```
#Importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

```
#Importing_CSV
Academic_Performance = pd.read_csv('/content/Academic Performance of Students.csv')
```

```
#Displaying data set
Academic_Performance.head(-1)
```

```
   Student_ID    Name  Age  Gender  Study_Hours  Attendance_Percentage  Midterm_Score  Final_Score  Extracurricular_Activities  Parental_
0         301   Alice   18  Female           15                     95             85           88                         Yes
1         302     Bob   19    Male           10                     80             78           82                          No
2         303 Charlie   17    Male           12                     90             80           84                         Yes
3         304   Daisy   18  Female           20                     98             92           95                         Yes
4         305   Ethan   20    Male            8                     75             70           74                          No
5         306   Fiona   17  Female           18                     92             89           93                         Yes
6         307  George   19    Male            7                     65             68           70                          No
7         308  Hannah   18  Female           14                     88             84           87                         Yes
8         309   Isaac   17    Male           16                     94             87           90                         Yes
```

Next steps: Generate code with Academic_Performance | View recommended plots | New interactive sheet

```
#Function assigning Grades
def assign_grade(score):
  if score >=90:
    return 'A'
  elif score >=80:
    return 'B'
  else:
    return 'C'


#Assigning grades as per function
Academic_Performance['Grade']=Academic_Performance['Final_Score'].apply(assign_grade)
```

```
Academic_Performance.head()
```

```
   Student_ID     Name  Age  Gender  Study_Hours  Attendance_Percentage  Midterm_Score  Final_Score  Extracurricular_Activities  Parental_
0         301    Alice   18  Female           15                     95             85           88                         Yes
1         302      Bob   19    Male           10                     80             78           82                          No
2         303  Charlie   17    Male           12                     90             80           84                         Yes
3         304    Daisy   18  Female           20                     98             92           95                         Yes
4         305    Ethan   20    Male            8                     75             70           74                          No
```

Next steps: Generate code with Academic_Performance | View recommended plots | New interactive sheet

```
#Filtered Students who perticipated in Extracurricular_Activities
Academic_Performance.loc[Academic_Performance['Extracurricular_Activities']=='Yes']
```

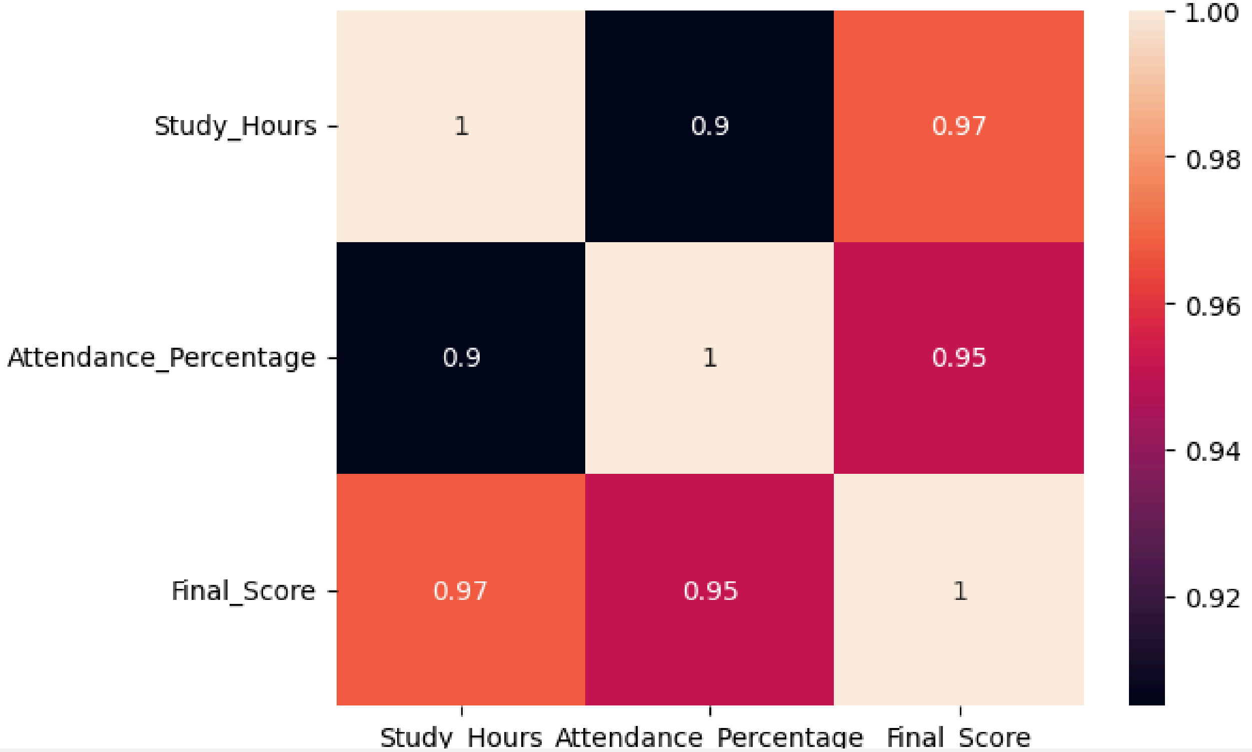
	Student_ID	Name	Age	Gender	Study_Hours	Attendance_Percentage	Midterm_Score	Final_Score	Extracurricular_Activities	Parental_
0	301	Alice	18	Female	15	95	85	88	Yes	
2	303	Charlie	17	Male	12	90	80	84	Yes	
3	304	Daisy	18	Female	20	98	92	95	Yes	
5	306	Fiona	17	Female	18	92	89	93	Yes	
7	308	Hannah	18	Female	14	88	84	87	Yes	
8	309	Isaac	17	Male	16	94	87	90	Yes	

