دوره دیتا ساینس کاربردی

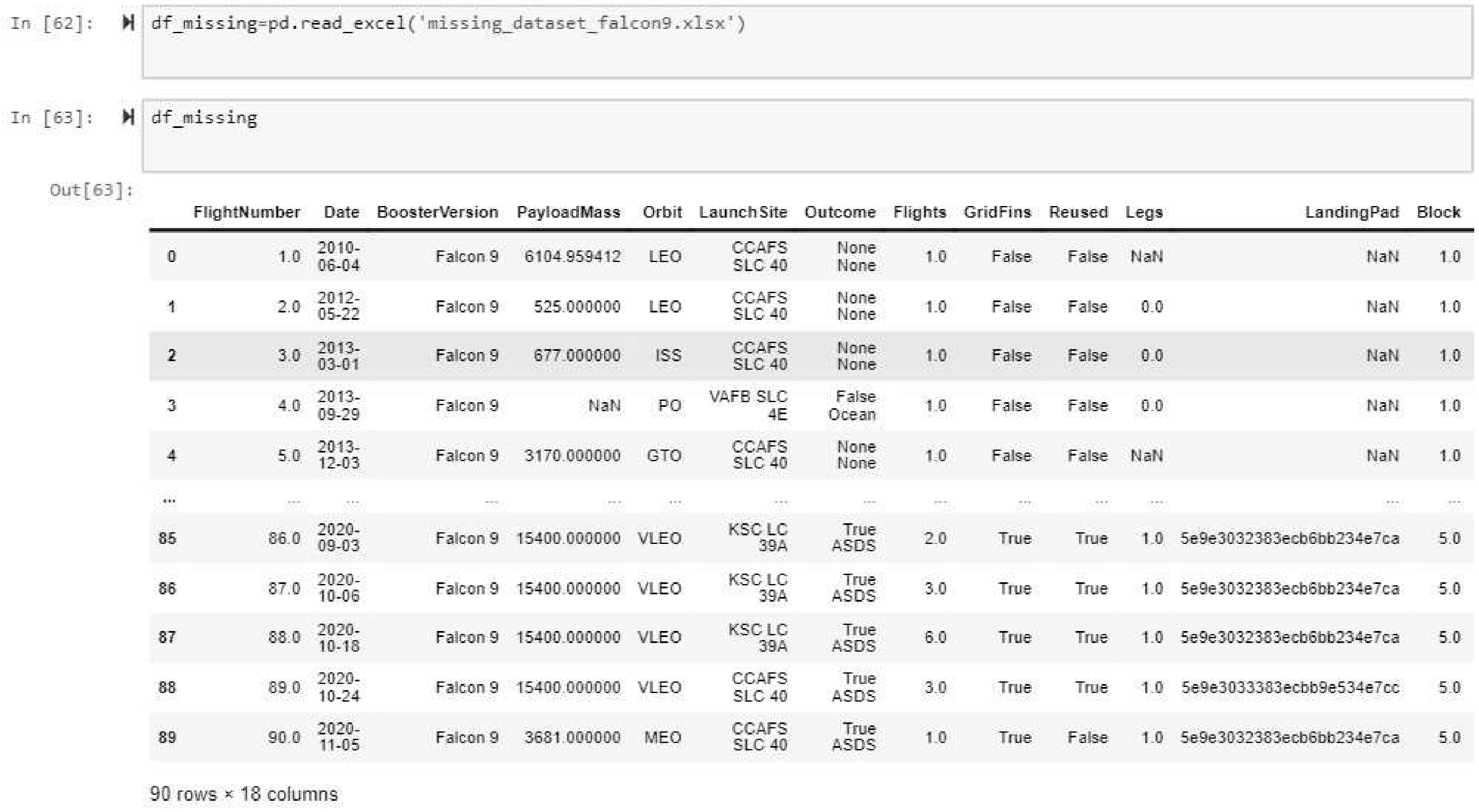
Data pre-processing

dataroadmap

مدرس: مونا حاتمی

جلسه پنجم

Read Excel file in Pandas



```
In [62]:  df_missing=pd.read_excel('missing_dataset_falcon9.xlsx')
```

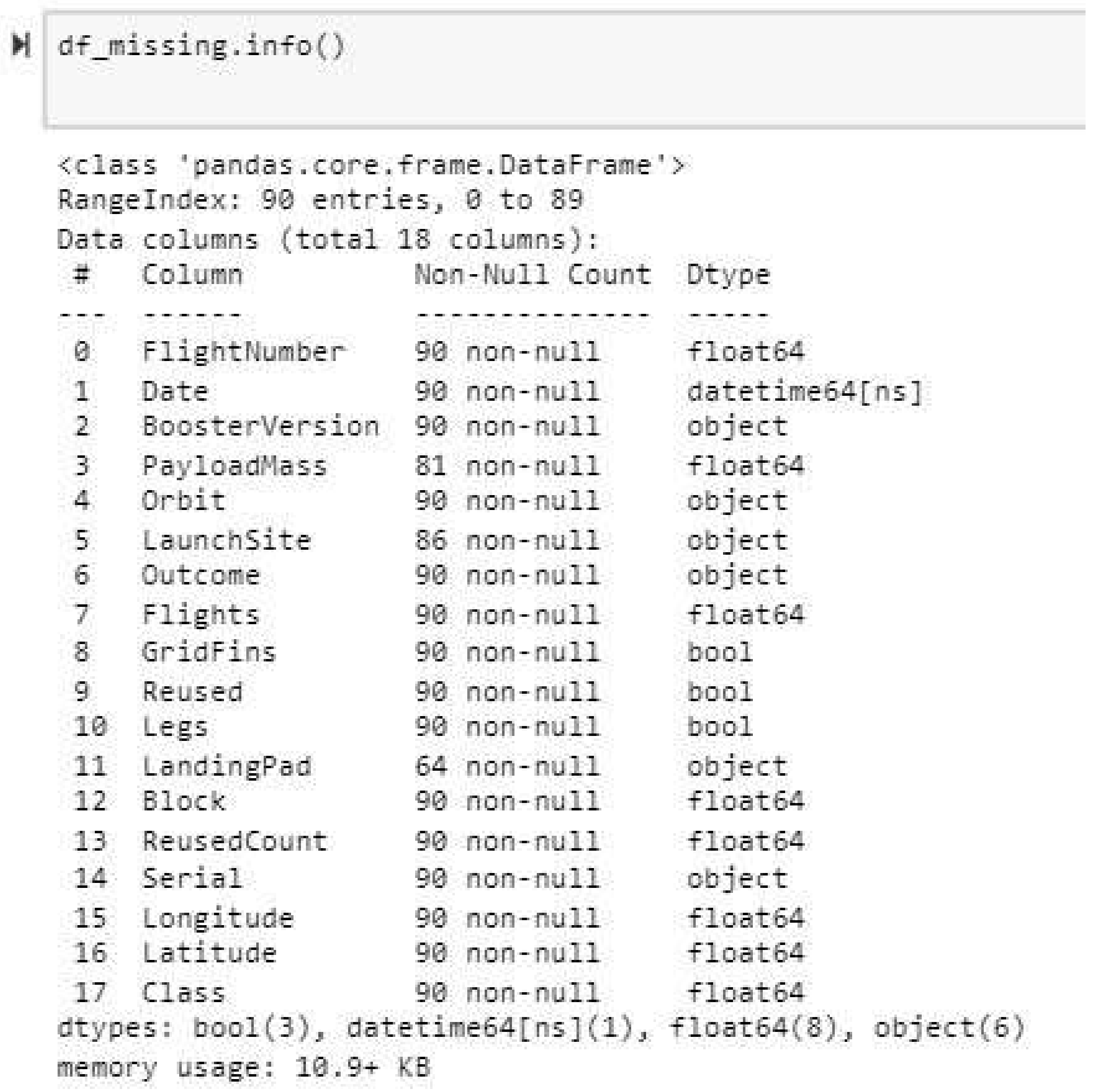
```
In [63]:  df_missing
```

Out[63]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
0	1.0	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1.0	False	False	NaN	NaN	1.0
1	2.0	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1.0	False	False	0.0	NaN	1.0
2	3.0	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1.0	False	False	0.0	NaN	1.0
3	4.0	2013-09-29	Falcon 9	NaN	PO	VAFB SLC 4E	False Ocean	1.0	False	False	0.0	NaN	1.0
4	5.0	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1.0	False	False	NaN	NaN	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
85	86.0	2020-09-03	Falcon 9	15400.000000	VLEO	KSC LC 39A	True ASDS	2.0	True	True	1.0	5e9e3032383ecb6bb234e7ca	5.0
86	87.0	2020-10-06	Falcon 9	15400.000000	VLEO	KSC LC 39A	True ASDS	3.0	True	True	1.0	5e9e3032383ecb6bb234e7ca	5.0
87	88.0	2020-10-18	Falcon 9	15400.000000	VLEO	KSC LC 39A	True ASDS	6.0	True	True	1.0	5e9e3032383ecb6bb234e7ca	5.0
88	89.0	2020-10-24	Falcon 9	15400.000000	VLEO	CCAFS SLC 40	True ASDS	3.0	True	True	1.0	5e9e3033383ecbb9e534e7cc	5.0
89	90.0	2020-11-05	Falcon 9	3681.000000	MEO	CCAFS SLC 40	True ASDS	1.0	True	False	1.0	5e9e3032383ecb6bb234e7ca	5.0

