Experiment Number 3

Name ::	Rishabh Anand	UID ::	19BCS4525
Branch ::	CSE - IoT	Sec/Grp ::	1/A
Semester ::	6th	Date ::	5th Mar, 2022
Subject ::	ML Lab	CODE ::	CSD-386

1. Aim :

To perform the EDA analysis on the given data set to make data set ready for the further processing and training.

2. Task :

1. Conditional Constraints
2. Loops

3A. Theory :

- EDA is simply data analysis techniqes to understand the various aspects of the data. Prepration of the clean data is the main aim of EDA.
- Data should be free from Redundancies, null values , extreme values, missing values etc.
- These things in the data may effect the model training and ultimately it will effect the performance of the trained model.
- Before going on to the more complex procedures in the data processing lifecycle, it is necessary to come to a conclusion with the data or just draw some conclusive insights from the data.

Why do we need to perform Exploratory Data Analysis?

1. To Maximise the insight into dataset.
2. To understand the connection between the variables and to uncover the underlying structure
3. To extract the import Variables
4. To detect anomalies
5. To test the underlying assumptions.

3B. Algorithm :

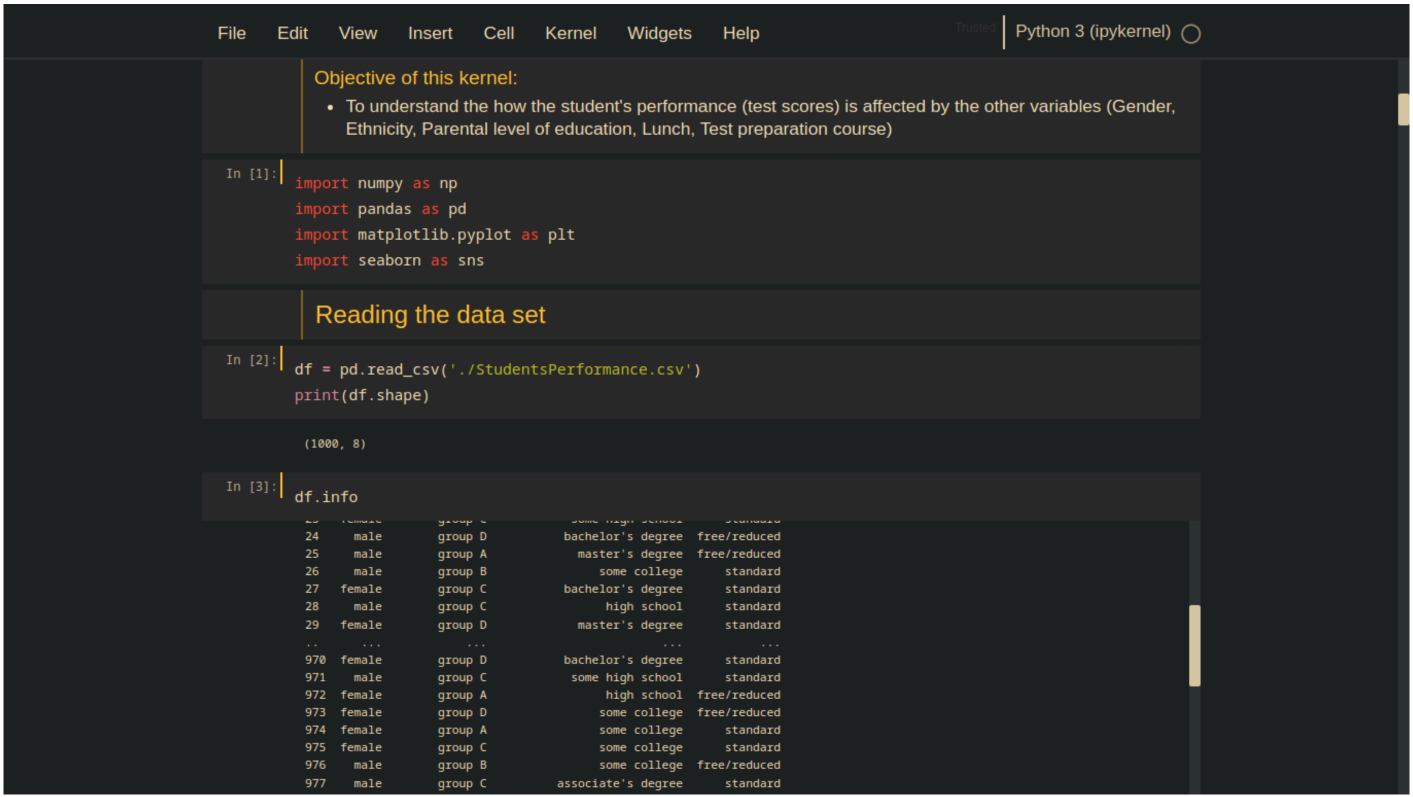
- Preview data
- Check total number of entries and column types
- Check any null values
- Check duplicate entries
- Plot distribution of numeric data (univariate and pairwise joint distribution)
- Plot count distribution of categorical data
- Analyse time series of numeric data by daily, monthly and yearly frequencies

4A. Source Code - A:

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

df = pd.read_csv(”./StudentsPerformance.csv”)
print(df.shape)
df.info
df.head()
df.tail()
df.describe
df.shape
df.isnull().sum()
plt.rcParams[”figure.figsize”] = (20, 5)
sns.countplot(df[”math score”], palette=”bright”)
plt.title(”Math Score”, fontsize=20)
plt.show()
plt.rcParams[”figure.figsize”] = (20, 5)
sns.countplot(df[”reading score”], palette=”hls”)
plt.title(”Reading Score”, fontsize=20)
plt.show()
plt.rcParams[”figure.figsize”] = (20, 5)
sns.countplot(df[”writing score”], palette=”prism”)
plt.title(”Writing Score”, fontsize=20)
plt.show()
plt.figure(figsize=(15, 5))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace
   =0.5, hspace=0.2)
plt.subplot(131)
plt.title(”Math Scores”)
sns.violinplot(y=”math score”, data=df, color=”b”, linewidth=2)
plt.subplot(132)
plt.title(”Reading Scores”)
sns.violinplot(y=”reading score”, data=df, color=”b”, linewidth=2)
plt.subplot(133)
plt.title(”Writing Scores”)
sns.violinplot(y=”writing score”, data=df, color=”b”, linewidth=2)
plt.show()
plt.figure(figsize=(20, 10))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace
   =0.5, hspace=0.2)
plt.subplot(141)
plt.title(”Gender”, fontsize=20)
df[”gender”].value_counts().plot.pie(autopct=”%1.1f%%”)
plt.subplot(142)
plt.title(”Ethinicity”, fontsize=20)
df[”race/ethnicity”].value_counts().plot.pie(autopct=”%1.1f%%”)
plt.subplot(143)
plt.title(”Lunch”, fontsize=20)
df[”lunch”].value_counts().plot.pie(autopct=”%1.1f%%”)
plt.subplot(144)
plt.title(”Parentel level of Education”, fontsize=20)
df[”parental level of education”].value_counts().plot.pie(autopct=”%1.1f
   %%”)
plt.show()
plt.figure(figsize=(15, 5))
plt.subplots_adjust(left=0.25, bottom=0.1, right=0.9, top=0.9, wspace
   =0.2, hspace=0.2)
plt.subplot(131)
plt.title(”Math Scores”)
sns.barplot(x=”gender”, y=”math score”, data=df)
plt.subplot(132)
plt.title(”Reading Scores”)
sns.barplot(x=”gender”, y=”reading score”, data=df)
plt.subplot(133)
plt.title(”Writing Scores”)
sns.barplot(x=”gender”, y=”writing score”, data=df)
plt.show()
plt.figure(figsize=(15, 5))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace
   =0.5, hspace=0.2)
plt.subplot(131)
plt.title(”Math Scores”)
sns.barplot(hue=”gender”, y=”math score”, x=”test preparation course”,
   data=df)
plt.subplot(132)
plt.title(”Reading Scores”)
sns.barplot(hue=”gender”, y=”reading score”, x=”test preparation course”,
    data=df)
plt.subplot(133)
plt.title(”Writing Scores”)
sns.barplot(hue=”gender”, y=”writing score”, x=”test preparation course”,
    data=df)
plt.show()
plt.figure(figsize=(15, 5))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace
    =0.5, hspace=0.9)
plt.subplot(131)
plt.title(”Math Scores”)
sns.barplot(x=”race/ethnicity”, y=”math score”, hue=”test preparation
    course”, data=df)
plt.subplot(132)
plt.title(”Reading Scores”)
sns.barplot(
     hue=”test preparation course”, y=”reading score”, x=”race/ethnicity”,
          data=df
)
plt.subplot(133)
plt.title(”Writing Scores”)
sns.barplot(
     hue=”test preparation course”, y=”writing score”, x=”race/ethnicity”,
          data=df
)
plt.show()
plt.figure(figsize=(30, 15))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace
    =0.5, hspace=0.2)
plt.subplot(251)
plt.title(”Test Preparation course Vs Gender”, fontsize=15)
sns.countplot(hue=”test preparation course”, x=”gender”, data=df)
plt.subplot(254)
plt.title(”Test Preparation course Vs Parental Level Of Education”,
    fontsize=15)
sns.countplot(hue=”test preparation course”, y=”parental level of
    education”, data=df)
plt.subplot(253)
plt.title(”Test Preparation course Vs Lunch”, fontsize=15)
sns.countplot(hue=”test preparation course”, x=”lunch”, data=df)
plt.subplot(252)
plt.title(”Test Preparation course Vs Ethnicity”, fontsize=15)
sns.countplot(hue=”test preparation course”, y=”race/ethnicity”, data=df)
plt.show()
plt.title(”Gender Vs Ethnicity”, fontsize=20)
sns.countplot(x=”gender”, hue=”race/ethnicity”, data=df)
plt.show()
plt.figure(figsize=(40, 10))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace
    =0.5, hspace=0.2)
plt.subplot(251)
plt.title(”Parental education and Gender”, fontsize=15)
```

```
sns.countplot(hue=”parental level of education”, x=”gender”, data=df)
plt.subplot(252)
plt.title(”Parental education and Lunch”, fontsize=15)
sns.countplot(hue=”parental level of education”, x=”lunch”, data=df)
plt.show()
plt.figure(figsize=(40, 10))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace
   =0.5, hspace=0.2)
plt.subplot(251)
plt.title(”Lunch and Gender”, fontsize=15)
sns.countplot(x=”lunch”, hue=”gender”, data=df)
plt.subplot(252)
plt.title(”Ethinicity and Lunch”, fontsize=15)
sns.countplot(x=”race/ethnicity”, hue=”lunch”, data=df)
plt.show()
df[”total marks”] = df[”math score”] + df[”reading score”] + df[”writing
   score”]
df[”percentage”] = df[”total marks”] / 300 * 100


def determine_grade(scores):
    if scores >= 85 and scores <= 100:
        return ”Grade A”
    elif scores >= 70 and scores < 85:
        return ”Grade B”
    elif scores >= 55 and scores < 70:
        return ”Grade C”
    elif scores >= 35 and scores < 55:
        return ”Grade D”
    elif scores >= 0 and scores < 35:
        return ”Grade E”


df[”grades”] = df[”percentage”].apply(determine_grade)
df.info
df.grades.value_counts().plot.bar()
plt.show()
plt.figure(figsize=(30, 10))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace
   =0.5, hspace=0.2)
plt.subplot(251)
plt.title(”Grades and Gender”)
sns.countplot(hue=”gender”, x=”grades”, data=df)
plt.subplot(252)
plt.title(”Grades and Lunch”)
sns.countplot(hue=”lunch”, x=”grades”, data=df)
plt.subplot(253)
plt.title(”Grades and Test preperation Course”)
sns.countplot(hue=”test preparation course”, x=”grades”, data=df)
plt.show()
plt.title(”Grades and Parental level of Education”, fontsize=20)
sns.countplot(x=”parental level of education”, hue=”grades”, data=df)
plt.show()
plt.title(”Grades and Ethinicity”, fontsize=20)
sns.countplot(x=”race/ethnicity”, hue=”grades”, data=df)
gr = pd.crosstab(df[”grades”], df[”race/ethnicity”], normalize=0)
gr.plot.bar(stacked=True)
plt.title(”Grades and Ethinicity”, fontsize=20)
plt.show()
```

5A. Observations - A:



Objective of this kernel:

- To understand the how the student's performance (test scores) is affected by the other variables (Gender, Ethnicity, Parental level of education, Lunch, Test preparation course)

```
In [1]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

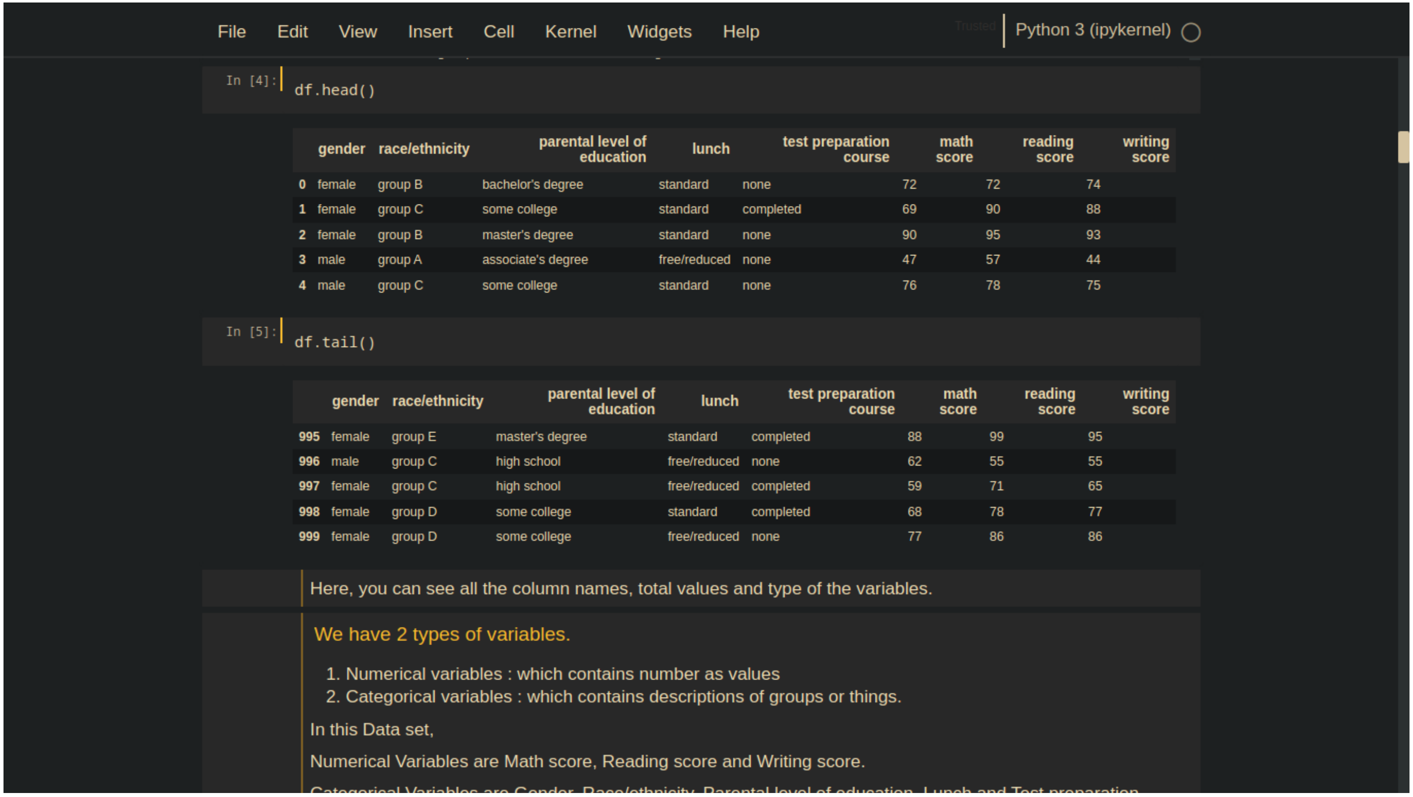
Reading the data set

```
In [2]:
df = pd.read_csv('./StudentsPerformance.csv')
print(df.shape)
```

```
(1000, 8)
```

```
In [3]:
df.info
```

```
24     male     group D   bachelor's degree  free/reduced
25     male     group A     master's degree  free/reduced
26     male     group B        some college      standard
27   female     group C   bachelor's degree      standard
28     male     group C         high school      standard
29   female     group D     master's degree      standard
..      ...         ...                 ...           ...
970  female     group D   bachelor's degree      standard
971    male     group C    some high school      standard
972  female     group A         high school  free/reduced
973  female     group D        some college  free/reduced
974  female     group A        some college      standard
975  female     group C        some college      standard
976    male     group B        some college  free/reduced
977    male     group C  associate's degree      standard
```



```
In [4]:
df.head()
```

	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 [5]:
df.tail()
```

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

Here, you can see all the column names, total values and type of the variables.

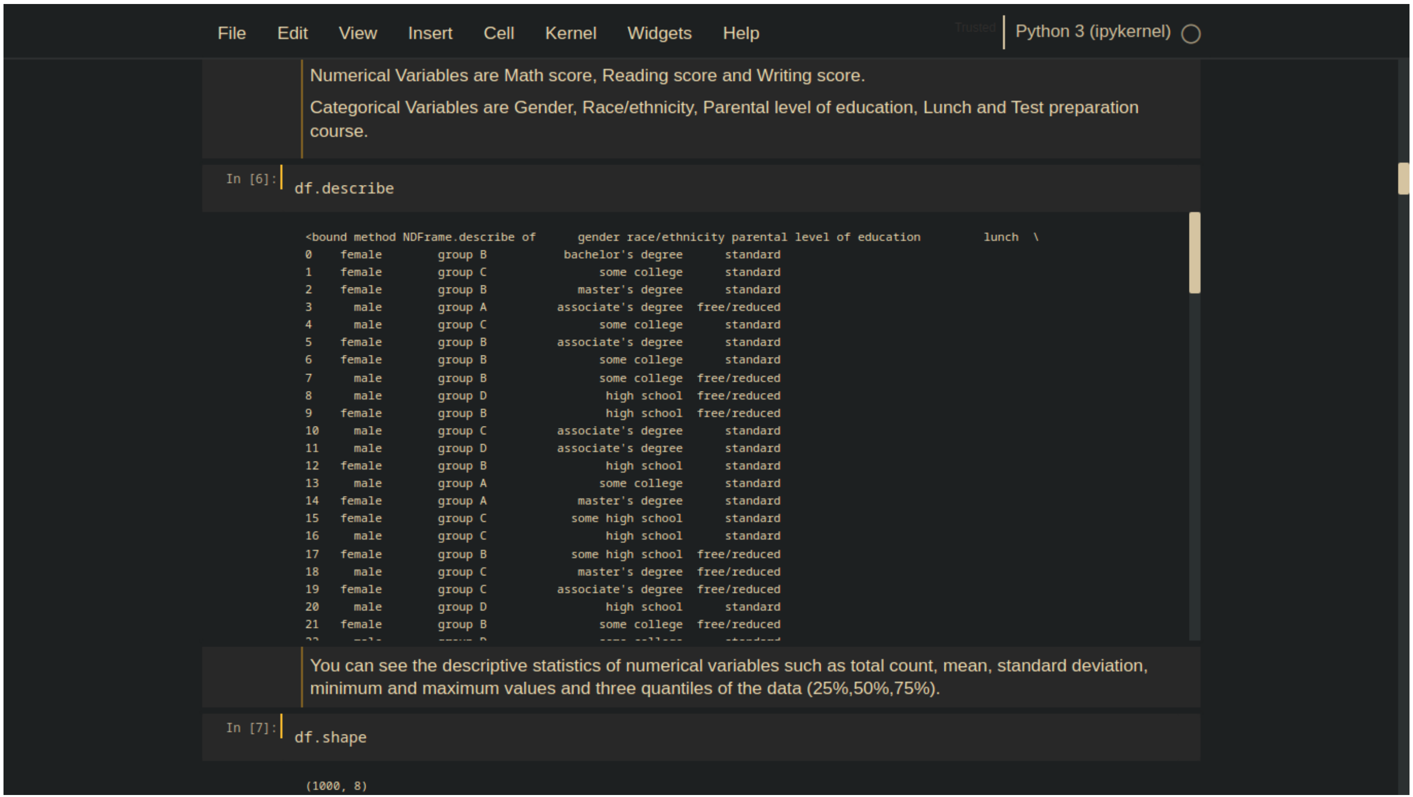
We have 2 types of variables.

1. Numerical variables : which contains number as values
2. Categorical variables : which contains descriptions of groups or things.

In this Data set,

Numerical Variables are Math score, Reading score and Writing score.

Categorical Variables are Gender, Race/ethnicity, Parental level of education, Lunch and Test preparation



Numerical Variables are Math score, Reading score and Writing score.

Categorical Variables are Gender, Race/ethnicity, Parental level of education, Lunch and Test preparation course.

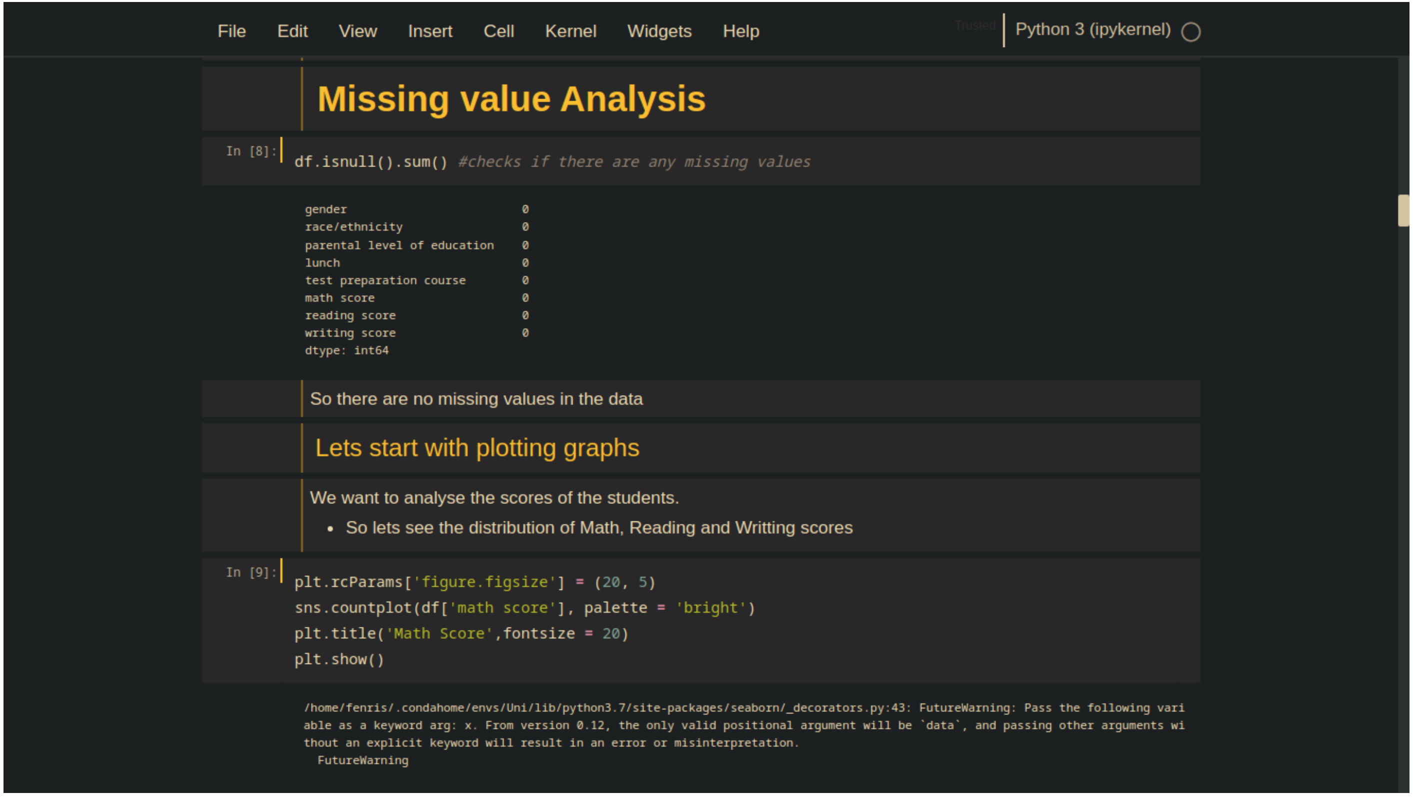
In [6]:
```
df.describe
```

