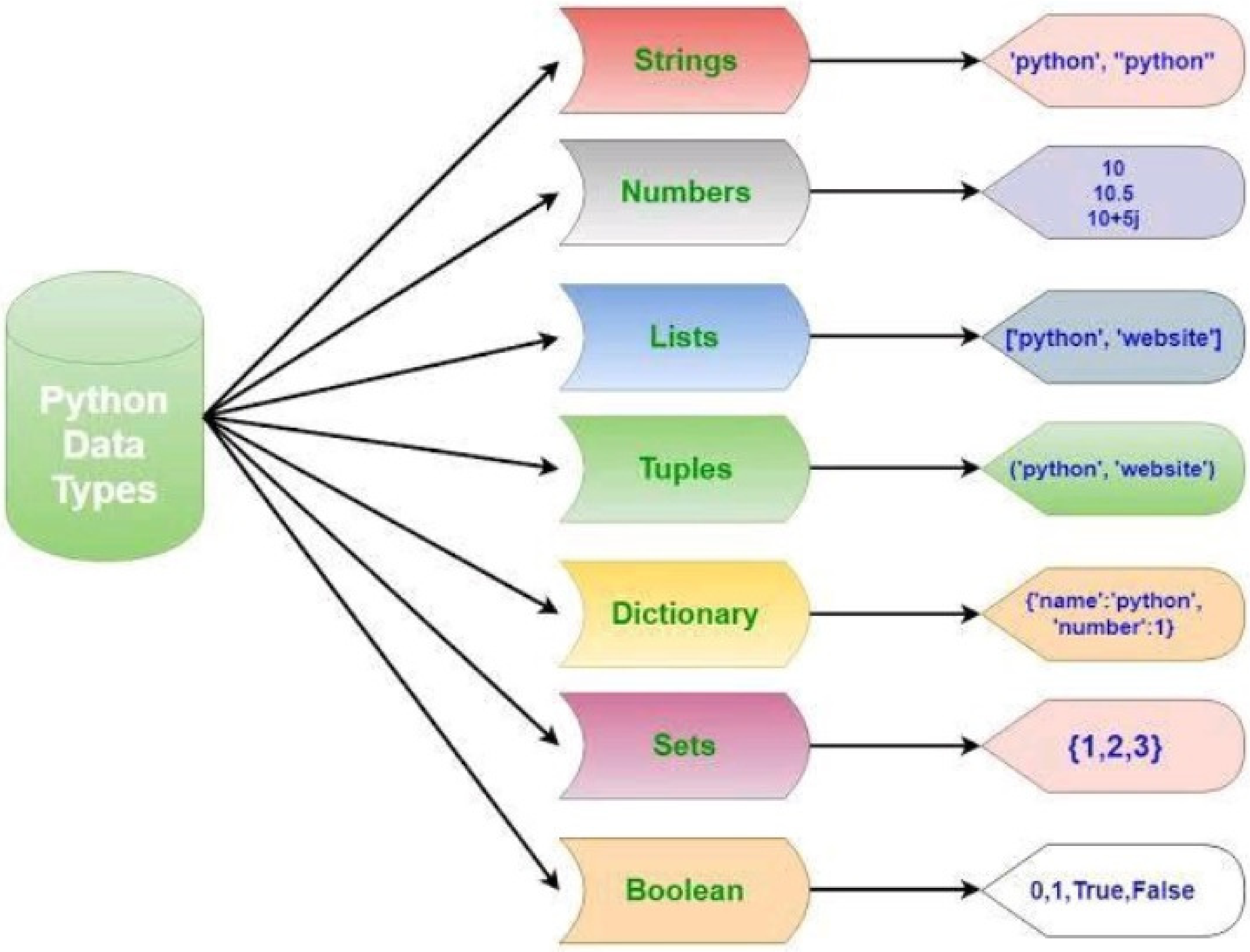
دوره دیتا ساینس کاربردی

Exploratory Data Analysis and Data Visualization

dataroadmap

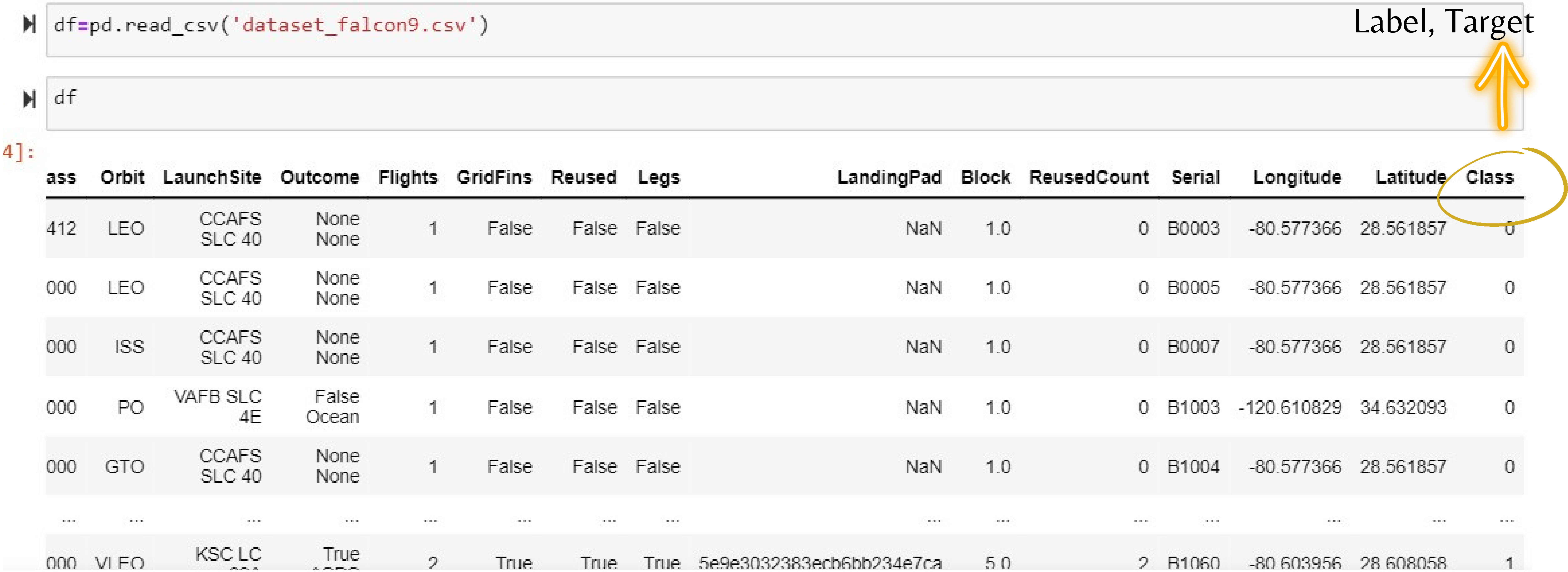
مدرس: مونا حاتمی

جلسه چهارم



Python Data Types

Read csv file in Pandas



```
df=pd.read_csv('dataset_falcon9.csv')
```

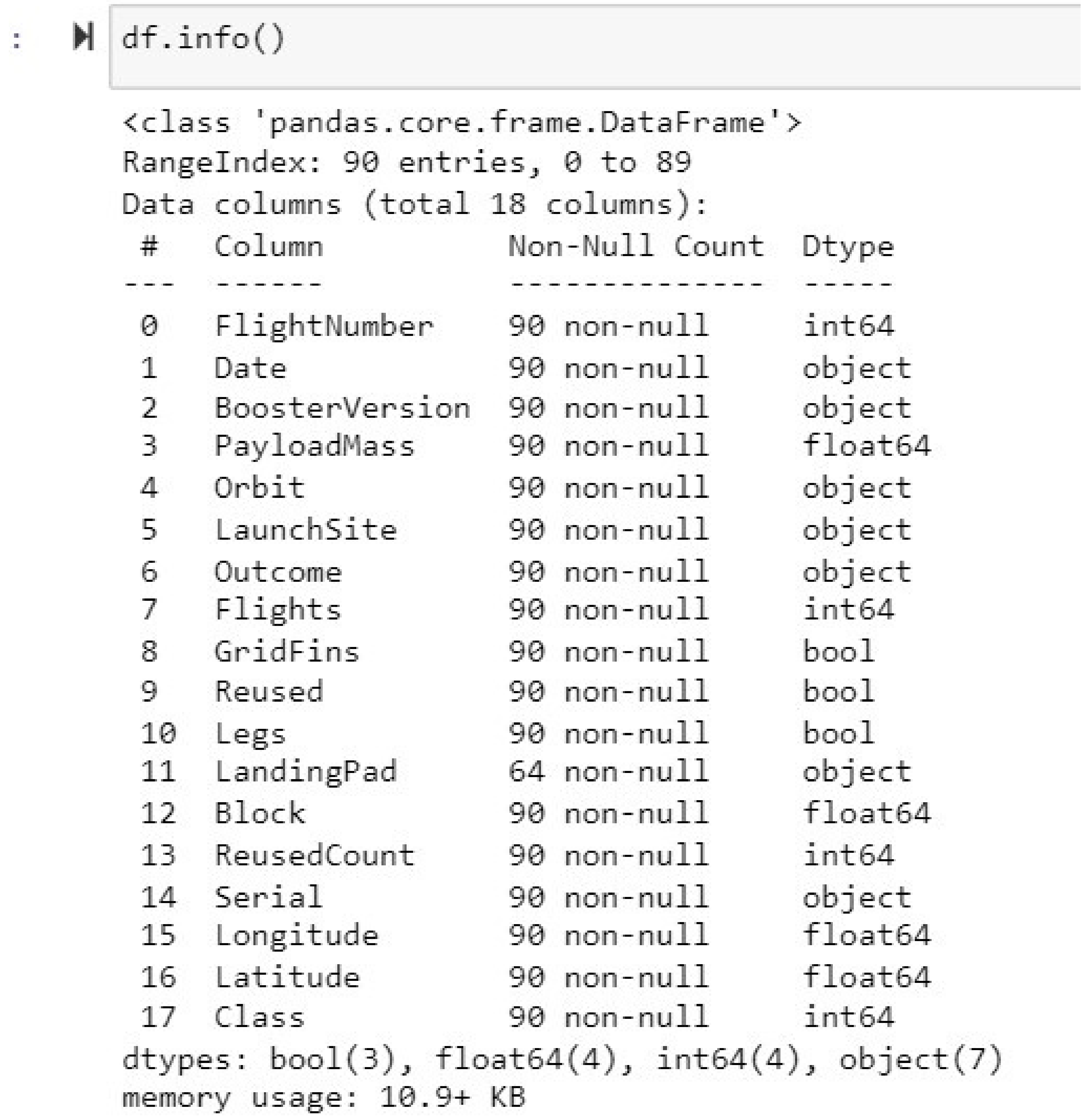
```
df
```

4]:

ass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
000	VLEO	KSC LC	True	2	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	2	B1060	-80.603956	28.608058	1

Label, Target

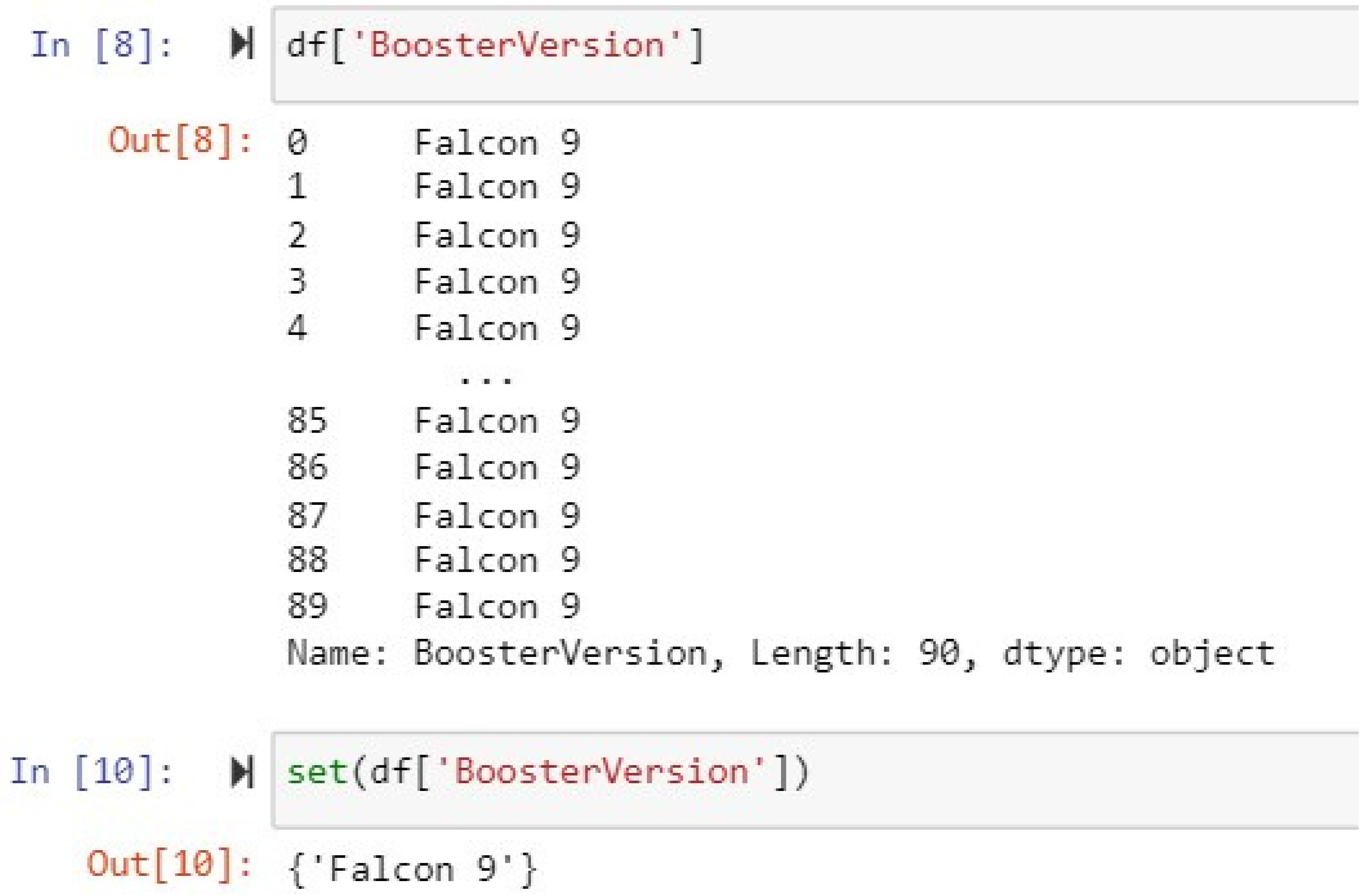
Exploratory data analysis



```
df.info()
```