90 rows × 18 columns

Exploratory data analysis



```
df_missing.info()
```

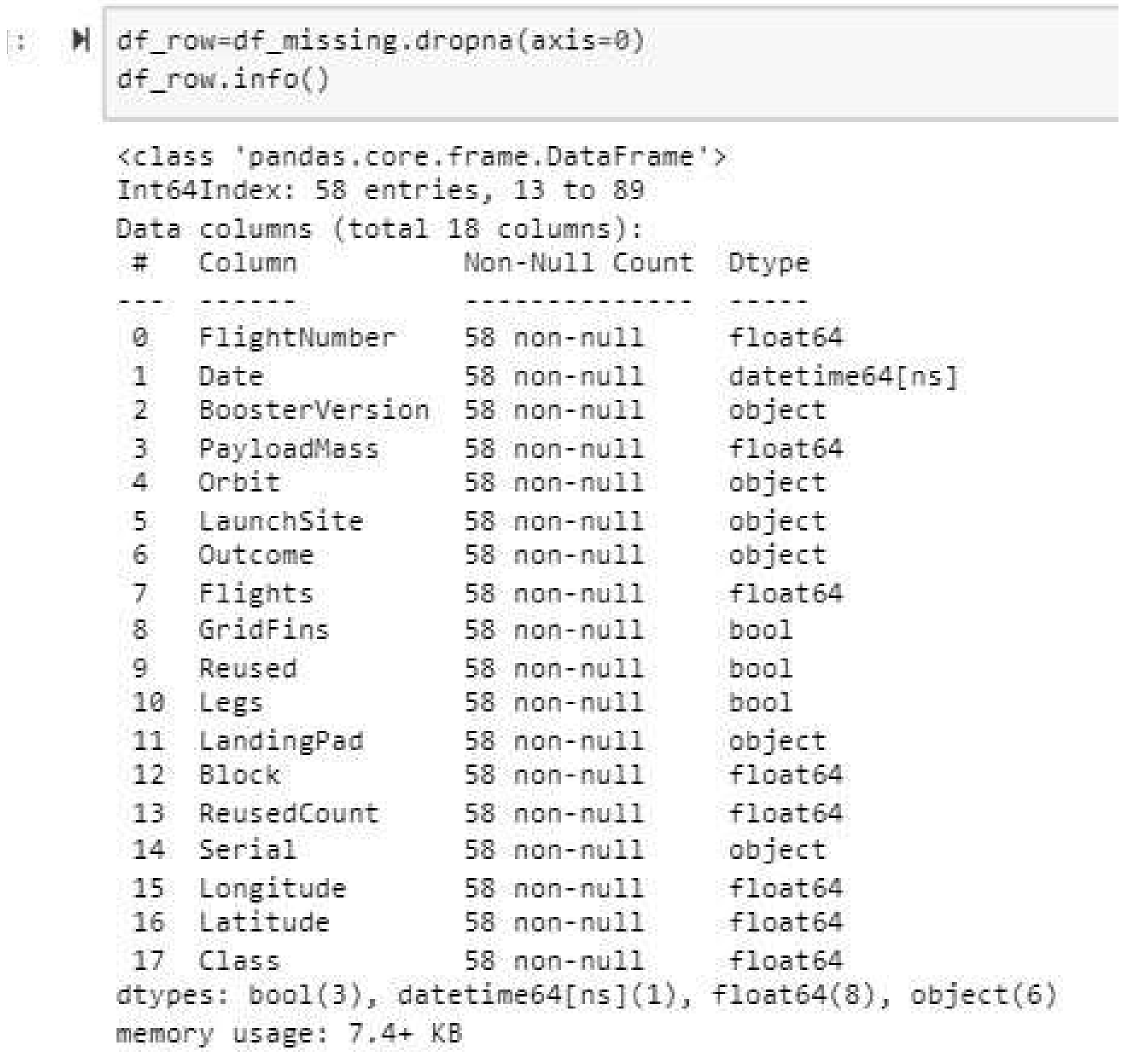
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 18 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   FlightNumber    90 non-null     float64
 1   Date            90 non-null     datetime64[ns]
 2   BoosterVersion  90 non-null     object
 3   PayloadMass     81 non-null     float64
 4   Orbit           90 non-null     object
 5   LaunchSite      86 non-null     object
 6   Outcome         90 non-null     object
 7   Flights         90 non-null     float64
 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     float64
 14  Serial          90 non-null     object
 15  Longitude       90 non-null     float64
 16  Latitude        90 non-null     float64
 17  Class           90 non-null     float64
dtypes: bool(3), datetime64[ns](1), float64(8), object(6)
memory usage: 10.9+ KB
```

How to deal with missing data?

Drop or Replace?

a. Drop the whole row

b. Drop the whole column



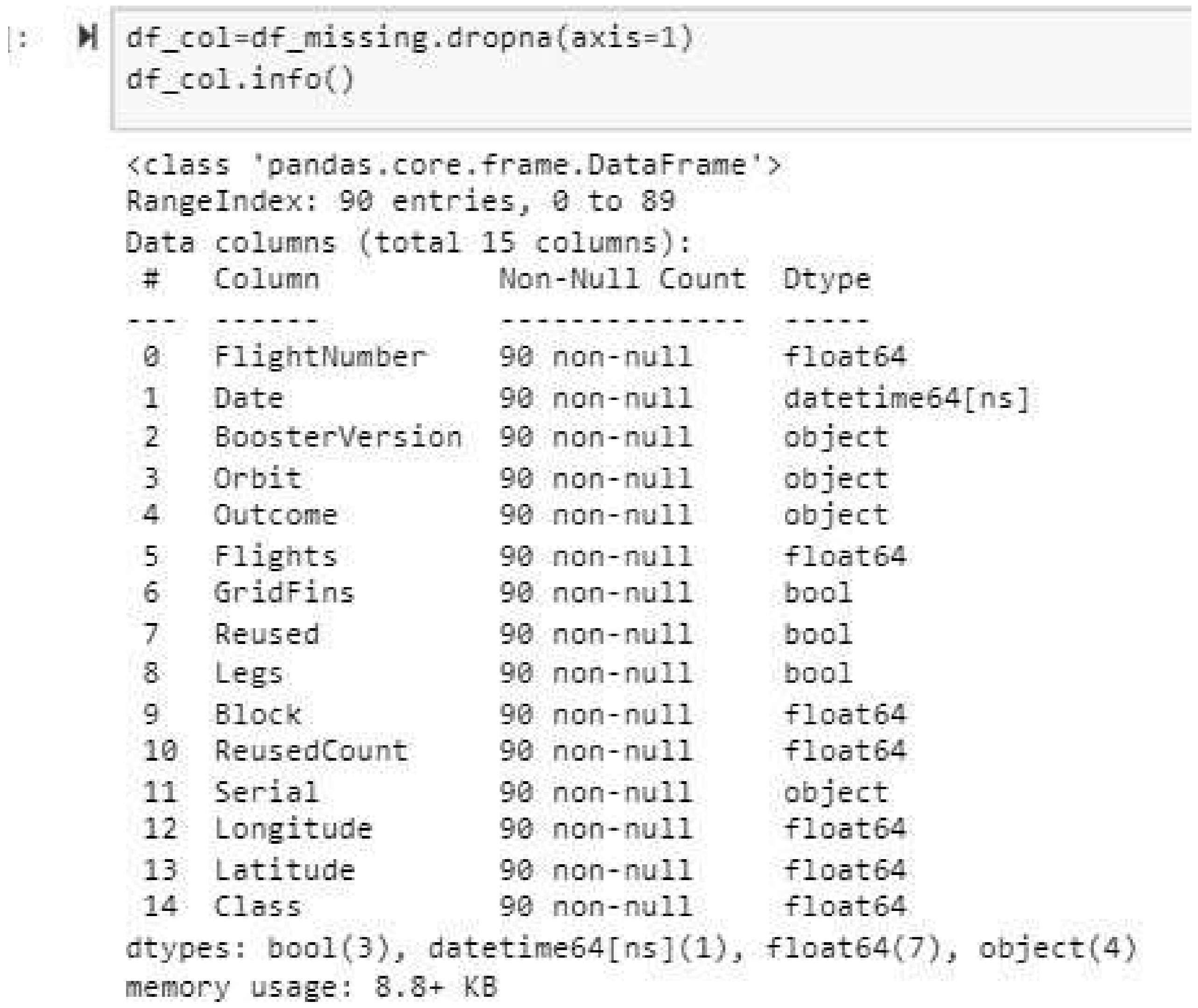
```
df_row=df_missing.dropna(axis=0)
df_row.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 58 entries, 13 to 89
Data columns (total 18 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   FlightNumber    58 non-null     float64
 1   Date            58 non-null     datetime64[ns]
 2   BoosterVersion  58 non-null     object
 3   PayloadMass     58 non-null     float64
 4   Orbit           58 non-null     object
 5   LaunchSite      58 non-null     object
 6   Outcome         58 non-null     object
 7   Flights         58 non-null     float64
 8   GridFins        58 non-null     bool
 9   Reused          58 non-null     bool
 10  Legs            58 non-null     bool
 11  LandingPad      58 non-null     object
 12  Block           58 non-null     float64
 13  ReusedCount     58 non-null     float64
 14  Serial          58 non-null     object
 15  Longitude       58 non-null     float64
 16  Latitude        58 non-null     float64
 17  Class           58 non-null     float64
dtypes: bool(3), datetime64[ns](1), float64(8), object(6)
memory usage: 7.4+ KB
```

How to deal with missing data? Drop or Replace?

a. Drop the whole row

b. Drop the whole column



