, and A achieve better results, but this is merely an impression.

Q5: lets check the 50 th percentile Score and distribution of datasets?

In [38]:
```
exams_df.mean_of_score.quantile(0.5)
```

Out[38]:
```
68.33333333333333
```

In [39]:
```
exams_df.query("mean_of_score <= 68.33")
```

Out[39]:

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_s
3	male	group A	associate's degree	free/reduced	none	
7	male	group B	some college	free/reduced	none	
8	male	group D	high school	free/reduced	completed	
9	female	group B	high school	free/reduced	none	
10	male	group C	associate's degree	standard	none	
...	...	...	...	...	...	
986	female	group C	associate's degree	standard	none	
988	female	group A	some high school	free/reduced	none	
994	male	group A	high school	standard	none	
996	male	group C	high school	free/reduced	none	
997	female	group C	high school	free/reduced	completed	

489 rows × 11 columns

In [40]:
```
# Divide the data by the 80th percentile of the data from the TotalCharges variable
score_under50 = exams_df.query("mean_of_score <=68.33")
score_above50 = exams_df.query("mean_of_score > 68.33")
```

In [41]:
```
# Show the distribution of people under the 50th percentile
score_under50.mean_of_score.describe()
```

Out[41]:
```
count    489.000000
mean      56.137696
std        9.377634
min        9.000000
25%       51.000000
50%       57.666667
75%       63.666667
max       68.000000
Name: mean_of_score, dtype: float64
```

In [42]:
```
# Show the distribution of people above the 50th percentile
score_above50.mean_of_score.describe()
```

Out[42]:
```
count    511.000000
mean      78.902805
std        7.752736
min       68.333333
25%       72.666667
50%       77.333333
75%       84.000000
max      100.000000
Name: mean_of_score, dtype: float64
```

The mean_of_score the 50% of score is 68.33.

Q6: whether it's true that students who receive reduced or free lunch cannot afford a preparation course?

The number of students who had a standard lunch and attended the preparation course, as well as the number of students who had a reduced/free lunch and attended the preparation course. To do this, I will use two for loops and two if statements.

In [43]:
```
exams_df
```

Out[43]:

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_s
0	female	group B	bachelor's degree	standard	none	
1	female	group C	some college	standard	completed	
2	female	group B	master's degree	standard	none	
3	male	group A	associate's degree	free/reduced	none	
4	male	group C	some college	standard	none	
...	...	...	...	...	...	
995	female	group E	master's degree	standard	completed	
996	male	group C	high school	free/reduced	none	
997	female	group C	high school	free/reduced	completed	
998	female	group D	some college	standard	completed	
999	female	group D	some college	free/reduced	none	

1000 rows × 11 columns

In [44]:
```
standard_lunch=exams_df[exams_df.lunch=='standard']
free_lunch=exams_df[exams_df.lunch=='free/reduced']
test_prep_standard=0
test_prep_free=0
for item in standard_lunch.test_preparation_course:
    if item=='completed':
        test_prep_standard+=1
for item in free_lunch.test_preparation_course:
    if item=='completed':
        test_prep_free+=1
```

In [45]:
```
print('The number of students who took the preparation test and had a standard lunch
```

```
The number of students who took the preparation test and had a standard lunch was 22
7, while the number of students who took the test and had a free/reduced lunch was 1
31.
```

Given the notable difference observed across the cases we're analyzing, we can reasonably infer that our hypothesis was correct.

In [46]:
```
import jovian
```

In [47]:
```
jovian.commit()
```

```
[jovian] Updating notebook "sethufleck/eda-students-performance-in-exams" on http
s://jovian.com
[jovian] Committed successfully! https://jovian.com/sethufleck/eda-students-performa
nce-in-exams
```

Out[47]:
```
'https://jovian.com/sethufleck/eda-students-performance-in-exams'
```

Inferences and Conclusion

After downloading and addressing some formal issues with the dataset, I conducted an initial examination of the tabular data.

1. From the findings, I suggest that there is no discernible association between gender and academic performance.
2. Although parental education level may have some impact on students' performance, it is not a crucial factor.
3. Taking a preparation course can be advantageous for students.
4. Eating lunch is crucial for students' performance and appears to be the most significant factor.
5. Finding raises the possibility that the cost of the test was prohibitive for students who received reduced-price or free meals.

To achieve good academic performance, students should ensure they are well-nourished and put effort into test preparation.However, this preliminary analysis is not comprehensive enough to yield definitive conclusions

In [48]:
```
import jovian
```

In [49]:
```
jovian.commit()
```

```
[jovian] Updating notebook "sethufleck/eda-students-performance-in-exams" on http
s://jovian.com
[jovian] Committed successfully! https://jovian.com/sethufleck/eda-students-performa
nce-in-exams
```

Out[49]:
```
'https://jovian.com/sethufleck/eda-students-performance-in-exams'
```

References and Future Work

TODO - During my analysis, I found the documentation for Python, Pandas, and NumPy to be extremely helpful. I consulted numerous functions and anticipate continuing to do so until I feel more comfortable. Although this initial project was straightforward and cursory, I am optimistic that as I acquire more expertise, I will be able to gain a deeper understanding of the underlying characteristics of this dataset.

In [50]:
```
import jovian
```

In [51]:
```
jovian.commit()
```

```
[jovian] Updating notebook "sethufleck/eda-students-performance-in-exams" on http
s://jovian.com
[jovian] Committed successfully! https://jovian.com/sethufleck/eda-students-performa
nce-in-exams
```

Out[51]:
```
'https://jovian.com/sethufleck/eda-students-performance-in-exams'
```

In [ ]:
```
jovian.submit(assignment="zero-to-pandas-project")
```