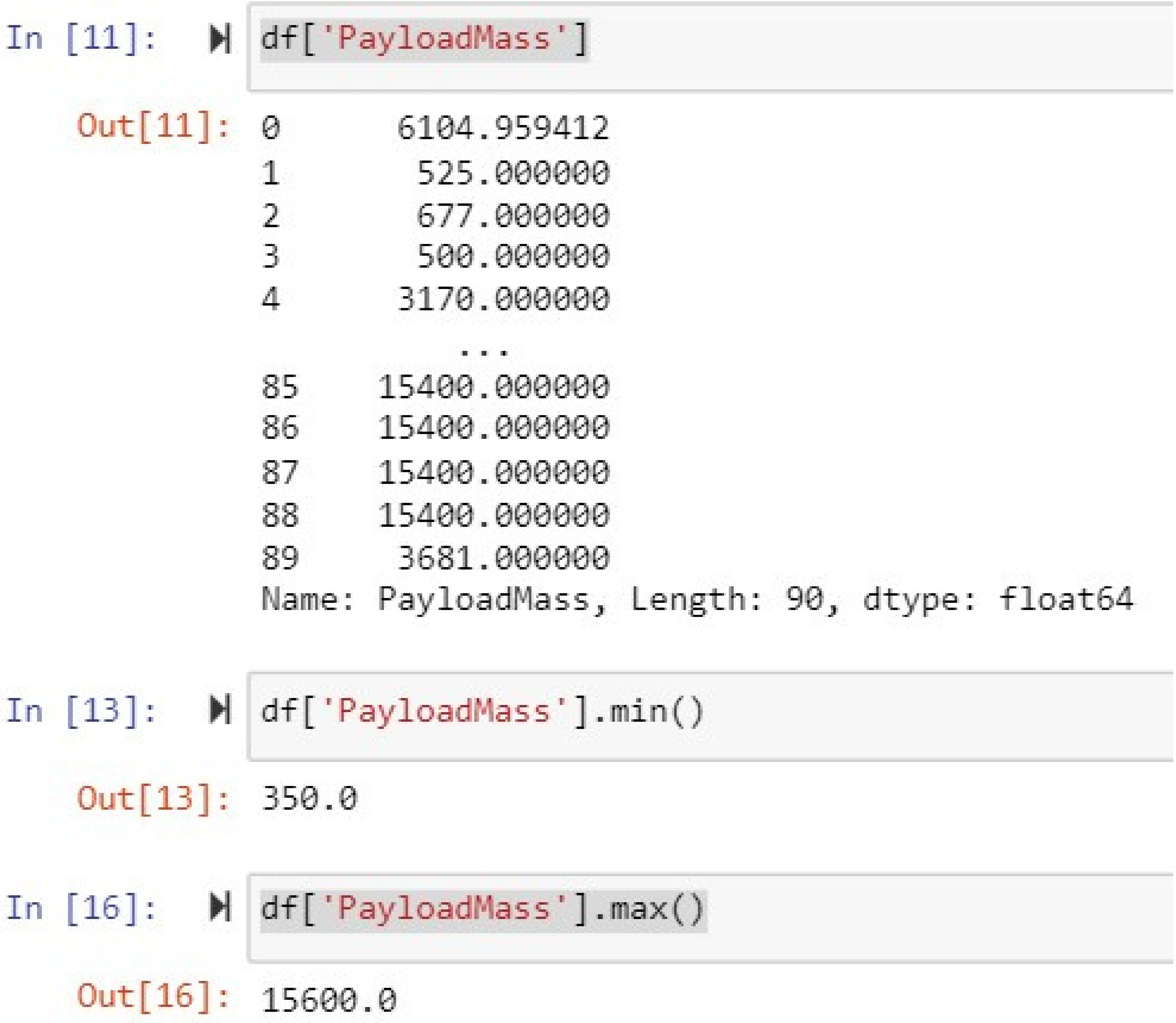
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 18 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   FlightNumber    90 non-null     int64  
 1   Date            90 non-null     object 
 2   BoosterVersion  90 non-null     object 
 3   PayloadMass     90 non-null     float64
 4   Orbit           90 non-null     object 
 5   LaunchSite      90 non-null     object 
 6   Outcome         90 non-null     object 
 7   Flights         90 non-null     int64  
 8   GridFins        90 non-null     bool   
 9   Reused          90 non-null     bool   
 10  Legs            90 non-null     bool   
 11  LandingPad      64 non-null     object 
 12  Block           90 non-null     float64
 13  ReusedCount     90 non-null     int64  
 14  Serial          90 non-null     object 
 15  Longitude       90 non-null     float64
 16  Latitude        90 non-null     float64
 17  Class           90 non-null     int64  
dtypes: bool(3), float64(4), int64(4), object(7)
memory usage: 10.9+ KB
```

Exploratory data analysis



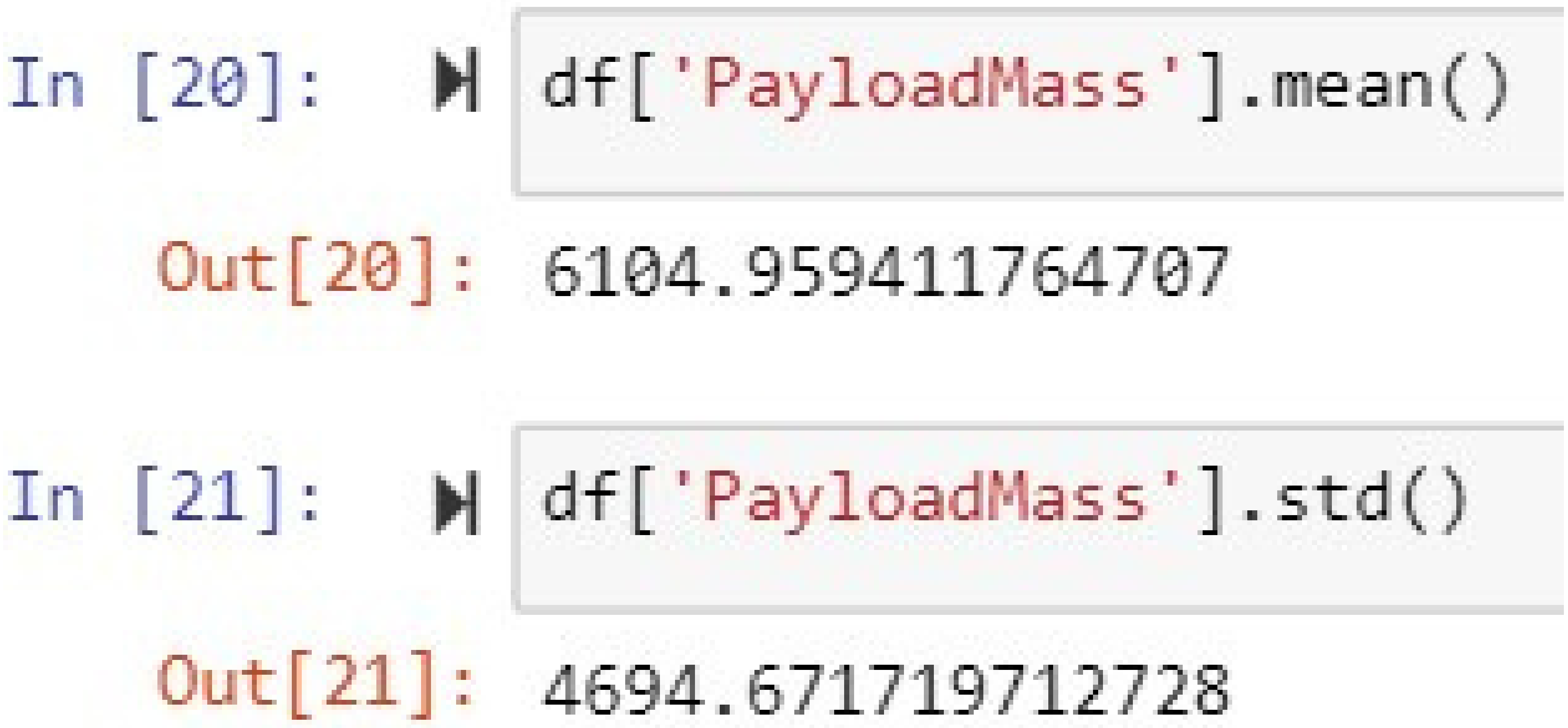
```
In [8]:  df['BoosterVersion']

Out[8]:  0     Falcon 9
         1     Falcon 9
         2     Falcon 9
         3     Falcon 9
         4     Falcon 9
                 ...
         85    Falcon 9
         86    Falcon 9
         87    Falcon 9
         88    Falcon 9
         89    Falcon 9
         Name: BoosterVersion, Length: 90, dtype: object

In [10]: set(df['BoosterVersion'])

Out[10]: {'Falcon 9'}
```

Exploratory data analysis



```
In [11]:    df['PayloadMass']

Out[11]:    0      6104.959412
            1       525.000000
            2       677.000000
            3       500.000000
            4      3170.000000
                      ...
            85    15400.000000
            86    15400.000000
            87    15400.000000
            88    15400.000000
            89     3681.000000
            Name: PayloadMass, Length: 90, dtype: float64

In [13]:    df['PayloadMass'].min()

Out[13]:    350.0

In [16]:    df['PayloadMass'].max()

Out[16]:    15600.0
```

Exploratory data analysis



```
In [20]:  df['PayloadMass'].mean()

Out[20]:  6104.959411764707

In [21]:  df['PayloadMass'].std()

Out[21]:  4694.671719712728
```

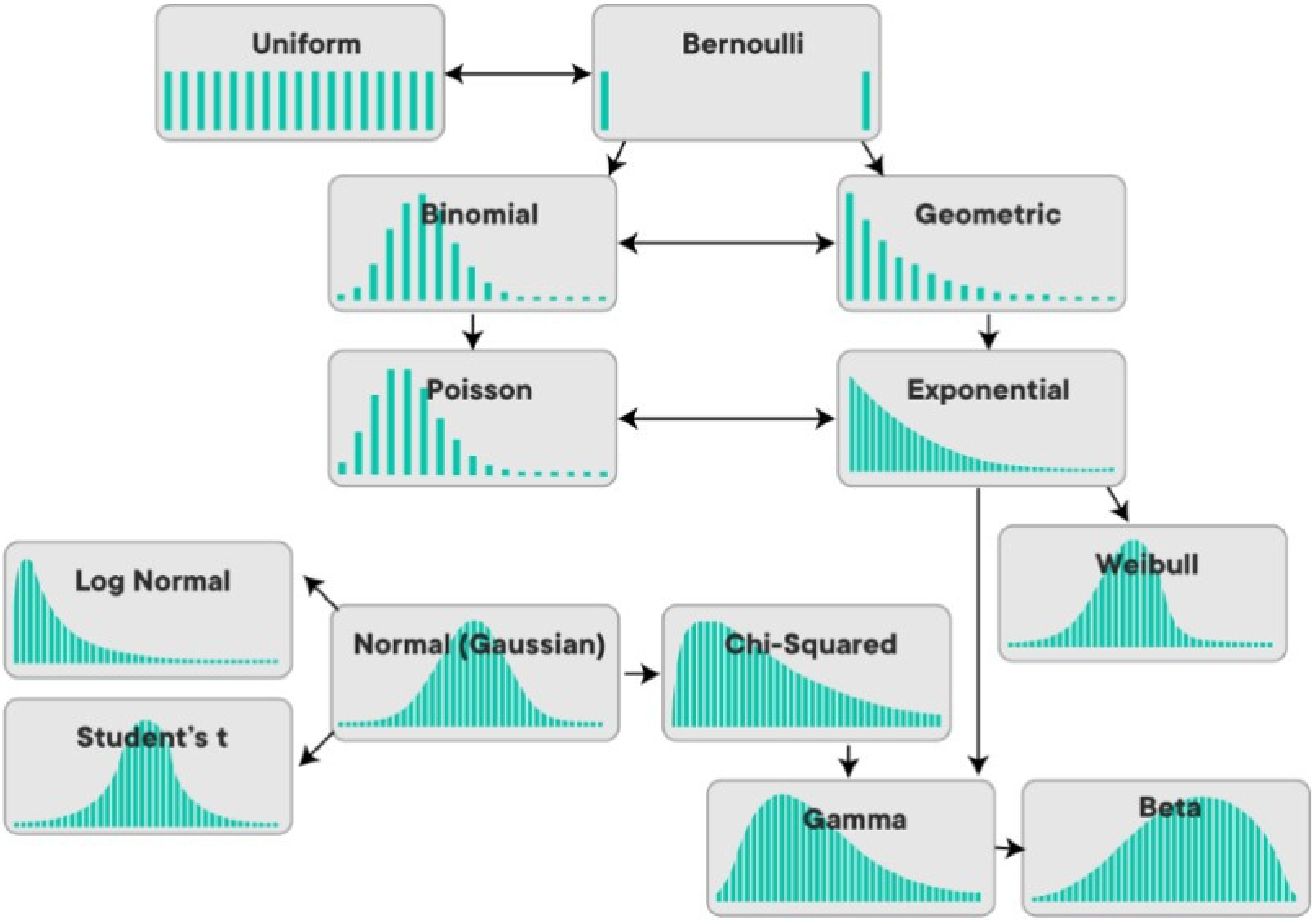
Mean and Standard deviation

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

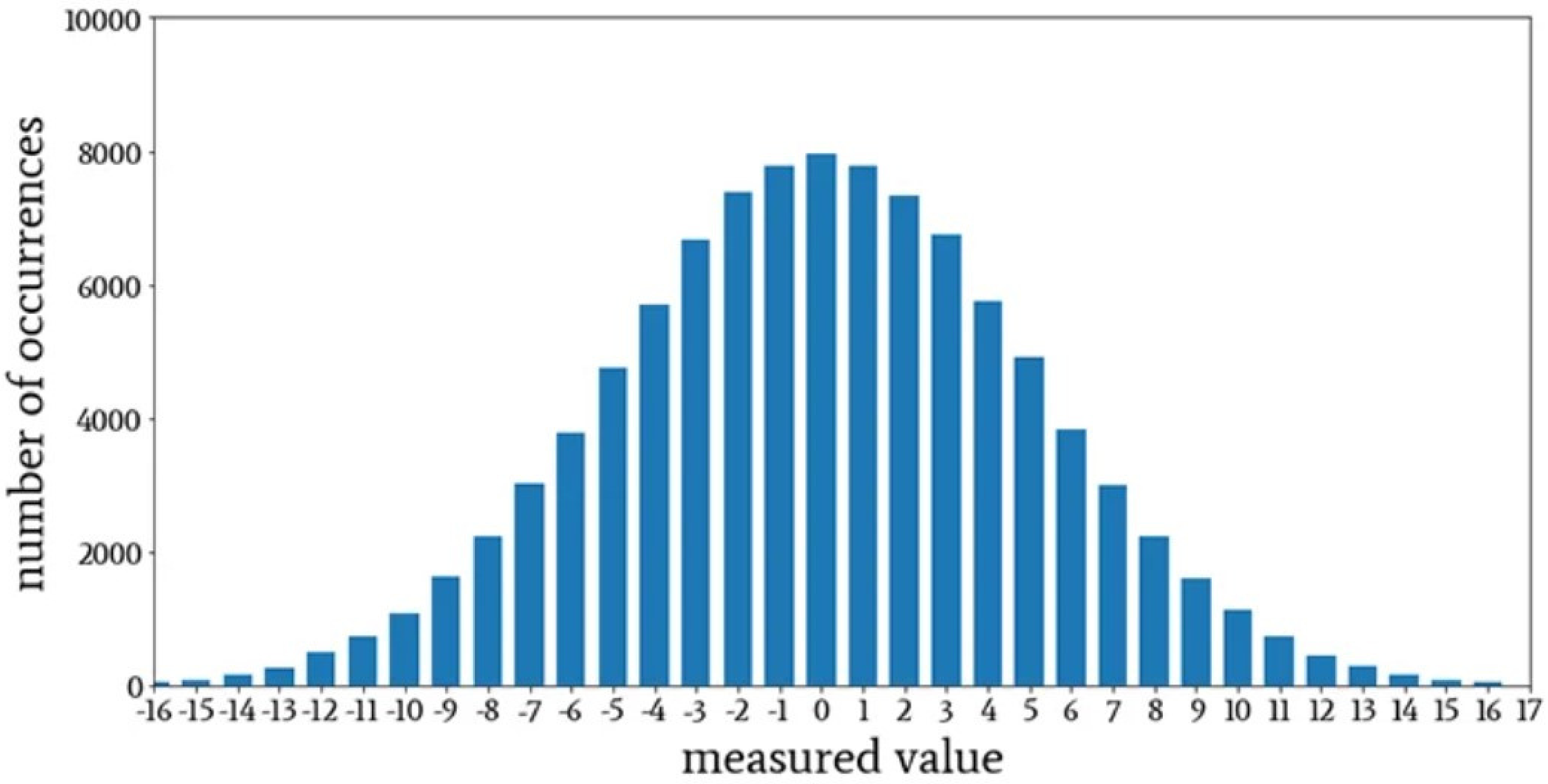
$$\sigma = \sqrt{\frac{\sum (x_i - \bar{x})^2}{N}}$$

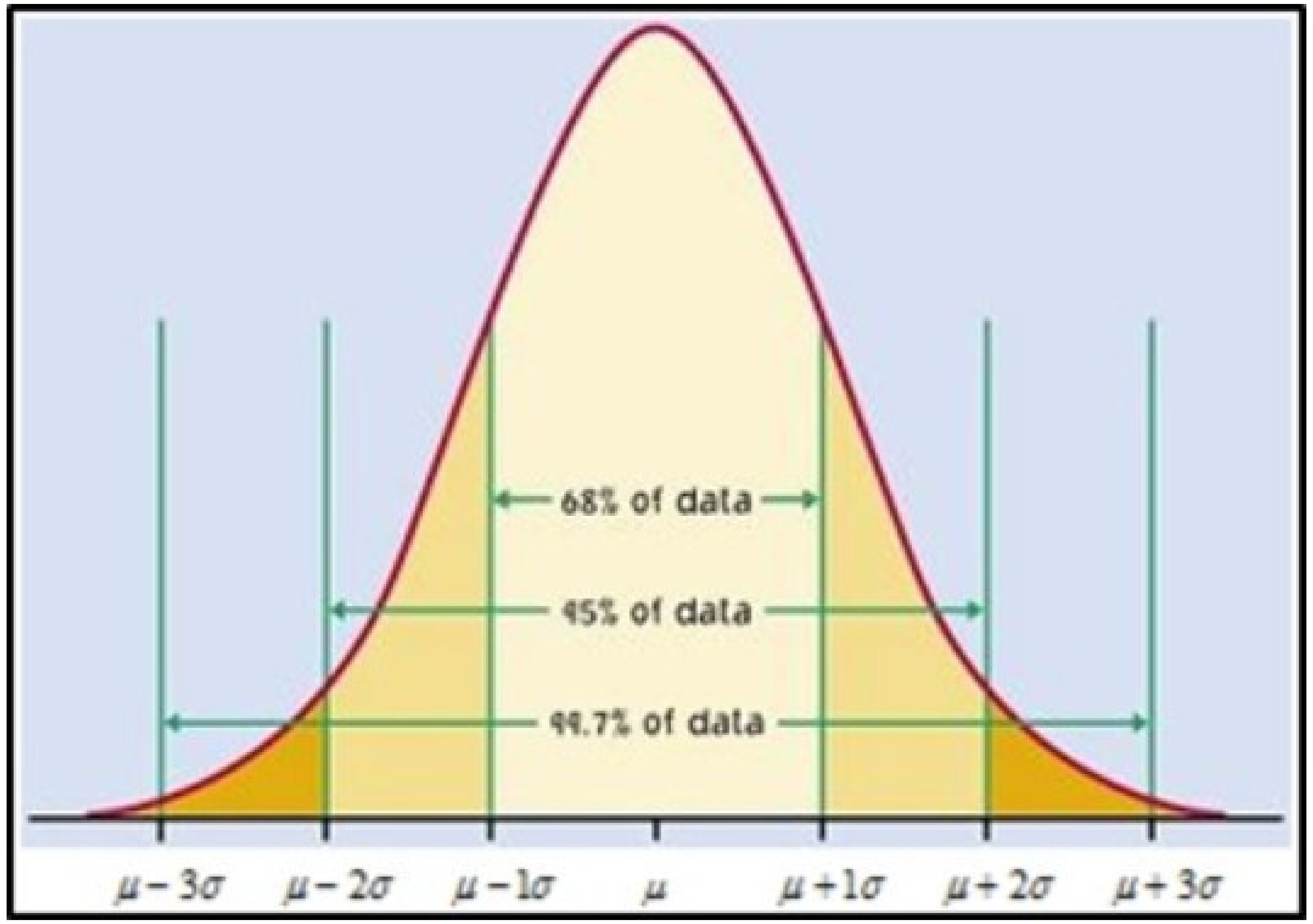
	$\bar{x}$	σ
5, 25	(5+ 25)/2=15	$\sqrt{\frac{(5-15)^2 + (25-15)^2}{2}}$ =10
14,16	(14+16)/2=15	$\sqrt{\frac{(14-15)^2 + (16-15)^2}{2}}$ =1

Data Distribution

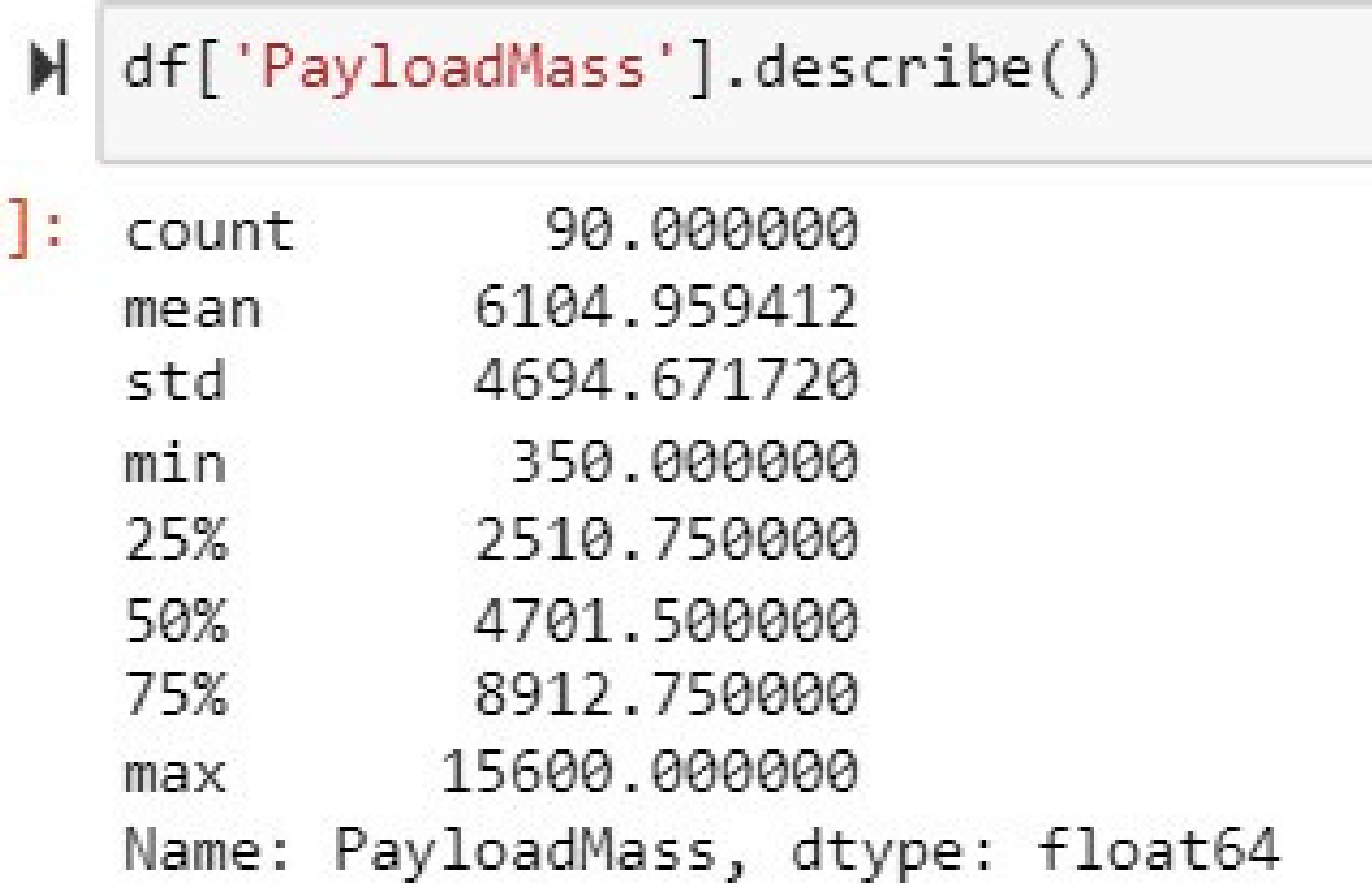


Normal Distribution



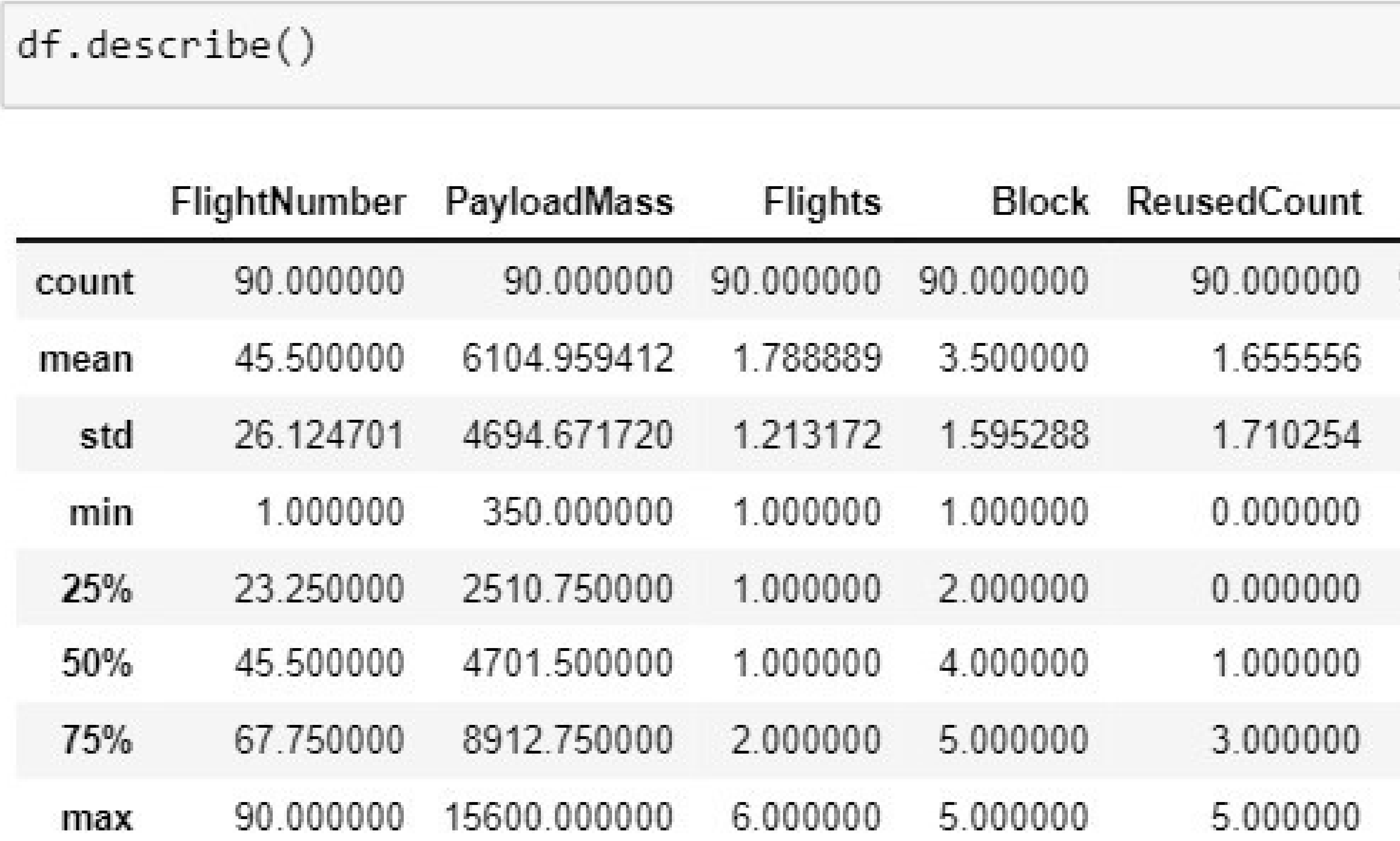


Exploratory data analysis