```
[ ]: df_col=df_missing.dropna(axis=1)
     df_col.info()
```

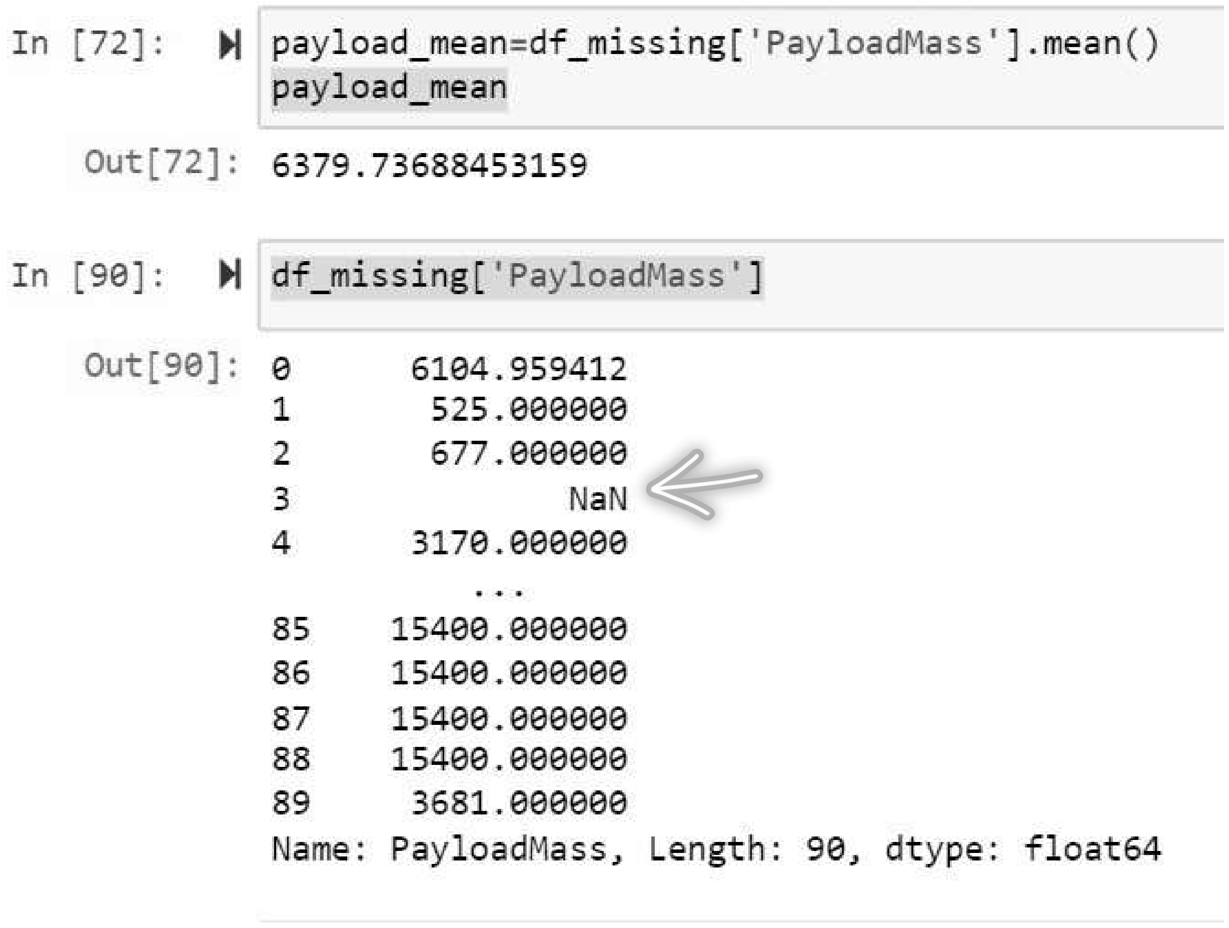
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   FlightNumber    90 non-null     float64
 1   Date            90 non-null     datetime64[ns]
 2   BoosterVersion  90 non-null     object
 3   Orbit           90 non-null     object
 4   Outcome         90 non-null     object
 5   Flights         90 non-null     float64
 6   GridFins        90 non-null     bool
 7   Reused          90 non-null     bool
 8   Legs            90 non-null     bool
 9   Block           90 non-null     float64
 10  ReusedCount     90 non-null     float64
 11  Serial          90 non-null     object
 12  Longitude       90 non-null     float64
 13  Latitude        90 non-null     float64
 14  Class           90 non-null     float64
dtypes: bool(3), datetime64[ns](1), float64(7), object(4)
memory usage: 8.8+ KB
```

How to deal with missing data?

