

- For solving the problem, I am using Pandas for reading and manipulating data and using matplotlib for creating graph
- To read csv file iam using read_csv() provided by pandas library
- Deleted the City Locations which are not available using dropna() method

jupyter 1st Last Checkpoint: 20 hours ago (autosaved)



Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted

Python 3 (ipykernel) ○

Run Stop Restart Code

```
In [1]: import pandas as pd
import matplotlib.pyplot as plt
```

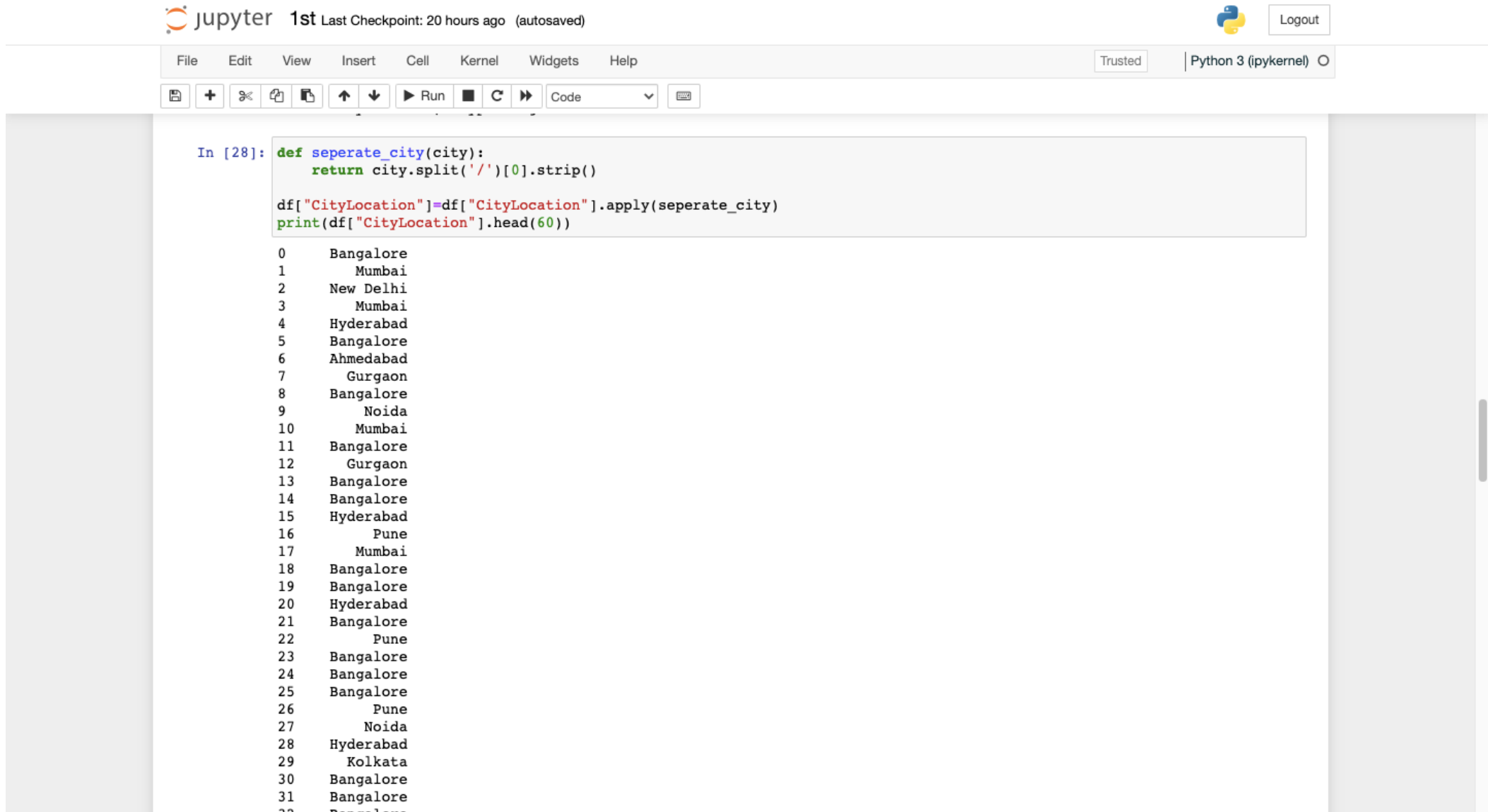
```
In [26]: dataset=pd.read_csv("../Datasets/startup_funding.csv")
df=dataset.copy()
df.dropna(subset="CityLocation",inplace=True)
print(df)
```