```
df['PayloadMass'].describe()
```

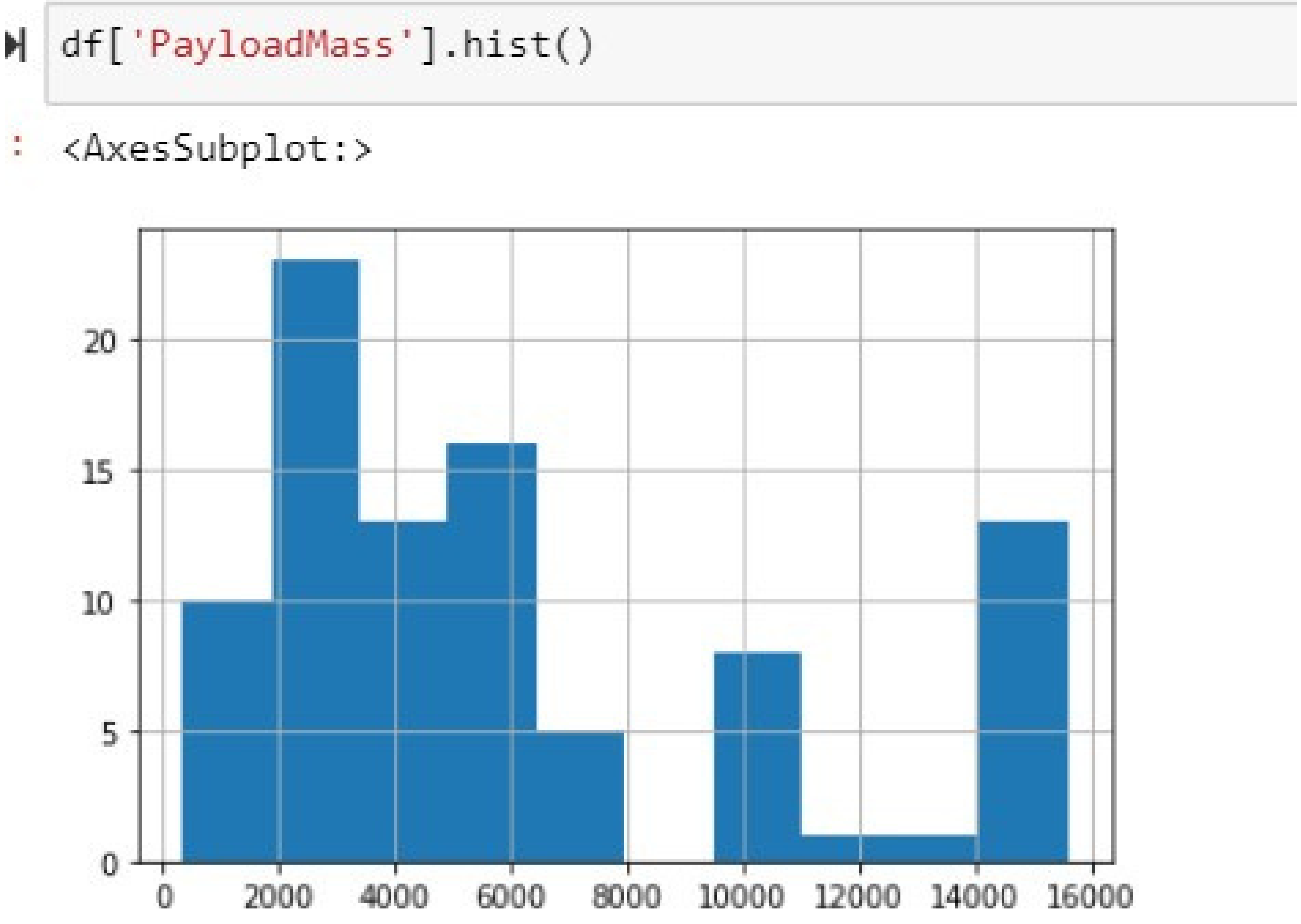
```
]:  count       90.000000
    mean      6104.959412
    std       4694.671720
    min        350.000000
    25%       2510.750000
    50%       4701.500000
    75%       8912.750000
    max      15600.000000
    Name: PayloadMass, dtype: float64
```



```
df.describe()
```

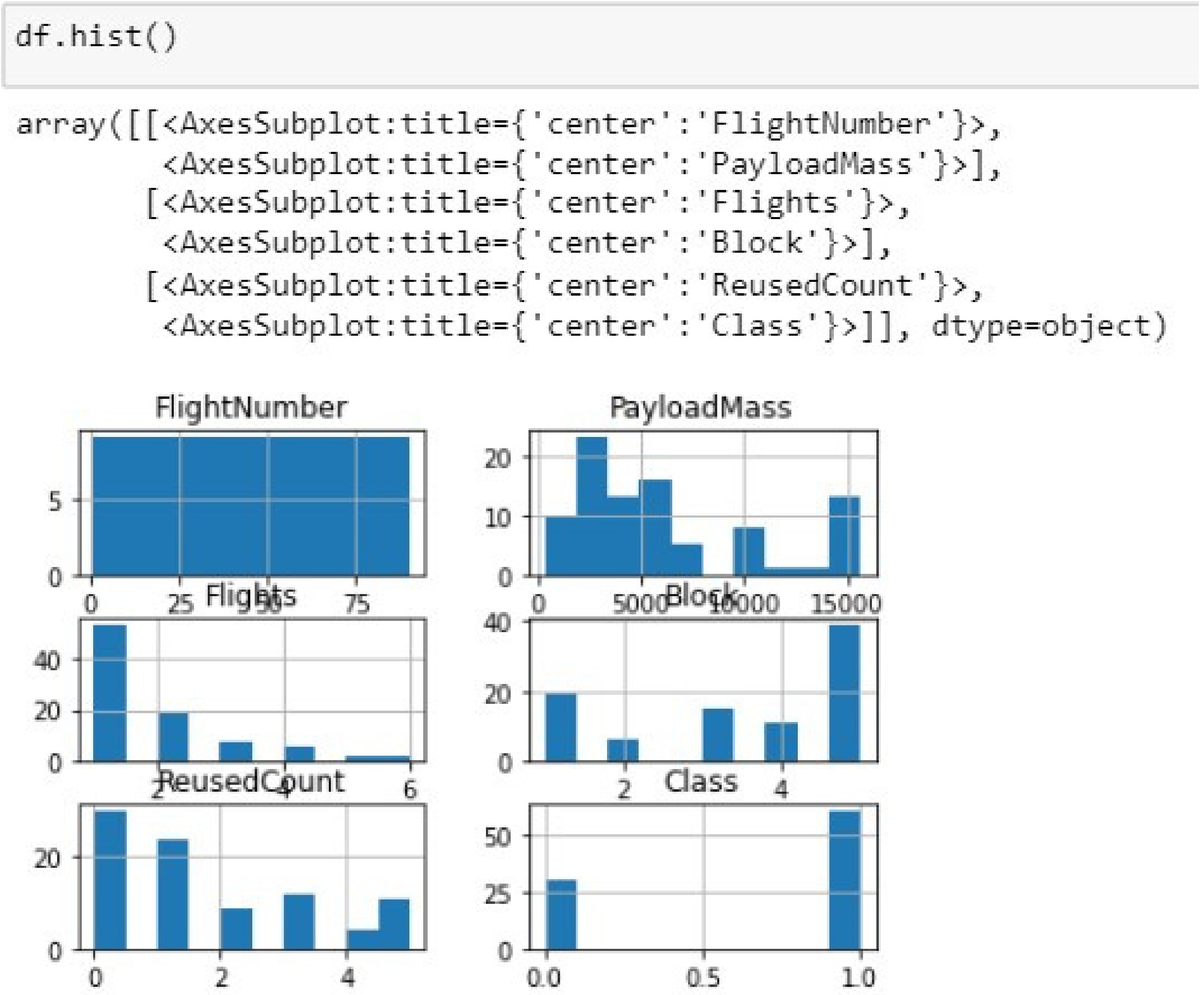
	FlightNumber	PayloadMass	Flights	Block	ReusedCount
count	90.000000	90.000000	90.000000	90.000000	90.000000
mean	45.500000	6104.959412	1.788889	3.500000	1.655556
std	26.124701	4694.671720	1.213172	1.595288	1.710254
min	1.000000	350.000000	1.000000	1.000000	0.000000
25%	23.250000	2510.750000	1.000000	2.000000	0.000000
50%	45.500000	4701.500000	1.000000	4.000000	1.000000
75%	67.750000	8912.750000	2.000000	5.000000	3.000000
max	90.000000	15600.000000	6.000000	5.000000	5.000000

Exploratory data analysis