Replace data

a. Replace it by mean

b. Replace it by frequency



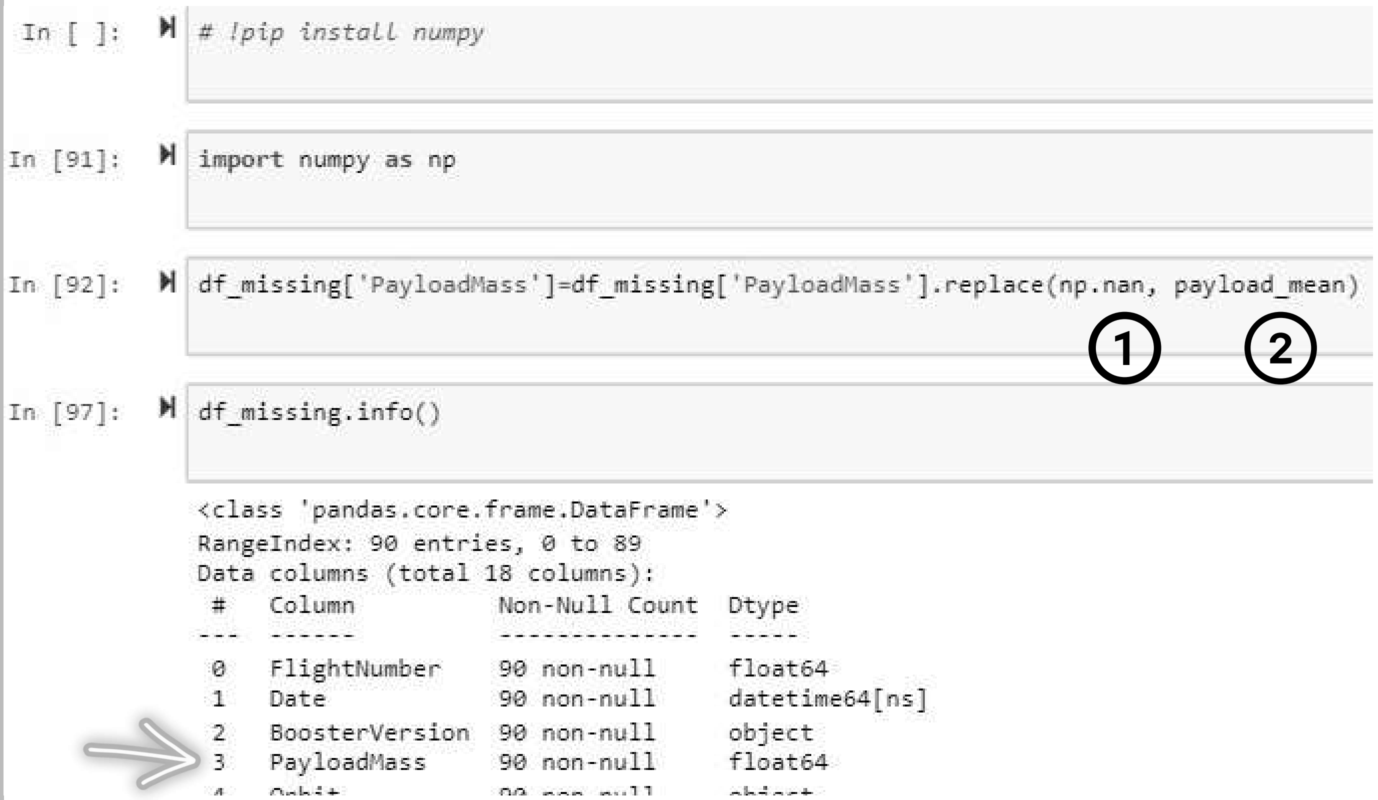
```
In [72]:   payload_mean=df_missing['PayloadMass'].mean()
           payload_mean

Out[72]:   6379.73688453159

In [90]:   df_missing['PayloadMass']

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

How to deal with missing data?

Replace data

a. Replace it by mean

b. Replace it by frequency



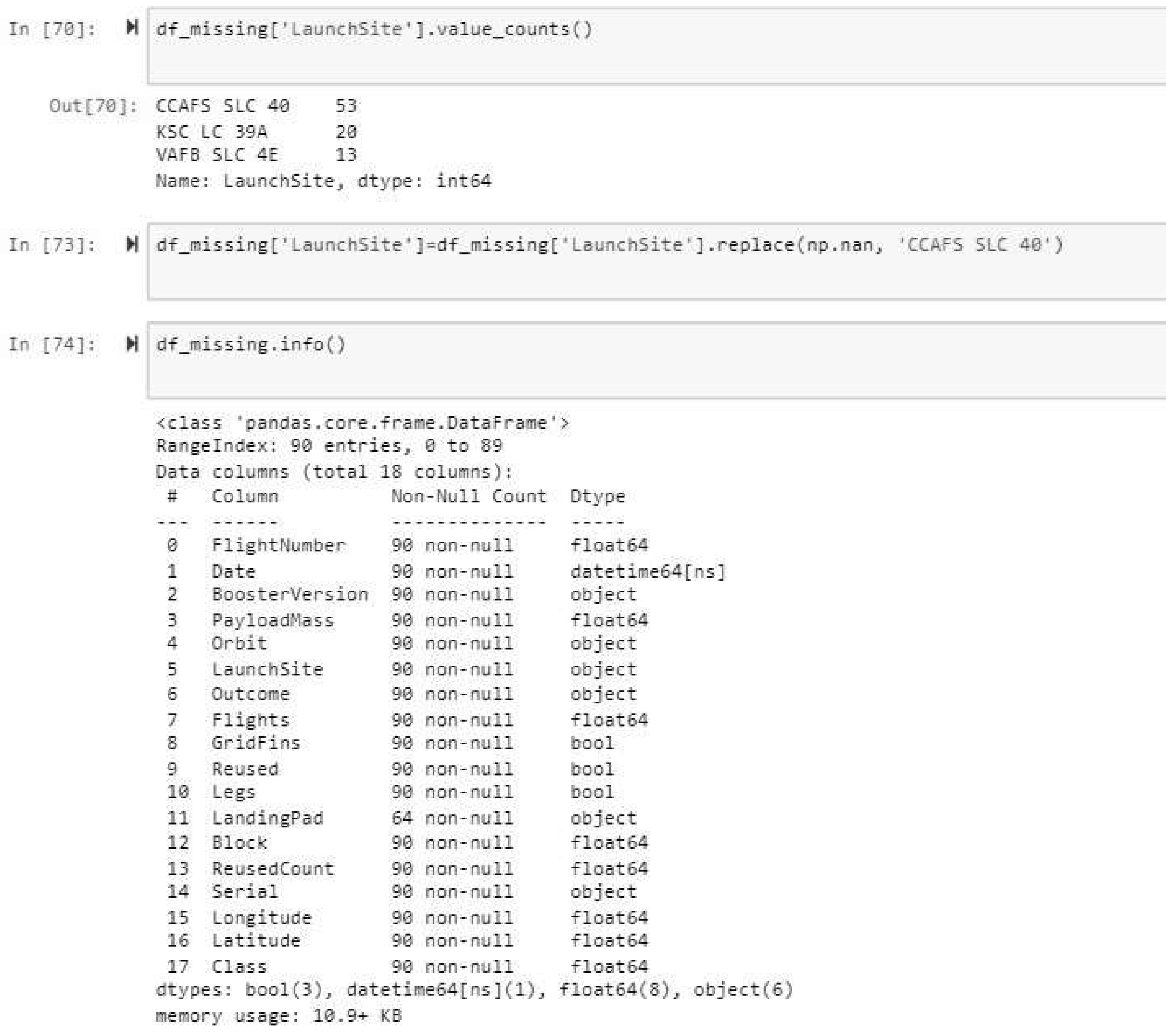
```
In [ ]:   # !pip install numpy

In [91]:  import numpy as np

In [92]:  df_missing['PayloadMass']=df_missing['PayloadMass'].replace(np.nan, payload_mean)

In [97]:  df_missing.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     float64
 1   Date            90 non-null     datetime64[ns]
 2   BoosterVersion  90 non-null     object
 3   PayloadMass     90 non-null     float64
 4   Orbit           90 non-null     object
```

How to deal with missing data?

Replace data

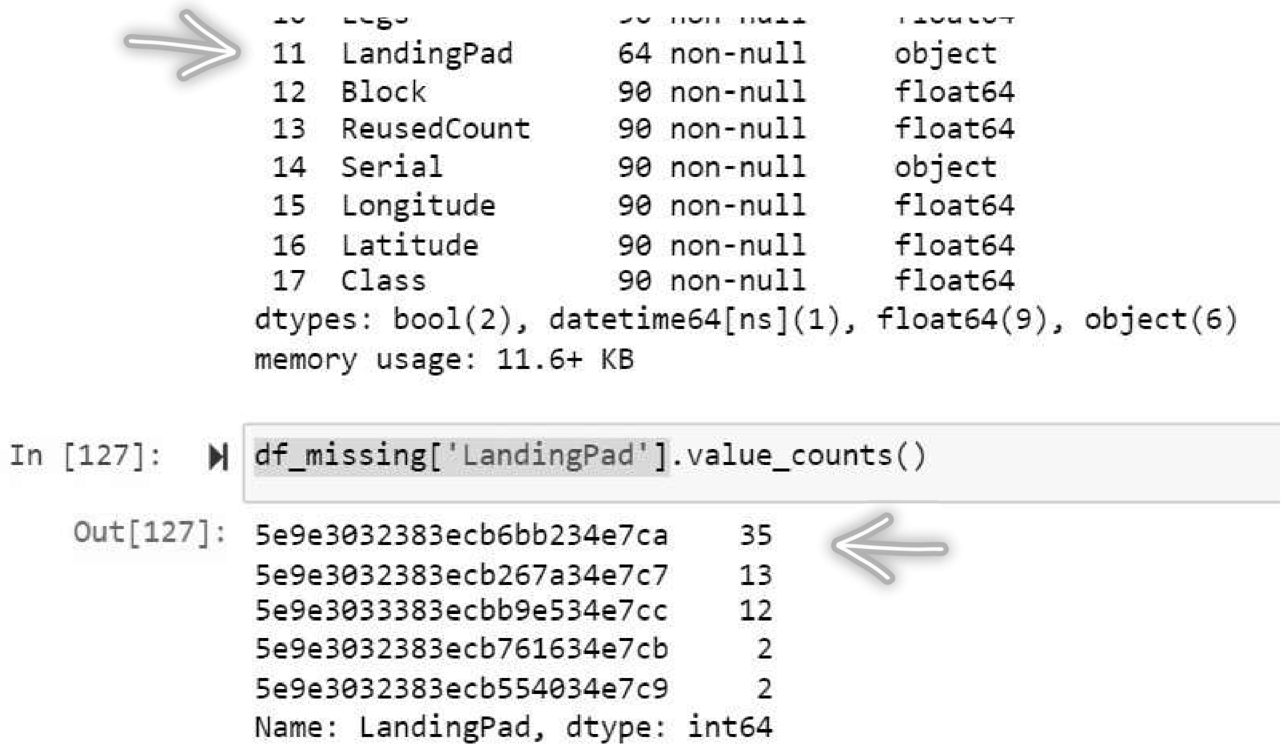
a. Replace it by mean

b. Replace it by frequency