	SNo	Date	StartupName	IndustryVertical \
0	0	01/08/2017	TouchKin	Technology
1	1	02/08/2017	Ethinos	Technology
2	2	02/08/2017	Leverage Edu	Consumer Internet
3	3	02/08/2017	Zepo	Consumer Internet
4	4	02/08/2017	Click2Clinic	Consumer Internet
...
2196	2196	29/04/2015	Tracxn	Startup Analytics platform
2197	2197	29/04/2015	Dazo	Mobile Food Ordering app
2198	2198	29/04/2015	Tradelab	Financial Markets Software
2199	2199	29/04/2015	PiQube	Hiring Analytics platform
2200	2200	29/04/2015	Travel Triangle	Online Travel Marketplace

		SubVertical	CityLocation \
0		Predictive Care Platform	Bangalore
1		Digital Marketing Agency	Mumbai
2	Online platform for Higher Education Services		New Delhi
3		DIY Ecommerce platform	Mumbai
4		healthcare service aggregator	Hyderabad
...	
2196		NaN	Bangalore
2197		NaN	Bangalore
2198		NaN	Bangalore
2199		NaN	Chennai
2200		NaN	Noida

		InvestorsName	InvestmentType \
0		Kae Capital	Private Equity
1		Triton Investment Advisors	Private Equity

For Some CityLocations there are two locations separated by '/' so splitting it using separate_city() method (Created by me)