```
df['PayloadMass'].hist()
```

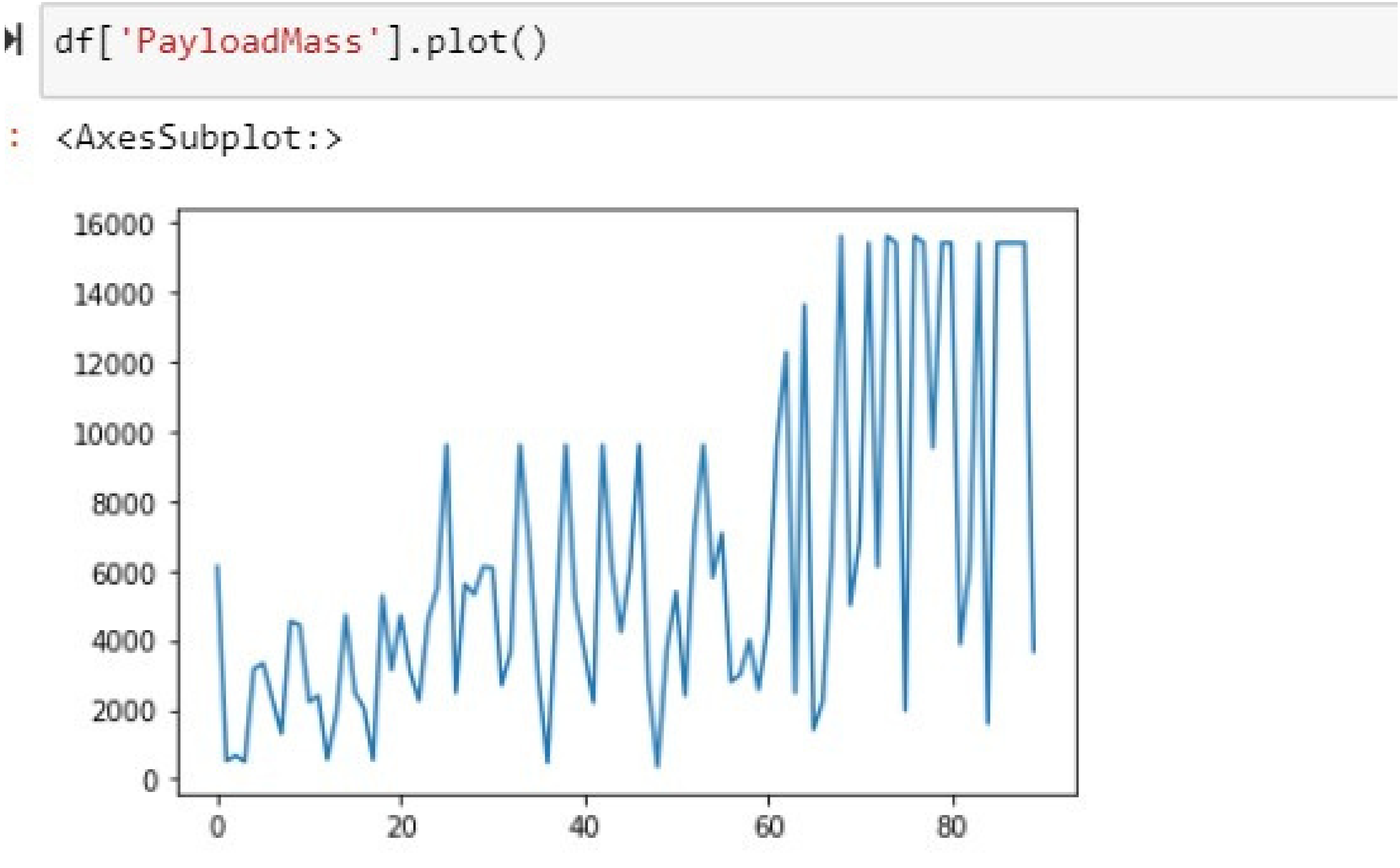
```
<AxesSubplot:>
```



```
df.hist()
```

```
array([[<AxesSubplot:title={'center':'FlightNumber'}>,
        <AxesSubplot:title={'center':'PayloadMass'}>],
       [<AxesSubplot:title={'center':'Flights'}>,
        <AxesSubplot:title={'center':'Block'}>],
       [<AxesSubplot:title={'center':'ReusedCount'}>,
        <AxesSubplot:title={'center':'Class'}>]], dtype=object)
```

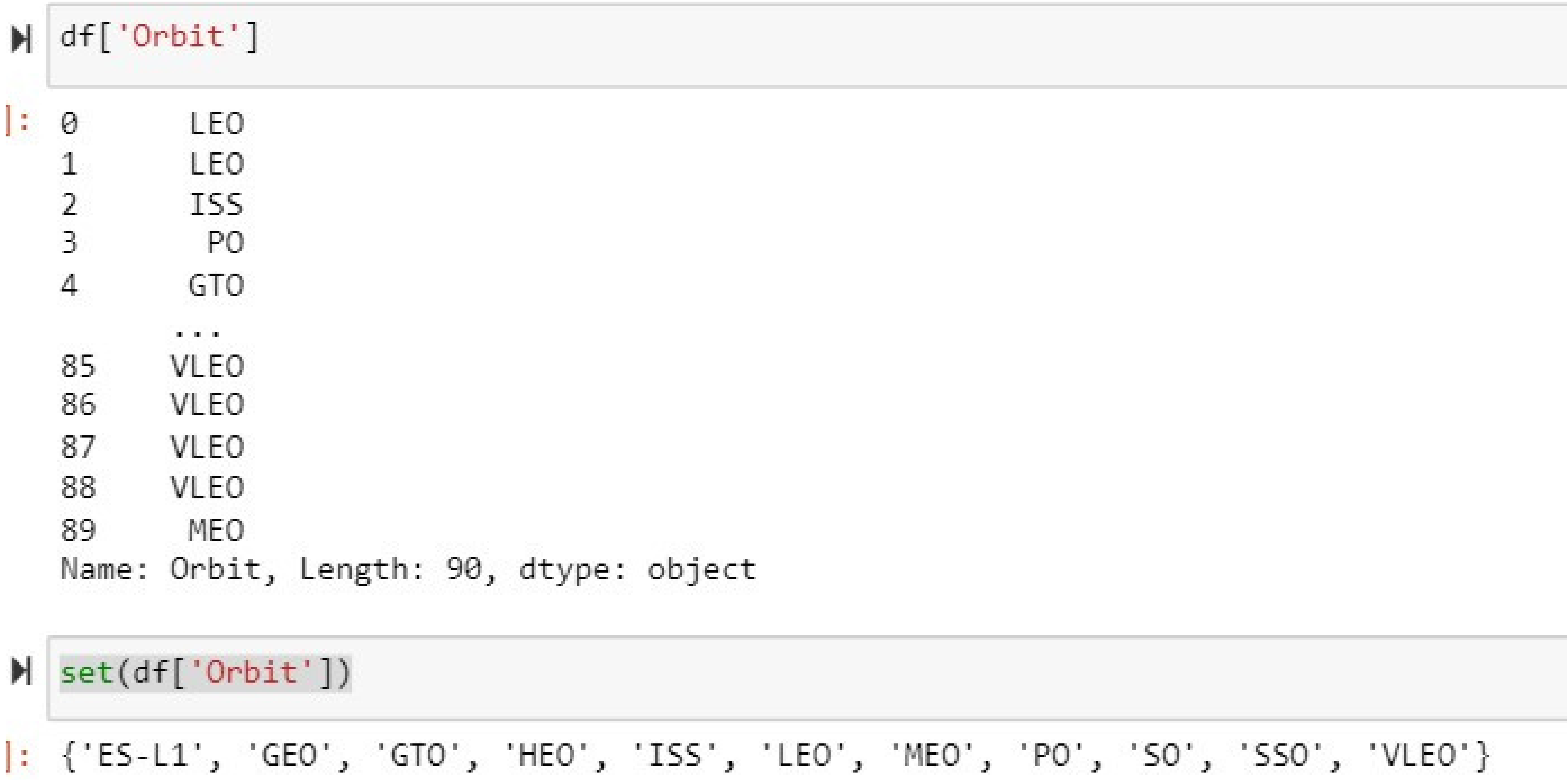
Exploratory data analysis



```
df['PayloadMass'].plot()
```

```
<AxesSubplot:>
```

Exploratory data analysis



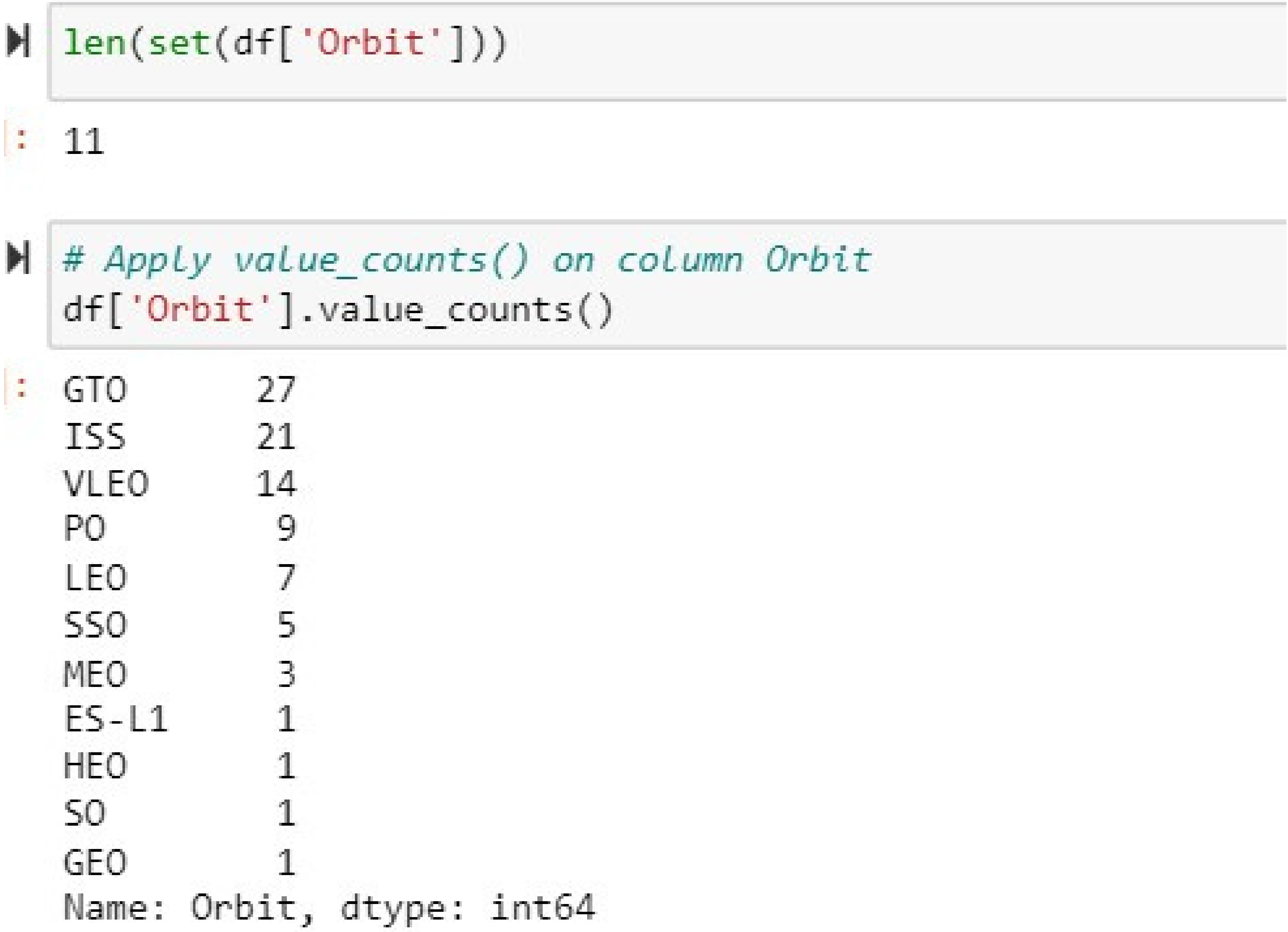
```
df['Orbit']
```

```
]: 0       LEO
   1       LEO
   2       ISS
   3        PO
   4       GTO
          ...
   85     VLEO
   86     VLEO
   87     VLEO
   88     VLEO
   89      MEO
   Name: Orbit, Length: 90, dtype: object
```

```
set(df['Orbit'])
```

```
]: {'ES-L1', 'GEO', 'GTO', 'HEO', 'ISS', 'LEO', 'MEO', 'PO', 'SO', 'SSO', 'VLEO'}
```

Exploratory data analysis



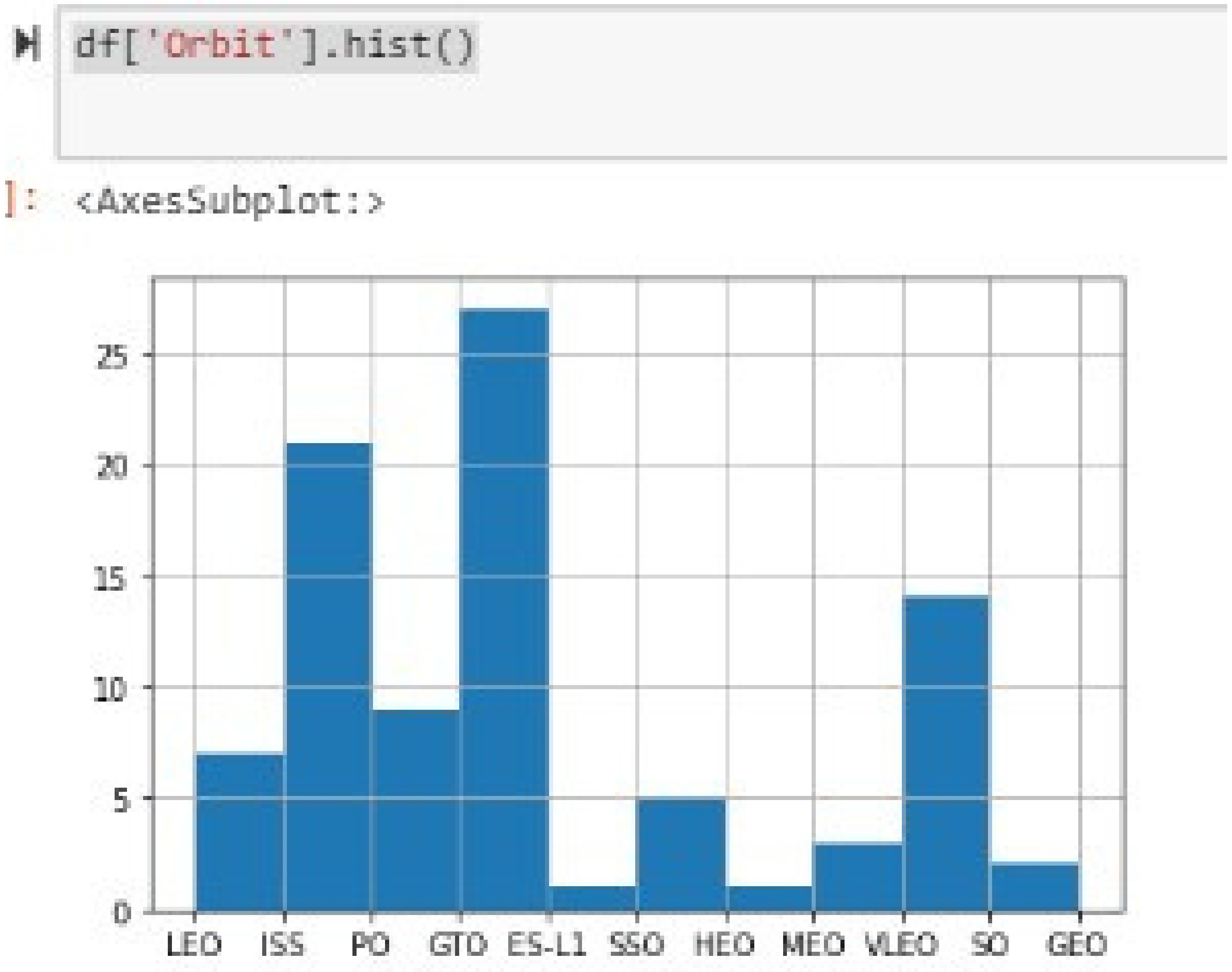
```
len(set(df['Orbit']))
```