```
In [70]:  df_missing['LaunchSite'].value_counts()

Out[70]: CCAFS SLC 40    53
          KSC LC 39A      20
          VAFB SLC 4E     13
          Name: LaunchSite, dtype: int64

In [73]:  df_missing['LaunchSite']=df_missing['LaunchSite'].replace(np.nan, 'CCAFS SLC 40')

In [74]:  df_missing.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     float64
 1   Date            90 non-null     datetime64[ns]
 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     float64
 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     float64
 14  Serial          90 non-null     object
 15  Longitude       90 non-null     float64
 16  Latitude        90 non-null     float64
 17  Class           90 non-null     float64
dtypes: bool(3), datetime64[ns](1), float64(8), object(6)
memory usage: 10.9+ KB
```

Exploratory data analysis

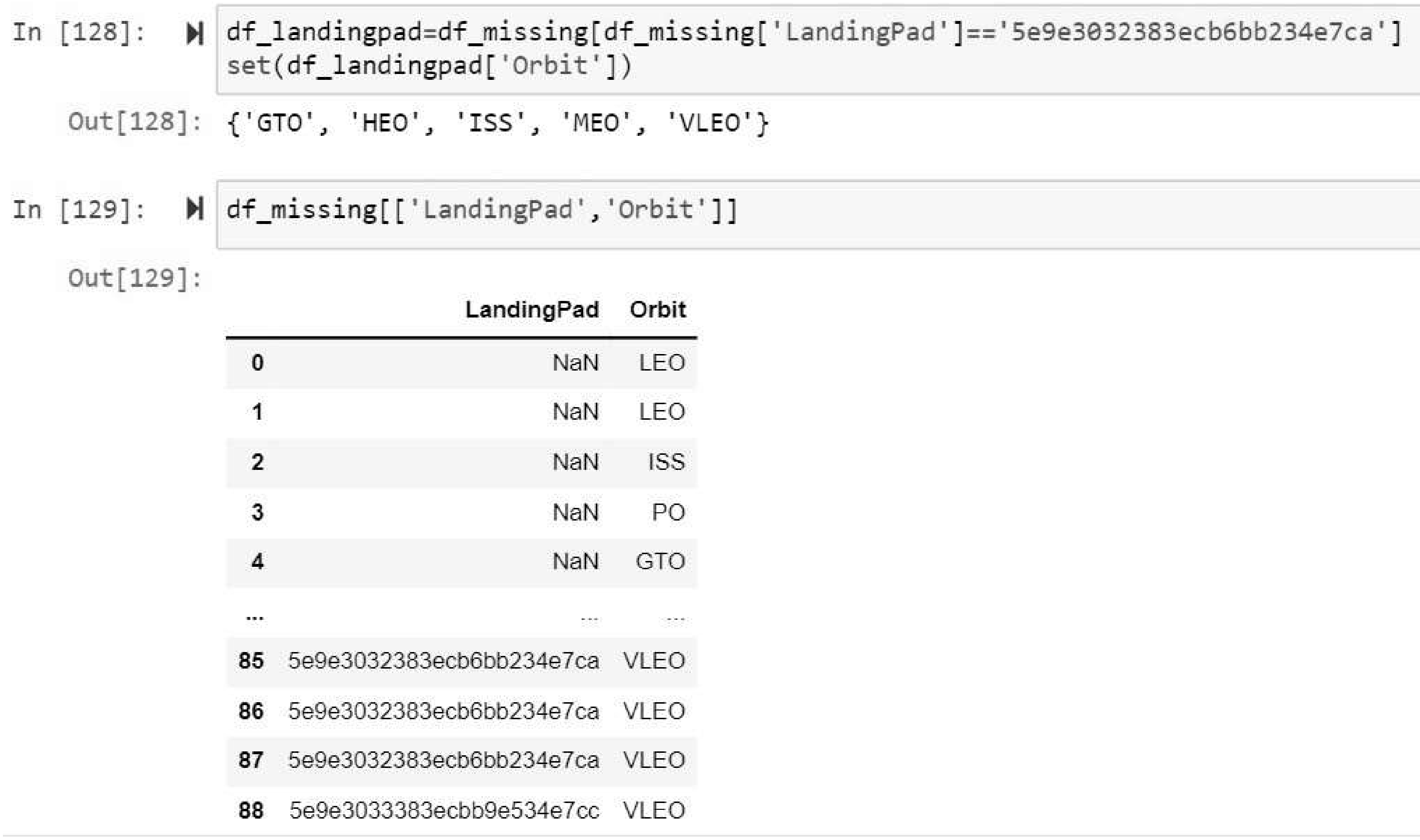


```
 11   LandingPad      64 non-null     object
 12   Block           90 non-null     float64
 13   ReusedCount     90 non-null     float64
 14   Serial          90 non-null     object
 15   Longitude       90 non-null     float64
 16   Latitude        90 non-null     float64
 17   Class           90 non-null     float64
dtypes: bool(2), datetime64[ns](1), float64(9), object(6)
memory usage: 11.6+ KB
```

In [127]:
```
df_missing['LandingPad'].value_counts()
```

Out[127]:
```
5e9e3032383ecb6bb234e7ca    35
5e9e3032383ecb267a34e7c7    13
5e9e3033383ecbb9e534e7cc    12
5e9e3032383ecb761634e7cb     2
5e9e3032383ecb554034e7c9     2
Name: LandingPad, dtype: int64
```

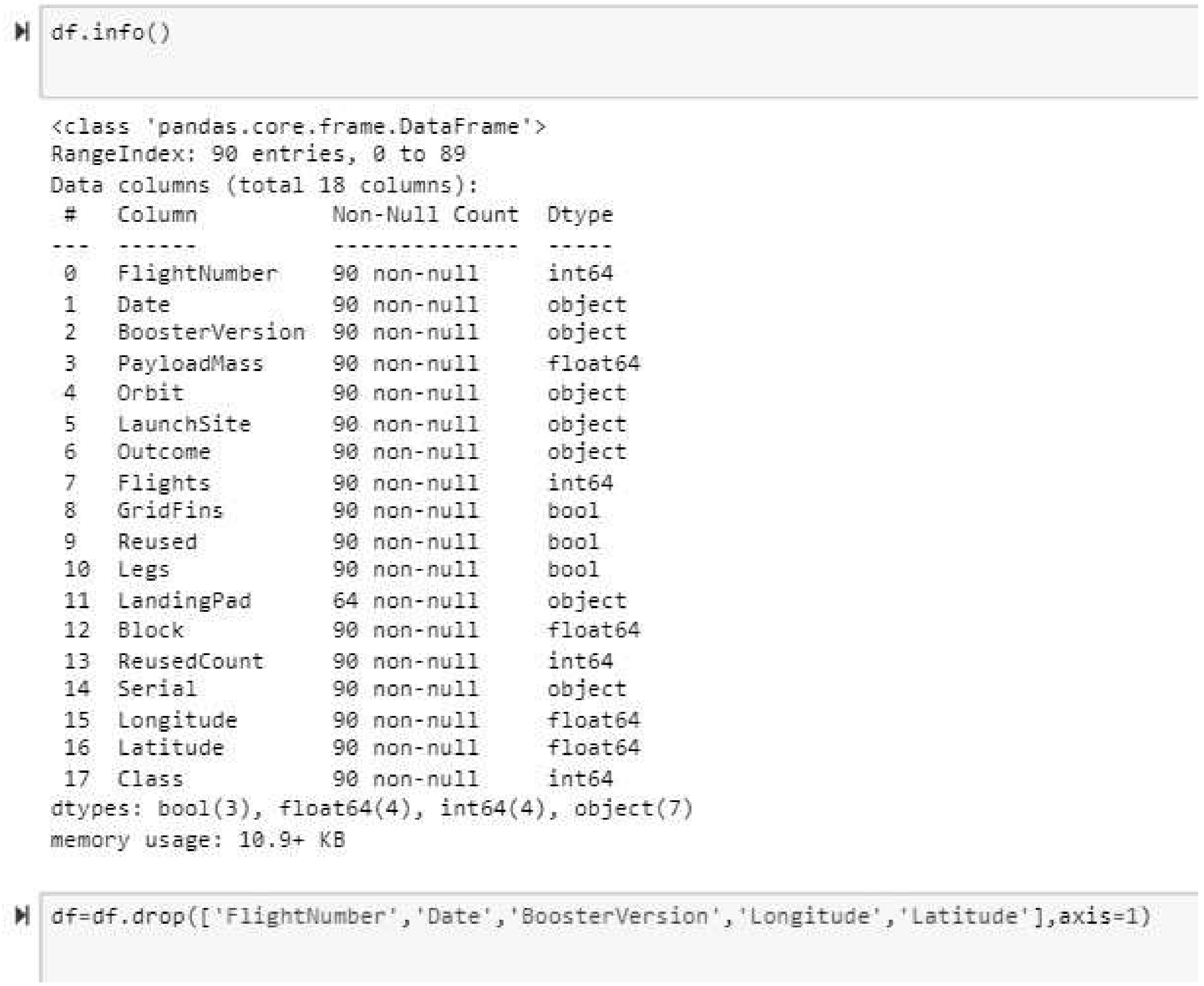
Exploratory data analysis