```
<bound method NDFrame.describe of      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
5    female        group B          associate's degree      standard
6    female        group B                some college      standard
7      male        group B                some college  free/reduced
8      male        group D                 high school  free/reduced
9    female        group B                 high school  free/reduced
10     male        group C          associate's degree      standard
11     male        group D          associate's degree      standard
12   female        group B                 high school      standard
13     male        group A                some college      standard
14   female        group A             master's degree      standard
15   female        group C            some high school      standard
16     male        group C                 high school      standard
17   female        group B            some high school  free/reduced
18     male        group C             master's degree  free/reduced
19   female        group C          associate's degree  free/reduced
20     male        group D                 high school      standard
21   female        group B                some college  free/reduced
```

You can see the descriptive statistics of numerical variables such as total count, mean, standard deviation, minimum and maximum values and three quantiles of the data (25%,50%,75%).

In [7]:
```
df.shape
```

```
(1000, 8)
```



Missing value Analysis

In [8]:
```
df.isnull().sum() #checks if there are any missing values
```

```
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
```

So there are no missing values in the data

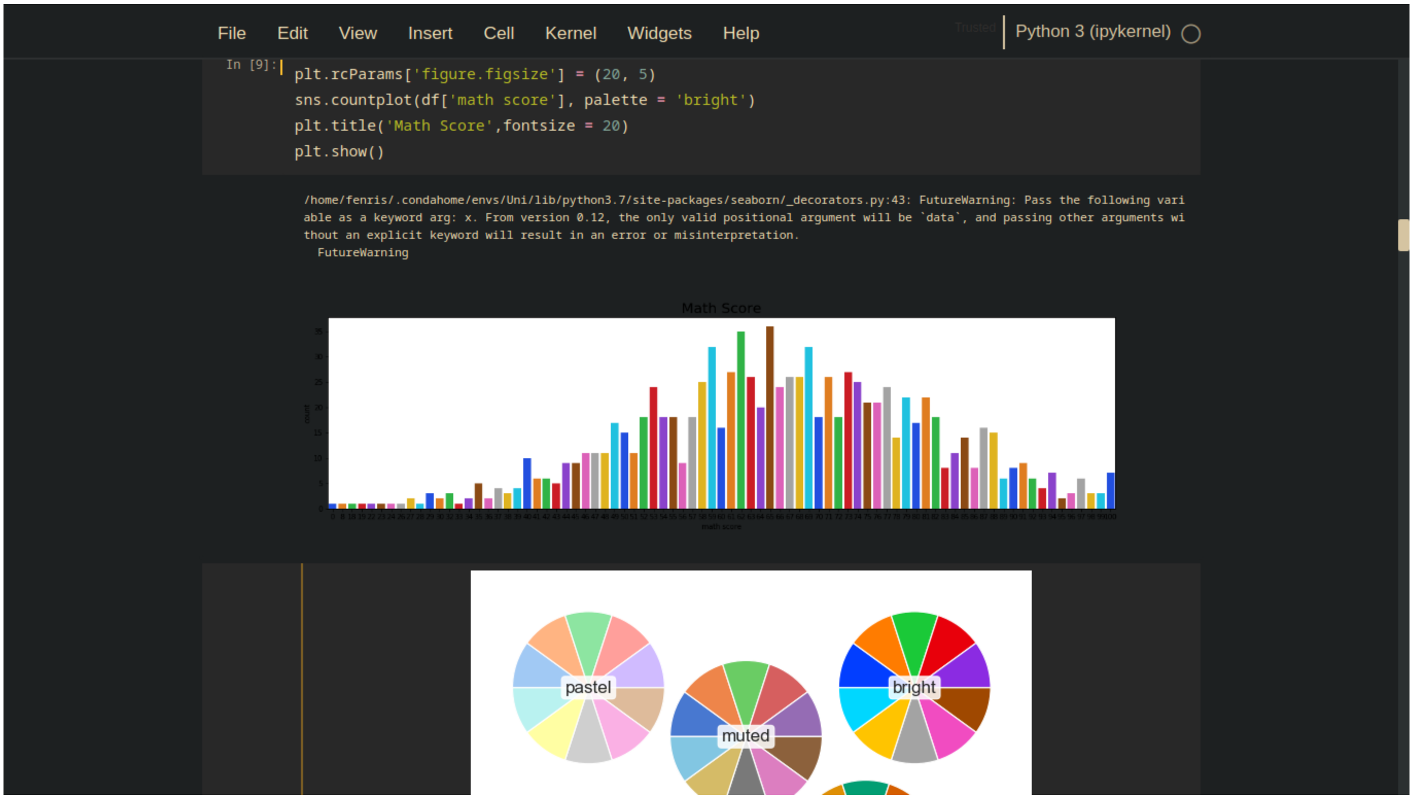
Lets start with plotting graphs

We want to analyse the scores of the students.

- So lets see the distribution of Math, Reading and Writting scores

In [9]:
```
plt.rcParams['figure.figsize'] = (20, 5)
sns.countplot(df['math score'], palette = 'bright')
plt.title('Math Score',fontsize = 20)
plt.show()
```

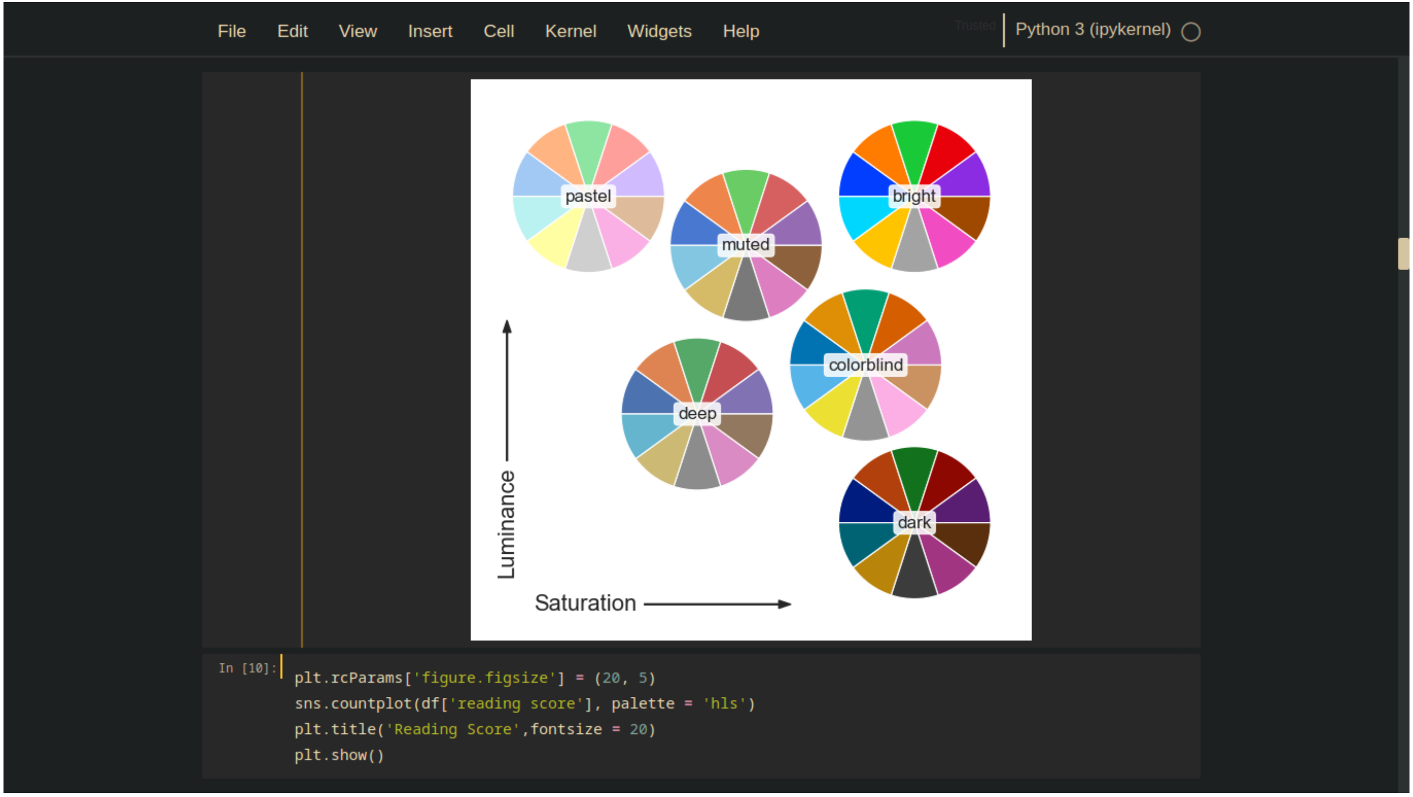
```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



In [9]:

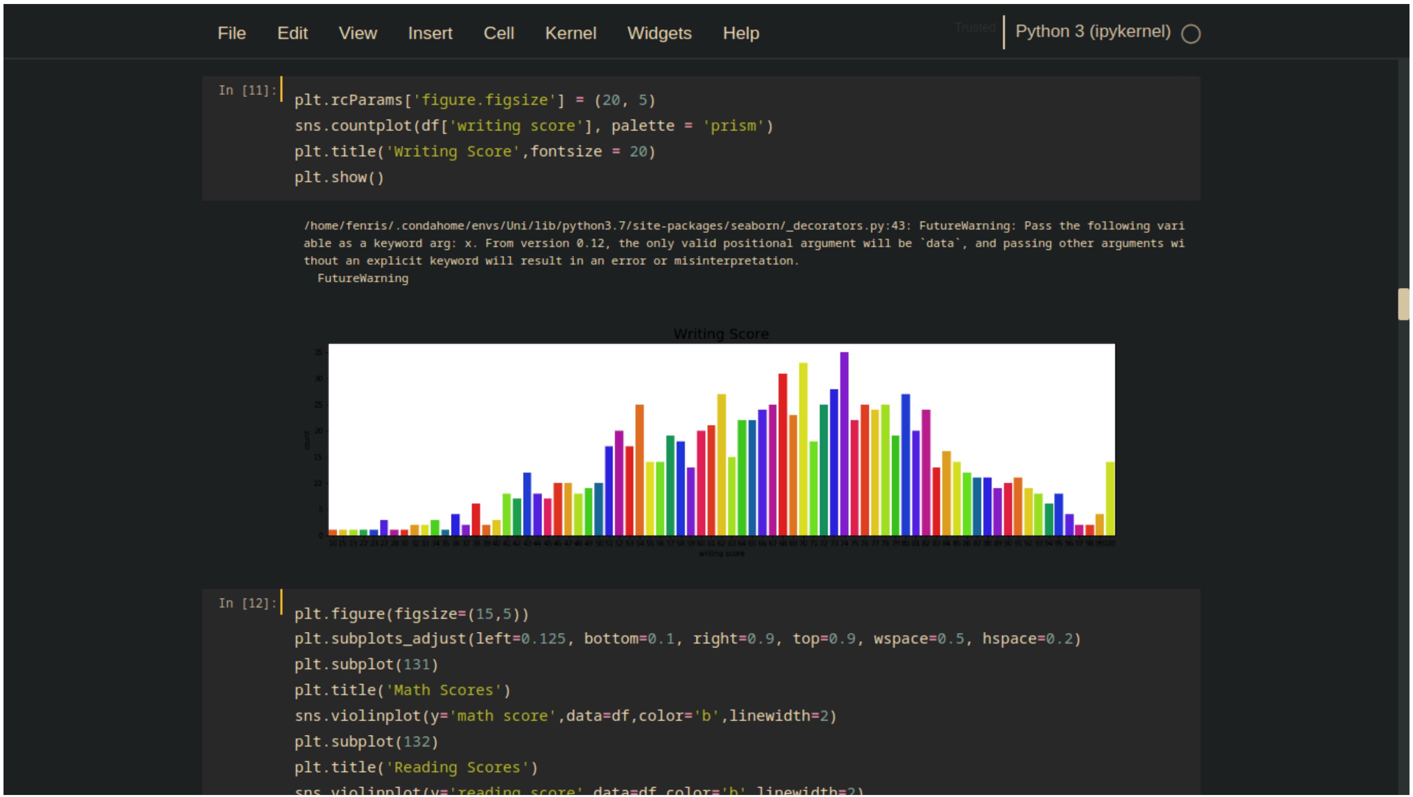
```
plt.rcParams['figure.figsize'] = (20, 5)
sns.countplot(df['math score'], palette = 'bright')
plt.title('Math Score',fontsize = 20)
plt.show()
```

```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



In [10]:

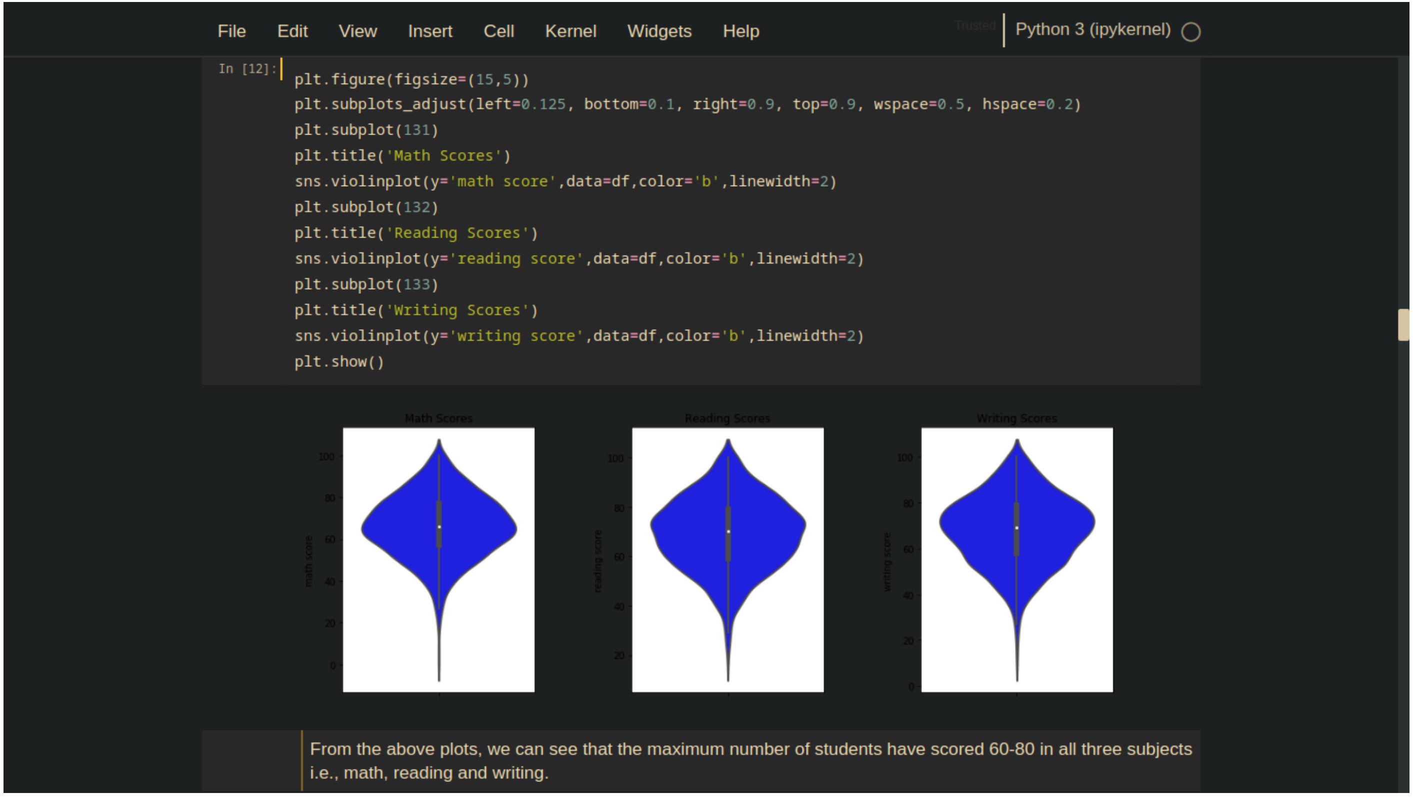
```
plt.rcParams['figure.figsize'] = (20, 5)
sns.countplot(df['reading score'], palette = 'hls')
plt.title('Reading Score',fontsize = 20)
plt.show()
```



In [11]:
```
plt.rcParams['figure.figsize'] = (20, 5)
sns.countplot(df['writing score'], palette = 'prism')
plt.title('Writing Score',fontsize = 20)
plt.show()
```

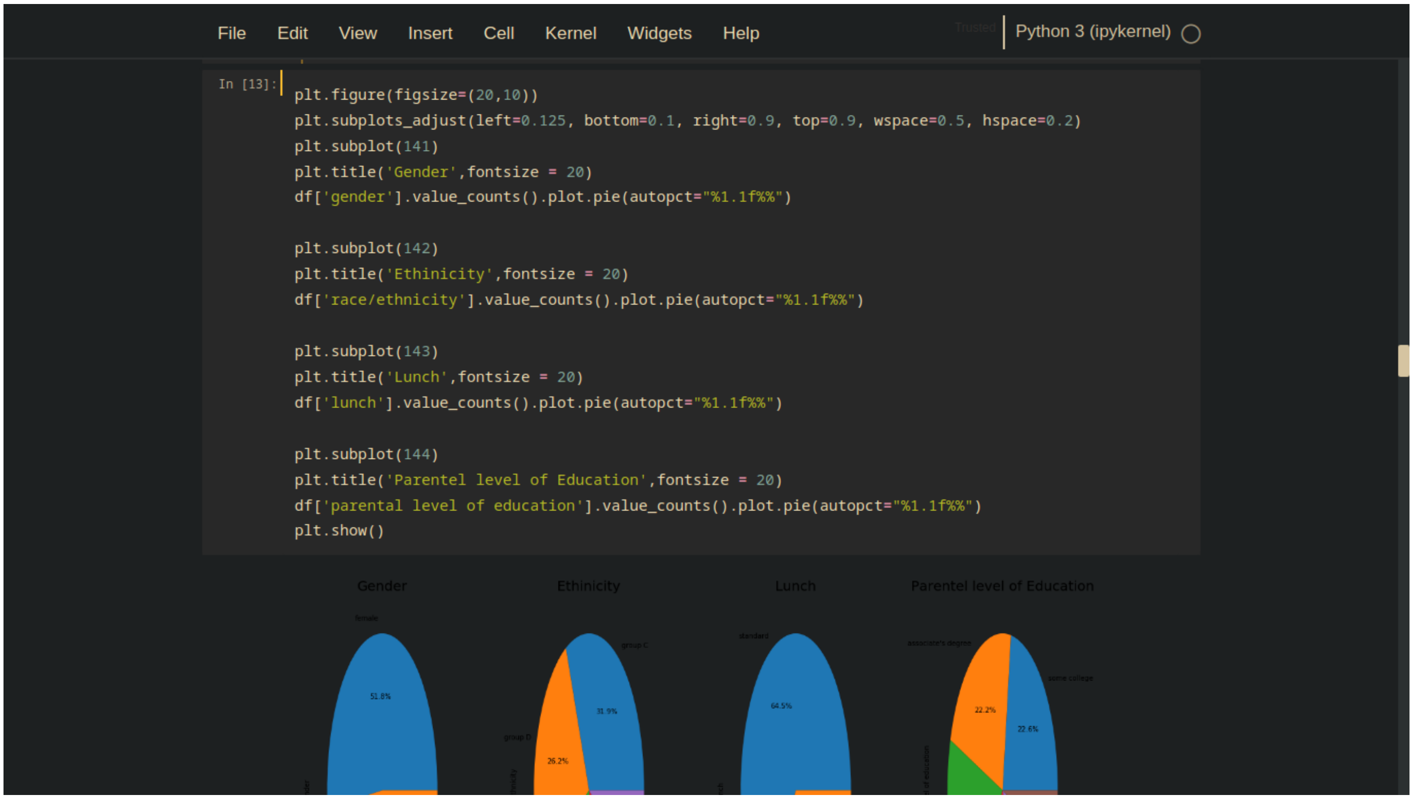
```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

In [12]:
```
plt.figure(figsize=(15,5))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.5, hspace=0.2)
plt.subplot(131)
plt.title('Math Scores')
sns.violinplot(y='math score',data=df,color='b',linewidth=2)
plt.subplot(132)
plt.title('Reading Scores')
sns.violinplot(y='reading score',data=df,color='b',linewidth=2)
```