```
11
```

```
# Apply value_counts() on column Orbit
df['Orbit'].value_counts()
```

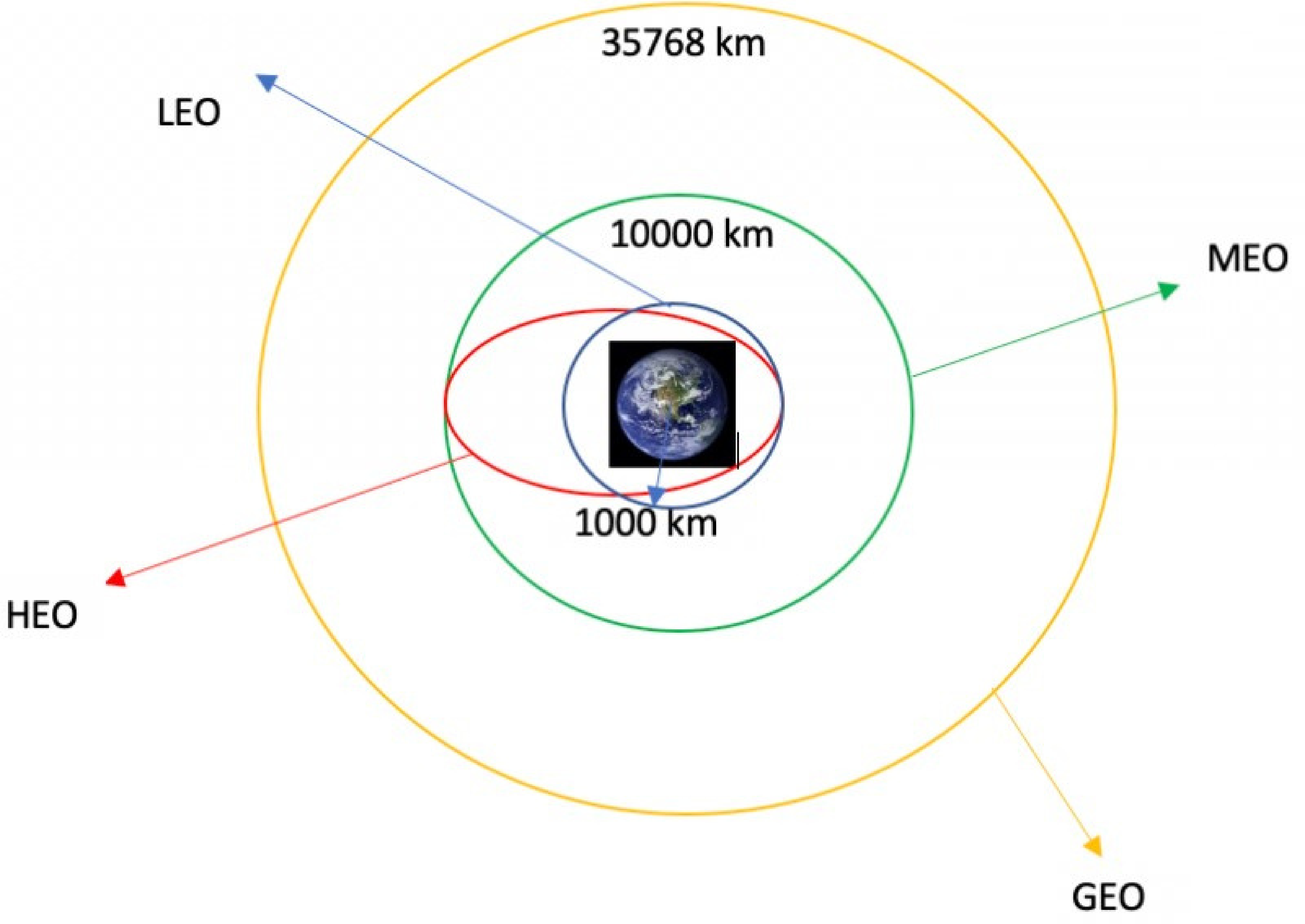
```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

Exploratory data analysis

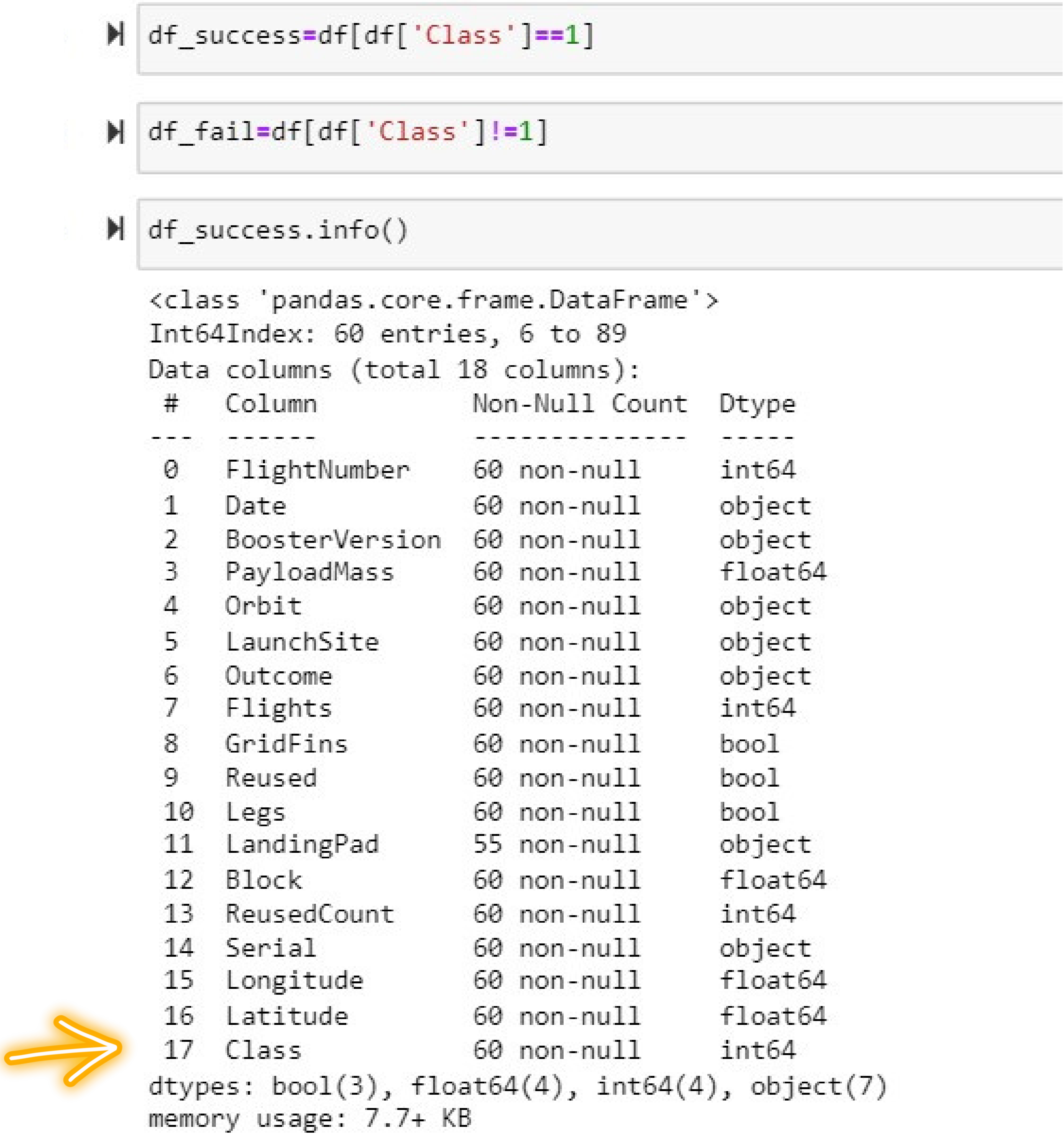


```
df['Orbit'].hist()
```

```
]: <AxesSubplot:>
```



Exploratory data analysis(EDA)



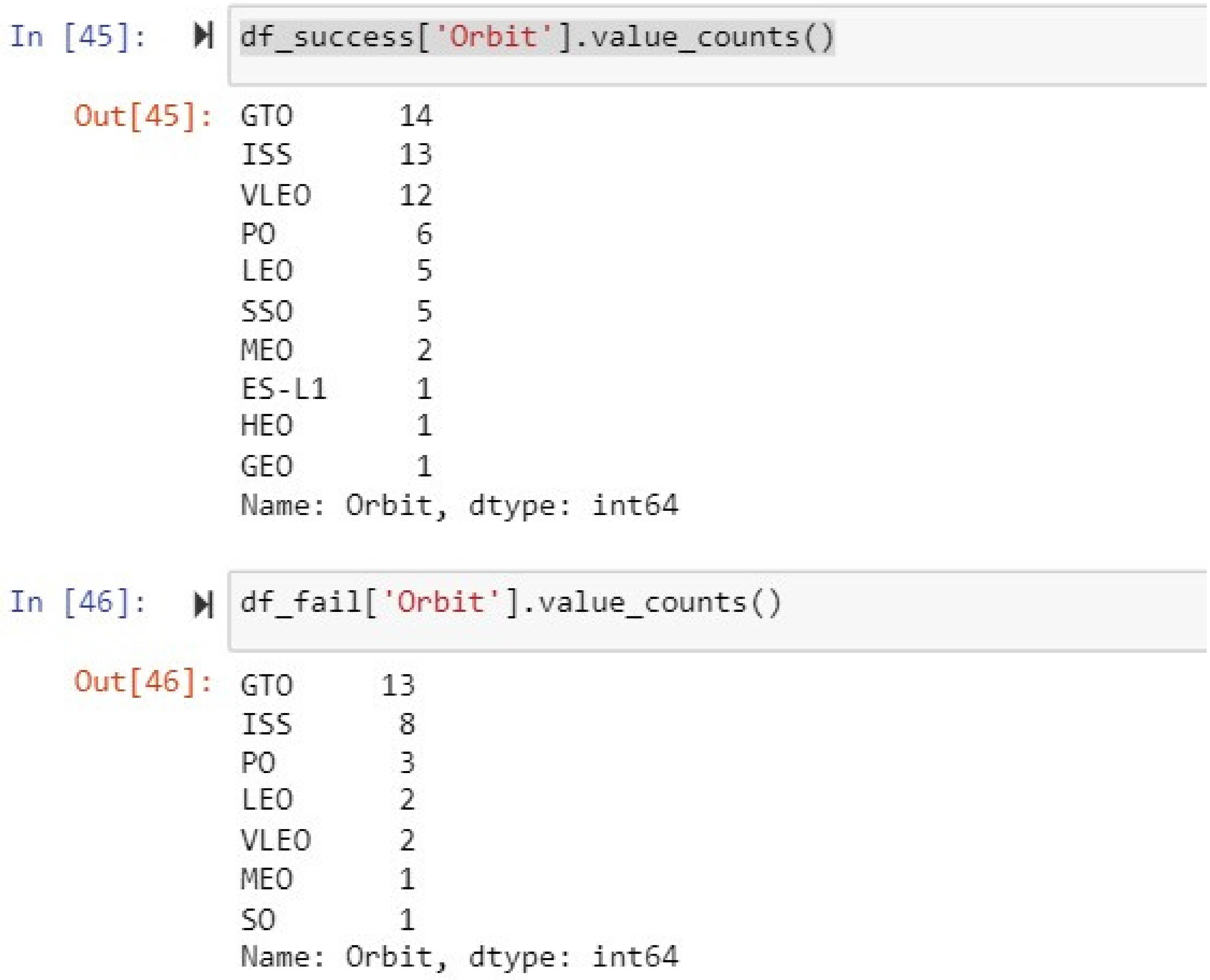
```
df_success=df[df['Class']==1]
```

```
df_fail=df[df['Class']!=1]
```

```
df_success.info()
```

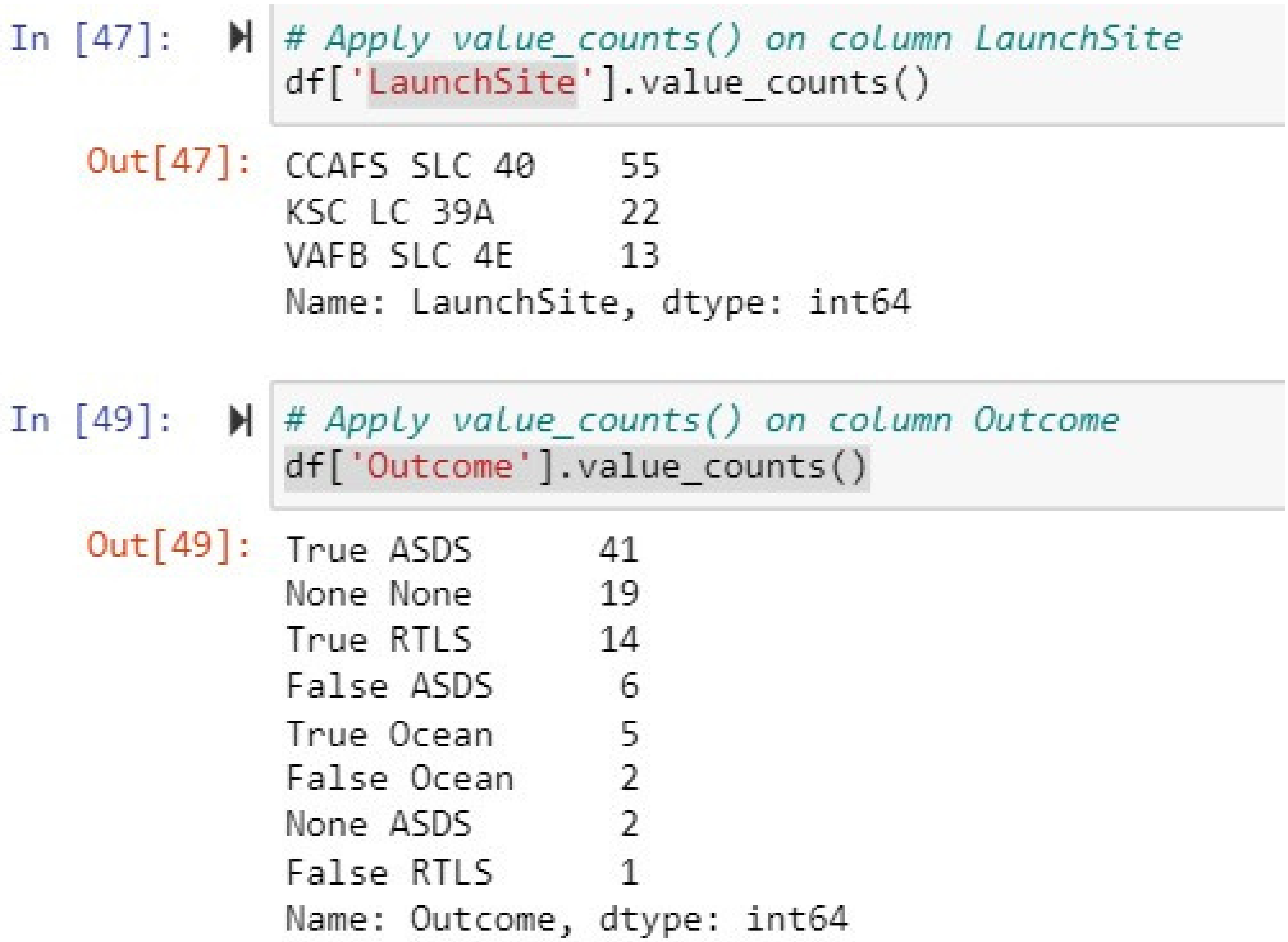
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 60 entries, 6 to 89
Data columns (total 18 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   FlightNumber    60 non-null     int64  
 1   Date            60 non-null     object 
 2   BoosterVersion  60 non-null     object 
 3   PayloadMass     60 non-null     float64
 4   Orbit           60 non-null     object 
 5   LaunchSite      60 non-null     object 
 6   Outcome         60 non-null     object 
 7   Flights         60 non-null     int64  
 8   GridFins        60 non-null     bool   
 9   Reused          60 non-null     bool   
 10  Legs            60 non-null     bool   
 11  LandingPad      55 non-null     object 
 12  Block           60 non-null     float64
 13  ReusedCount     60 non-null     int64  
 14  Serial          60 non-null     object 
 15  Longitude       60 non-null     float64
 16  Latitude        60 non-null     float64
 17  Class           60 non-null     int64  
dtypes: bool(3), float64(4), int64(4), object(7)
memory usage: 7.7+ KB
```

Exploratory data analysis(EDA)



```
In [45]: df_success['Orbit'].value_counts()

Out[45]: GTO      14
         ISS      13
         VLEO     12
         PO        6
         LEO       5
         SSO       5
         MEO       2
         ES-L1     1
         HEO       1
         GEO       1
         Name: Orbit, dtype: int64

In [46]: df_fail['Orbit'].value_counts()