```
In [128]:  df_landingpad=df_missing[df_missing['LandingPad']=='5e9e3032383ecb6bb234e7ca']
           set(df_landingpad['Orbit'])

Out[128]:  {'GTO', 'HEO', 'ISS', 'MEO', 'VLEO'}

In [129]:  df_missing[['LandingPad','Orbit']]

Out[129]:
```

	LandingPad	Orbit
0	NaN	LEO
1	NaN	LEO
2	NaN	ISS
3	NaN	PO
4	NaN	GTO
...	...	...
85	5e9e3032383ecb6bb234e7ca	VLEO
86	5e9e3032383ecb6bb234e7ca	VLEO
87	5e9e3032383ecb6bb234e7ca	VLEO
88	5e9e3033383ecbb9e534e7cc	VLEO

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
```

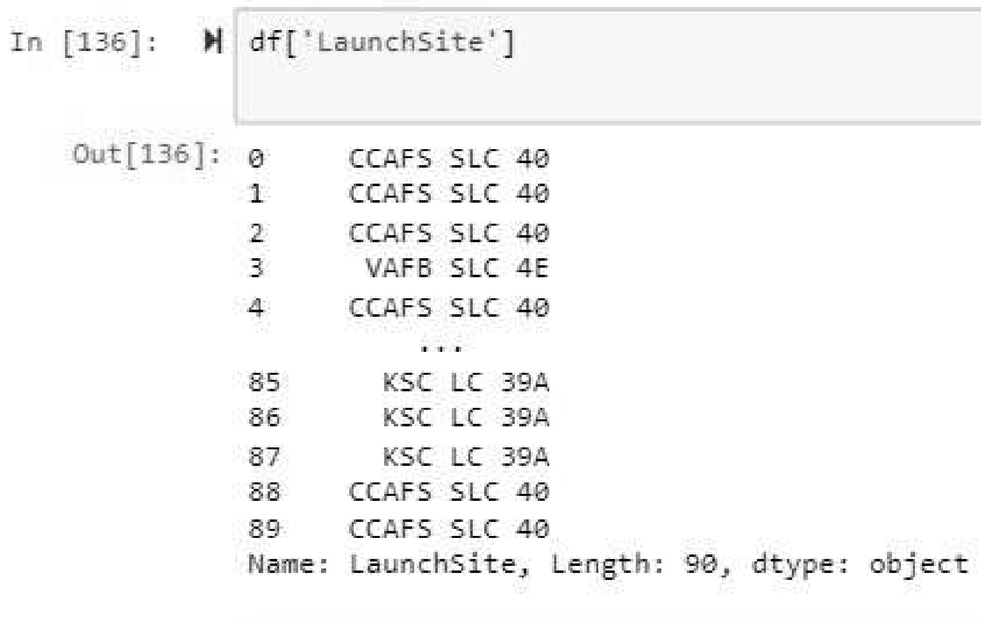
```
df=df.drop(['FlightNumber','Date','BoosterVersion','Longitude','Latitude'],axis=1)
```

Dummy Variable

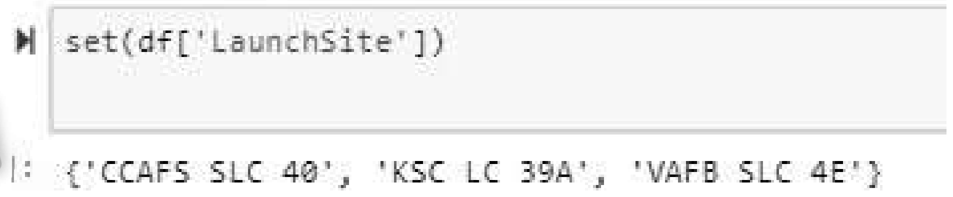
An indicator variable (or dummy variable) is a numerical variable used to label categories. They are called 'dummies' because the numbers themselves don't have inherent meaning.

متغییر دامی(ساختگی) یک متغییر عددی ساختگی است که به یک مقدار دسته بندی شده نسبت داده میشود.

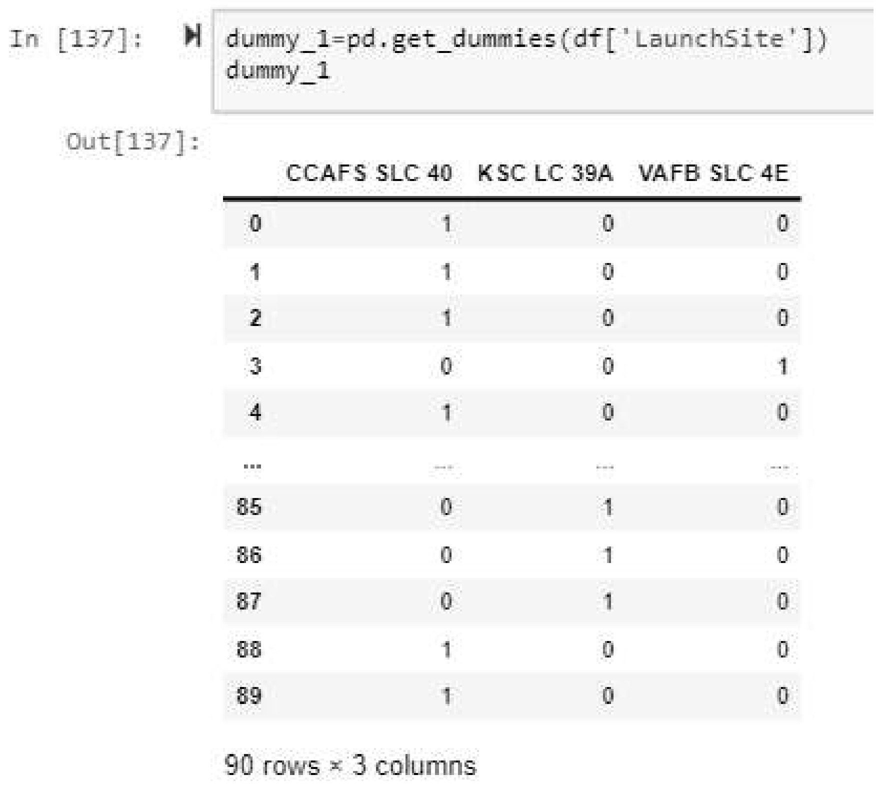
Dummy Variable



```
In [136]:  df['LaunchSite']

Out[136]: 0      CCAFS SLC 40
          1      CCAFS SLC 40
          2      CCAFS SLC 40
          3       VAFB SLC 4E
          4      CCAFS SLC 40
                     ...
          85       KSC LC 39A
          86       KSC LC 39A
          87       KSC LC 39A
          88     CCAFS SLC 40
          89     CCAFS SLC 40
          Name: LaunchSite, Length: 90, dtype: object
```