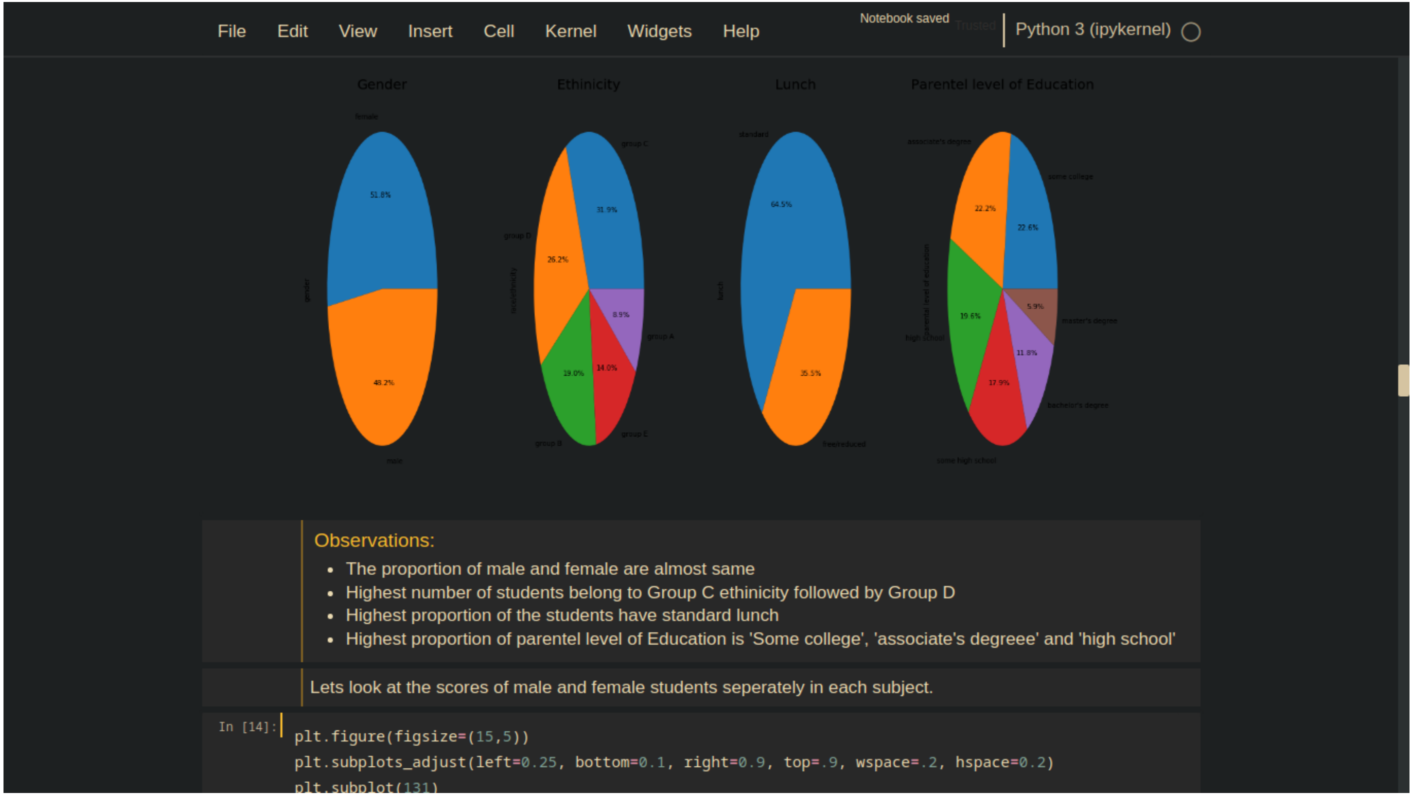


In [12]:
```
plt.figure(figsize=(15,5))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.5, hspace=0.2)
plt.subplot(131)
plt.title('Math Scores')
sns.violinplot(y='math score',data=df,color='b',linewidth=2)
plt.subplot(132)
plt.title('Reading Scores')
sns.violinplot(y='reading score',data=df,color='b',linewidth=2)
plt.subplot(133)
plt.title('Writing Scores')
sns.violinplot(y='writing score',data=df,color='b',linewidth=2)
plt.show()
```

From the above plots, we can see that the maximum number of students have scored 60-80 in all three subjects i.e., math, reading and writing.



```
plt.figure(figsize=(20,10))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.5, hspace=0.2)
plt.subplot(141)
plt.title('Gender',fontsize = 20)
df['gender'].value_counts().plot.pie(autopct="%1.1f%%")

plt.subplot(142)
plt.title('Ethinicity',fontsize = 20)
df['race/ethnicity'].value_counts().plot.pie(autopct="%1.1f%%")

plt.subplot(143)
plt.title('Lunch',fontsize = 20)
df['lunch'].value_counts().plot.pie(autopct="%1.1f%%")

plt.subplot(144)
plt.title('Parentel level of Education',fontsize = 20)
df['parental level of education'].value_counts().plot.pie(autopct="%1.1f%%")
plt.show()
```

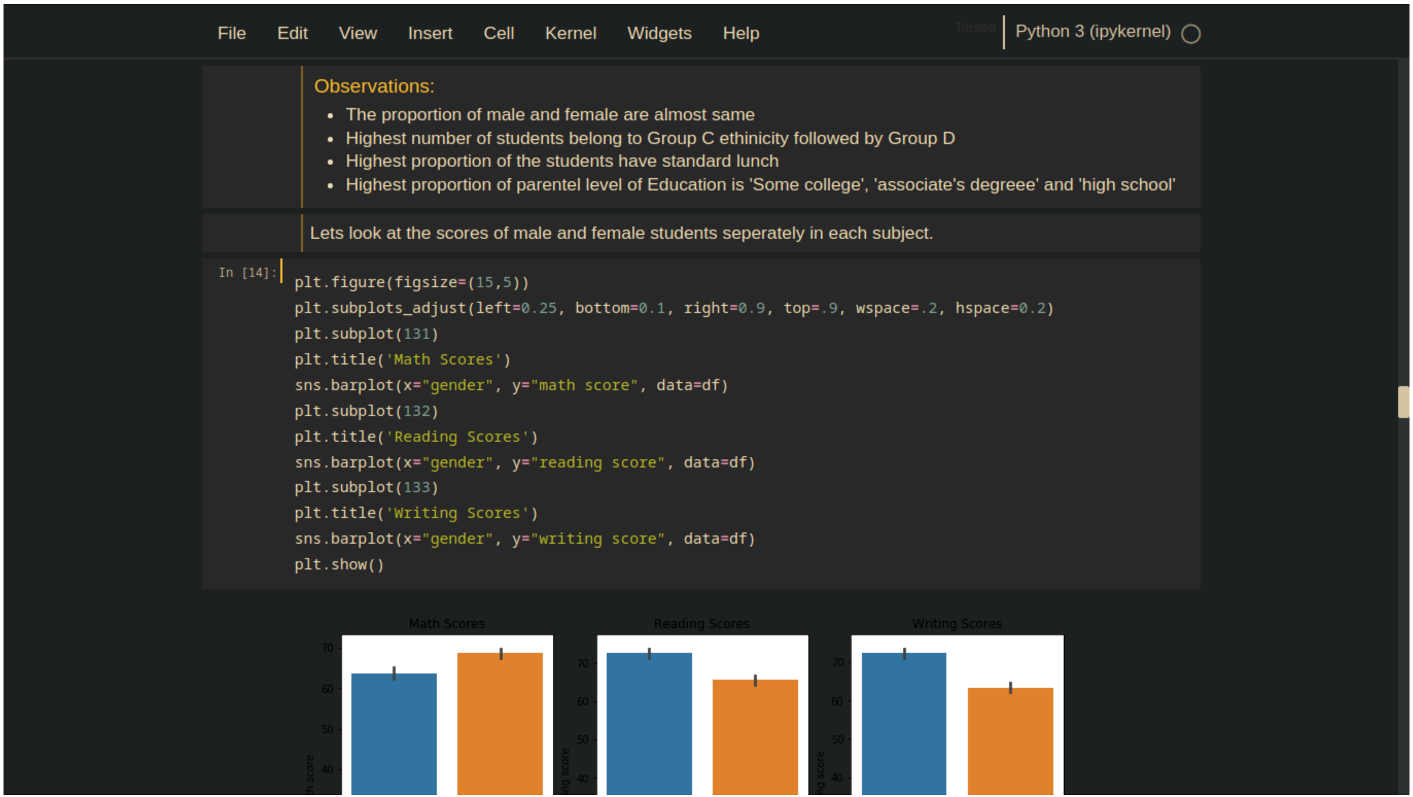


Observations:

- The proportion of male and female are almost same
- Highest number of students belong to Group C ethinicity followed by Group D
- Highest proportion of the students have standard lunch
- Highest proportion of parentel level of Education is 'Some college', 'associate's degreee' and 'high school'

Lets look at the scores of male and female students seperately in each subject.

```
plt.figure(figsize=(15,5))
plt.subplots_adjust(left=0.25, bottom=0.1, right=0.9, top=.9, wspace=.2, hspace=0.2)
plt.subplot(131)
```



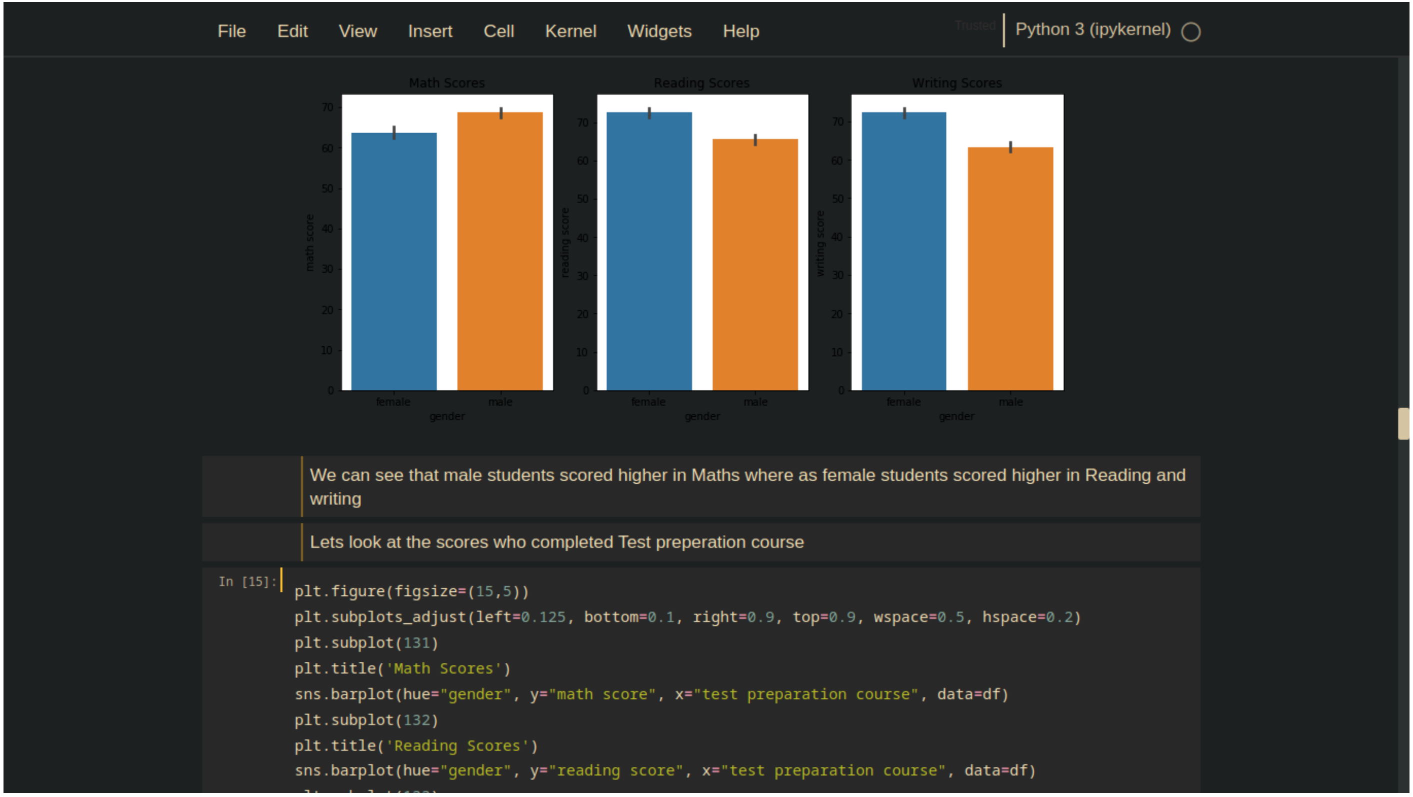
Observations:

- The proportion of male and female are almost same
- Highest number of students belong to Group C ethinicity followed by Group D
- Highest proportion of the students have standard lunch
- Highest proportion of parentel level of Education is 'Some college', 'associate's degreee' and 'high school'

Lets look at the scores of male and female students seperately in each subject.

In [14]:

```
plt.figure(figsize=(15,5))
plt.subplots_adjust(left=0.25, bottom=0.1, right=0.9, top=.9, wspace=.2, hspace=0.2)
plt.subplot(131)
plt.title('Math Scores')
sns.barplot(x="gender", y="math score", data=df)
plt.subplot(132)
plt.title('Reading Scores')
sns.barplot(x="gender", y="reading score", data=df)
plt.subplot(133)
plt.title('Writing Scores')
sns.barplot(x="gender", y="writing score", data=df)
plt.show()
```

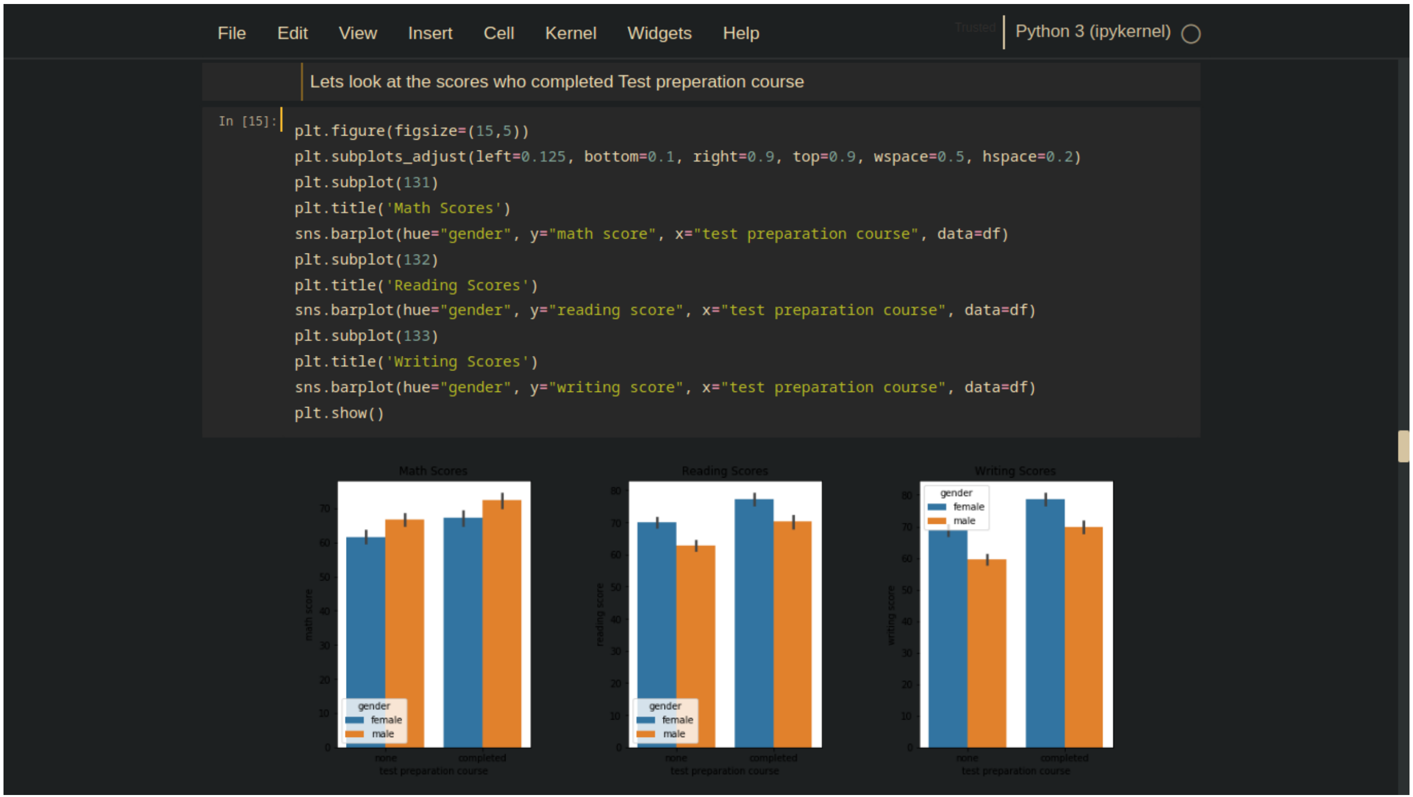


We can see that male students scored higher in Maths where as female students scored higher in Reading and writing

Lets look at the scores who completed Test preperation course

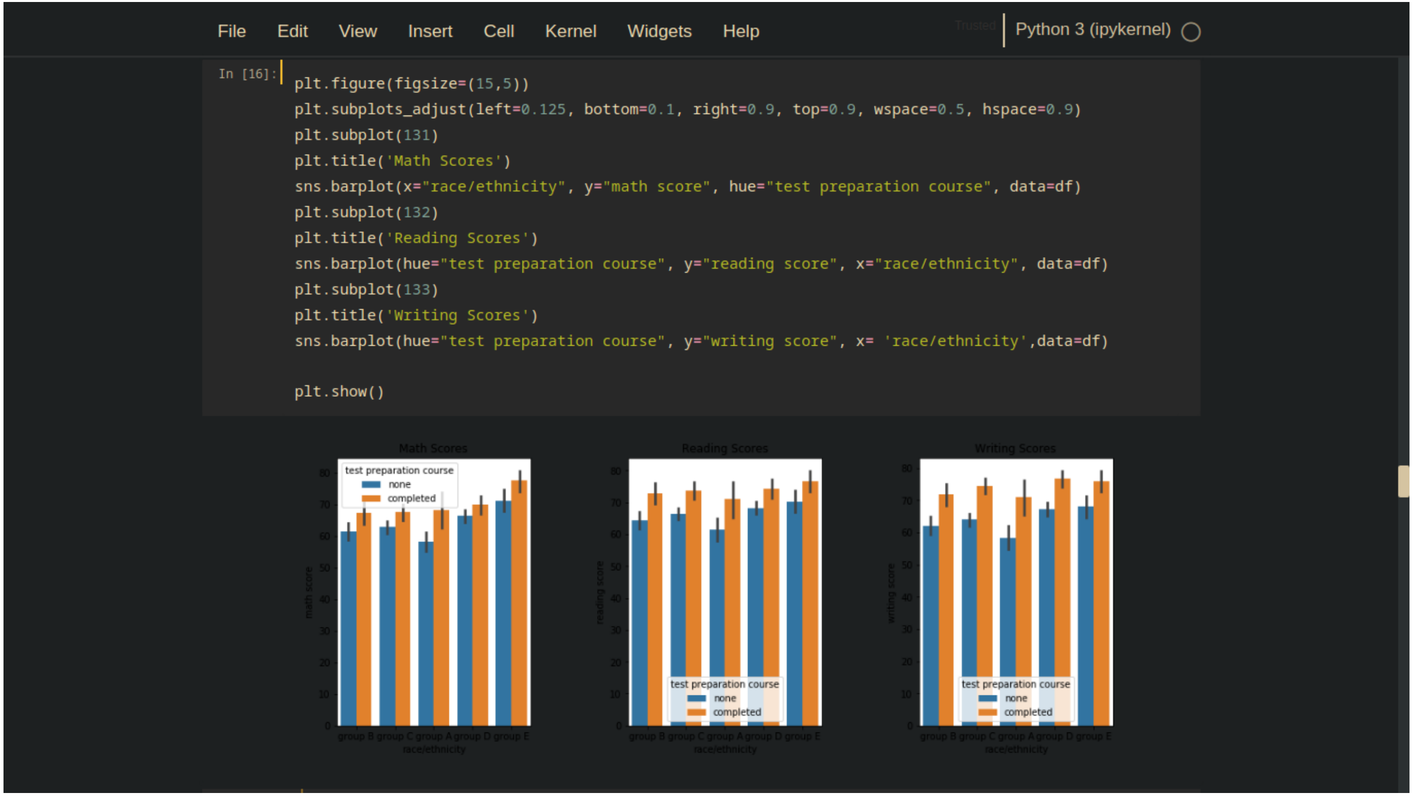
In [15]:

```
plt.figure(figsize=(15,5))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.5, hspace=0.2)
plt.subplot(131)
plt.title('Math Scores')
sns.barplot(hue="gender", y="math score", x="test preparation course", data=df)
plt.subplot(132)
plt.title('Reading Scores')
sns.barplot(hue="gender", y="reading score", x="test preparation course", data=df)
```