Out[46]: GTO     13
         ISS      8
         PO       3
         LEO      2
         VLEO     2
         MEO      1
         SO       1
         Name: Orbit, dtype: int64
```

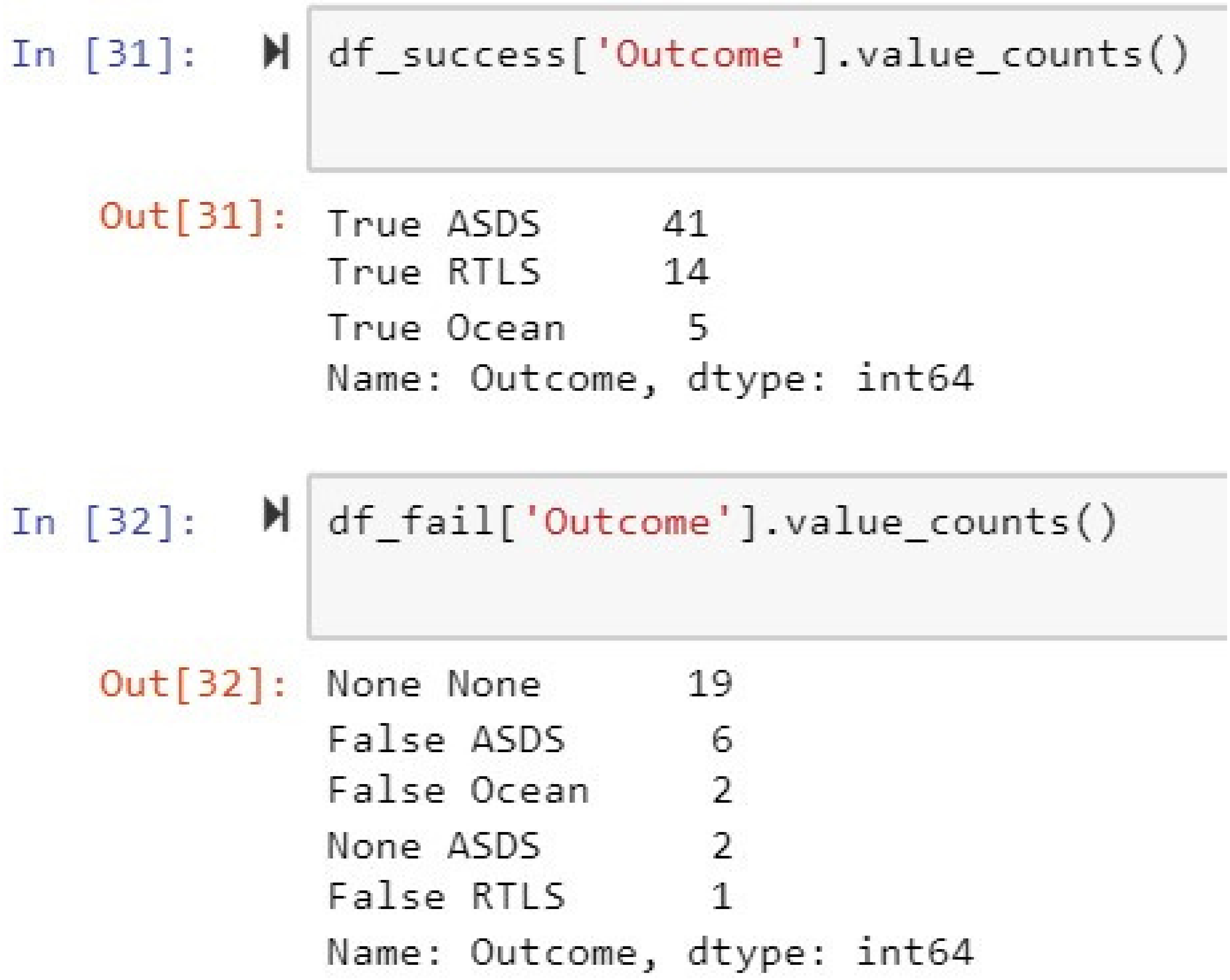
Exploratory data analysis(EDA)



```
In [47]:  # Apply value_counts() on column LaunchSite
          df['LaunchSite'].value_counts()

Out[47]:  CCAFS SLC 40    55
          KSC LC 39A      22
          VAFB SLC 4E     13
          Name: LaunchSite, dtype: int64
```

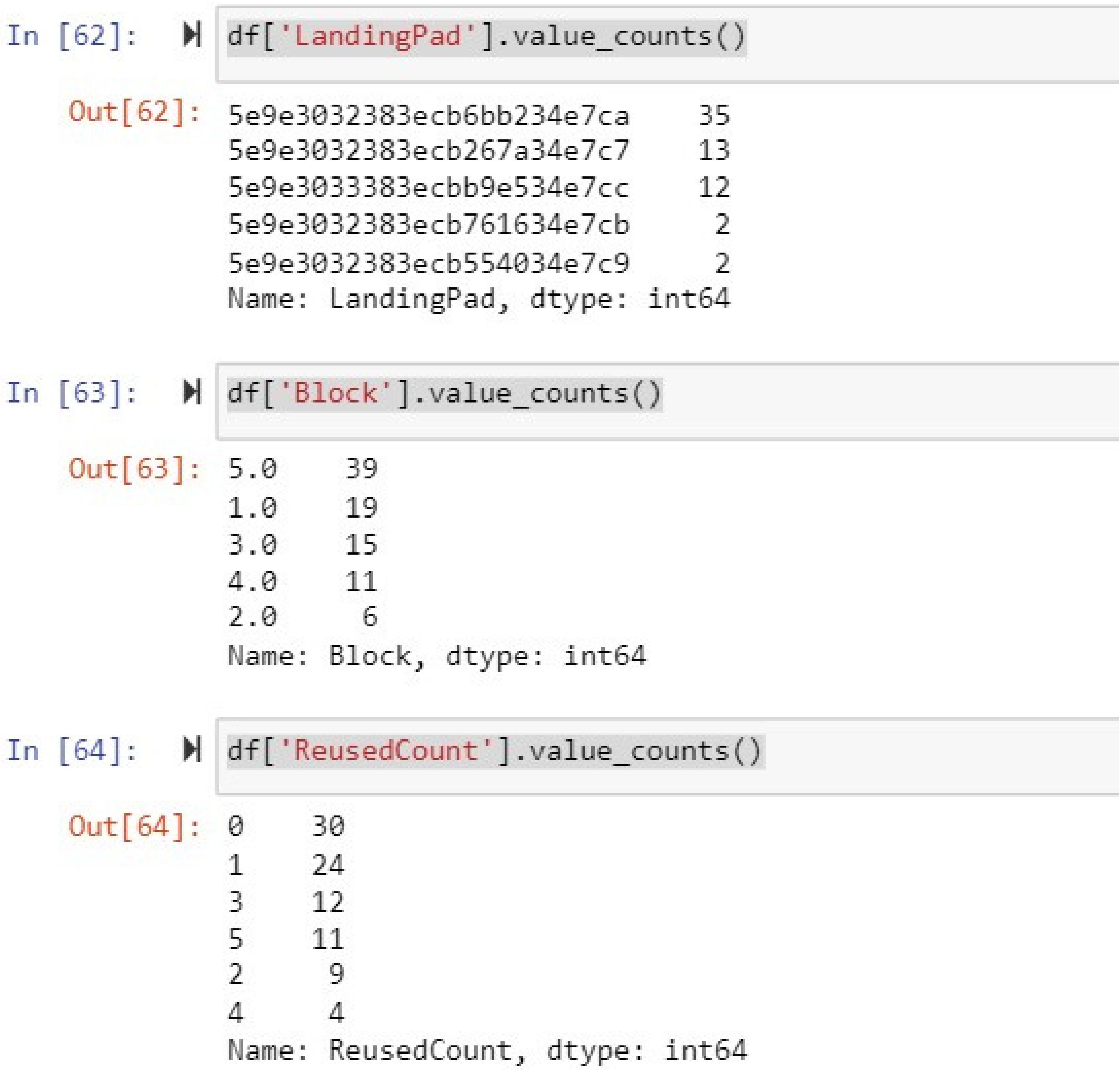
```
In [49]:  # Apply value_counts() on column Outcome
          df['Outcome'].value_counts()

Out[49]:  True ASDS      41
          None None      19
          True RTLS      14
          False ASDS      6
          True Ocean      5
          False Ocean     2
          None ASDS       2
          False RTLS      1
          Name: Outcome, dtype: int64
```

Exploratory data analysis(EDA)



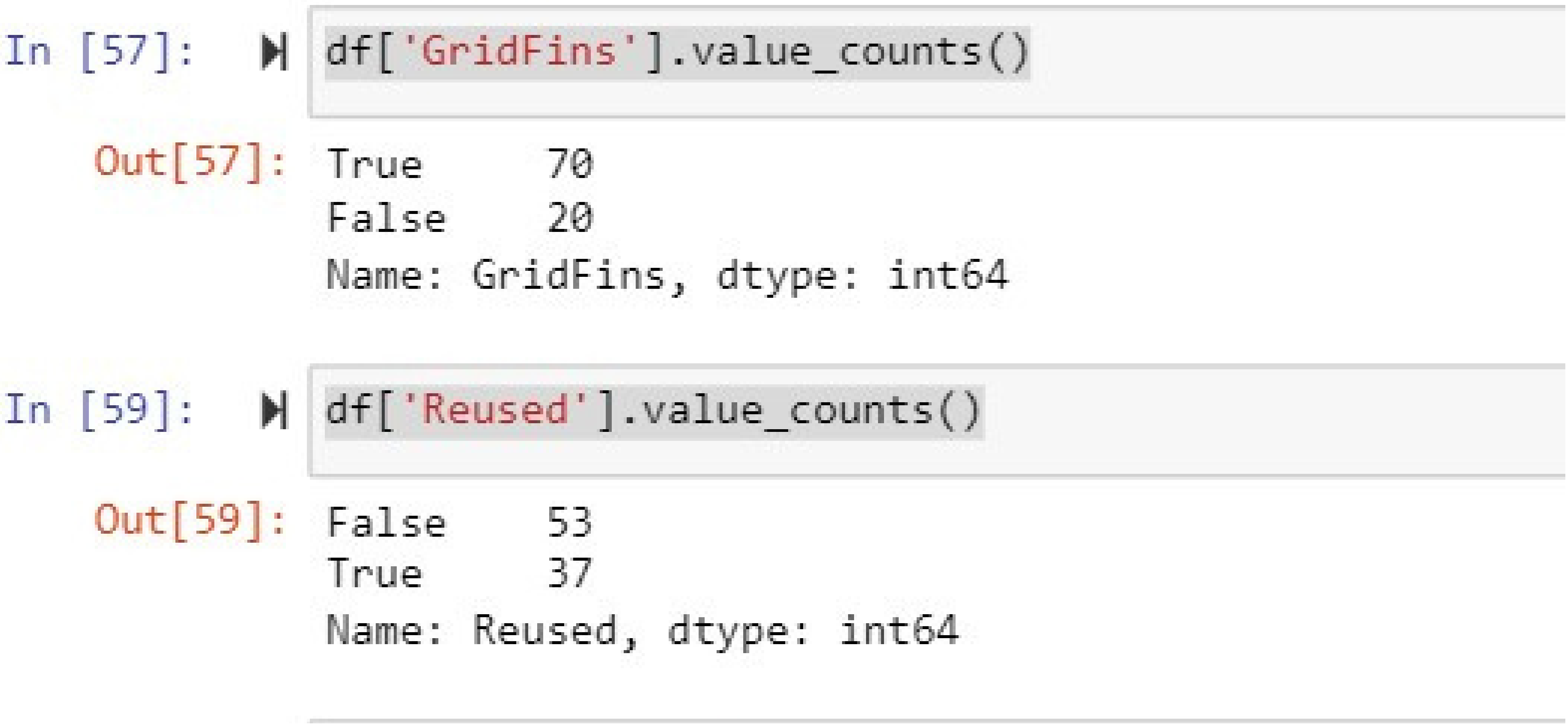
```
In [31]:  df_success['Outcome'].value_counts()

Out[31]: True ASDS     41
         True RTLS     14
         True Ocean     5
         Name: Outcome, dtype: int64

In [32]:  df_fail['Outcome'].value_counts()

Out[32]: None None      19
         False ASDS      6
         False Ocean     2
         None ASDS       2
         False RTLS      1
         Name: Outcome, dtype: int64
```

Exploratory data analysis(EDA)

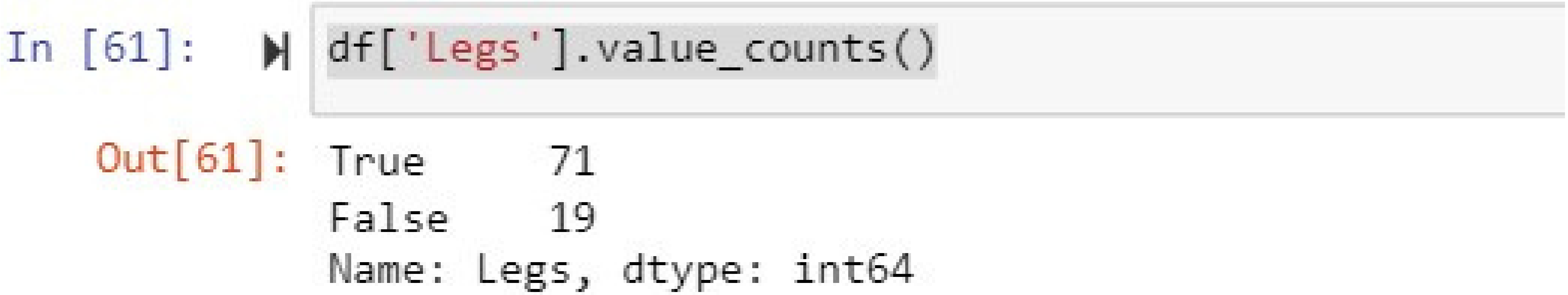


```
In [62]:  df['LandingPad'].value_counts()

Out[62]:  5e9e3032383ecb6bb234e7ca    35
          5e9e3032383ecb267a34e7c7    13
          5e9e3033383ecbb9e534e7cc    12
          5e9e3032383ecb761634e7cb     2
          5e9e3032383ecb554034e7c9     2
          Name: LandingPad, dtype: int64

In [63]:  df['Block'].value_counts()

Out[63]:  5.0    39
          1.0    19
          3.0    15
          4.0    11
          2.0     6
          Name: Block, dtype: int64

In [64]:  df['ReusedCount'].value_counts()

Out[64]:  0    30
          1    24
          3    12
          5    11
          2     9
          4     4
          Name: ReusedCount, dtype: int64
```

Exploratory data analysis(EDA)

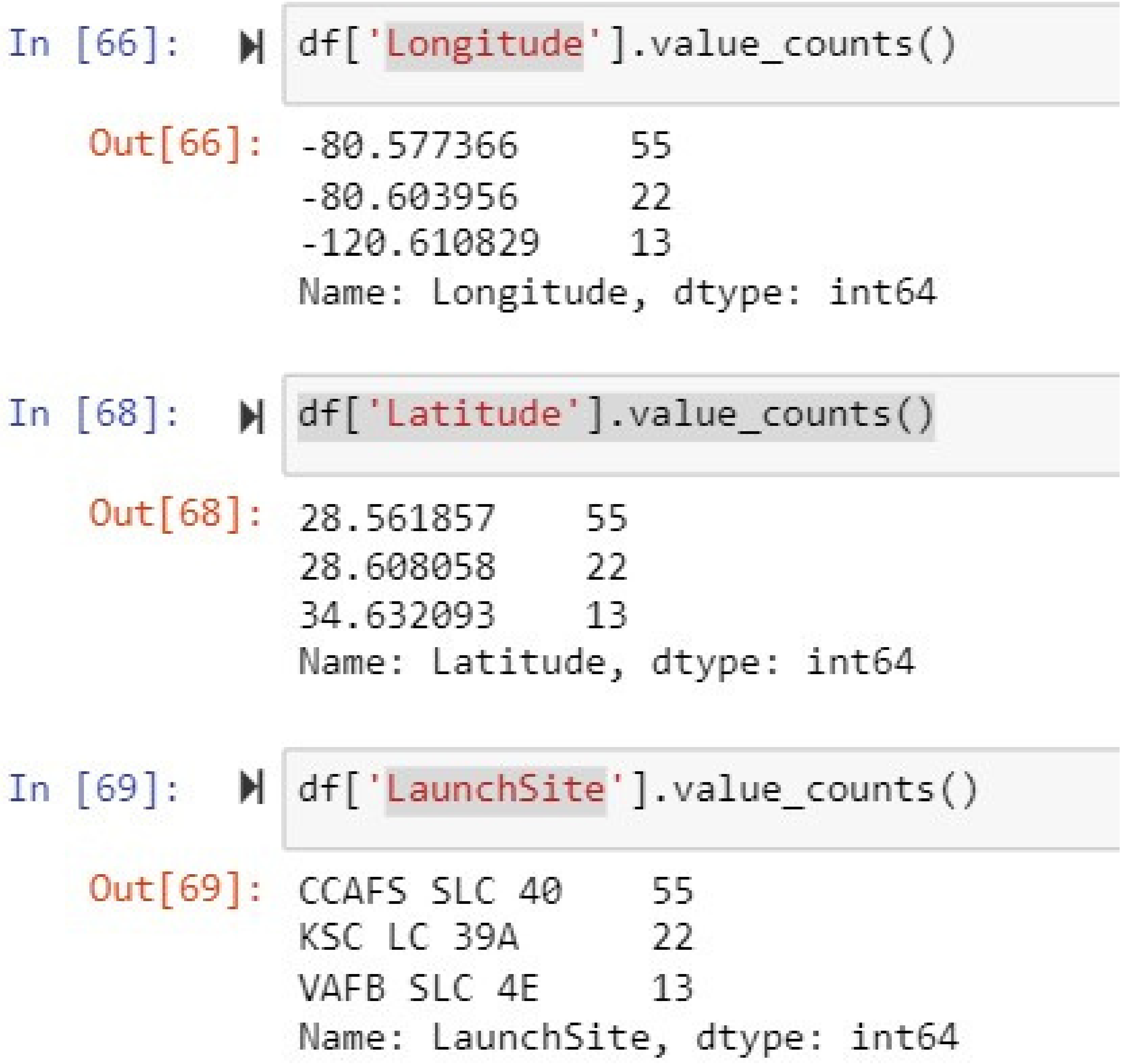


```
In [57]:  df['GridFins'].value_counts()

Out[57]:  True     70
          False    20
          Name: GridFins, dtype: int64

In [59]:  df['Reused'].value_counts()