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

]: {'CCAFS SLC 40', 'KSC LC 39A', 'VAFB SLC 4E'}
```

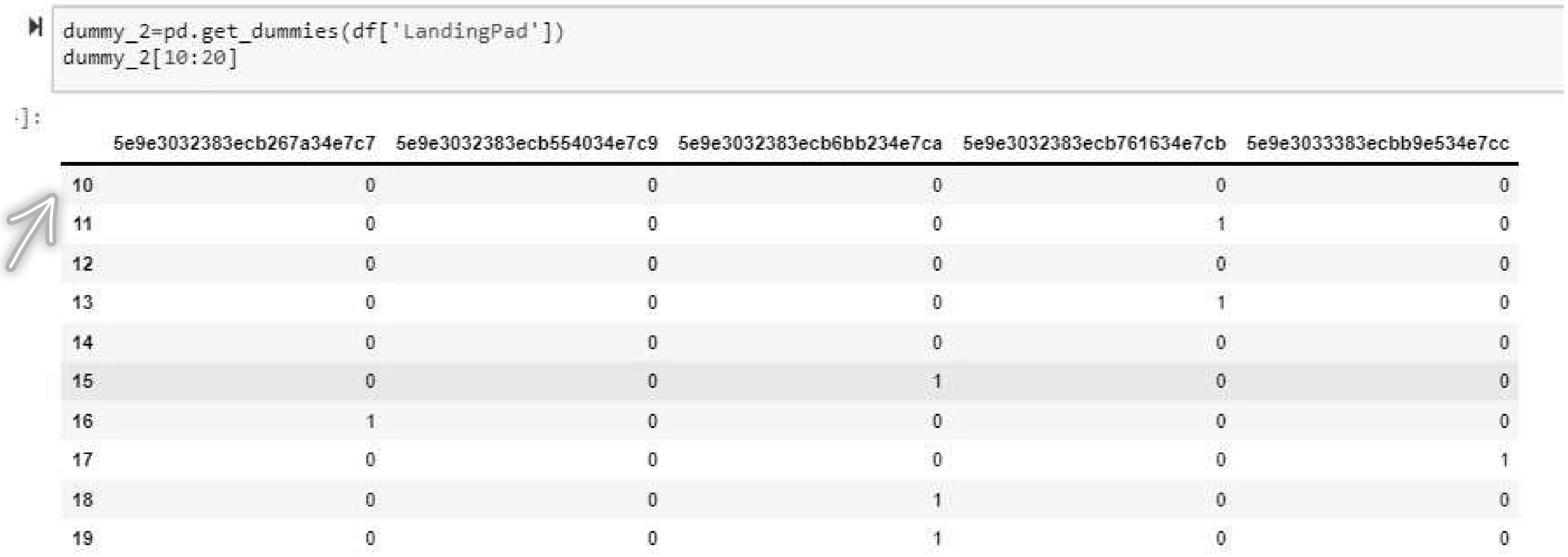


```
In [137]:  dummy_1=pd.get_dummies(df['LaunchSite'])
           dummy_1

Out[137]:
```

	CCAFS SLC 40	KSC LC 39A	VAFB SLC 4E
0	1	0	0
1	1	0	0
2	1	0	0
3	0	0	1
4	1	0	0
...	...	...	...
85	0	1	0
86	0	1	0
87	0	1	0
88	1	0	0
89	1	0	0

90 rows × 3 columns

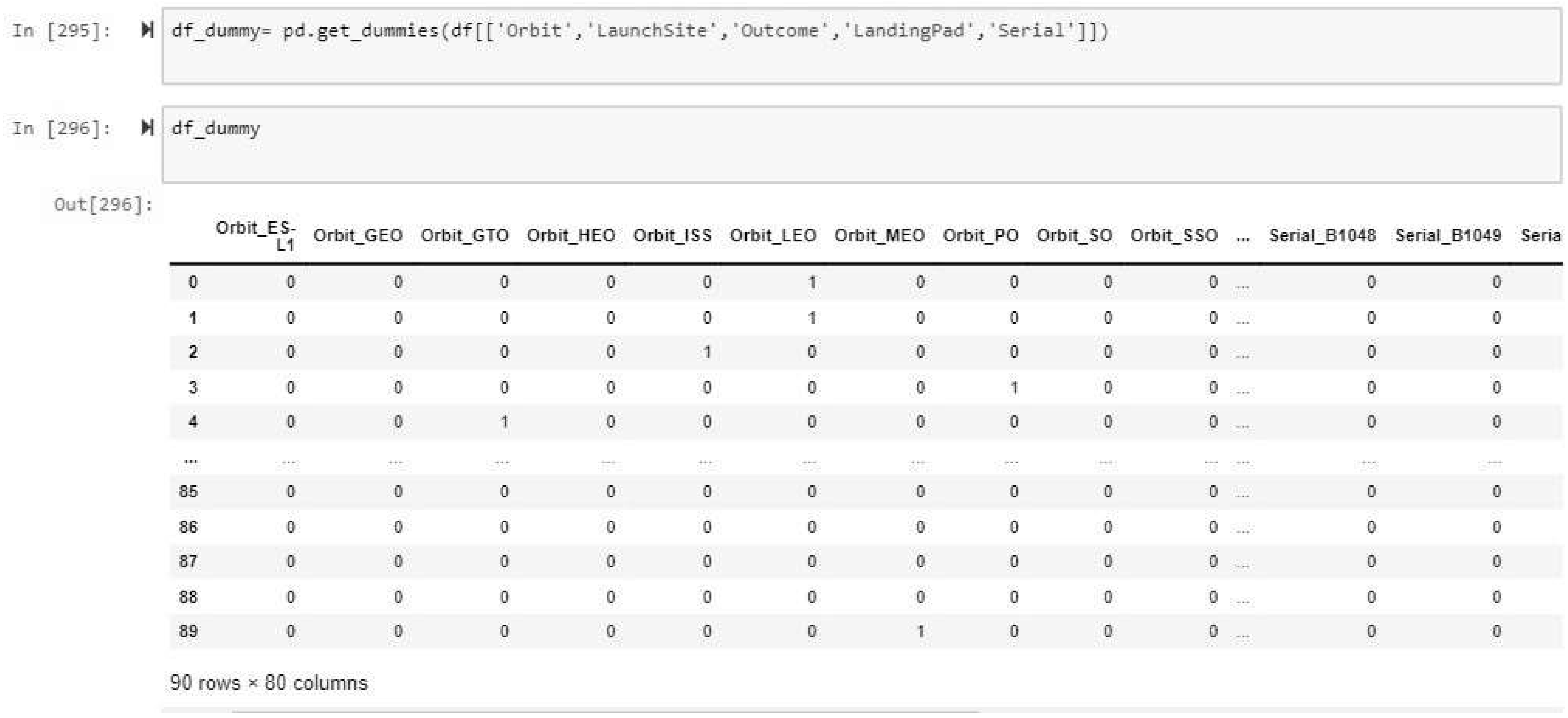
Dummy Variable



```
dummy_2=pd.get_dummies(df['LandingPad'])
dummy_2[10:20]
```

	5e9e3032383ecb267a34e7c7	5e9e3032383ecb554034e7c9	5e9e3032383ecb6bb234e7ca	5e9e3032383ecb761634e7cb	5e9e3033383ecbb9e534e7cc
10	0	0	0	0	0
11	0	0	0	1	0
12	0	0	0	0	0
13	0	0	0	1	0
14	0	0	0	0	0
15	0	0	1	0	0
16	1	0	0	0	0
17	0	0	0	0	1
18	0	0	1	0	0
19	0	0	1	0	0

Dummy Variable