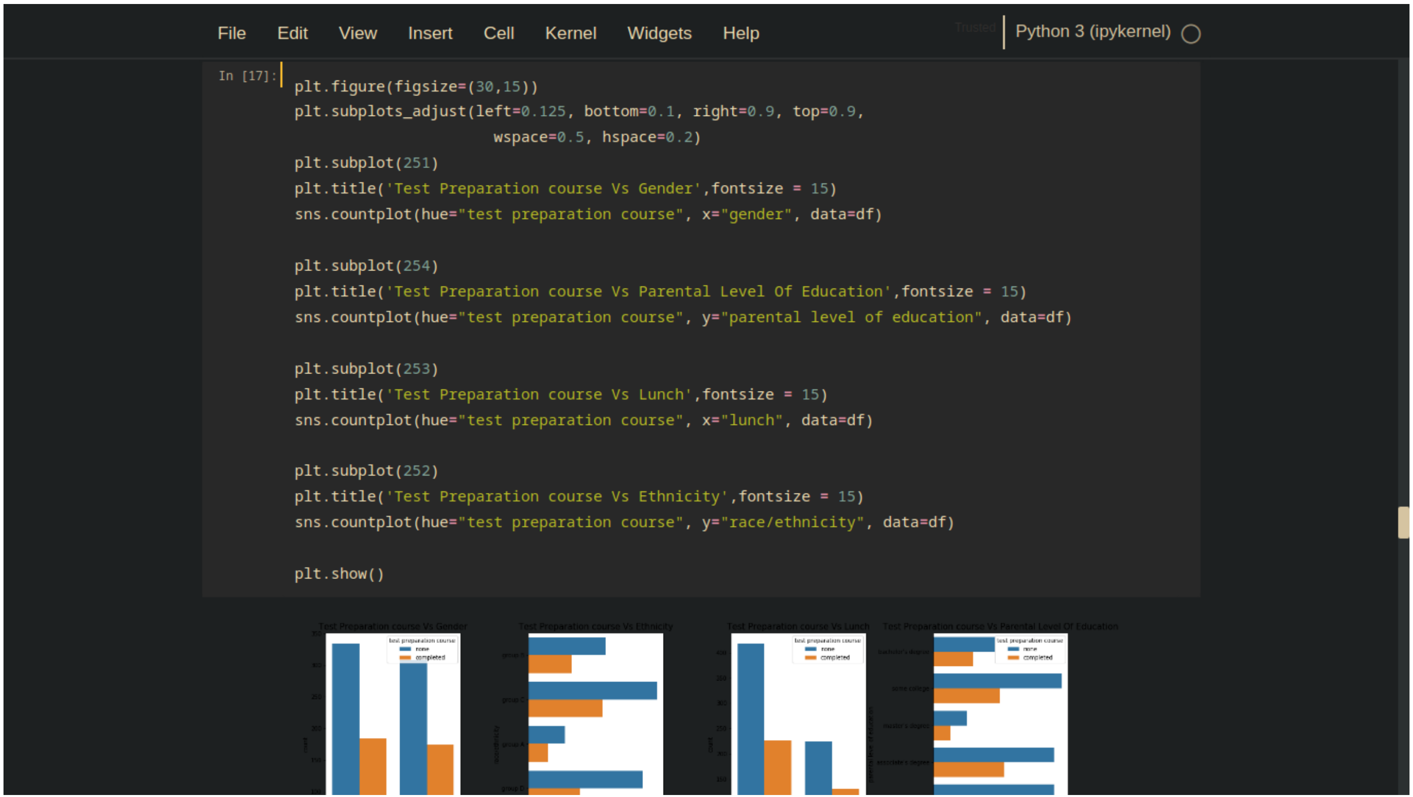


Lets look at the scores who completed Test preperation course

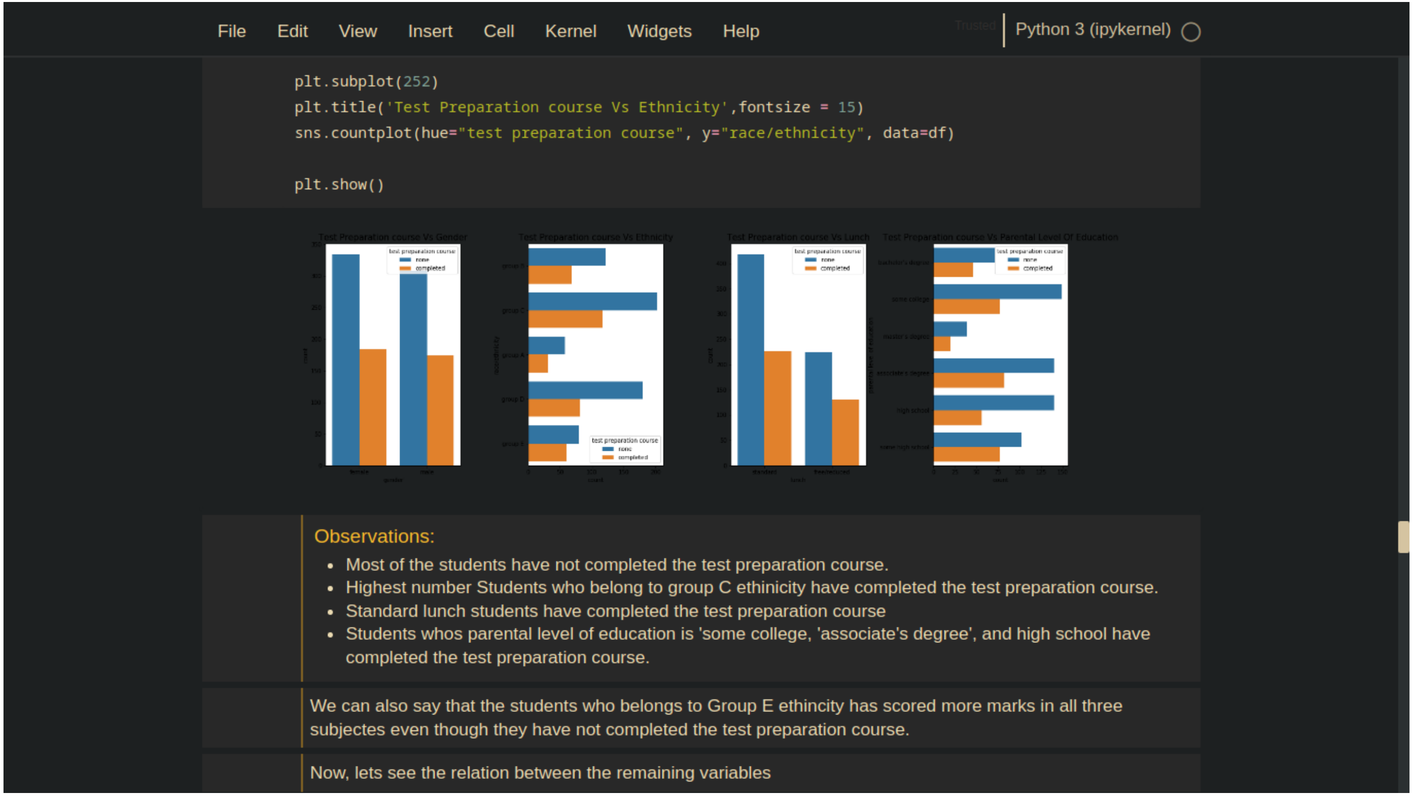
```
plt.figure(figsize=(15,5))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.5, hspace=0.2)
plt.subplot(131)
plt.title('Math Scores')
sns.barplot(hue="gender", y="math score", x="test preparation course", data=df)
plt.subplot(132)
plt.title('Reading Scores')
sns.barplot(hue="gender", y="reading score", x="test preparation course", data=df)
plt.subplot(133)
plt.title('Writing Scores')
sns.barplot(hue="gender", y="writing score", x="test preparation course", data=df)
plt.show()
```



```
plt.figure(figsize=(15,5))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.5, hspace=0.9)
plt.subplot(131)
plt.title('Math Scores')
sns.barplot(x="race/ethnicity", y="math score", hue="test preparation course", data=df)
plt.subplot(132)
plt.title('Reading Scores')
sns.barplot(hue="test preparation course", y="reading score", x="race/ethnicity", data=df)
plt.subplot(133)
plt.title('Writing Scores')
sns.barplot(hue="test preparation course", y="writing score", x= 'race/ethnicity',data=df)

plt.show()
```


File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel) ○
In [17]:
Test Preparation course Vs Gender
Test Preparation course Vs Ethnicity
Test Preparation course Vs Lunch
Test Preparation course Vs Parental Level Of Education
test preparation course
none
completed

```
plt.figure(figsize=(30,15))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9,
                      wspace=0.5, hspace=0.2)
plt.subplot(251)
plt.title('Test Preparation course Vs Gender',fontsize = 15)
sns.countplot(hue="test preparation course", x="gender", data=df)

plt.subplot(254)
plt.title('Test Preparation course Vs Parental Level Of Education',fontsize = 15)
sns.countplot(hue="test preparation course", y="parental level of education", data=df)

plt.subplot(253)
plt.title('Test Preparation course Vs Lunch',fontsize = 15)
sns.countplot(hue="test preparation course", x="lunch", data=df)

plt.subplot(252)
plt.title('Test Preparation course Vs Ethnicity',fontsize = 15)
sns.countplot(hue="test preparation course", y="race/ethnicity", data=df)

plt.show()
```


File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3 (ipykernel) ○
Test Preparation course Vs Gender
Test Preparation course Vs Ethnicity
Test Preparation course Vs Lunch
Test Preparation course Vs Parental Level Of Education
test preparation course
none
completed

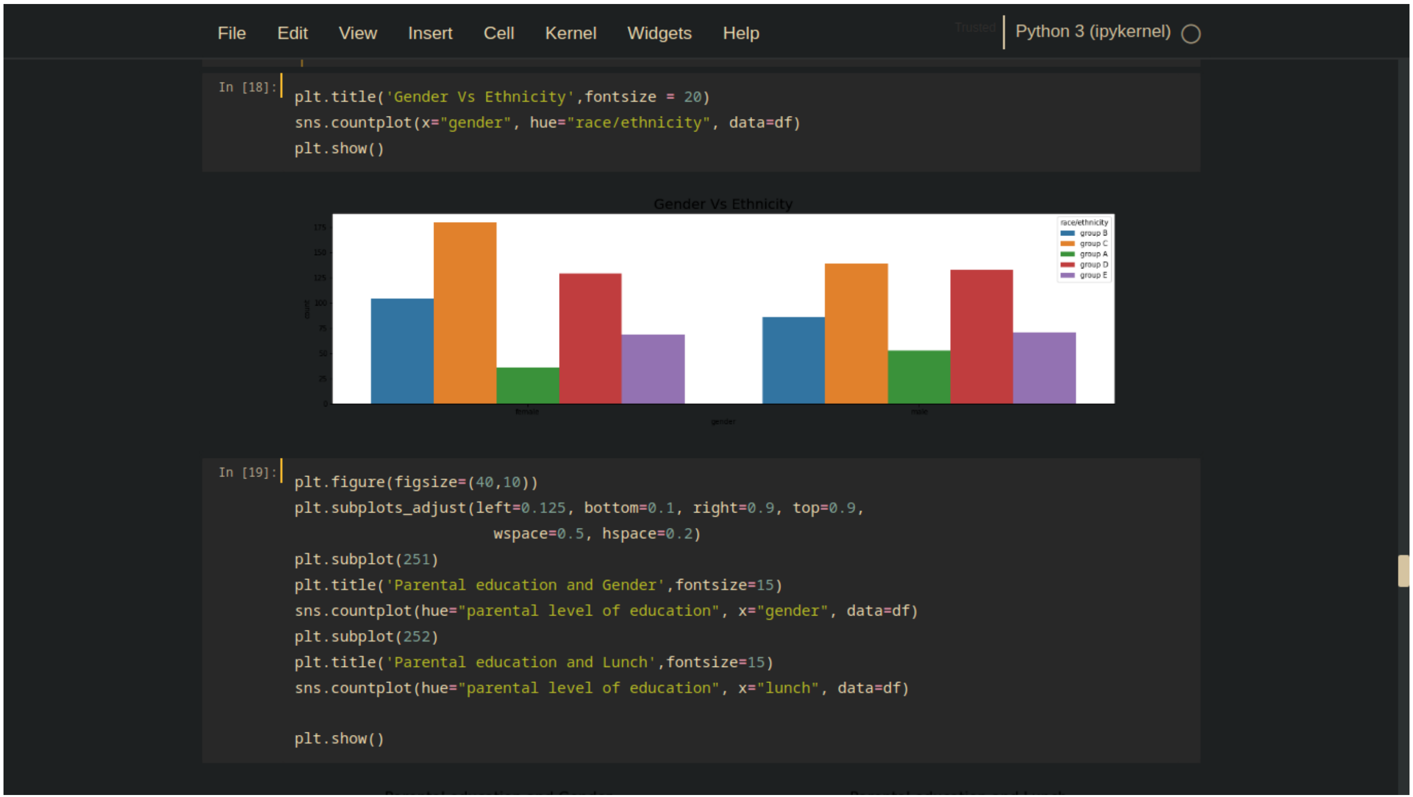
```
plt.subplot(252)
plt.title('Test Preparation course Vs Ethnicity',fontsize = 15)
sns.countplot(hue="test preparation course", y="race/ethnicity", data=df)

plt.show()
```

Observations:

- Most of the students have not completed the test preparation course.
- Highest number Students who belong to group C ethinicity have completed the test preparation course.
- Standard lunch students have completed the test preparation course
- Students whos parental level of education is 'some college, 'associate's degree', and high school have completed the test preparation course.

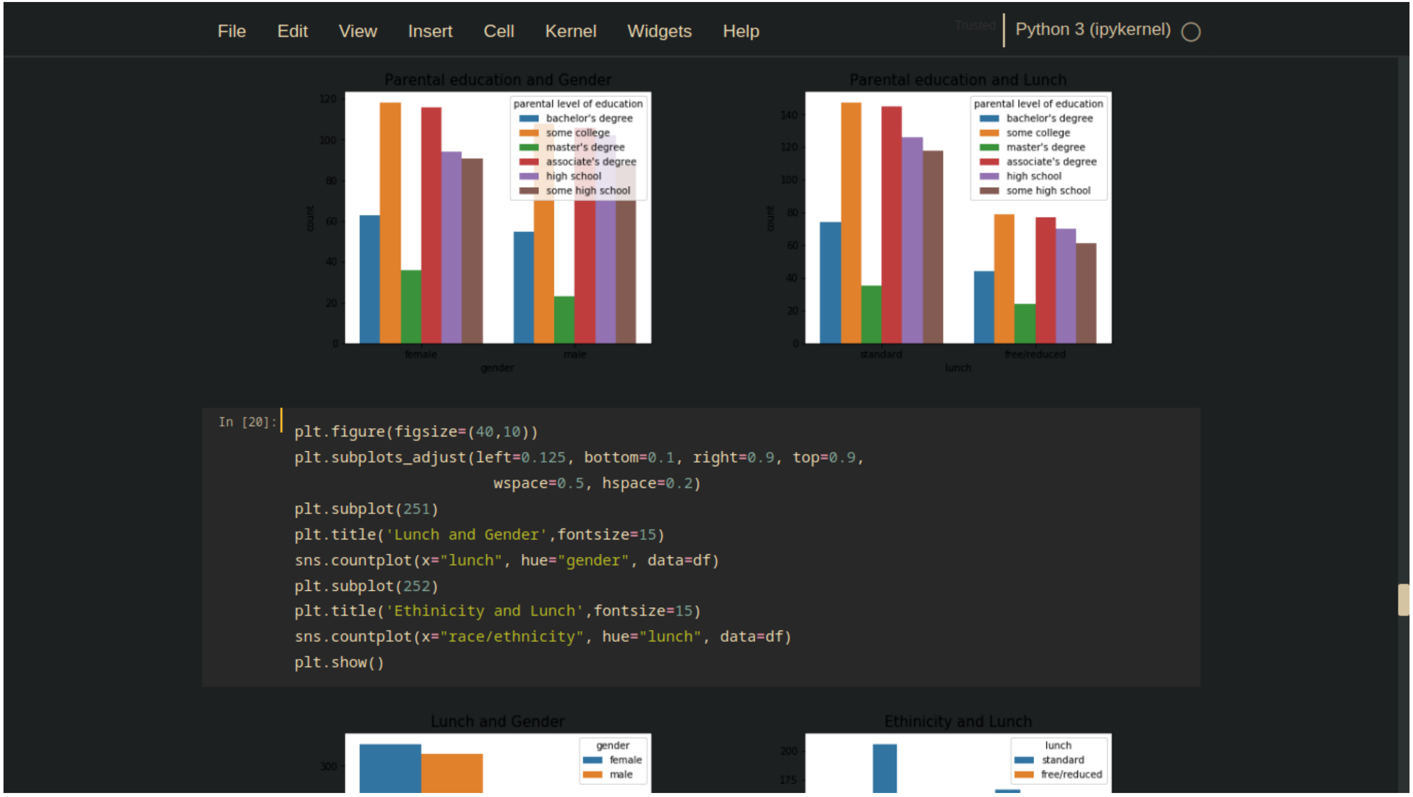
We can also say that the students who belongs to Group E ethincity has scored more marks in all three subjectes even though they have not completed the test preparation course.

Now, lets see the relation between the remaining variables

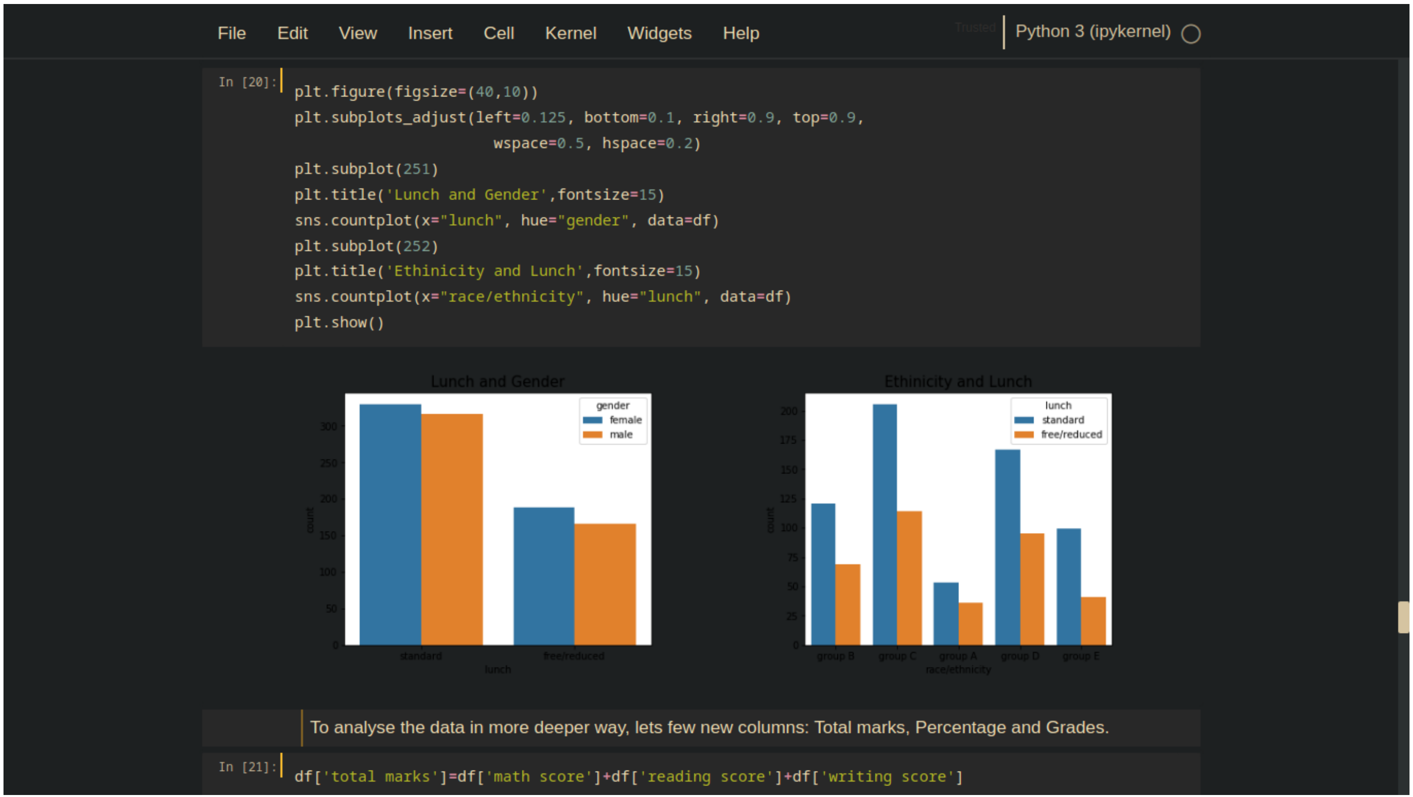


```
In [18]: plt.title('Gender Vs Ethnicity',fontsize = 20)
         sns.countplot(x="gender", hue="race/ethnicity", data=df)
         plt.show()
```

```
In [19]: plt.figure(figsize=(40,10))
         plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9,
                             wspace=0.5, hspace=0.2)
         plt.subplot(251)
         plt.title('Parental education and Gender',fontsize=15)
         sns.countplot(hue="parental level of education", x="gender", data=df)
         plt.subplot(252)
         plt.title('Parental education and Lunch',fontsize=15)
         sns.countplot(hue="parental level of education", x="lunch", data=df)

         plt.show()
```