Out[59]:  False    53
          True     37
          Name: Reused, dtype: int64
```



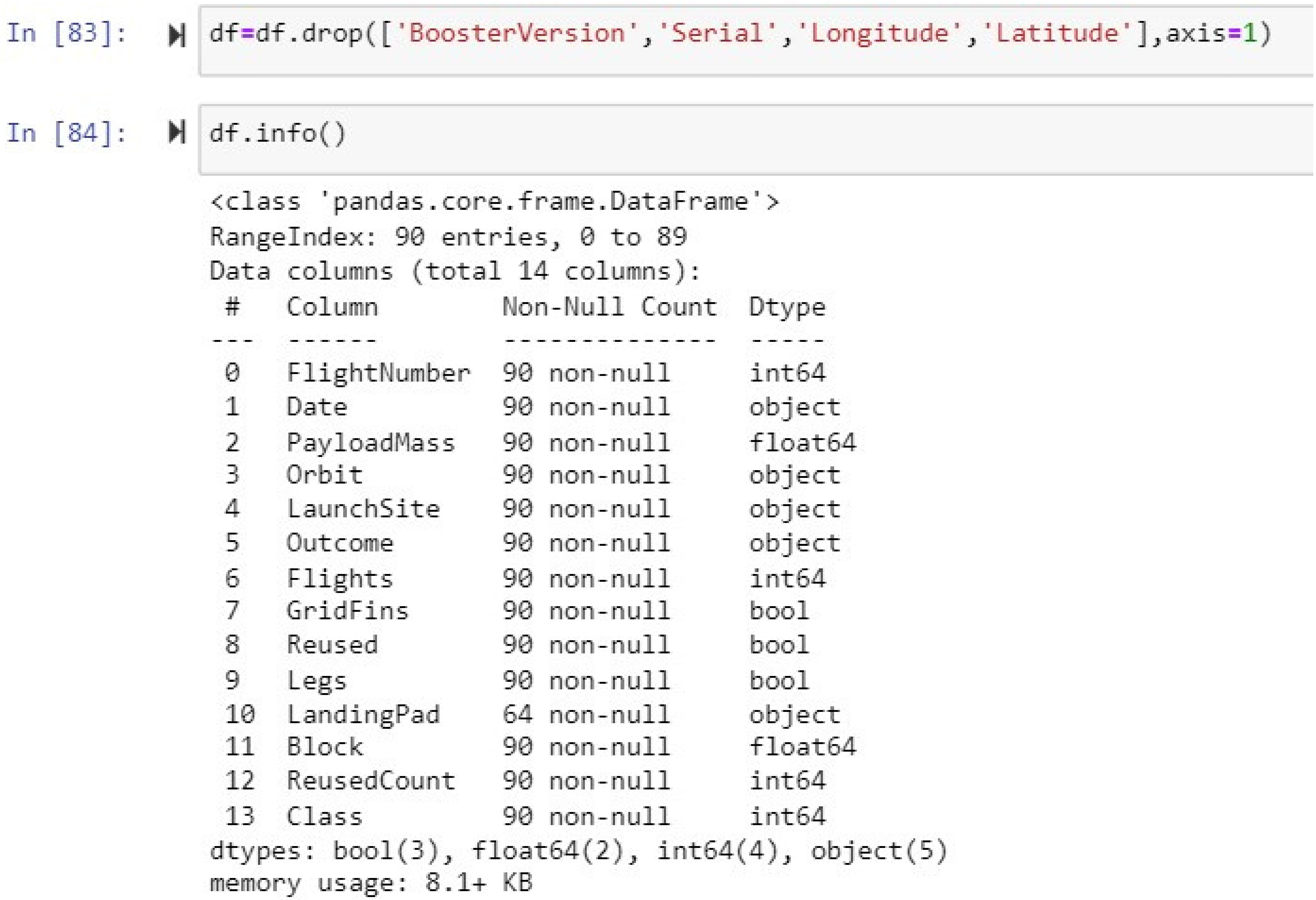
```
In [61]:  df['Legs'].value_counts()

Out[61]:  True     71
          False    19
          Name: Legs, dtype: int64
```

Exploratory data analysis(EDA)



```
In [66]:  df['Longitude'].value_counts()

Out[66]:  -80.577366     55
          -80.603956     22
          -120.610829    13
          Name: Longitude, dtype: int64

In [68]:  df['Latitude'].value_counts()

Out[68]:  28.561857    55
          28.608058    22
          34.632093    13
          Name: Latitude, dtype: int64

In [69]:  df['LaunchSite'].value_counts()

Out[69]:  CCAFS SLC 40    55
          KSC LC 39A      22
          VAFB SLC 4E     13
          Name: LaunchSite, dtype: int64
```

Exploratory data analysis(EDA)

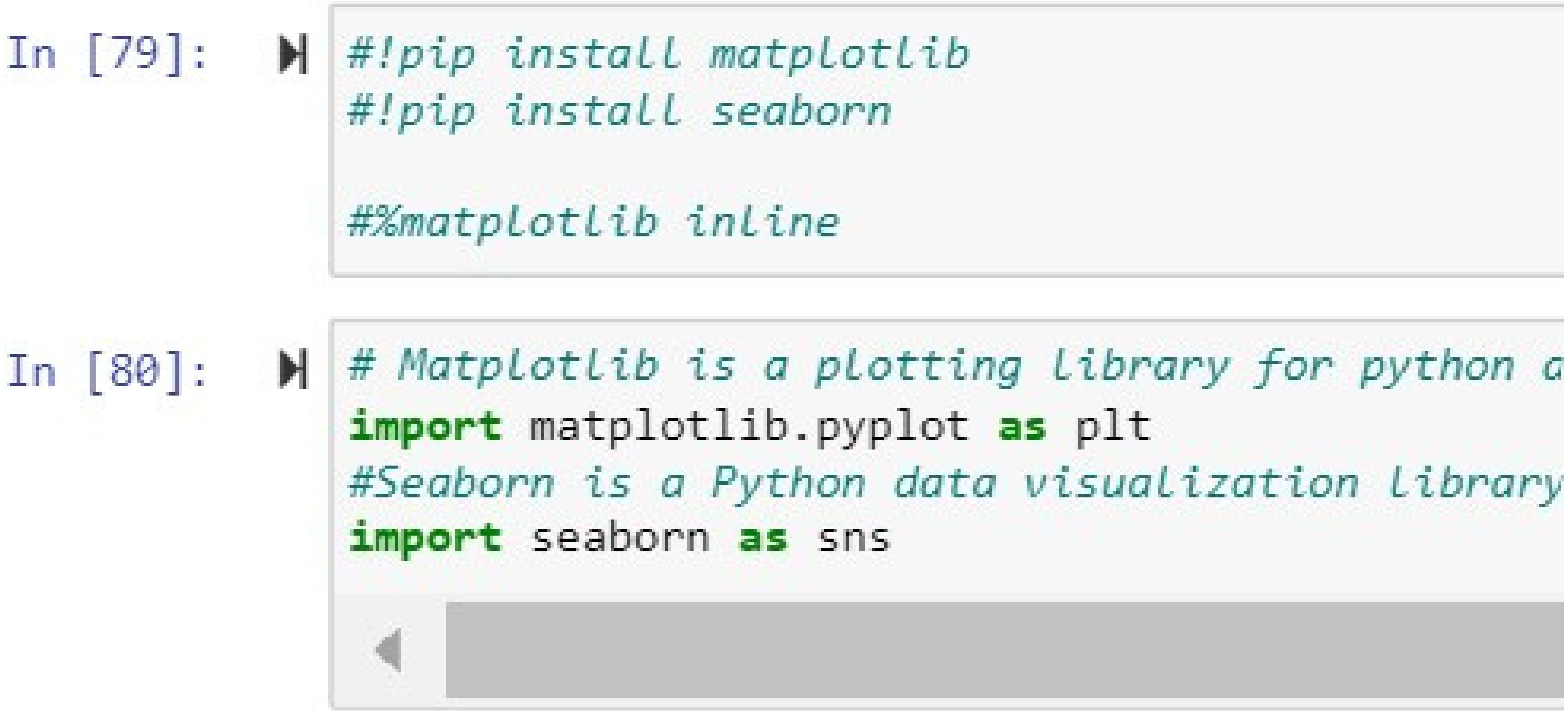


```
In [83]:  df=df.drop(['BoosterVersion','Serial','Longitude','Latitude'],axis=1)
```

```
In [84]:  df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   FlightNumber   90 non-null     int64  
 1   Date           90 non-null     object 
 2   PayloadMass    90 non-null     float64
 3   Orbit          90 non-null     object 
 4   LaunchSite     90 non-null     object 
 5   Outcome        90 non-null     object 
 6   Flights        90 non-null     int64  
 7   GridFins       90 non-null     bool   
 8   Reused         90 non-null     bool   
 9   Legs           90 non-null     bool   
 10  LandingPad     64 non-null     object 
 11  Block          90 non-null     float64
 12  ReusedCount    90 non-null     int64  
 13  Class          90 non-null     int64  
dtypes: bool(3), float64(2), int64(4), object(5)
memory usage: 8.1+ KB
```

Visualization Libraries in Python



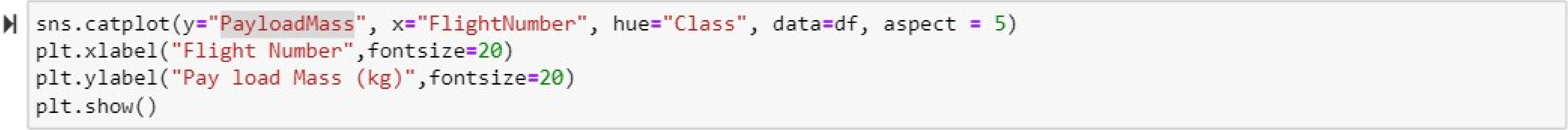
In [79]:

```
#!pip install matplotlib
#!pip install seaborn

#%matplotlib inline
```

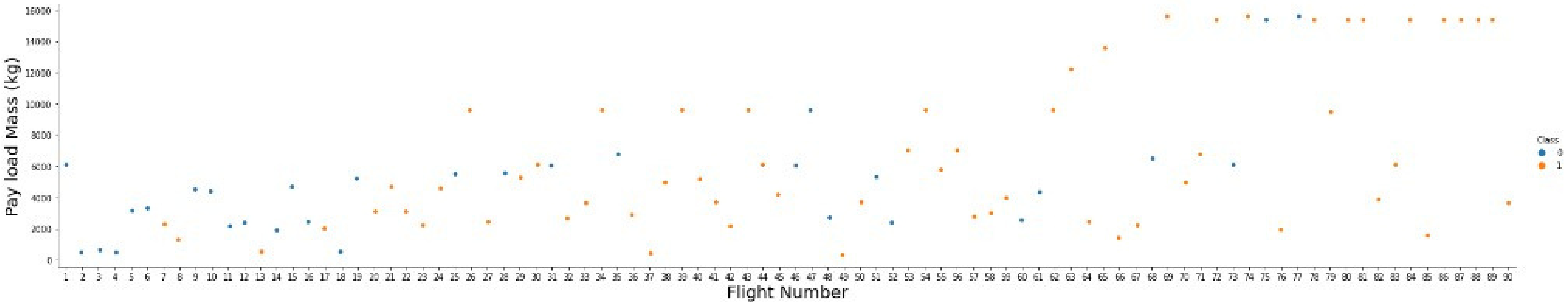
In [80]:

```
# Matplotlib is a plotting library for python a
import matplotlib.pyplot as plt
#Seaborn is a Python data visualization library
import seaborn as sns
```

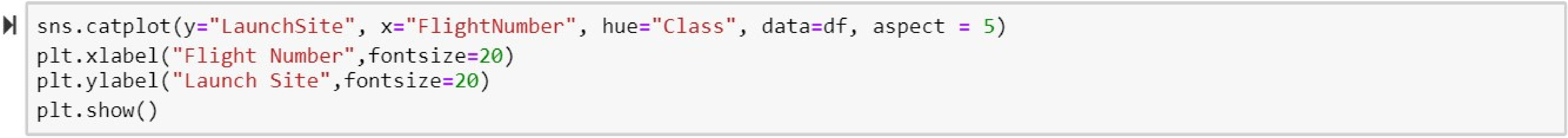
Visualize the relationship between Flight Number and Payload Mass



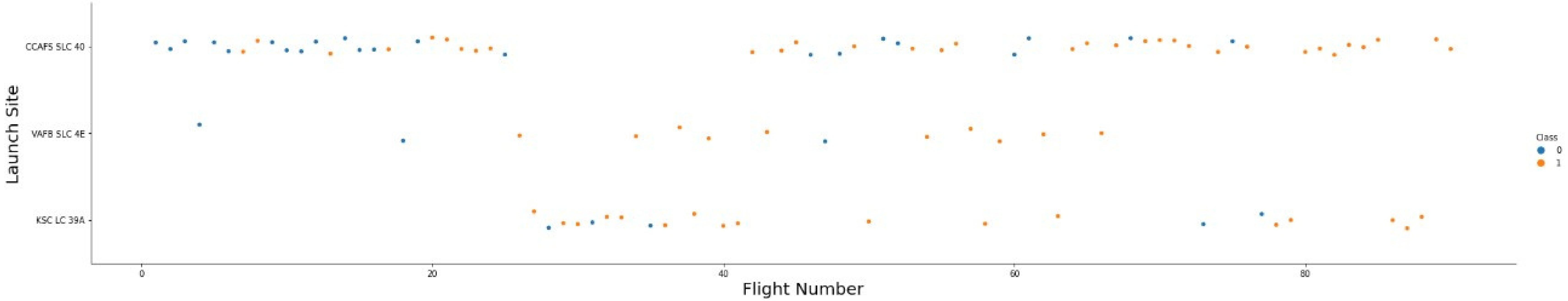
```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



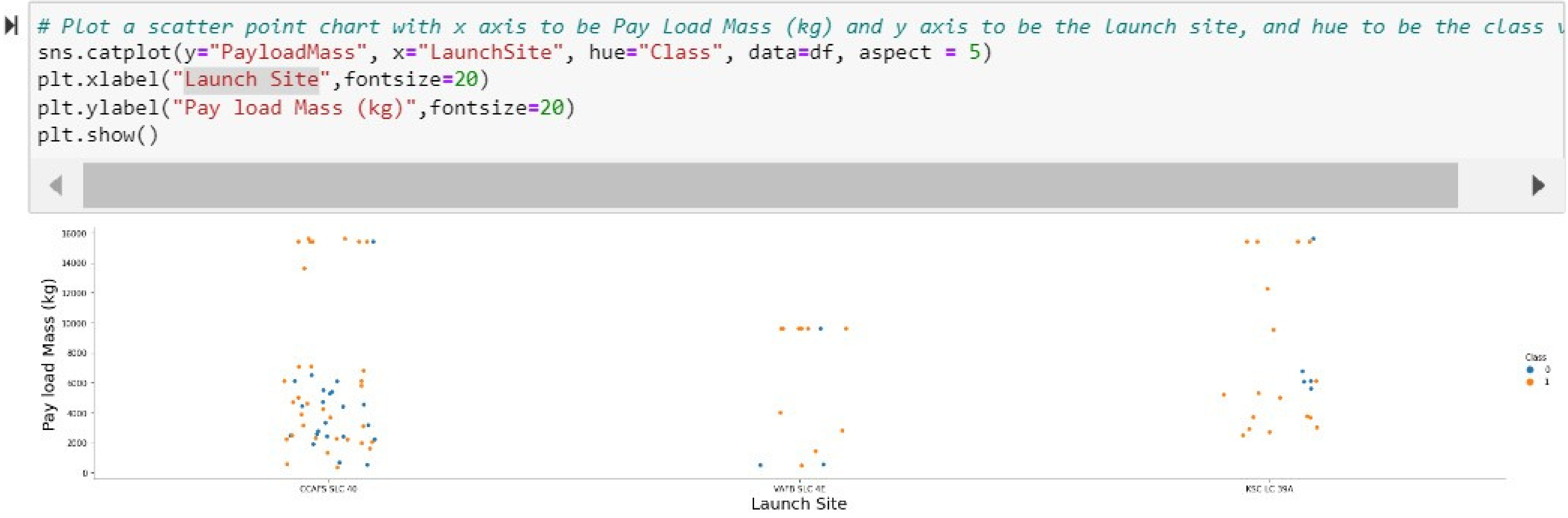
Visualize the relationship between Flight Number and Launch Site



```
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

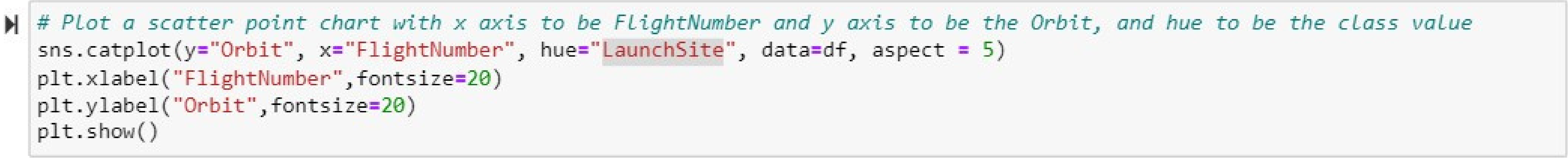


Visualize the relationship between Payload and Launch Site

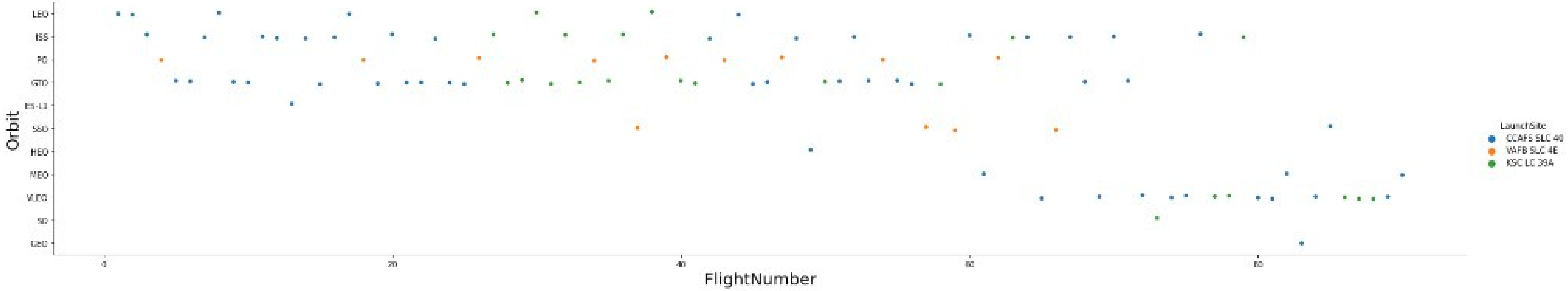


