Pandas Dataframes
Part III

GITAA
Transforming careers

Pandas Dataframes - Recap

In the previous lecture, we have seen about

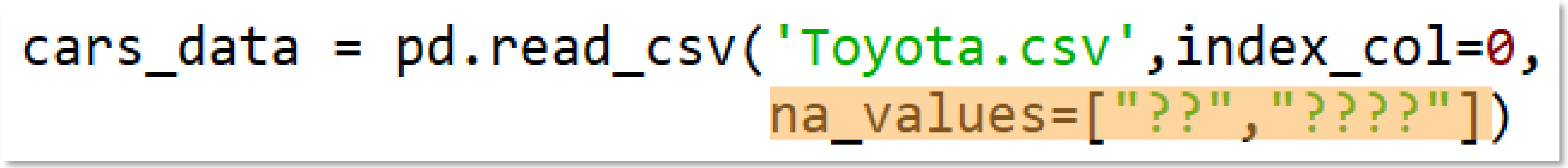
- Data types
  - Numeric
  - Character
- Checking data types of each column
- Count of unique data types
- Selecting data based on data types
- Concise summary of dataframe
- Checking format of each column
- Getting unique elements of each column

In this lecture

- Importing data
- Concise summary of dataframe
- Converting variable’s data types
- Category vs Object data type
- Cleaning column ‘Doors
- Getting count of missing values

Importing data

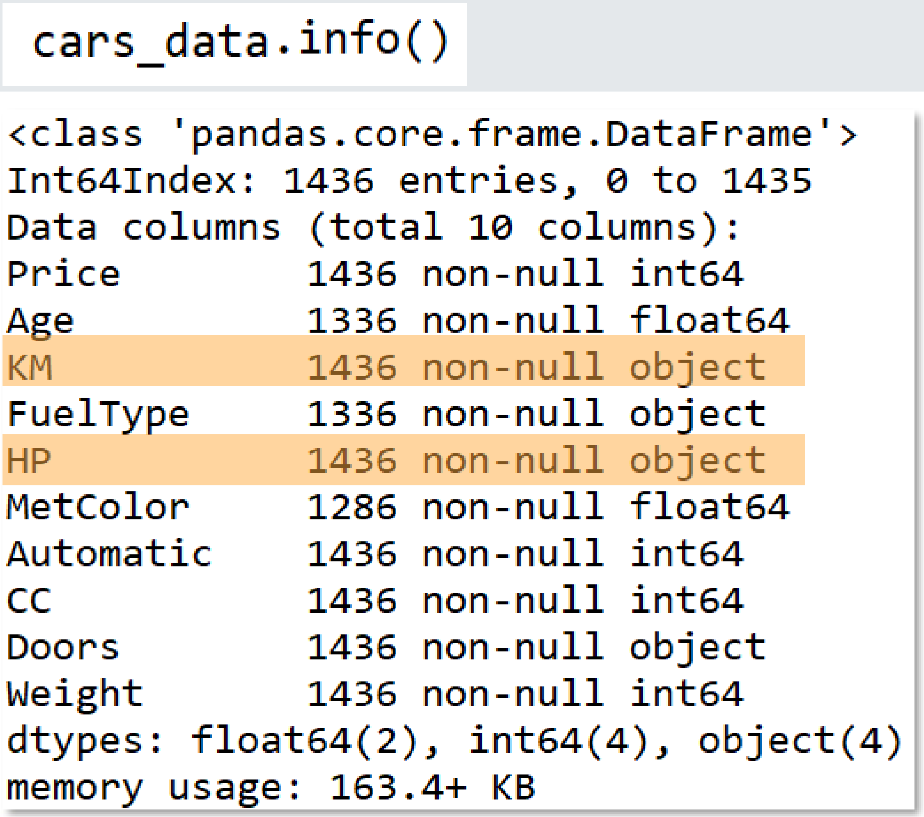
- We need to know how missing values are represented in the dataset in order to make reasonable decisions
- The missing values exist in the form of ‘nan’ '??' '????'
  - Python, by default replace blank values with ‘nan’
- Now, importing the data considering other forms of missing values in a dataframe



```
cars_data = pd.read_csv('Toyota.csv',index_col=0,
                         na_values=["??","????"])
```