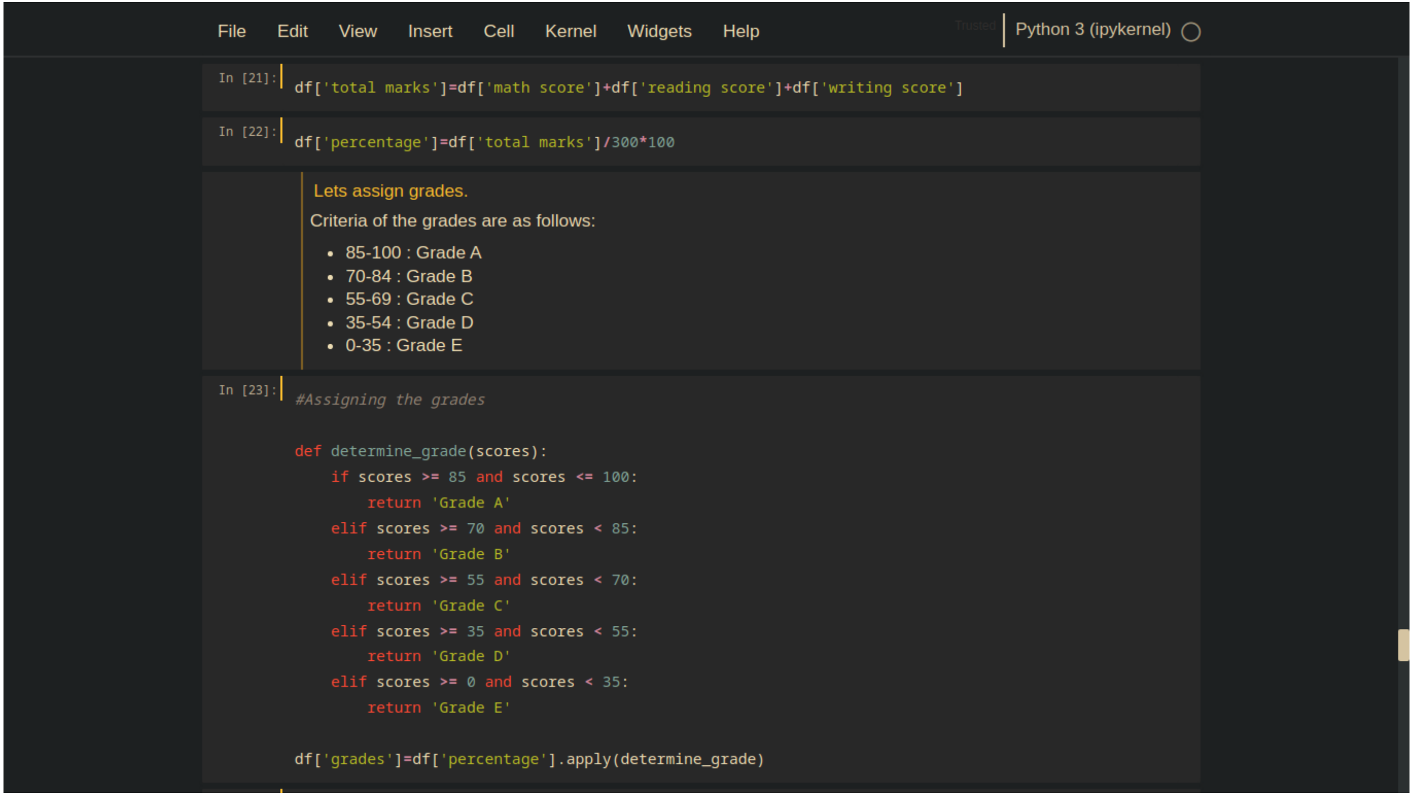
```
In [20]: plt.figure(figsize=(40,10))
         plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9,
                             wspace=0.5, hspace=0.2)
         plt.subplot(251)
         plt.title('Lunch and Gender',fontsize=15)
         sns.countplot(x="lunch", hue="gender", data=df)
         plt.subplot(252)
         plt.title('Ethinicity and Lunch',fontsize=15)
         sns.countplot(x="race/ethnicity", hue="lunch", data=df)
         plt.show()
```



```
In [20]: plt.figure(figsize=(40,10))
         plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9,
                             wspace=0.5, hspace=0.2)
         plt.subplot(251)
         plt.title('Lunch and Gender',fontsize=15)
         sns.countplot(x="lunch", hue="gender", data=df)
         plt.subplot(252)
         plt.title('Ethinicity and Lunch',fontsize=15)
         sns.countplot(x="race/ethnicity", hue="lunch", data=df)
         plt.show()
```

To analyse the data in more deeper way, lets few new columns: Total marks, Percentage and Grades.

```
In [21]: df['total marks']=df['math score']+df['reading score']+df['writing score']
```



```
In [21]: df['total marks']=df['math score']+df['reading score']+df['writing score']
```

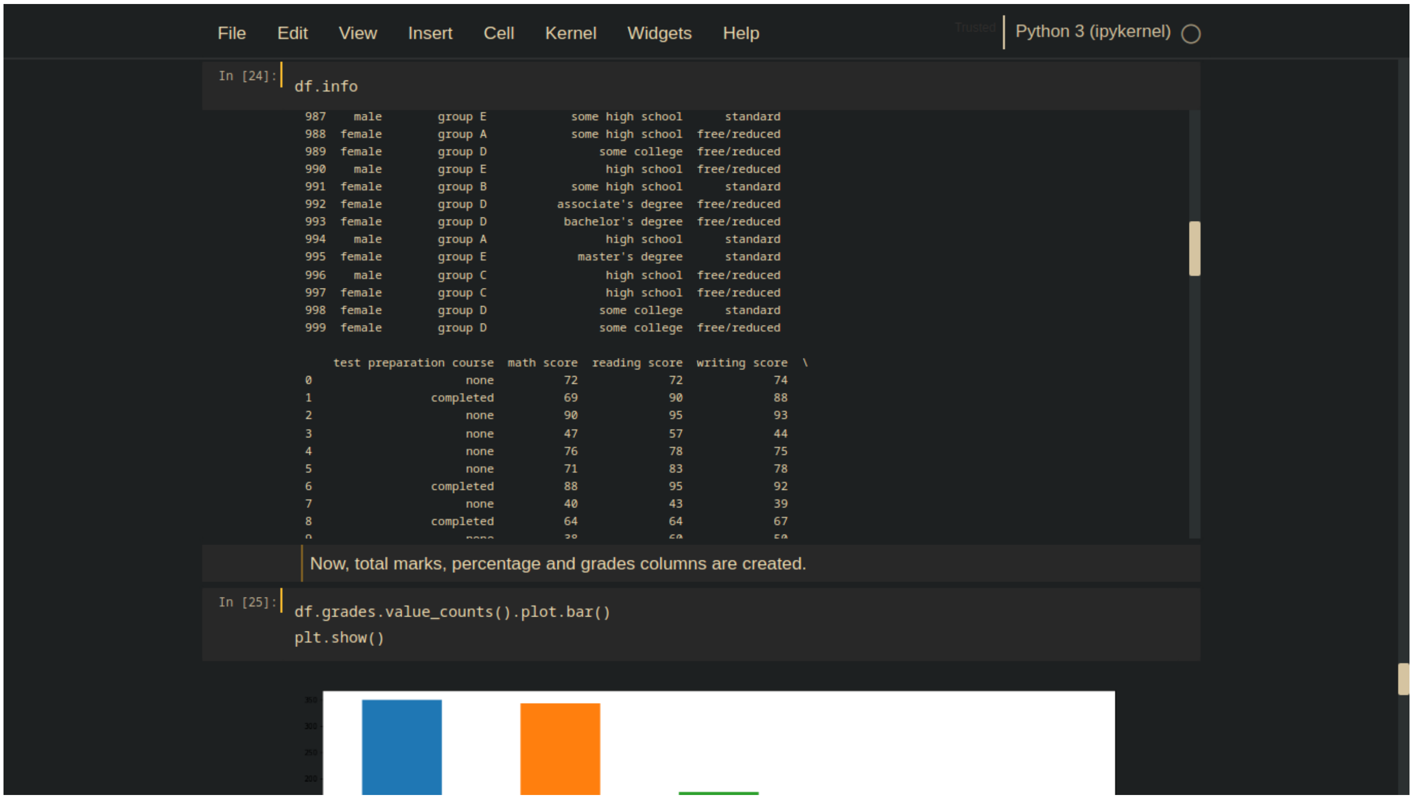
```
In [22]: df['percentage']=df['total marks']/300*100
```

Lets assign grades.

Criteria of the grades are as follows:

- 85-100 : Grade A
- 70-84 : Grade B
- 55-69 : Grade C
- 35-54 : Grade D
- 0-35 : Grade E

```
In [23]: #Assigning the grades

         def determine_grade(scores):
             if scores >= 85 and scores <= 100:
                 return 'Grade A'
             elif scores >= 70 and scores < 85:
                 return 'Grade B'
             elif scores >= 55 and scores < 70:
                 return 'Grade C'
             elif scores >= 35 and scores < 55:
                 return 'Grade D'
             elif scores >= 0 and scores < 35:
                 return 'Grade E'

         df['grades']=df['percentage'].apply(determine_grade)
```



In [24]:

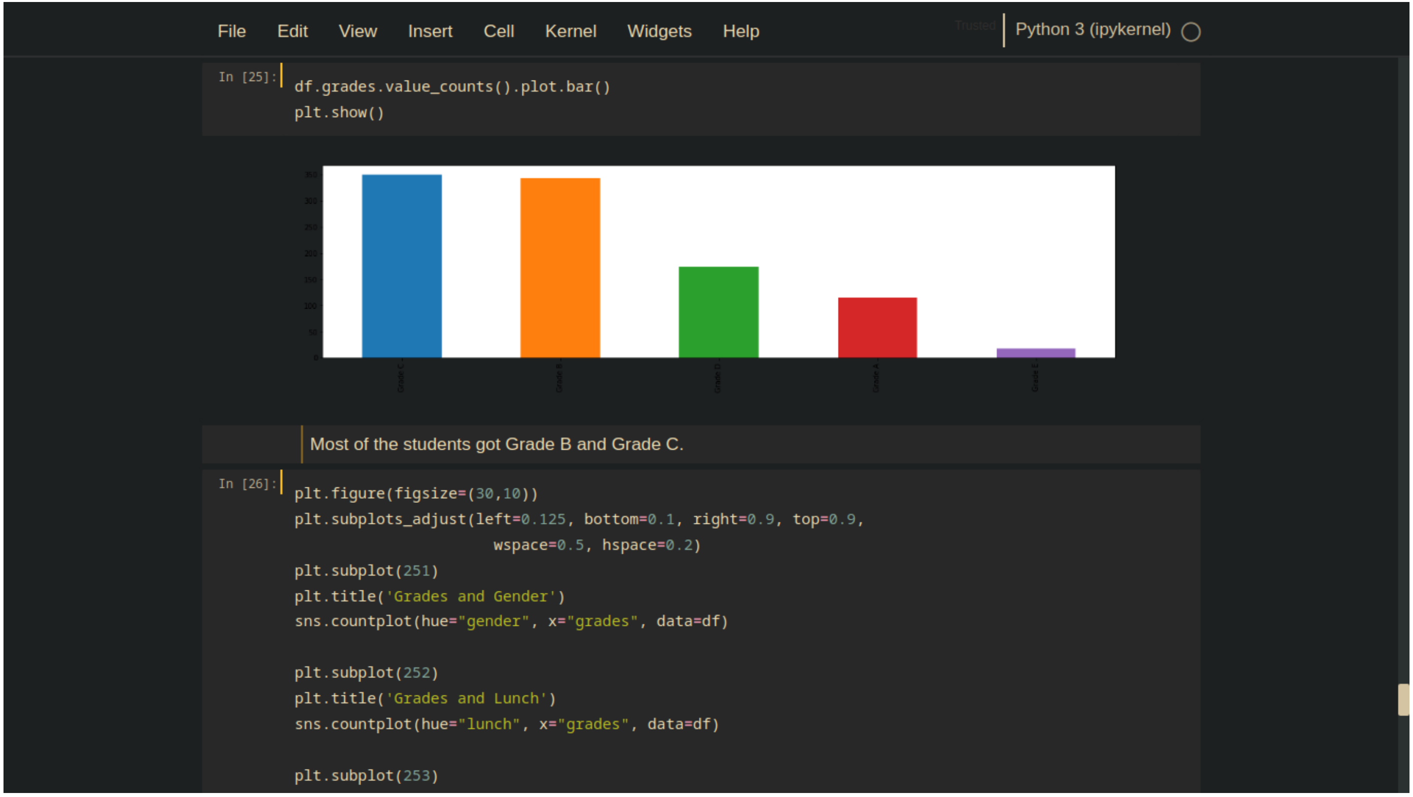
```
df.info
```

```
 987    male        group E         some high school      standard
 988  female        group A         some high school  free/reduced
 989  female        group D             some college  free/reduced
 990    male        group E              high school  free/reduced
 991  female        group B         some high school      standard
 992  female        group D       associate's degree  free/reduced
 993  female        group D        bachelor's degree  free/reduced
 994    male        group A              high school      standard
 995  female        group E          master's degree      standard
 996    male        group C              high school  free/reduced
 997  female        group C              high school  free/reduced
 998  female        group D             some college      standard
 999  female        group D             some college  free/reduced

    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
 5                     none          71             83             78
 6                completed          88             95             92
 7                     none          40             43             39
 8                completed          64             64             67
```

Now, total marks, percentage and grades columns are created.

In [25]:

```
df.grades.value_counts().plot.bar()
plt.show()
```



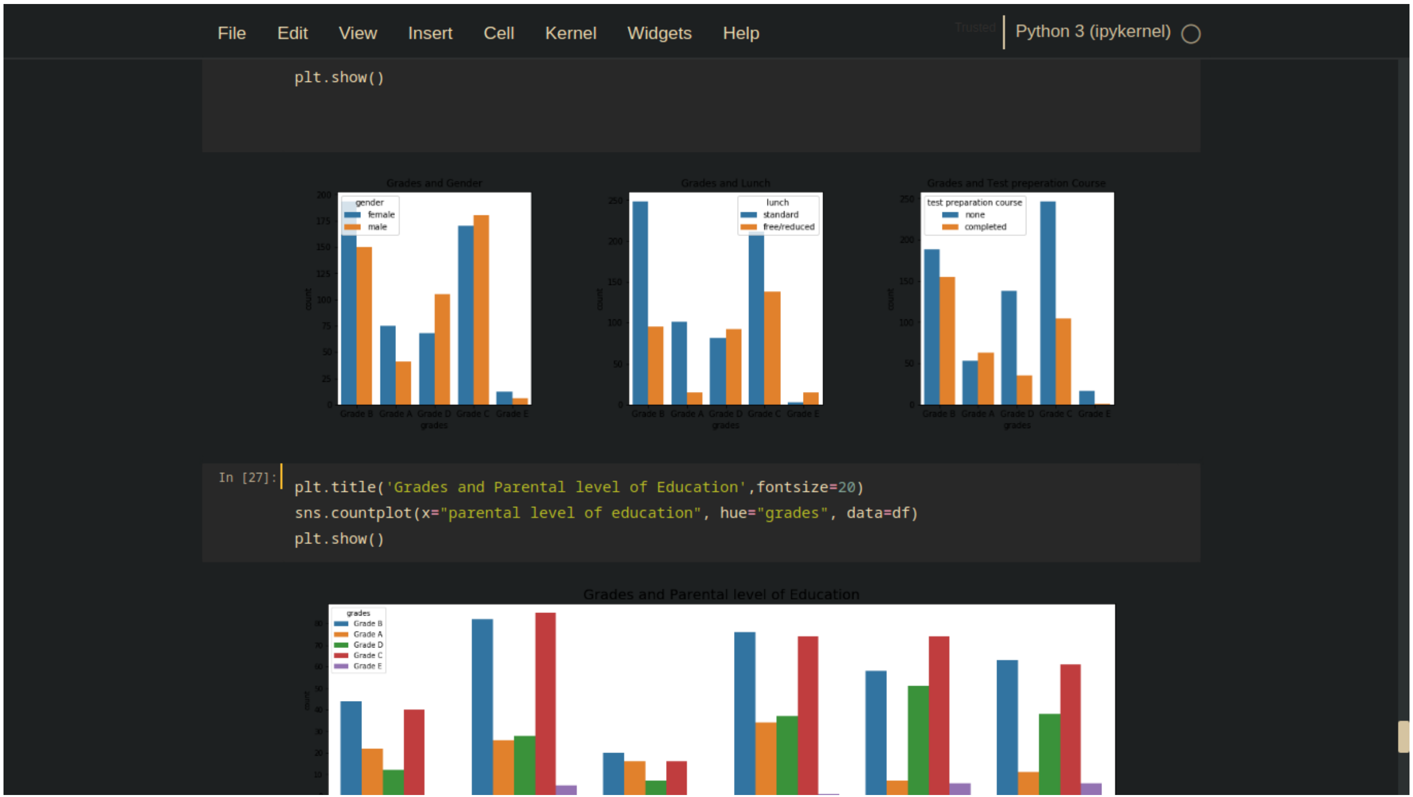
In [25]:

```
df.grades.value_counts().plot.bar()
plt.show()
```

Most of the students got Grade B and Grade C.

In [26]:

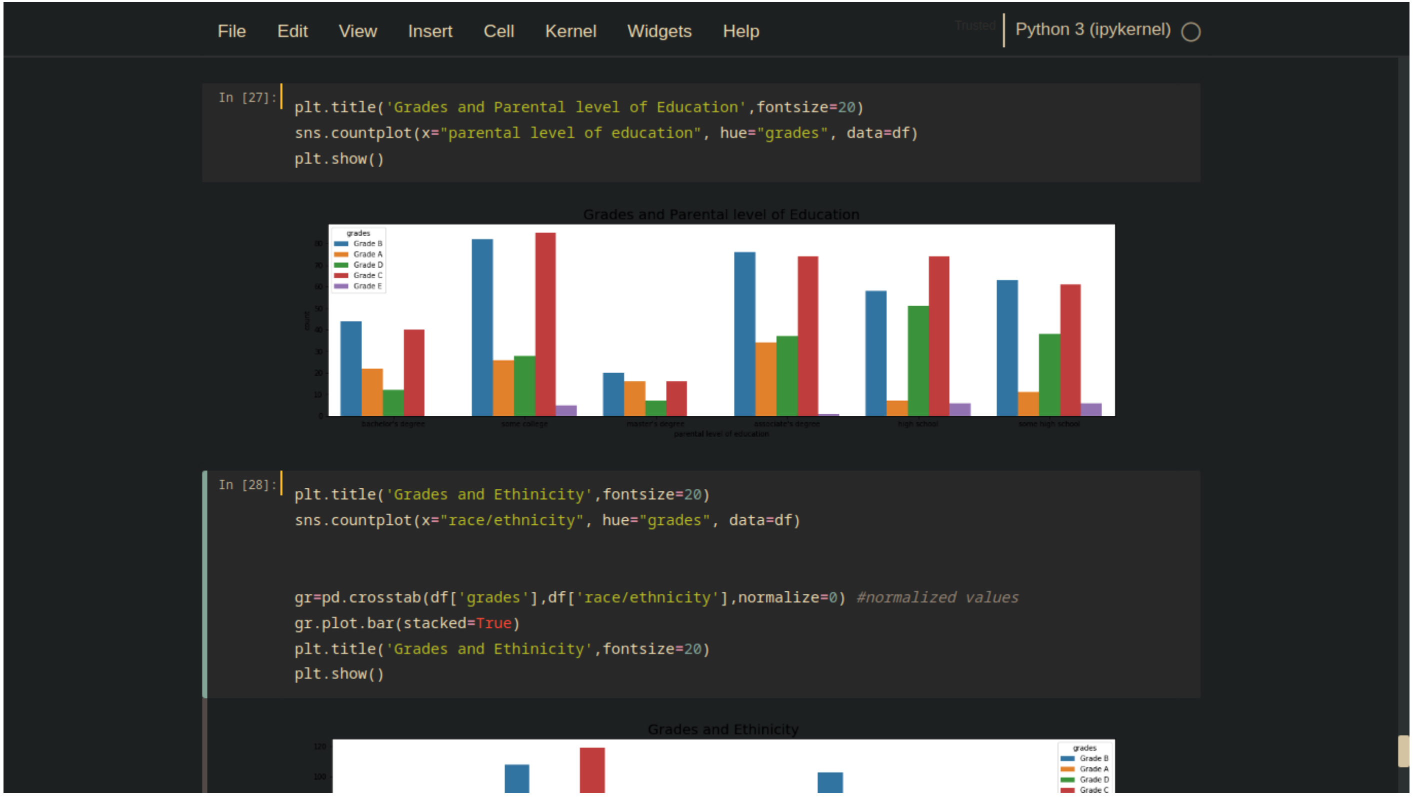
```
plt.figure(figsize=(30,10))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9,
                    wspace=0.5, hspace=0.2)
plt.subplot(251)
plt.title('Grades and Gender')
sns.countplot(hue="gender", x="grades", data=df)

plt.subplot(252)
plt.title('Grades and Lunch')
sns.countplot(hue="lunch", x="grades", data=df)

plt.subplot(253)
```



```
plt.show()
```

In [27]:

```
plt.title('Grades and Parental level of Education',fontsize=20)
sns.countplot(x="parental level of education", hue="grades", data=df)
plt.show()
```

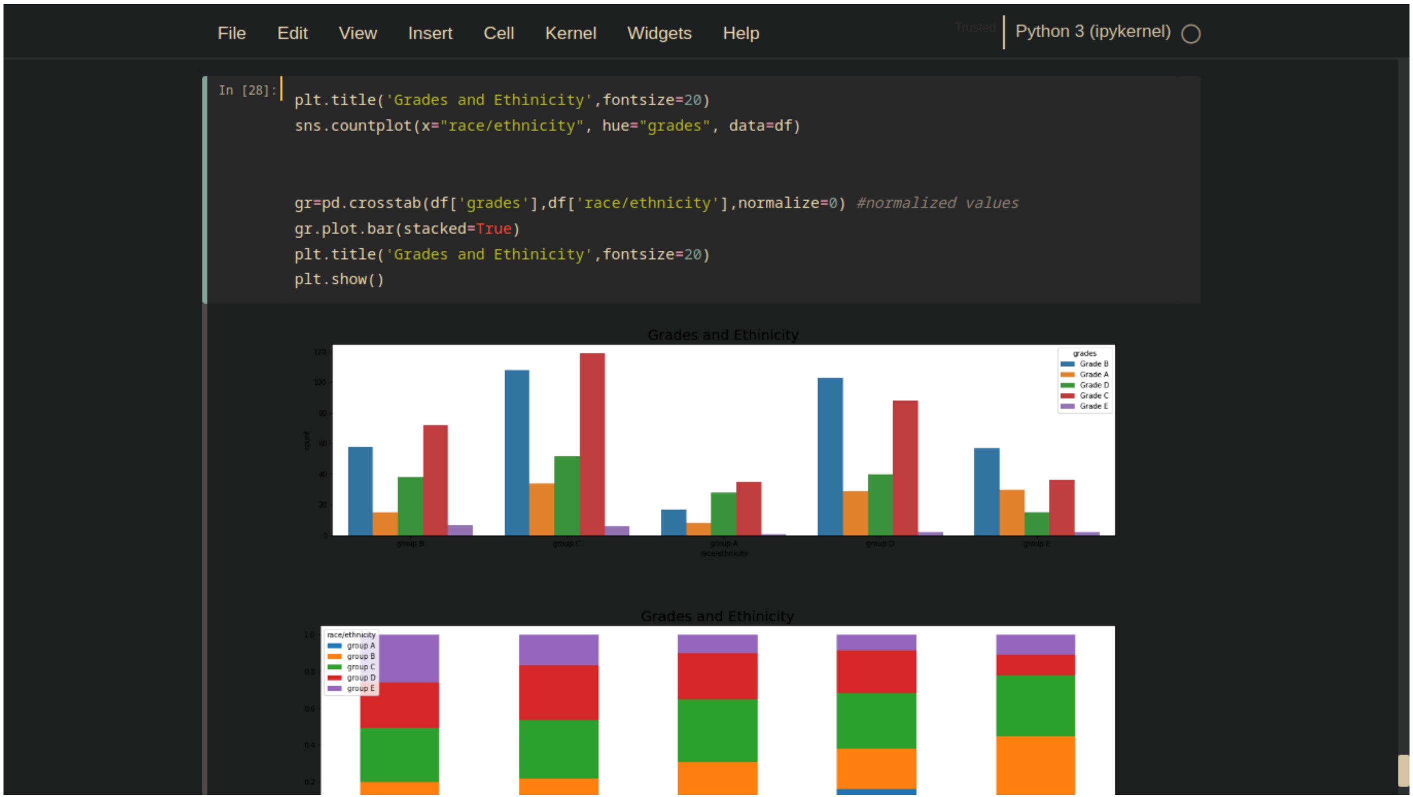


In [27]:

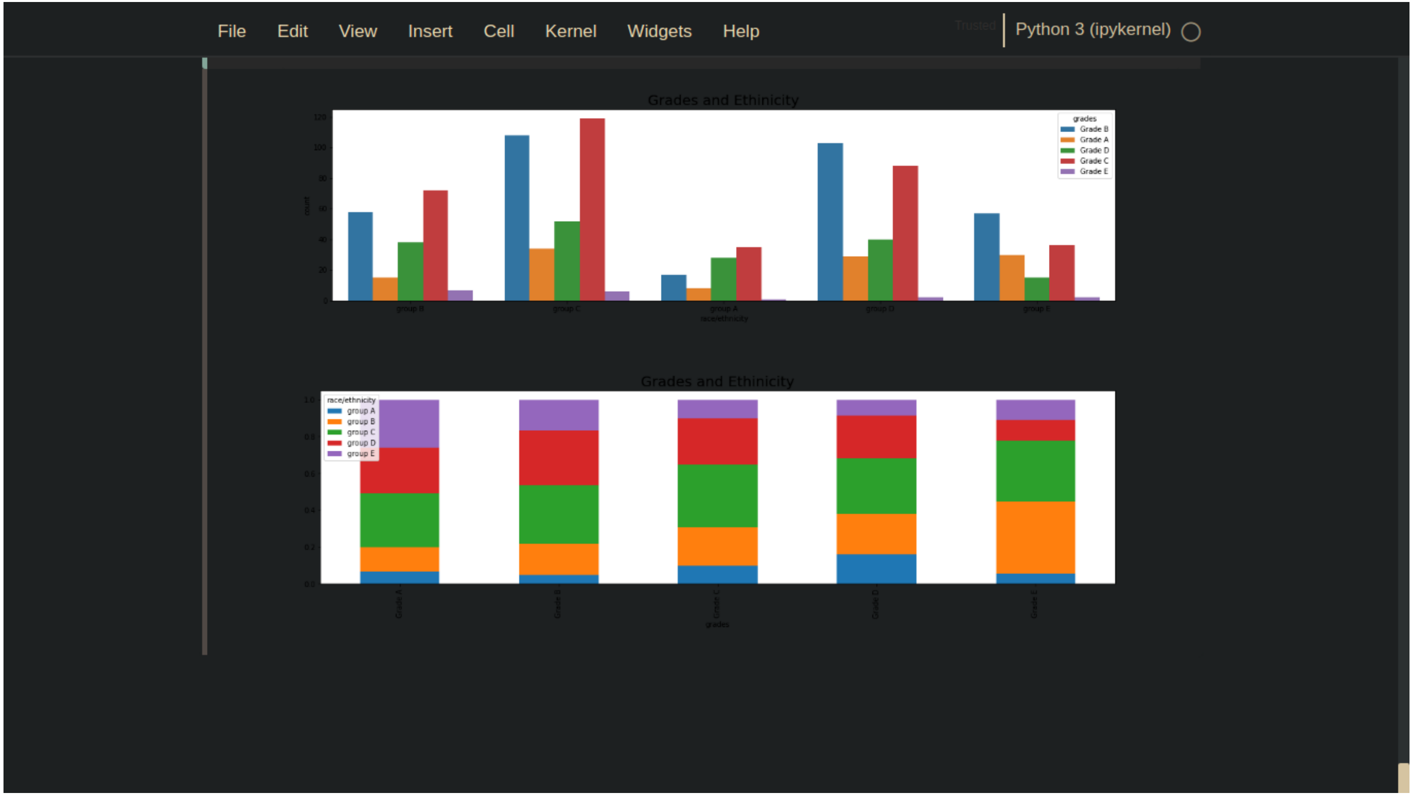
```
plt.title('Grades and Parental level of Education',fontsize=20)
sns.countplot(x="parental level of education", hue="grades", data=df)
plt.show()
```

In [28]:

```
plt.title('Grades and Ethinicity',fontsize=20)
sns.countplot(x="race/ethnicity", hue="grades", data=df)


gr=pd.crosstab(df['grades'],df['race/ethnicity'],normalize=0) #normalized values
gr.plot.bar(stacked=True)
plt.title('Grades and Ethinicity',fontsize=20)
plt.show()
```



```
plt.title('Grades and Ethinicity',fontsize=20)
sns.countplot(x="race/ethnicity", hue="grades", data=df)


gr=pd.crosstab(df['grades'],df['race/ethnicity'],normalize=0) #normalized values
gr.plot.bar(stacked=True)
plt.title('Grades and Ethinicity',fontsize=20)
plt.show()
```

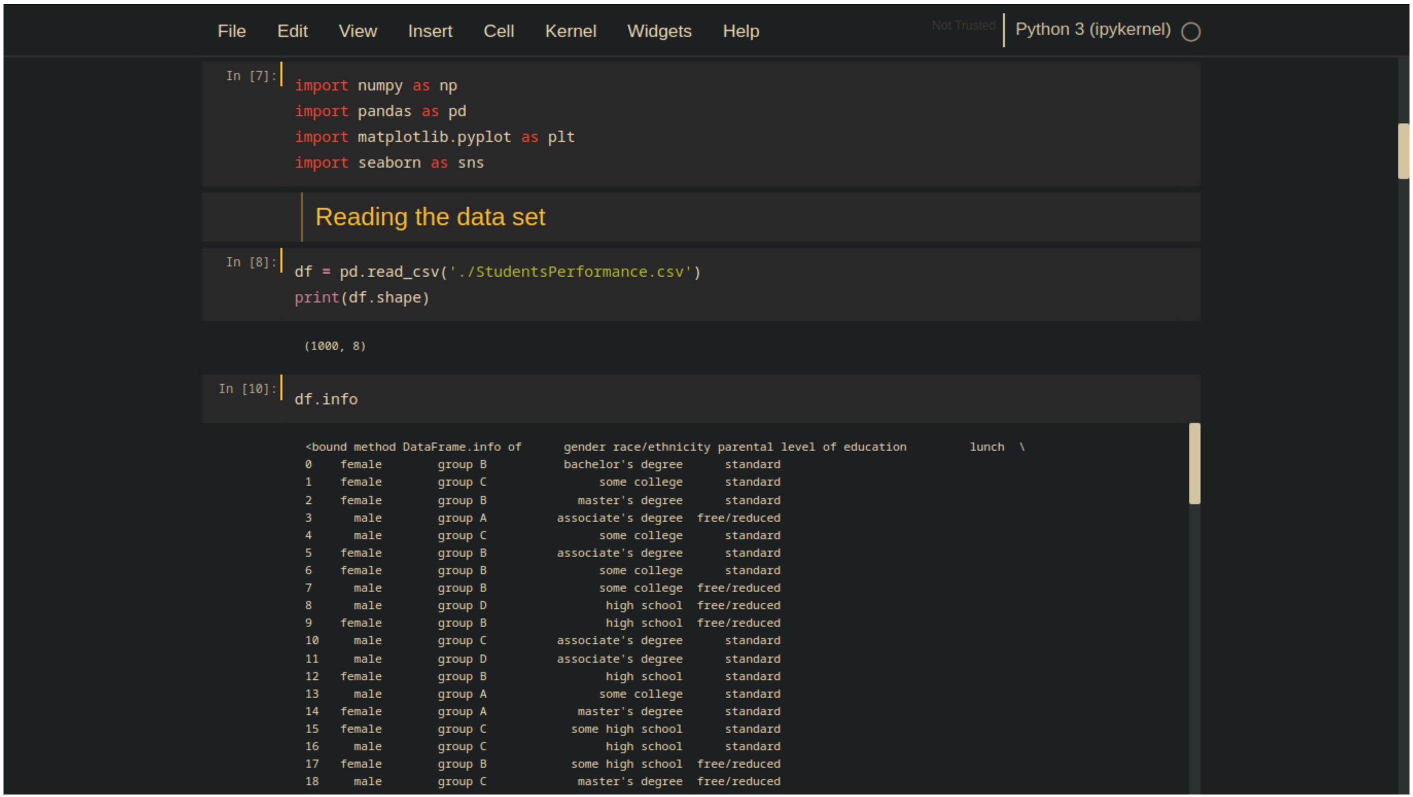


4B. Source Code - B:

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

data = pd.read_csv(”./Marketing_Analysis.csv”)
data.shape
data.info()
data.head()
data.tail()
data
data.describe()
data_new = data.iloc[2::]
data_new.shape
data_new
data_again = pd.read_csv(”./Marketing_Analysis.csv”, skiprows=2)
data_again
data_again.isnull().sum()
data_again.info()
mode = data_again[”age”].mode().values[0]
print(mode)
data_again[”age”] = data_again[”age”].replace(np.nan, mode)
data_again.isnull().sum()
data_again.shape
data_again[”response”].fillna(”no response”, inplace=True)
print(data_again.isnull().sum())
print(data_again.shape)
data_again = data_again.dropna(axis=0, how=”any”)
print(data_again.isnull().sum())
print(data_again.shape)
data_again.head()
data_again.drop[””]
plt.rcParams[”figure.figsize”] = (25, 10)
sns.countplot(data_again[”age”], palette=”bright”)
plt.title(”Age”, fontsize=28)
plt.show()
sns.histplot(
    x=”salary”,
    data=data_again,
)
plt.show()
plt.rcParams[”figure.figsize”] = (25, 5)
```

```
sns.countplot(data_again["salary"], palette="bright")
plt.title("Salary", fontsize=20)
plt.show()
sns.boxplot(data_again["pdays"])
sns.boxplot(data_again["balance"])
```

5B. Observations - B:

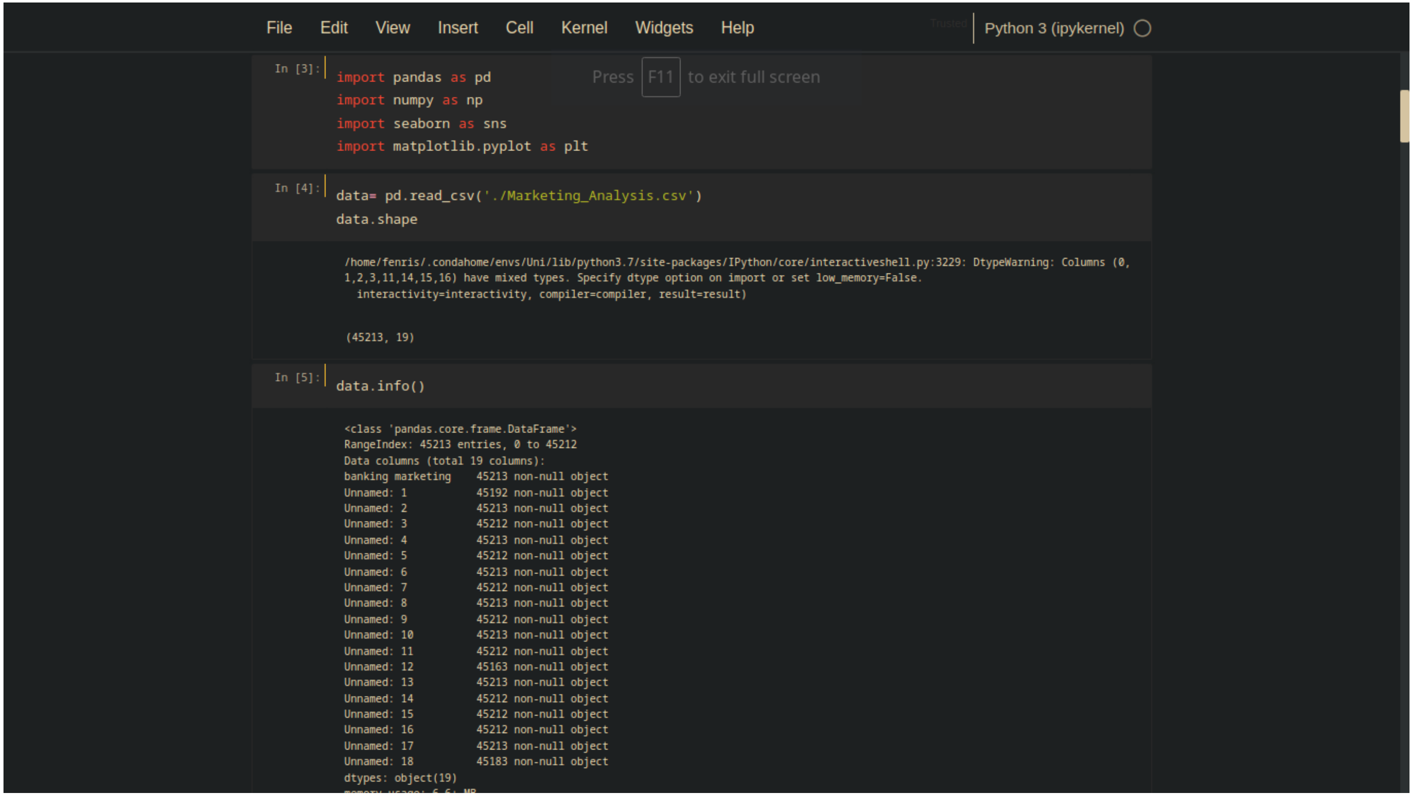


```
In [7]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
```

Reading the data set

```
In [8]: df = pd.read_csv('./StudentsPerformance.csv')
        print(df.shape)

(1000, 8)
```

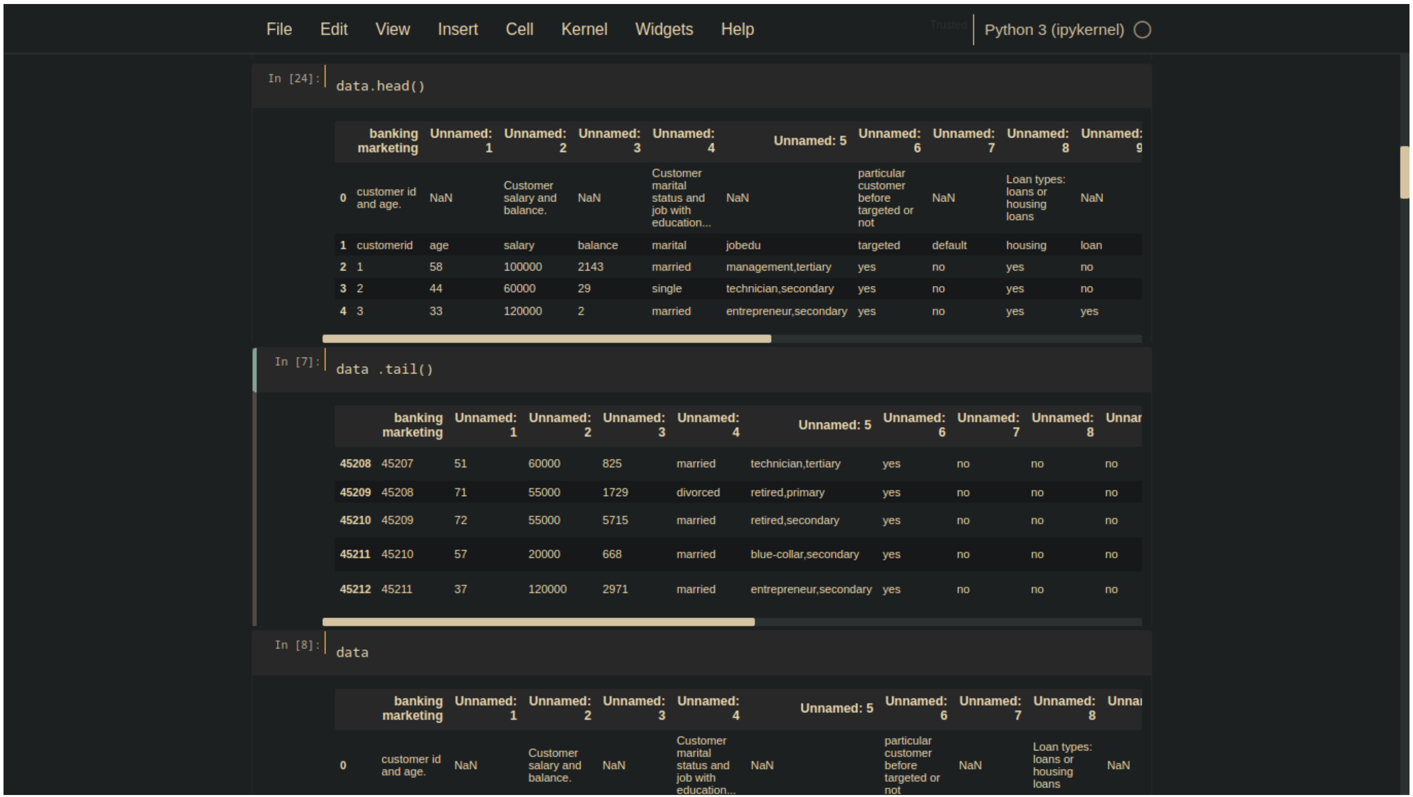
```
In [10]: df.info

<bound method DataFrame.info of      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
5    female        group B          associate's degree      standard
6    female        group B                some college      standard
7      male        group B                some college  free/reduced
8      male        group D                 high school  free/reduced
9    female        group B                 high school  free/reduced
10     male        group C          associate's degree      standard
11     male        group D          associate's degree      standard
12   female        group B                 high school      standard
13     male        group A                some college      standard
14   female        group A             master's degree      standard
15   female        group C            some high school      standard
16     male        group C                 high school      standard
17   female        group B            some high school  free/reduced
18     male        group C             master's degree  free/reduced
```



```
In [3]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
```

```
In [4]: data= pd.read_csv('./Marketing_Analysis.csv')
        data.shape

/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/IPython/core/interactiveshell.py:3229: DtypeWarning: Columns (0,
1,2,3,11,14,15,16) have mixed types. Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)