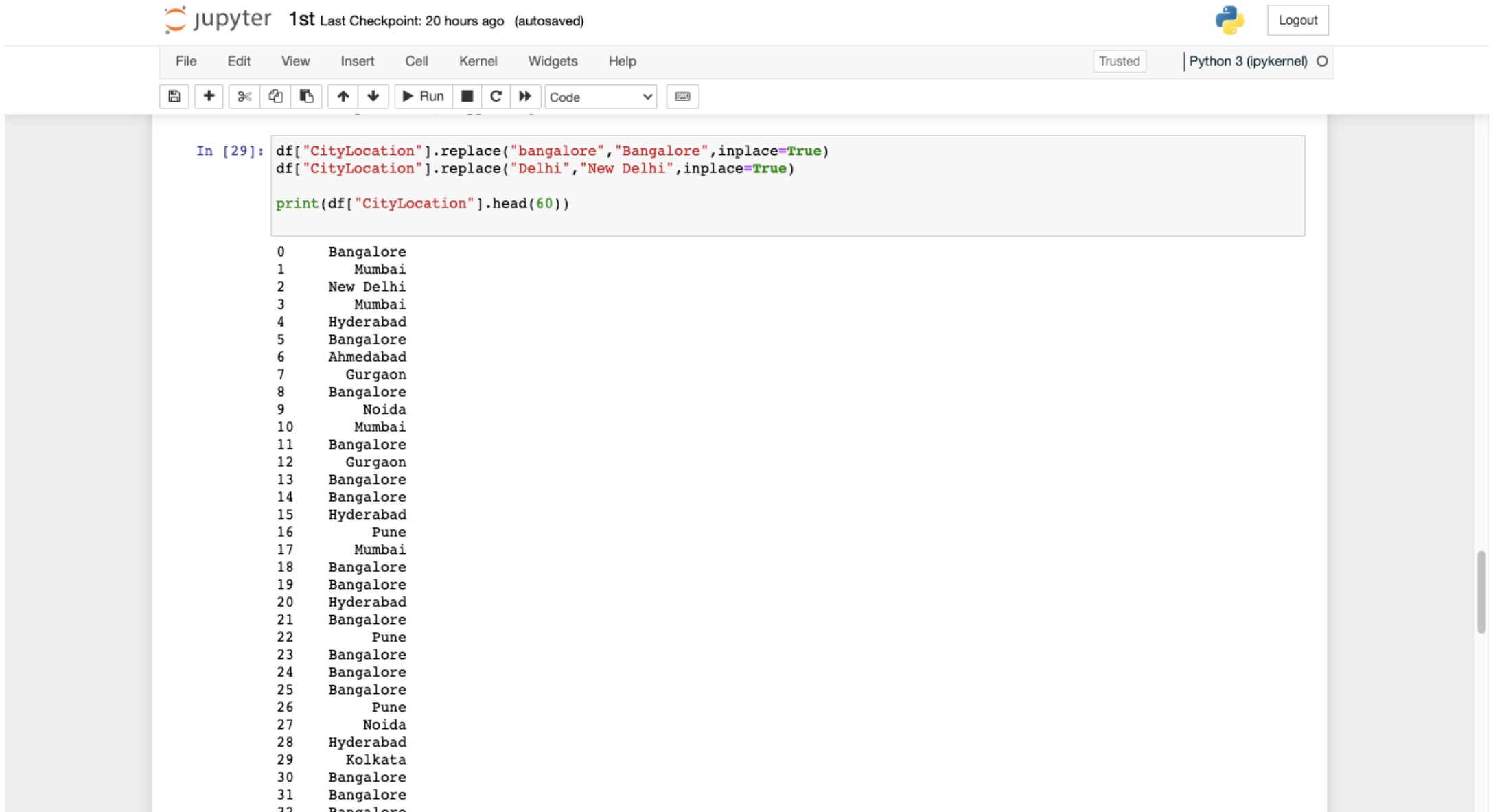


The image shows a Jupyter Notebook interface. At the top, the Jupyter logo is followed by '1st Last Checkpoint: 20 hours ago (autosaved)'. On the right, there is a Python logo and a 'Logout' button. Below this is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. To the right of the menu bar are 'Trusted' and 'Python 3 (ipykernel)' labels. Below the menu bar is a toolbar with icons for saving, adding cells, zooming, copying, pasting, undo, redo, running, and a dropdown menu currently set to 'Code'. The main area contains a code cell with the following Python code:

```
In [28]: def seperate_city(city):  
         return city.split('/')[0].strip()  
  
df["CityLocation"] = df["CityLocation"].apply(seperate_city)  
print(df["CityLocation"].head(60))
```

The output of the code is a list of 60 city names, each preceded by an index from 0 to 59. The cities are: Bangalore, Mumbai, New Delhi, Mumbai, Hyderabad, Bangalore, Ahmedabad, Gurgaon, Bangalore, Noida, Mumbai, Bangalore, Gurgaon, Bangalore, Bangalore, Hyderabad, Pune, Mumbai, Bangalore, Bangalore, Hyderabad, Bangalore, Pune, Bangalore, Bangalore, Bangalore, Pune, Noida, Hyderabad, Kolkata, Bangalore, Bangalore, and Bangalore.

Some locations have spelling mistakes so we are replacing it (purifying the data) like bangalore with Bangalore etc.



The image shows a Jupyter Notebook interface. At the top, the Jupyter logo is followed by the text "1st Last Checkpoint: 20 hours ago (autosaved)". On the right, there is a Python logo and a "Logout" button. Below this is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". To the right of the menu bar are "Trusted" and "Python 3 (ipykernel)" labels. Below the menu bar is a toolbar with icons for saving, adding, copying, pasting, undo, redo, running, and other functions. The main area contains a code cell with the following Python code:

```
In [29]: df["CityLocation"].replace("bangalore", "Bangalore", inplace=True)
df["CityLocation"].replace("Delhi", "New Delhi", inplace=True)

print(df["CityLocation"].head(60))
```

The output of the code cell is a list of 60 city names, indexed from 0 to 59. The cities are: Bangalore, Mumbai, New Delhi, Mumbai, Hyderabad, Bangalore, Ahmedabad, Gurgaon, Bangalore, Noida, Mumbai, Bangalore, Gurgaon, Bangalore, Bangalore, Hyderabad, Pune, Mumbai, Bangalore, Bangalore, Hyderabad, Bangalore, Pune, Bangalore, Bangalore, Bangalore, Pune, Noida, Hyderabad, Kolkata, Bangalore, Bangalore, and Bangalore.