```
#Group by Parental_Education_Level & Final_Score
FinalScore=Academic_Performance.groupby('Parental_Education_Level')['Final_Score'].mean()
print(FinalScore)
```

```
Parental_Education_Level
Bachelor's     90.00
High School    77.25
Master's       88.50
Name: Final_Score, dtype: float64
```

```
#Corelation between Study_Hours,Attendance_Percentage,Final_Score
Corelation =Academic_Performance [['Study_Hours','Attendance_Percentage','Final_Score']].corr()
print(Corelation)
#Plotting heatmap
sns.heatmap(Corelation, annot=True)
plt.show()
```

```
                         Study_Hours  Attendance_Percentage  Final_Score
Study_Hours                 1.000000               0.904959     0.967760
Attendance_Percentage       0.904959               1.000000     0.951344
Final_Score                 0.967760               0.951344     1.000000
```



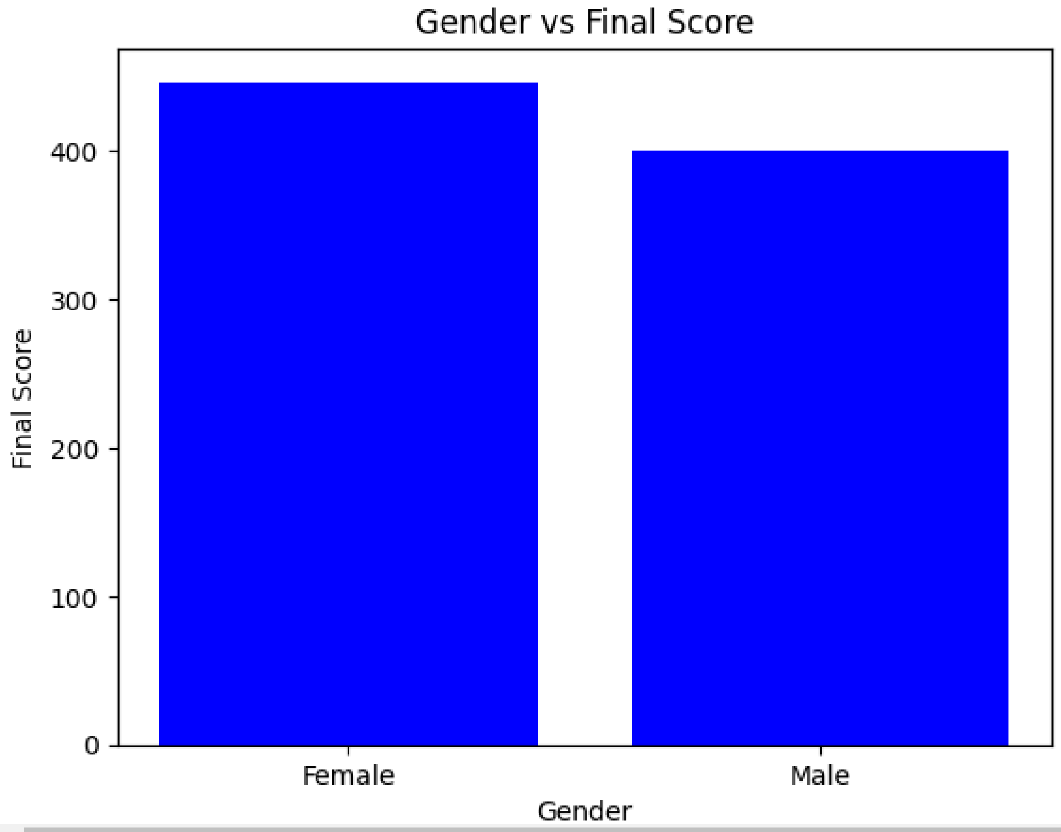
Double-click (or enter) to edit