(45213, 19)
```

```
In [5]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45213 entries, 0 to 45212
Data columns (total 19 columns):
banking marketing    45213 non-null object
Unnamed: 1           45192 non-null object
Unnamed: 2           45213 non-null object
Unnamed: 3           45212 non-null object
Unnamed: 4           45213 non-null object
Unnamed: 5           45212 non-null object
Unnamed: 6           45213 non-null object
Unnamed: 7           45212 non-null object
Unnamed: 8           45213 non-null object
Unnamed: 9           45212 non-null object
Unnamed: 10          45213 non-null object
Unnamed: 11          45212 non-null object
Unnamed: 12          45163 non-null object
Unnamed: 13          45213 non-null object
Unnamed: 14          45212 non-null object
Unnamed: 15          45212 non-null object
Unnamed: 16          45212 non-null object
Unnamed: 17          45213 non-null object
Unnamed: 18          45183 non-null object
dtypes: object(19)
memory usage: 6.6+ MB
```



In [24]:
```
data.head()
```

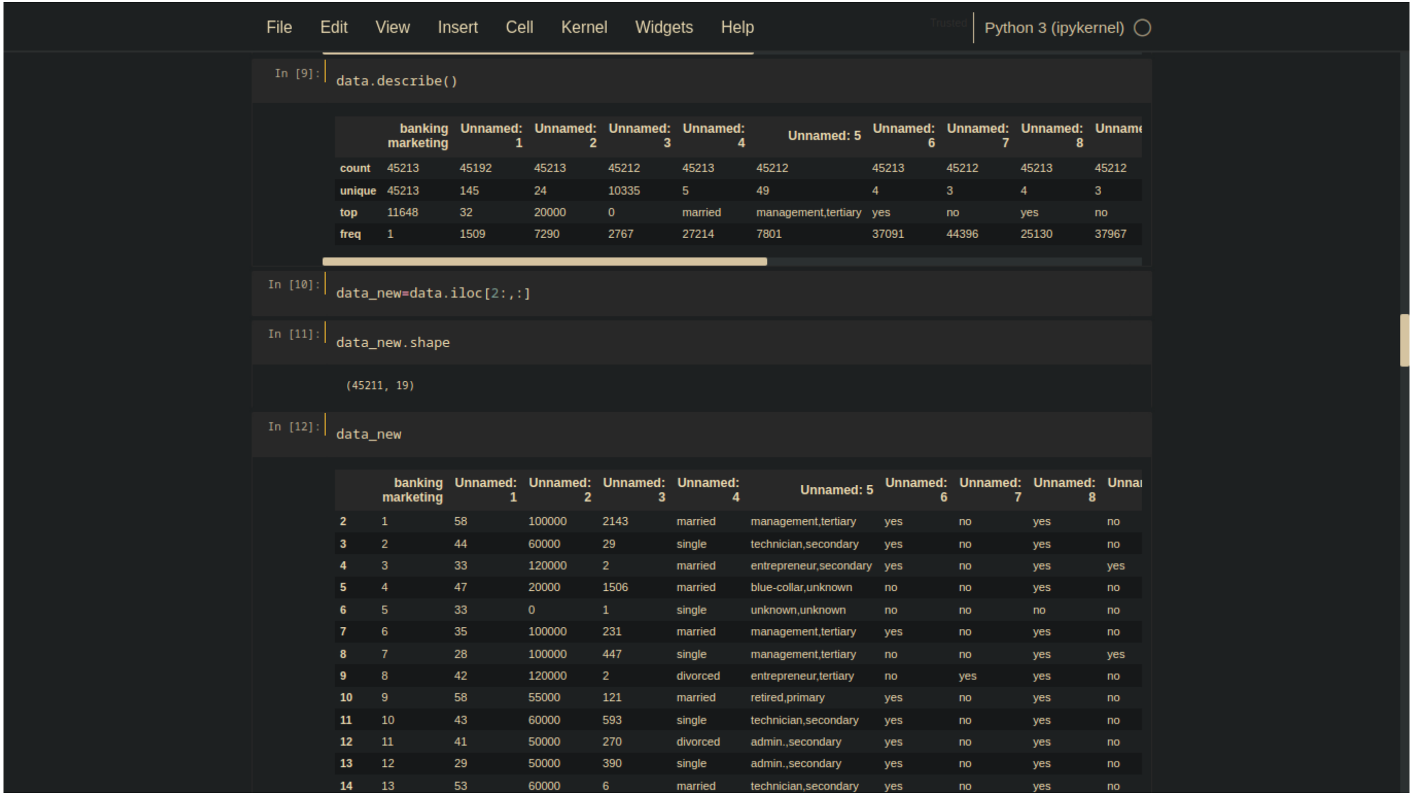
	banking marketing	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9
0	customer id and age.	NaN	Customer salary and balance.	NaN	Customer marital status and job with education...	NaN	particular customer before targeted or not	NaN	Loan types: loans or housing loans	NaN
1	customerid	age	salary	balance	marital	jobedu	targeted	default	housing	loan
2	1	58	100000	2143	married	management,tertiary	yes	no	yes	no
3	2	44	60000	29	single	technician,secondary	yes	no	yes	no
4	3	33	120000	2	married	entrepreneur,secondary	yes	no	yes	yes

In [7]:
```
data .tail()
```

	banking marketing	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnar
45208	45207	51	60000	825	married	technician,tertiary	yes	no	no	no
45209	45208	71	55000	1729	divorced	retired,primary	yes	no	no	no
45210	45209	72	55000	5715	married	retired,secondary	yes	no	no	no
45211	45210	57	20000	668	married	blue-collar,secondary	yes	no	no	no
45212	45211	37	120000	2971	married	entrepreneur,secondary	yes	no	no	no

In [8]:
```
data
```

	banking marketing	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnai
0	customer id and age.	NaN	Customer salary and balance.	NaN	Customer marital status and job with education...	NaN	particular customer before targeted or not	NaN	Loan types: loans or housing loans	NaN



In [9]:
```
data.describe()
```

	banking marketing	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unname
count	45213	45192	45213	45212	45213	45212	45213	45212	45213	45212
unique	45213	145	24	10335	5	49	4	3	4	3
top	11648	32	20000	0	married	management,tertiary	yes	no	yes	no
freq	1	1509	7290	2767	27214	7801	37091	44396	25130	37967

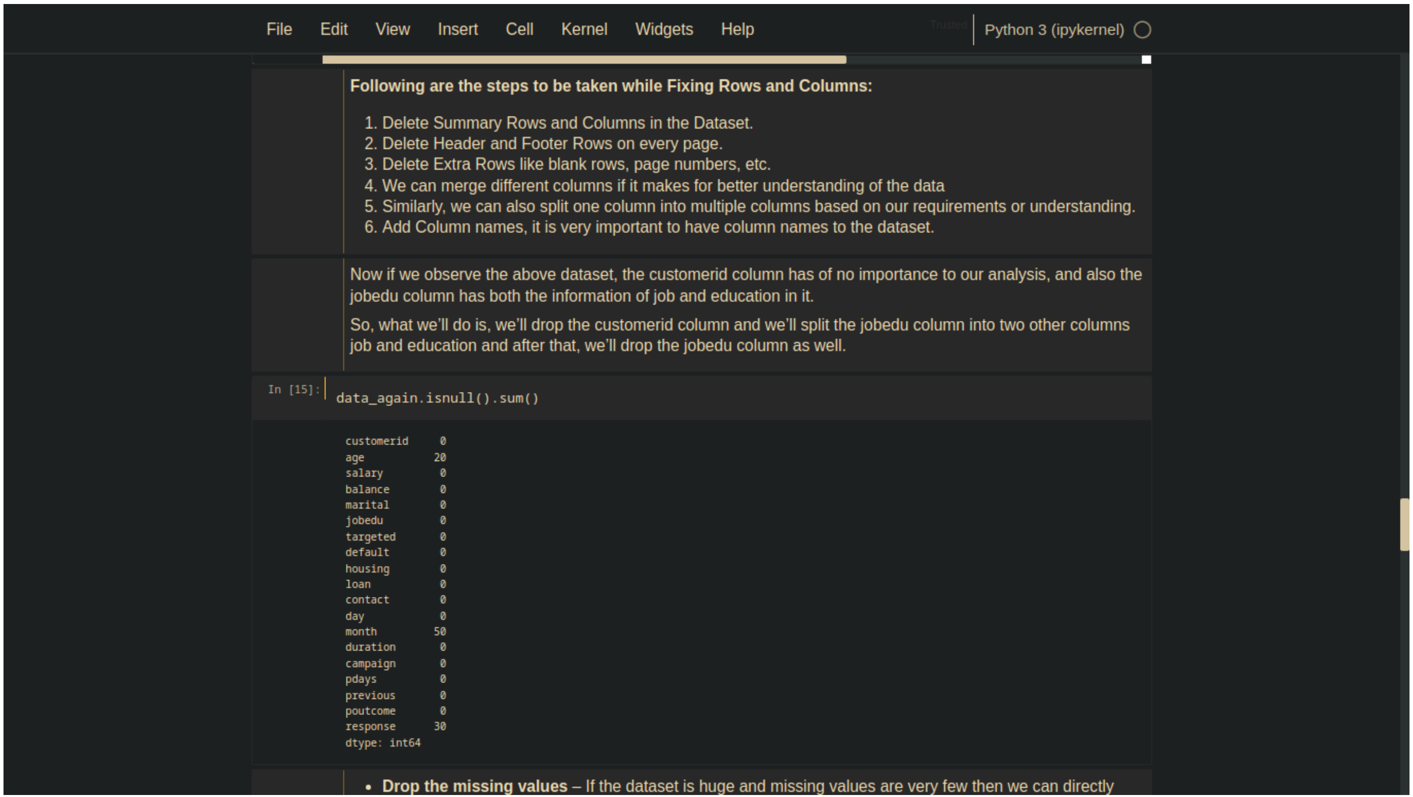
In [10]:
```
data_new=data.iloc[2:,:]
```

In [11]:
```
data_new.shape
```

```
(45211, 19)
```

In [12]:
```
data_new
```

	banking marketing	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnai
2	1	58	100000	2143	married	management,tertiary	yes	no	yes	no
3	2	44	60000	29	single	technician,secondary	yes	no	yes	no
4	3	33	120000	2	married	entrepreneur,secondary	yes	no	yes	yes
5	4	47	20000	1506	married	blue-collar,unknown	no	no	yes	no
6	5	33	0	1	single	unknown,unknown	no	no	no	no
7	6	35	100000	231	married	management,tertiary	yes	no	yes	no
8	7	28	100000	447	single	management,tertiary	no	no	yes	yes
9	8	42	120000	2	divorced	entrepreneur,tertiary	no	yes	yes	no
10	9	58	55000	121	married	retired,primary	yes	no	yes	no
11	10	43	60000	593	single	technician,secondary	yes	no	yes	no
12	11	41	50000	270	divorced	admin.,secondary	yes	no	yes	no
13	12	29	50000	390	single	admin.,secondary	yes	no	yes	no
14	13	53	60000	6	married	technician,secondary	yes	no	yes	no



Following are the steps to be taken while Fixing Rows and Columns:

1. Delete Summary Rows and Columns in the Dataset.
2. Delete Header and Footer Rows on every page.
3. Delete Extra Rows like blank rows, page numbers, etc.
4. We can merge different columns if it makes for better understanding of the data
5. Similarly, we can also split one column into multiple columns based on our requirements or understanding.
6. Add Column names, it is very important to have column names to the dataset.

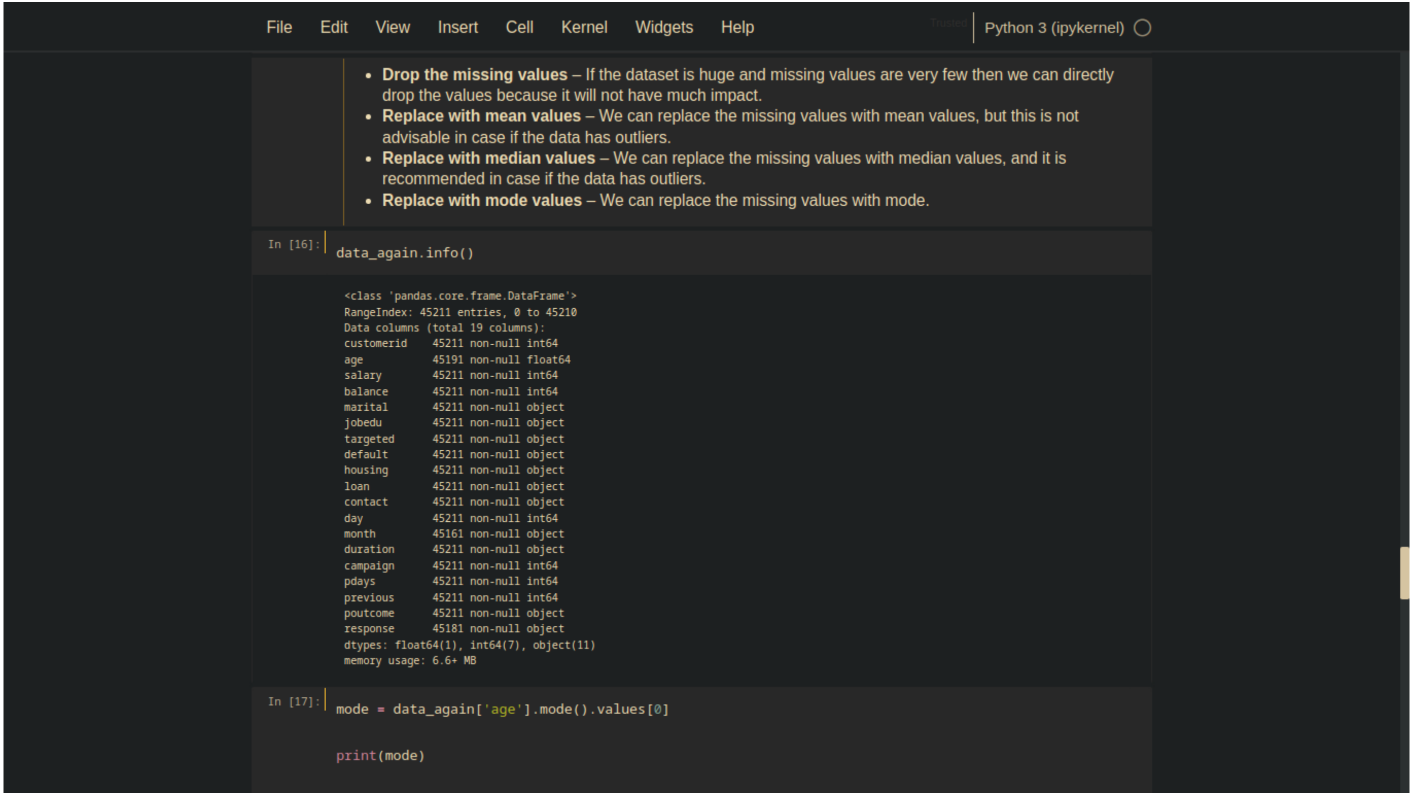
Now if we observe the above dataset, the customerid column has of no importance to our analysis, and also the jobedu column has both the information of job and education in it.

So, what we'll do is, we'll drop the customerid column and we'll split the jobedu column into two other columns job and education and after that, we'll drop the jobedu column as well.

In [15]:
```
data_again.isnull().sum()
```

```
customerid     0
age           20
salary         0
balance        0
marital        0
jobedu         0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response      30
dtype: int64
```

- **Drop the missing values** – If the dataset is huge and missing values are very few then we can directly

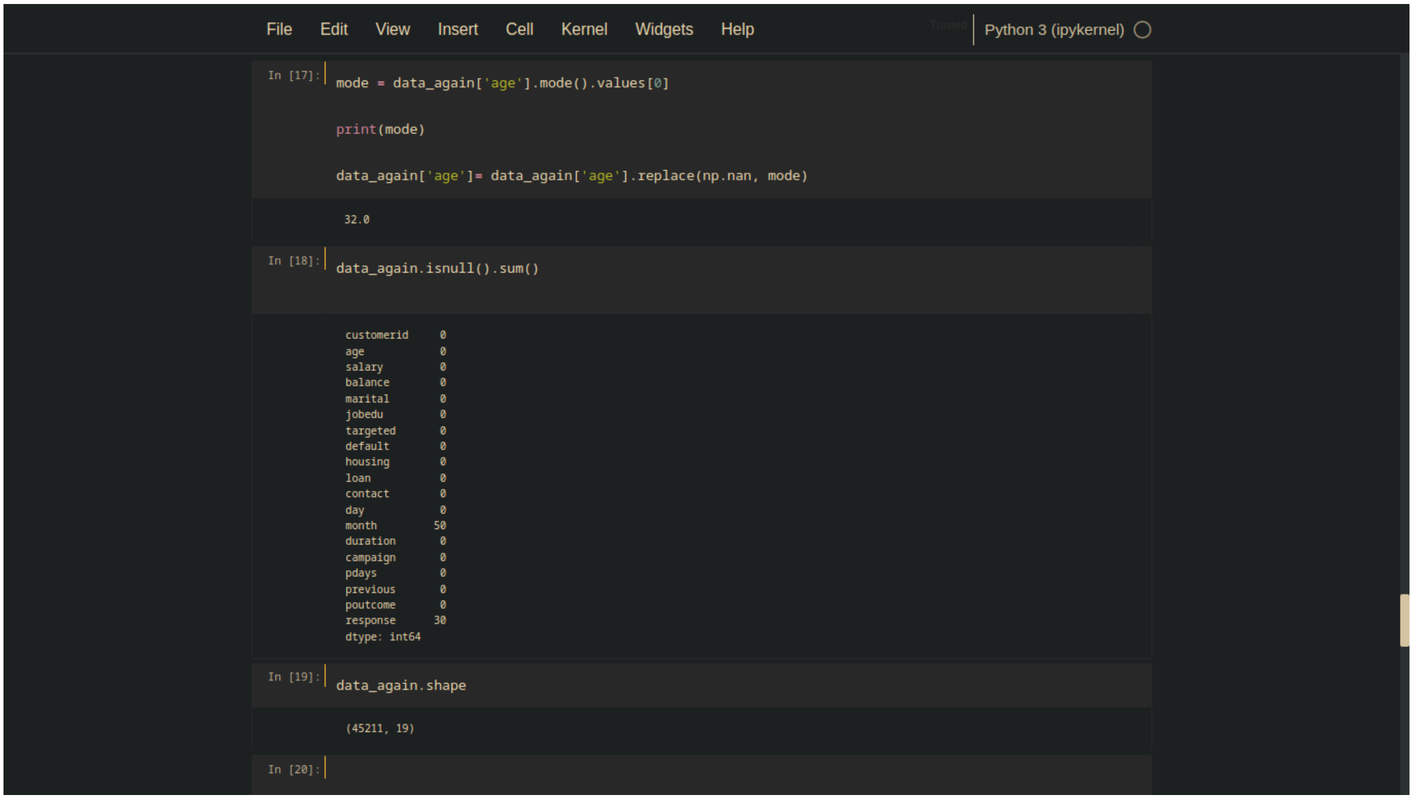


- **Drop the missing values** – If the dataset is huge and missing values are very few then we can directly drop the values because it will not have much impact.
- **Replace with mean values** – We can replace the missing values with mean values, but this is not advisable in case if the data has outliers.
- **Replace with median values** – We can replace the missing values with median values, and it is recommended in case if the data has outliers.
- **Replace with mode values** – We can replace the missing values with mode.

In [16]:
```
data_again.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 19 columns):
customerid    45211 non-null int64
age           45191 non-null float64
salary        45211 non-null int64
balance       45211 non-null int64
marital       45211 non-null object
jobedu        45211 non-null object
targeted      45211 non-null object
default       45211 non-null object
housing       45211 non-null object
loan          45211 non-null object
contact       45211 non-null object
day           45211 non-null int64
month         45161 non-null object
duration      45211 non-null object
campaign      45211 non-null int64
pdays         45211 non-null int64
previous      45211 non-null int64
poutcome      45211 non-null object
response      45181 non-null object
dtypes: float64(1), int64(7), object(11)
memory usage: 6.6+ MB
```

In [17]:
```
mode = data_again['age'].mode().values[0]

print(mode)
```



In [17]:
```
mode = data_again['age'].mode().values[0]

print(mode)

data_again['age']= data_again['age'].replace(np.nan, mode)
```

```
32.0
```

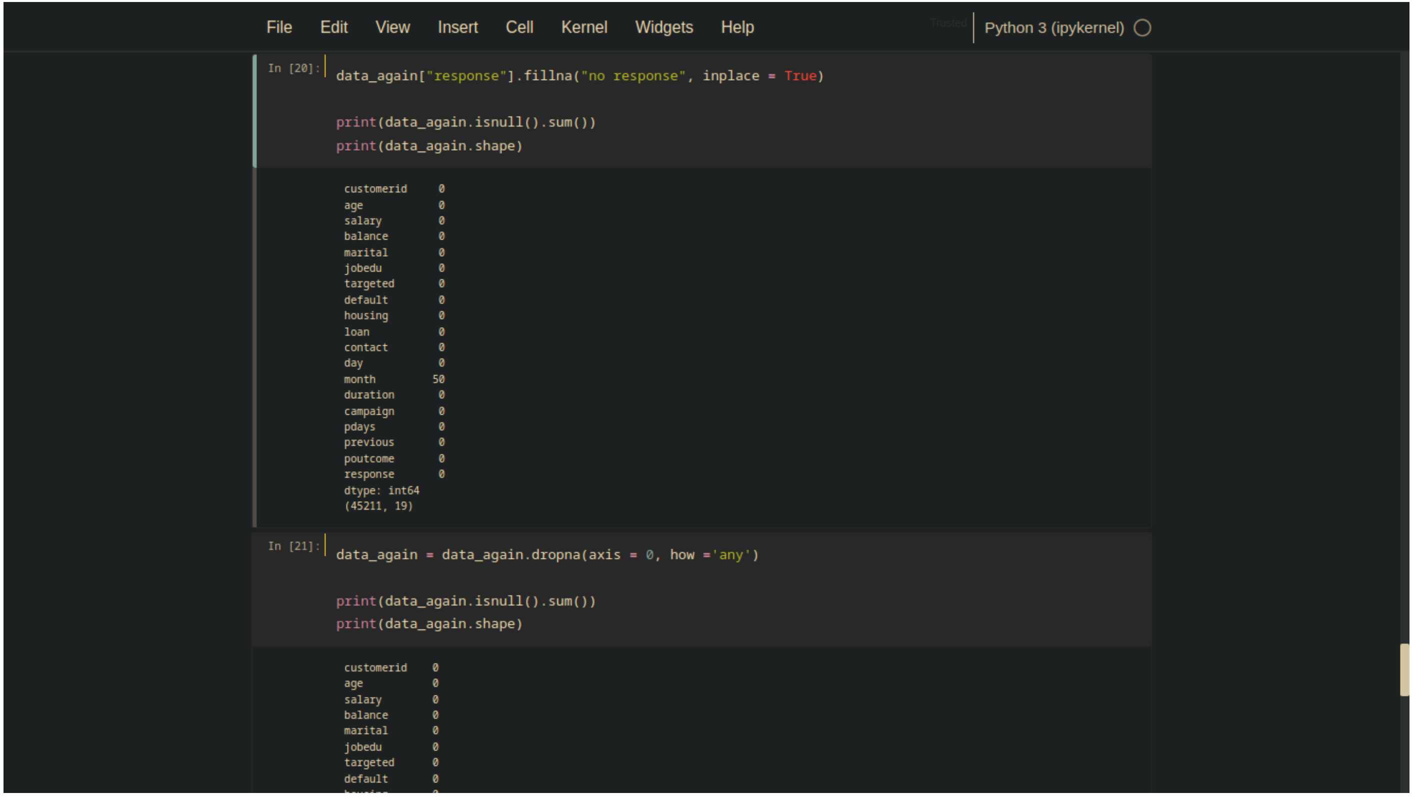
In [18]:
```
data_again.isnull().sum()
```

```
customerid     0
age            0
salary         0
balance        0
marital        0
jobedu         0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response      30
dtype: int64
```

In [19]:
```
data_again.shape
```