```
In [295]:  df_dummy= pd.get_dummies(df[['Orbit','LaunchSite','Outcome','LandingPad','Serial']])
```

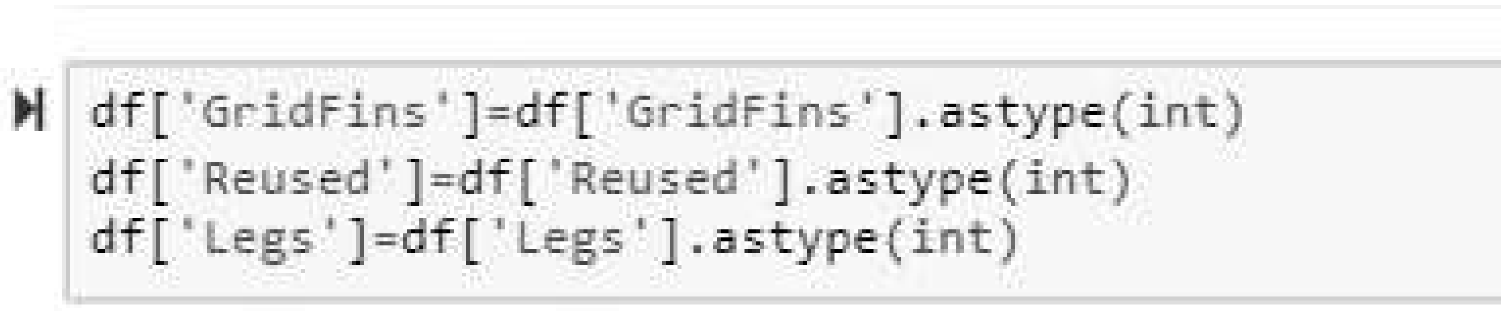
```
In [296]:  df_dummy
```

Out[296]:

	Orbit_ES-L1	Orbit_GEO	Orbit_GTO	Orbit_HEO	Orbit_ISS	Orbit_LEO	Orbit_MEO	Orbit_PO	Orbit_SO	Orbit_SSO	...	Serial_B1048	Serial_B1049	Seria
0	0	0	0	0	0	1	0	0	0	0	...	0	0	
1	0	0	0	0	0	1	0	0	0	0	...	0	0	
2	0	0	0	0	1	0	0	0	0	0	...	0	0	
3	0	0	0	0	0	0	0	1	0	0	...	0	0	
4	0	0	1	0	0	0	0	0	0	0	...	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
85	0	0	0	0	0	0	0	0	0	0	...	0	0	
86	0	0	0	0	0	0	0	0	0	0	...	0	0	
87	0	0	0	0	0	0	0	0	0	0	...	0	0	
88	0	0	0	0	0	0	0	0	0	0	...	0	0	
89	0	0	0	0	0	0	1	0	0	0	...	0	0	

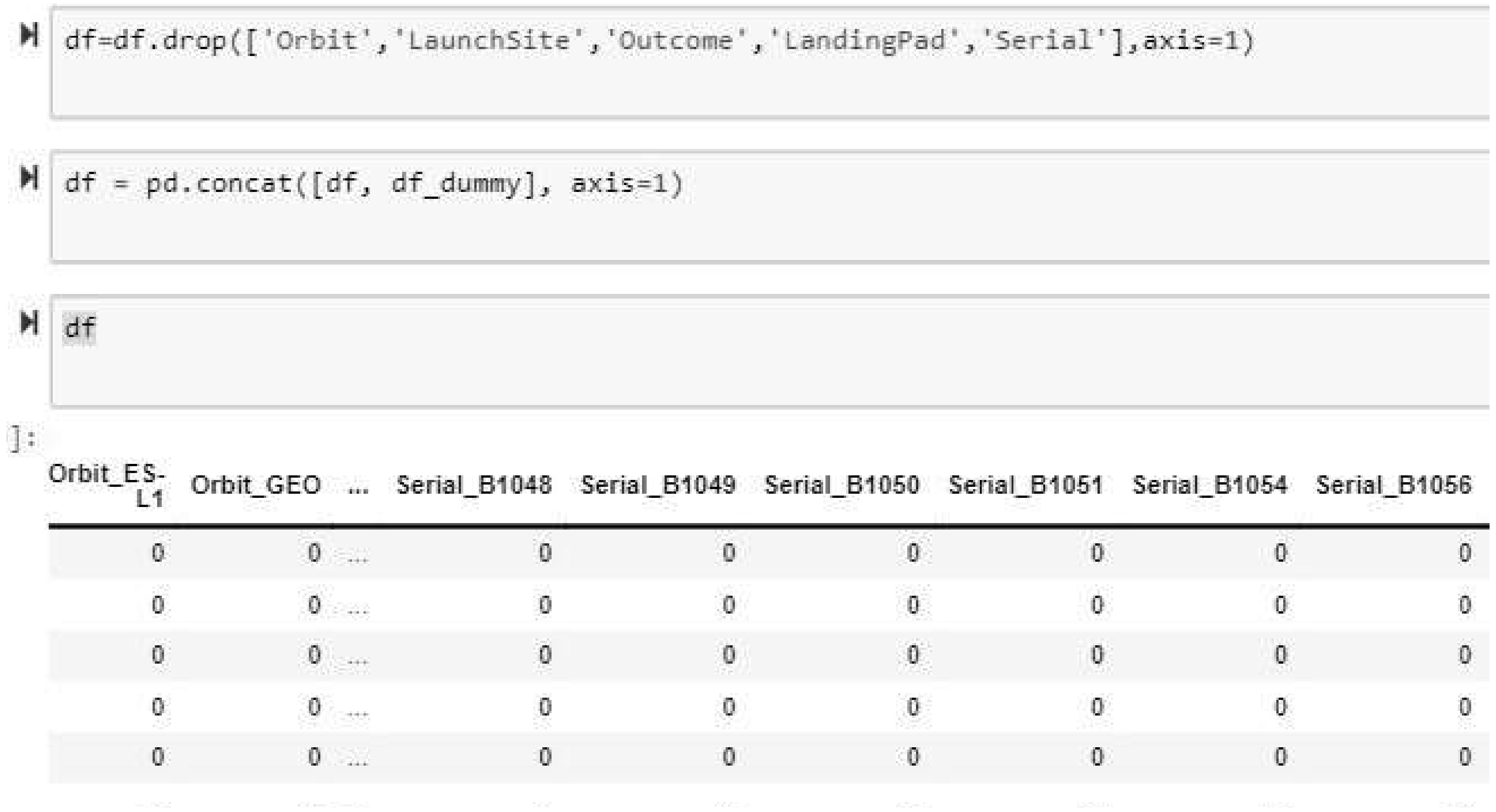
90 rows × 80 columns

Boolean Variable



```
df['GridFins']=df['GridFins'].astype(int)
df['Reused']=df['Reused'].astype(int)
df['Legs']=df['Legs'].astype(int)
```

.concat()



```
df=df.drop(['Orbit','LaunchSite','Outcome','LandingPad','Serial'],axis=1)
```

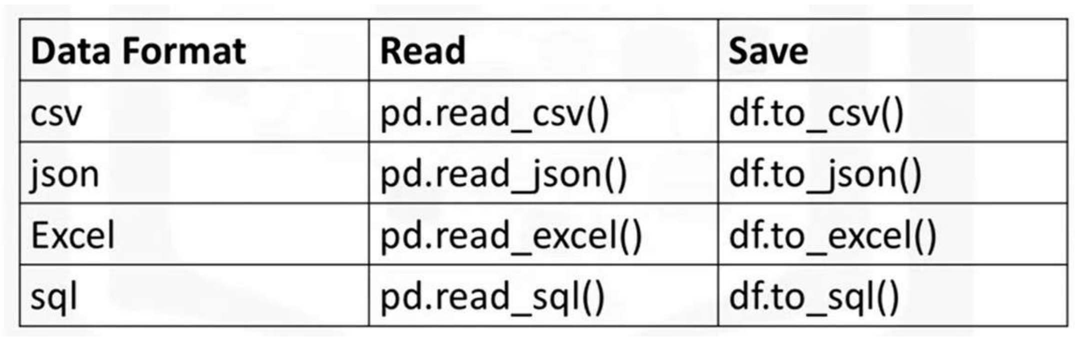
```
df = pd.concat([df, df_dummy], axis=1)
```

```
df
```

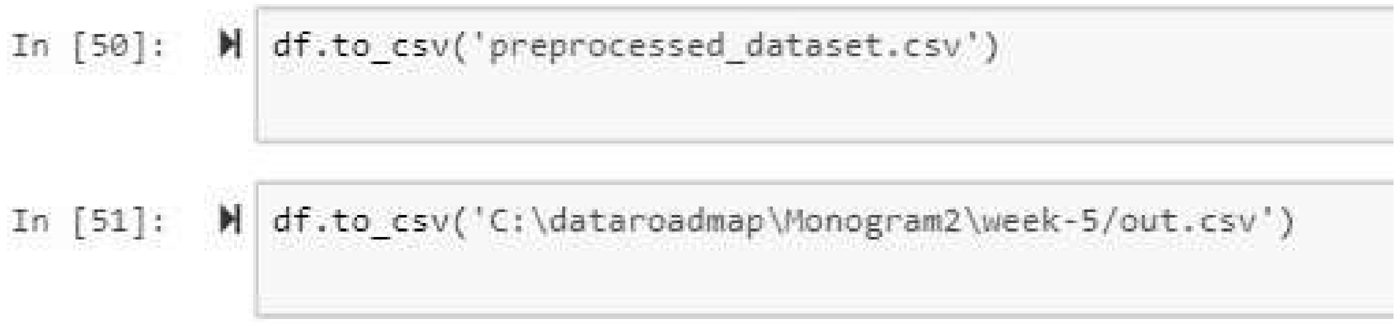
]:

Orbit_ES-L1	Orbit_GEO	...	Serial_B1048	Serial_B1049	Serial_B1050	Serial_B1051	Serial_B1054	Serial_B1056
0	0	...	0	0	0	0	0	0
0	0	...	0	0	0	0	0	0
0	0	...	0	0	0	0	0	0
0	0	...	0	0	0	0	0	0
0	0	...	0	0	0	0	0	0

Save Dataframe



Data Format	Read	Save
csv	pd.read_csv()	df.to_csv()
json	pd.read_json()	df.to_json()
Excel	pd.read_excel()	df.to_excel()
sql	pd.read_sql()	df.to_sql()



```
In [50]:  df.to_csv('preprocessed_dataset.csv')
```

```
In [51]:  df.to_csv('C:\dataroadmap\Monogram2\week-5/out.csv')
```

Assignment:

تمرین:

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

در صورتیکه علاقمند به تمرین بیشتر هستید جلسه هفتم دوره منتورینگ دیتاساینس را در کانال یوتیوب ملاحظه کنید.

 آپشنهای منو Kernel در ژوپیتر یا Runtime در کولب را بررسی کرده و بر روی نوتبوک خود اعمال کنید.