```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class v
sns.catplot(y="PayloadMass", x="LaunchSite", hue="Class", data=df, aspect = 5)
plt.xlabel("Launch Site",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

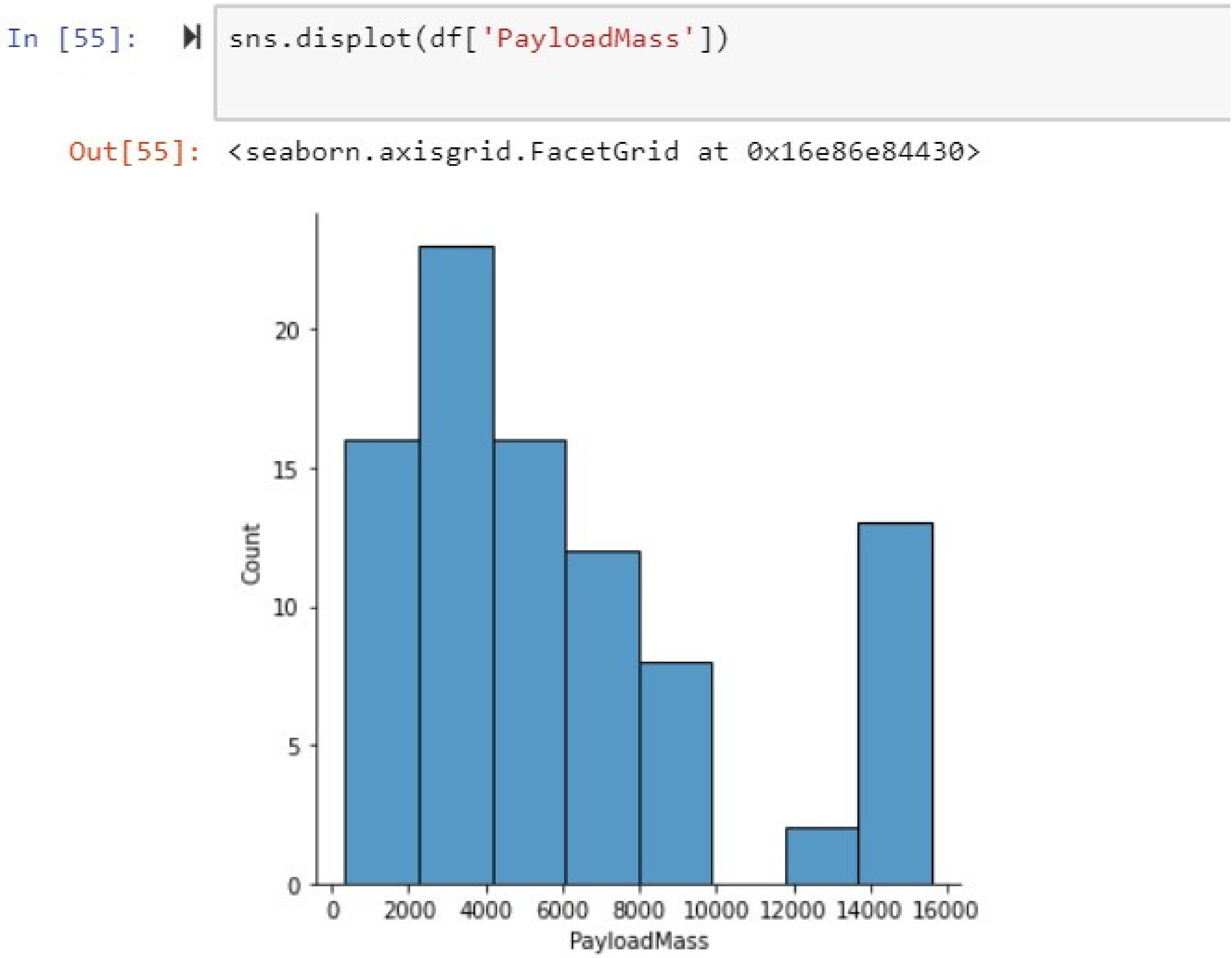
Visualize the relationship between FlightNumber and Orbit type



```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="LaunchSite", data=df, aspect = 5)
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

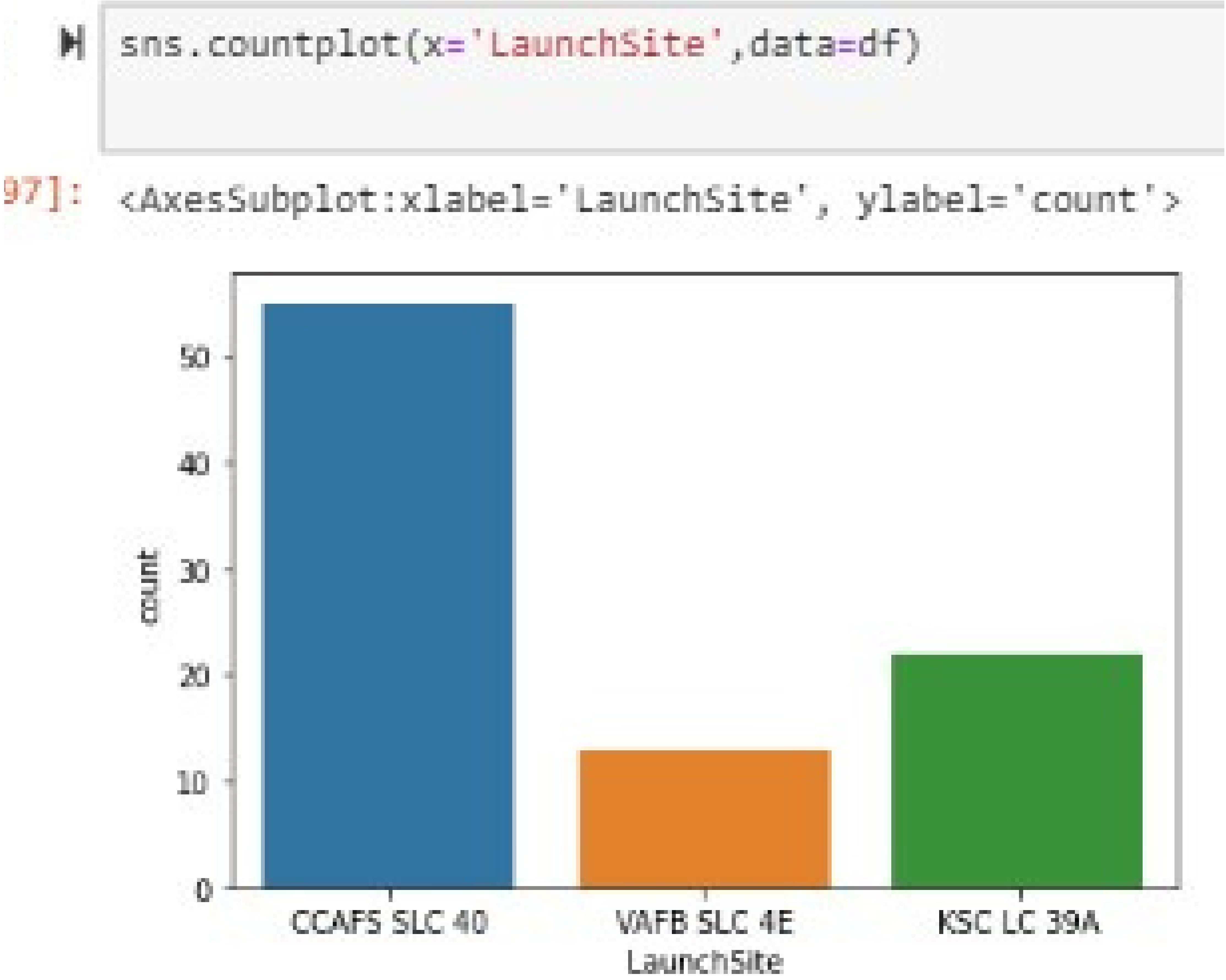


Histogram



```
In [55]:  sns.displot(df['PayloadMass'])

Out[55]: <seaborn.axisgrid.FacetGrid at 0x16e86e84430>
```

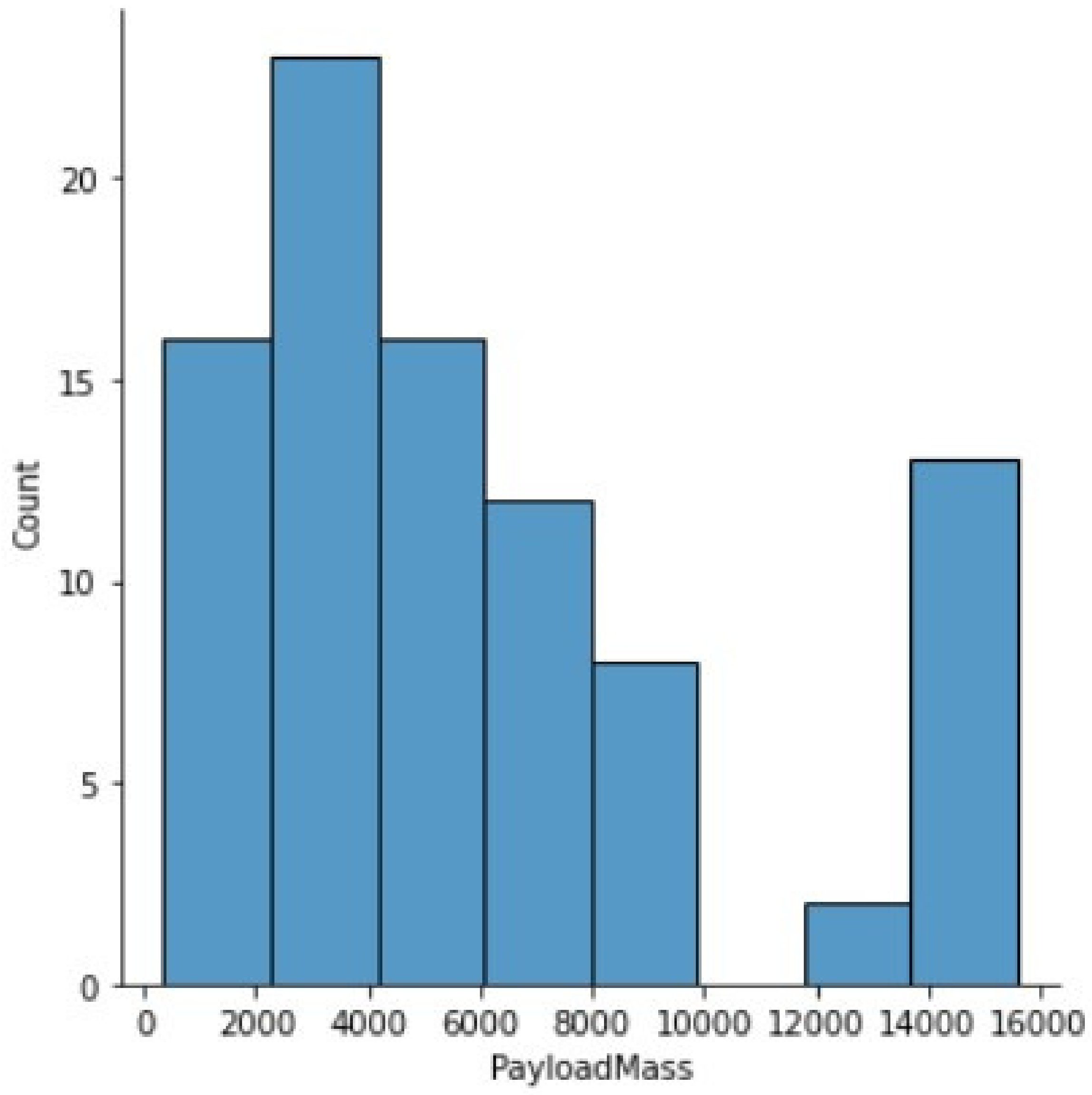
Bar Chart

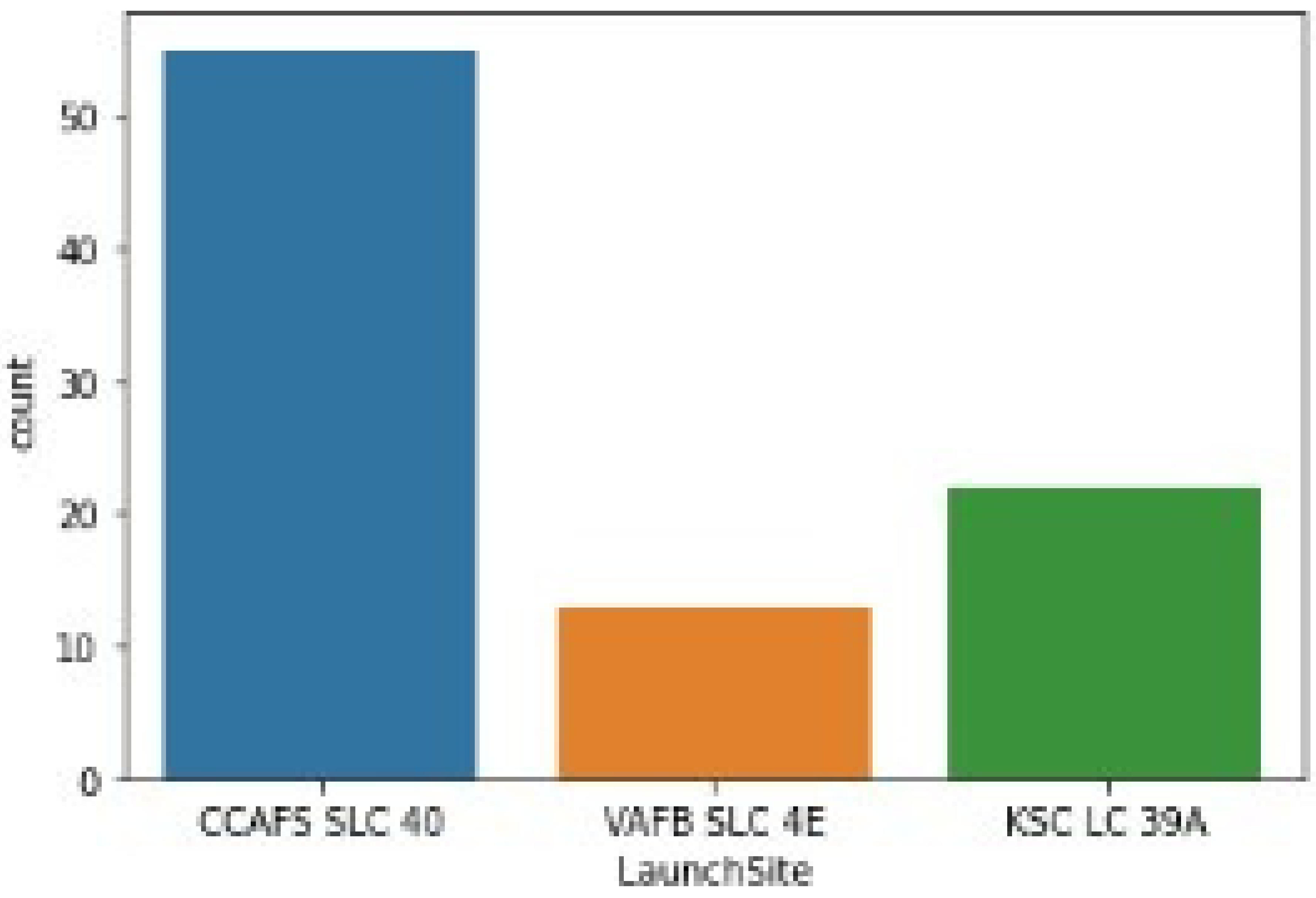


```
sns.countplot(x='LaunchSite',data=df)
```

```
97]:  <AxesSubplot:xlabel='LaunchSite', ylabel='count'>
```

Histogram vs Bar Plot





Assignment:

تمرین:

کدهای ارائه شده در درس را در نوتبوک جدیدی انجام داده و در صورت نیاز از نوتبوک هفته چهارم استفاده کنید.

در صورتیکه علاقمند به تمرین بیشتر با کتابخانه های matplotlib , seaborn هستید جلسه ششم دوره منتورینگ دیتاساینس را در کانال یوتیوب ملاحظه کنید.