```
(45211, 19)
```

In [20]:

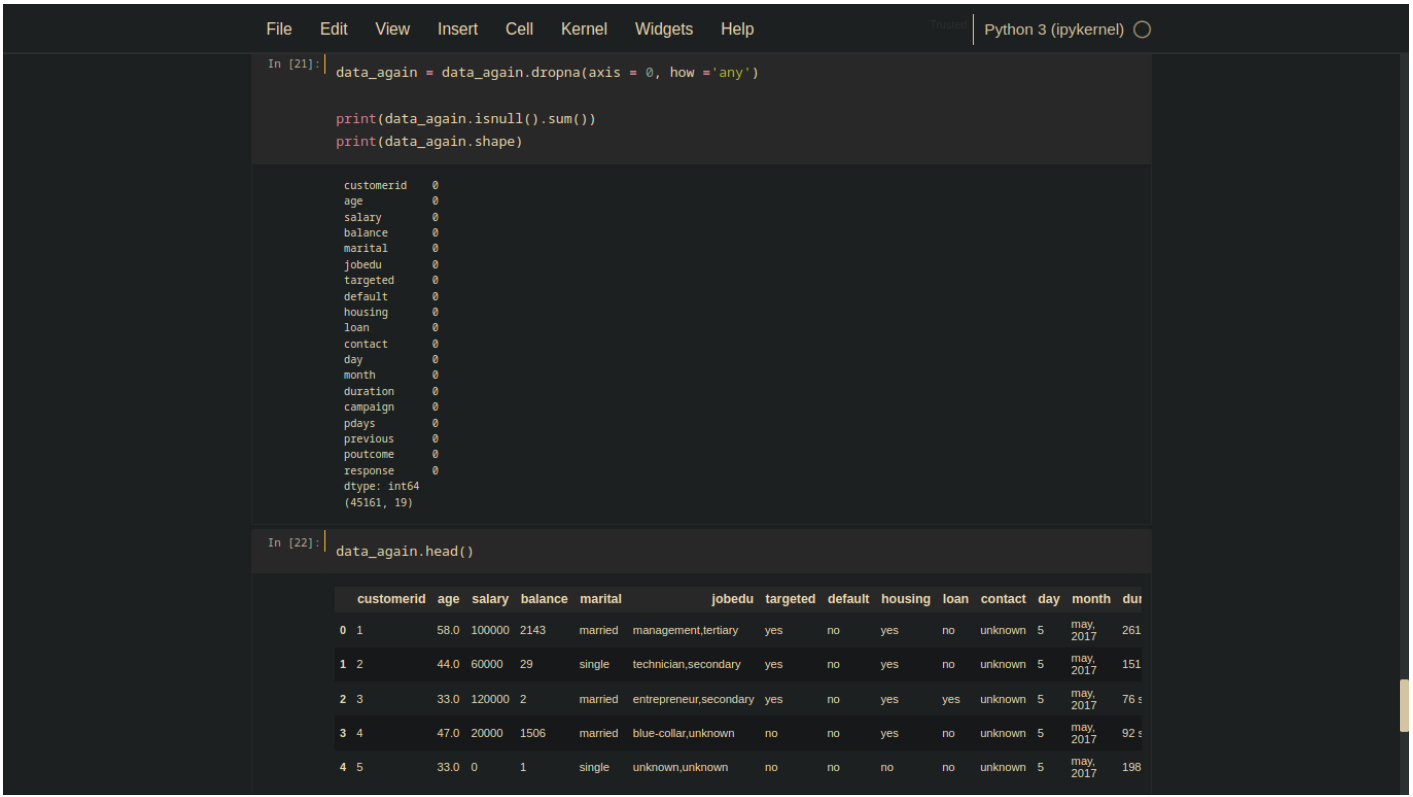


In [20]:
```
data_again["response"].fillna("no response", inplace = True)

print(data_again.isnull().sum())
print(data_again.shape)
```

```
customerid     0
age            0
salary         0
balance        0
marital        0
jobedu         0
targeted       0
default        0
housing        0
loan           0
contact        0
day            0
month         50
duration       0
campaign       0
pdays          0
previous       0
poutcome       0
response       0
dtype: int64
(45211, 19)
```

In [21]:
```
data_again = data_again.dropna(axis = 0, how ='any')

print(data_again.isnull().sum())
print(data_again.shape)
```

```
customerid    0
age           0
salary        0
balance       0
marital       0
jobedu        0
targeted      0
default       0
```



In [21]:

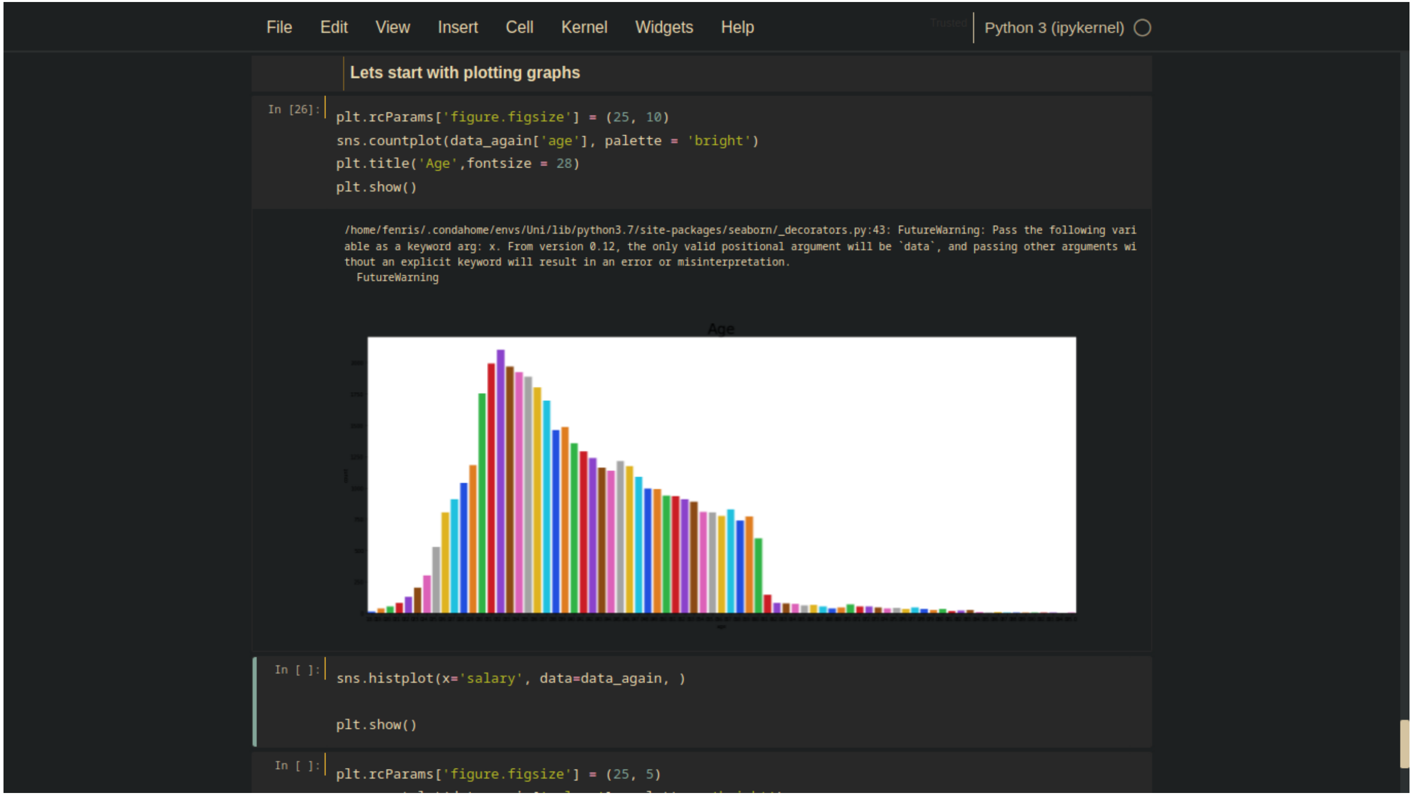
```
data_again = data_again.dropna(axis = 0, how ='any')

print(data_again.isnull().sum())
print(data_again.shape)
```

```
customerid    0
age           0
salary        0
balance       0
marital       0
jobedu        0
targeted      0
default       0
housing       0
loan          0
contact       0
day           0
month         0
duration      0
campaign      0
pdays         0
previous      0
poutcome      0
response      0
dtype: int64
(45161, 19)
```

In [22]:

```
data_again.head()
```

	customerid	age	salary	balance	marital	jobedu	targeted	default	housing	loan	contact	day	month	dur
0	1	58.0	100000	2143	married	management,tertiary	yes	no	yes	no	unknown	5	may, 2017	261
1	2	44.0	60000	29	single	technician,secondary	yes	no	yes	no	unknown	5	may, 2017	151
2	3	33.0	120000	2	married	entrepreneur,secondary	yes	no	yes	yes	unknown	5	may, 2017	76 s
3	4	47.0	20000	1506	married	blue-collar,unknown	no	no	yes	no	unknown	5	may, 2017	92 s
4	5	33.0	0	1	single	unknown,unknown	no	no	no	no	unknown	5	may, 2017	198



Lets start with plotting graphs

In [26]:

```
plt.rcParams['figure.figsize'] = (25, 10)
sns.countplot(data_again['age'], palette = 'bright')
plt.title('Age',fontsize = 28)
plt.show()
```

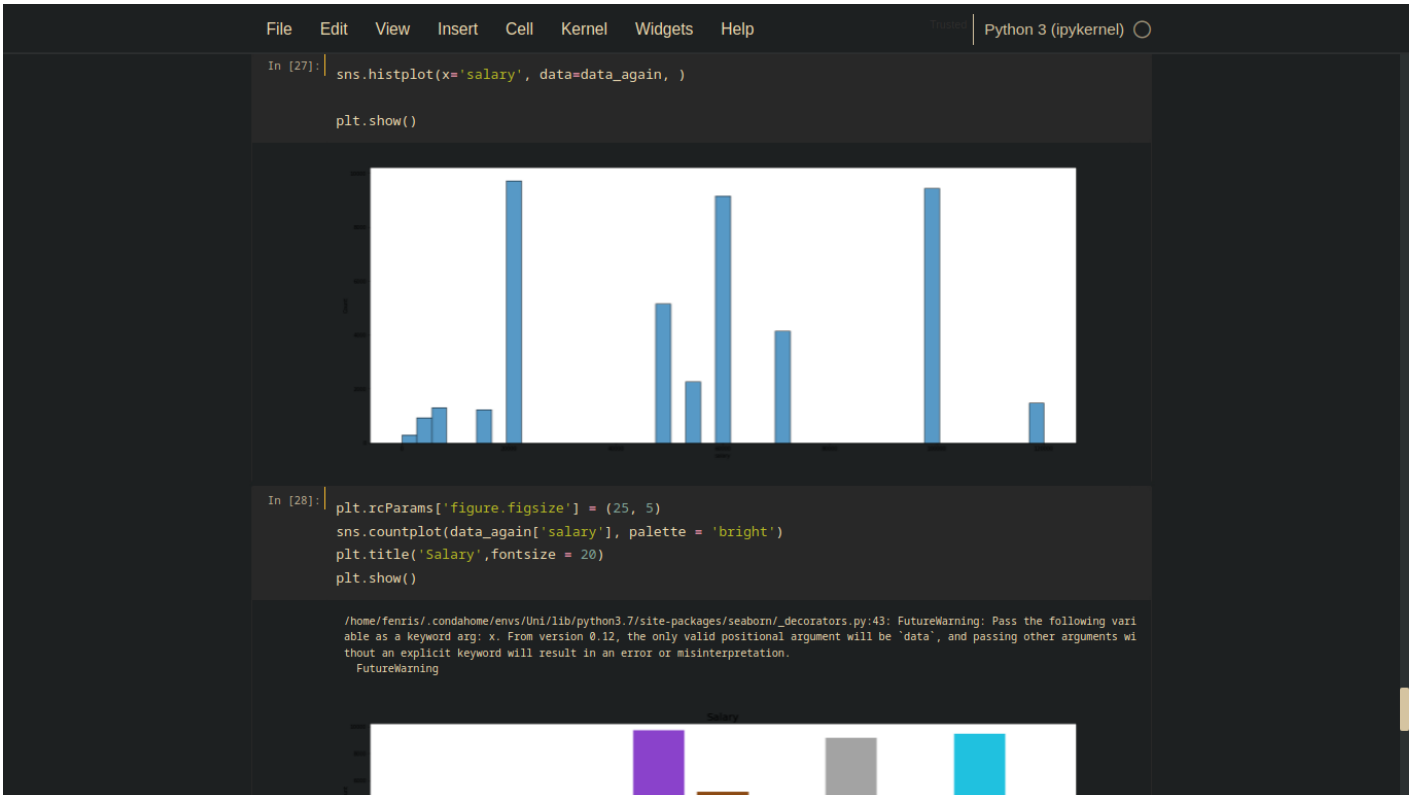
```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

In [ ]:

```
sns.histplot(x='salary', data=data_again, )

plt.show()
```

In [ ]:

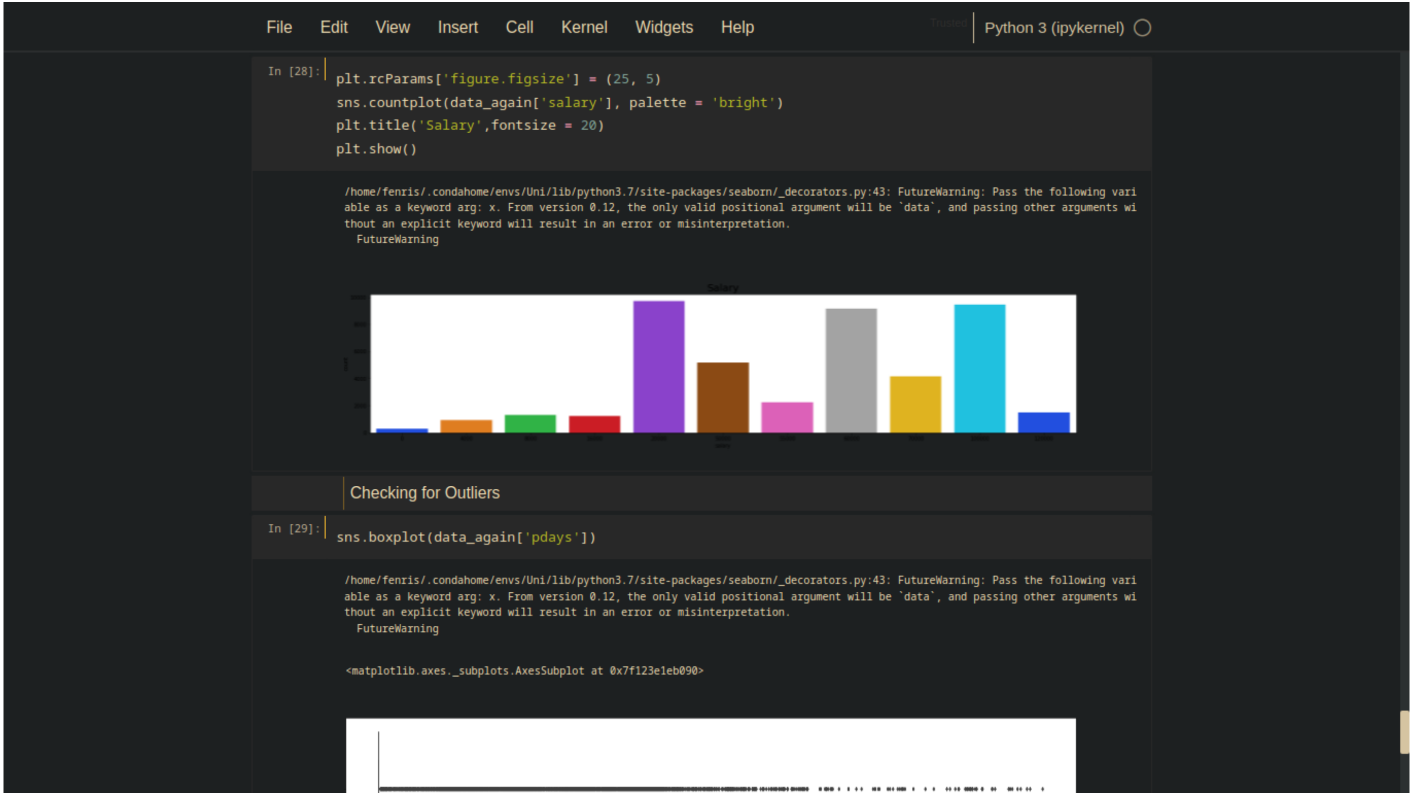
```
plt.rcParams['figure.figsize'] = (25, 5)
```



```
In [27]: sns.histplot(x='salary', data=data_again, )

         plt.show()
```

```
In [28]: plt.rcParams['figure.figsize'] = (25, 5)
         sns.countplot(data_again['salary'], palette = 'bright')
         plt.title('Salary',fontsize = 20)
         plt.show()
```

```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



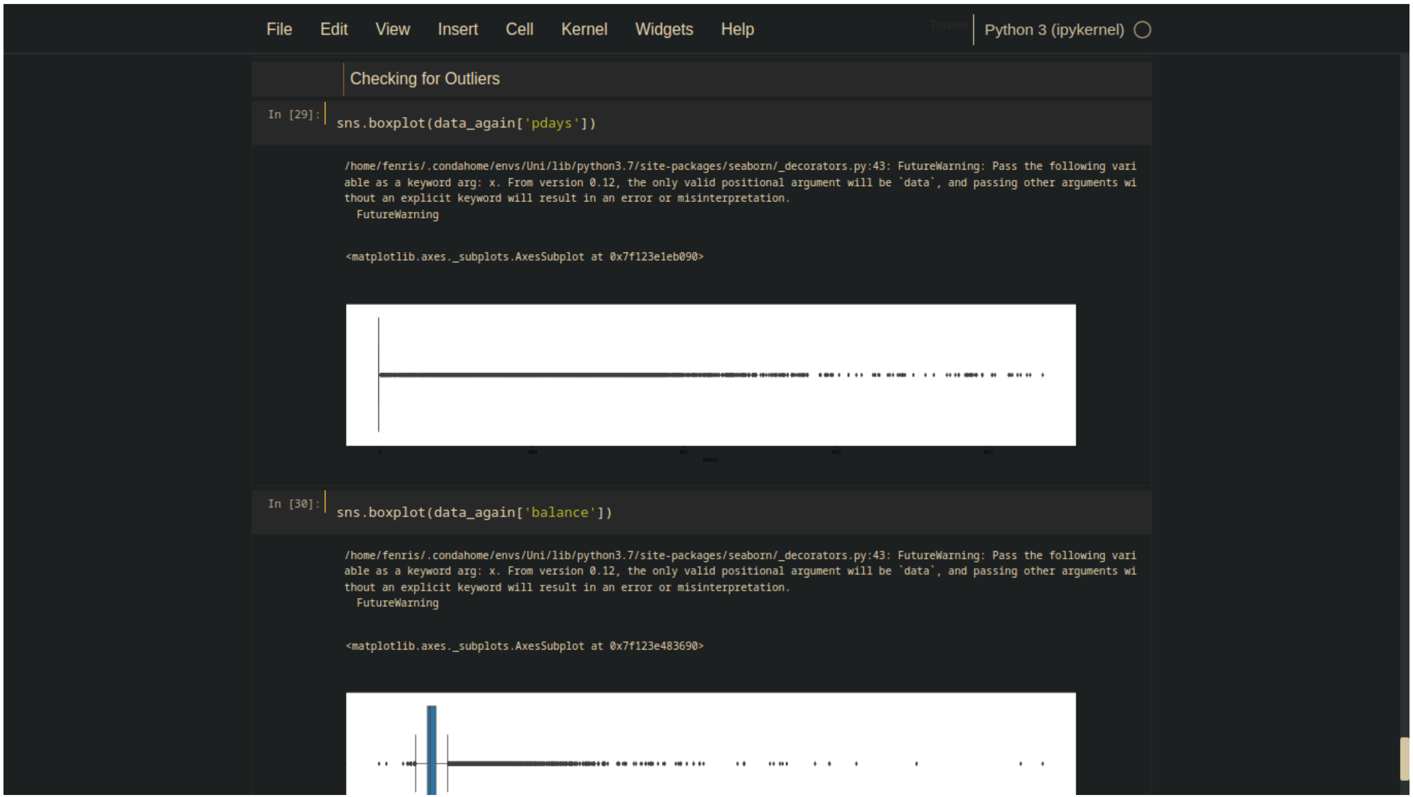
```
In [28]: plt.rcParams['figure.figsize'] = (25, 5)
         sns.countplot(data_again['salary'], palette = 'bright')
         plt.title('Salary',fontsize = 20)
         plt.show()
```

```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Checking for Outliers

```
In [29]: sns.boxplot(data_again['pdays'])
```

```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7f123e1eb090>
```



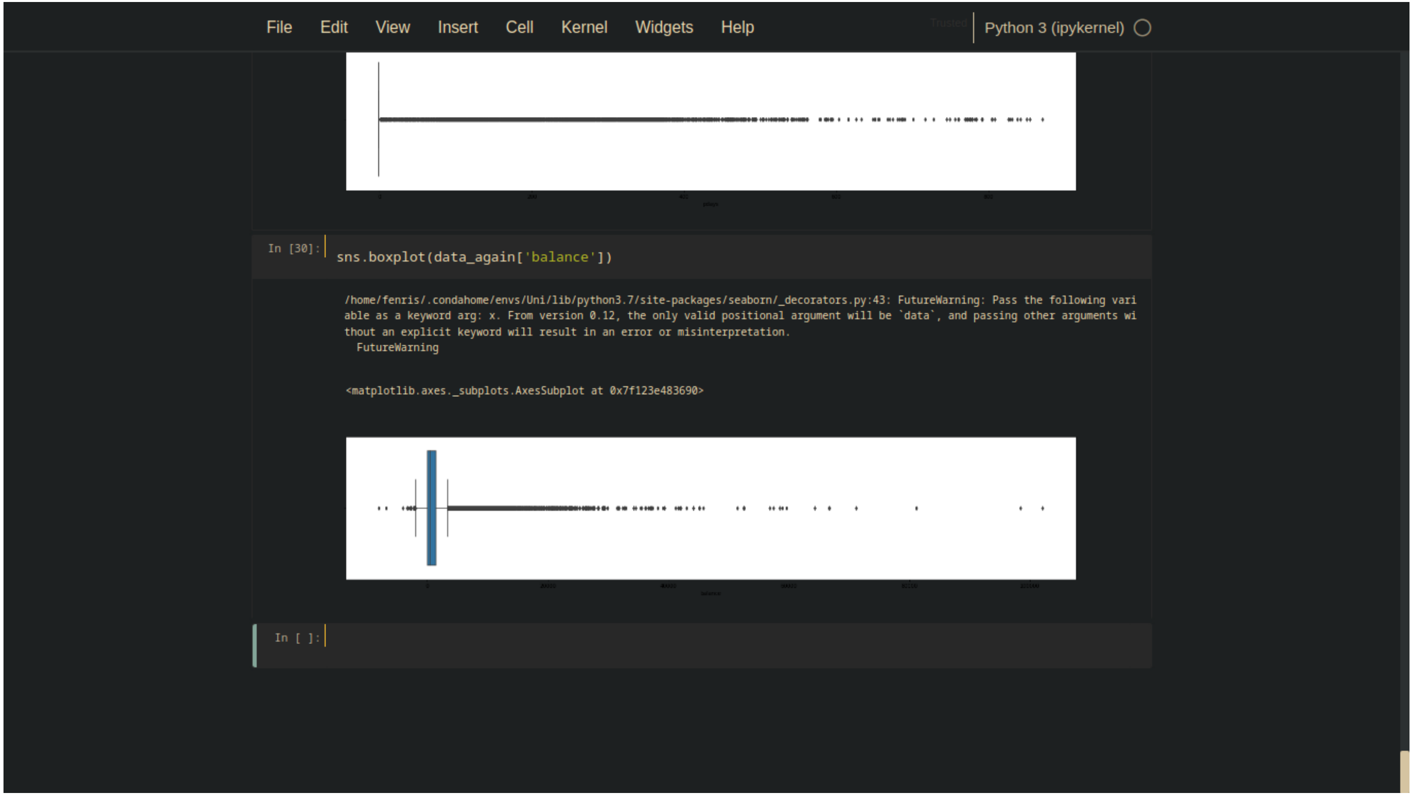
Checking for Outliers

```
In [29]: sns.boxplot(data_again['pdays'])
```

```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7f123e1eb090>
```

```
In [30]: sns.boxplot(data_again['balance'])
```

```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7f123e483690>
```



```
In [30]: sns.boxplot(data_again['balance'])
```

```
/home/fenris/.condahome/envs/Uni/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following vari
able as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments wi
thout an explicit keyword will result in an error or misinterpretation.
  FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7f123e483690>
```

```
In [ ]:
```

Learning Outcomes :

- Learnt to do EDA analysis on dataset
- Learnt to remove null and duplicates from dataset
- Learnt to drop rows in dataset
- Learnt to load dataset

S. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			