```
#Corelation between SAT_Score,Attendance_Percentage

Sat_att=Academic_Performance[['SAT_Score','Attendance_Percentage']].corr()
print(Sat_att)
```

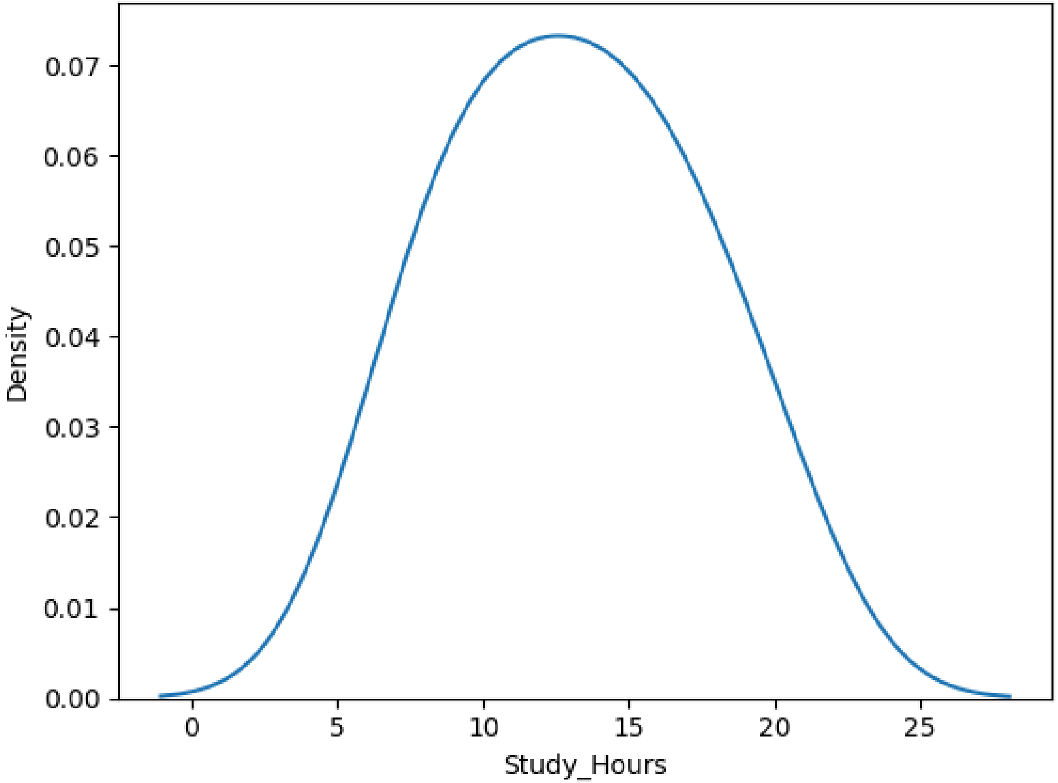
```
                       SAT_Score  Attendance_Percentage
SAT_Score                1.00000                0.96273
Attendance_Percentage    0.96273                1.00000
```

```
#Gender wise comparison of final score & bar chart

GENDER_SCORE=Academic_Performance.groupby('Gender')['Final_Score'].sum()
print(GENDER_SCORE)
plt.bar(GENDER_SCORE.index,GENDER_SCORE.values, color='blue',)
plt.xlabel('Gender')
plt.ylabel('Final Score')
plt.title('Gender vs Final Score')
plt.show()
```

```
Gender
Female    446
Male      400
Name: Final_Score, dtype: int64
```



```
#Visulization of study_hours using density and histrogram

sns.kdeplot(Academic_Performance['Study_Hours'])
plt.show()
sns.distplot(Academic_Performance['Study_Hours'])
plt.show()
```



```
<ipython-input-30-e0dc6b880851>:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.
```

Start coding or generate with AI.

```
For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
```