Index	City
0	Bangalore
1	Mumbai
2	New Delhi
3	Mumbai
4	Hyderabad
5	Bangalore
6	Ahmedabad
7	Gurgaon
8	Bangalore
9	Noida
10	Mumbai
11	Bangalore
12	Gurgaon
13	Bangalore
14	Bangalore
15	Hyderabad
16	Pune
17	Mumbai
18	Bangalore
19	Bangalore
20	Hyderabad
21	Bangalore
22	Pune
23	Bangalore
24	Bangalore
25	Bangalore
26	Pune
27	Noida
28	Hyderabad
29	Kolkata
30	Bangalore
31	Bangalore
32	Bangalore

- Finally Selecting only those cityLocations which my friend wanted
- And Then Finding each city fundings using value_counts() method

jupyter 1st Last Checkpoint: 20 hours ago (autosaved)



Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted

Python 3 (ipykernel) O

Save + Close Copy Paste Undo Redo Run Stop Restart Code

```
56 New Delhi
57 New Delhi
58 Gurgaon
59 New Delhi
Name: CityLocation, dtype: object
```

```
In [30]: df=df[(df["CityLocation"]=="Bangalore") | (df["CityLocation"]=="New Delhi") | (df["CityLocation"]=="Mumbai") | (df["CityLocation"]=="Gurgaon") | (df["CityLocation"]=="Noida")]
city_fundings=df["CityLocation"].value_counts()
print(city_fundings)
```

```
Bangalore    635
Mumbai       449
New Delhi    389
Gurgaon      241
Noida         79
Name: CityLocation, dtype: int64
```

Plotted a bar graph between cityLocation and funding

jupyter 1st Last Checkpoint: 20 hours ago (autosaved)



Logout

File Edit View Insert Cell Kernel Widgets Help

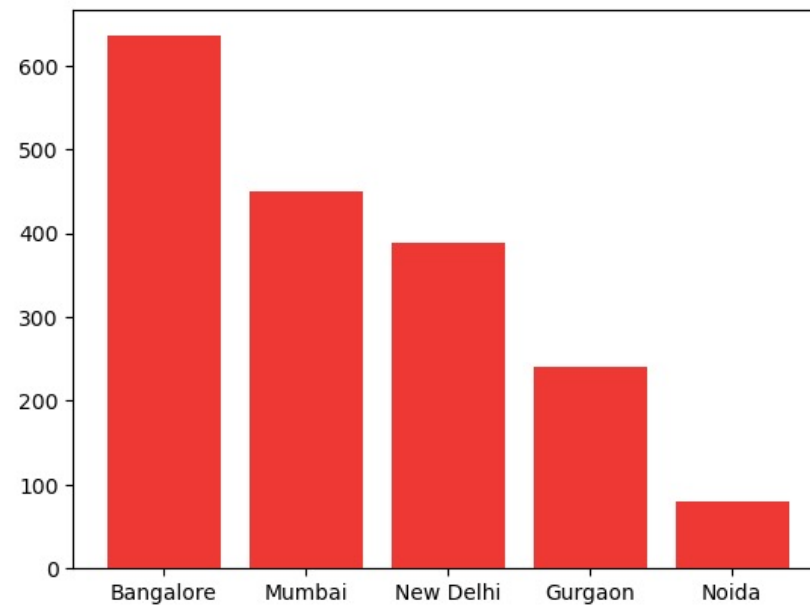
Trusted

Python 3 (ipykernel)

Run Code

```
Gurgaon      241  
Noida         79  
Name: CityLocation, dtype: int64
```

```
In [35]: plt.bar(city_fundings.index,city_fundings,color="red")  
plt.show()
```



As We can see in the graph Bangalore city has received maximum number of fundings compared to other cities