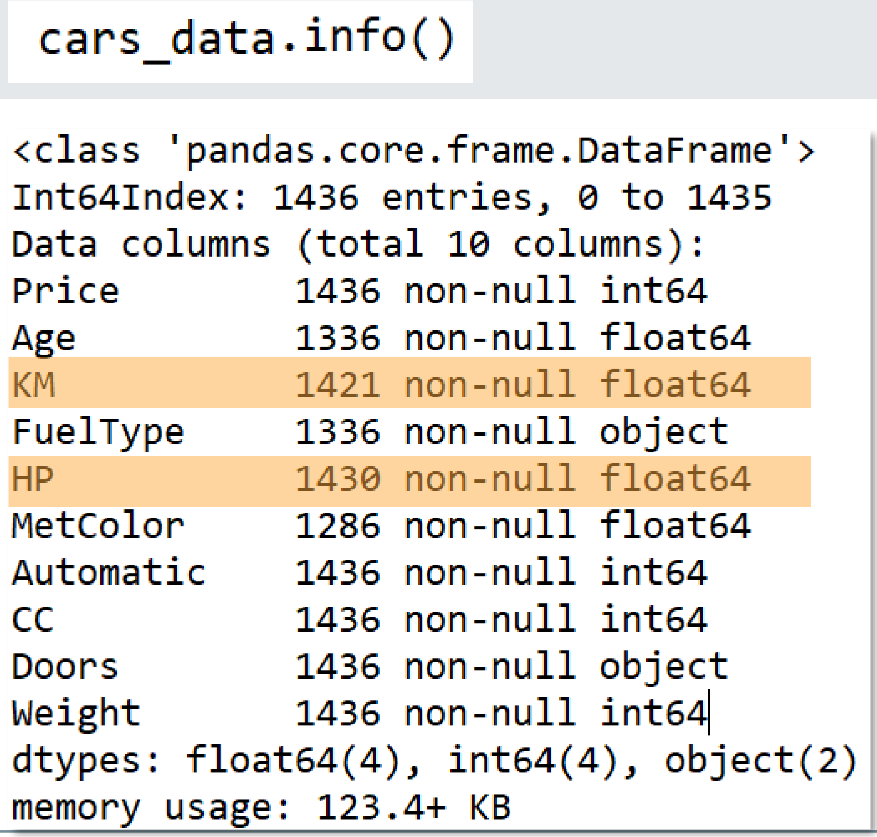
Concise summary of dataframe

Summary - before replacing special characters with nan	Summary - after replacing special characters with nan



```
cars_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
Price        1436 non-null int64
Age          1336 non-null float64
KM           1436 non-null object
FuelType     1336 non-null object
HP           1436 non-null object
MetColor     1286 non-null float64
Automatic    1436 non-null int64
CC           1436 non-null int64
Doors        1436 non-null object
Weight       1436 non-null int64
dtypes: float64(2), int64(4), object(4)
memory usage: 163.4+ KB
```



```
cars_data.info()
```

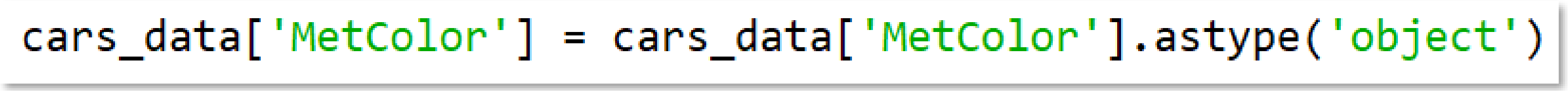
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
Price        1436 non-null int64
Age          1336 non-null float64
KM           1421 non-null float64
FuelType     1336 non-null object
HP           1430 non-null float64
MetColor     1286 non-null float64
Automatic    1436 non-null int64
CC           1436 non-null int64
Doors        1436 non-null object
Weight       1436 non-null int64
dtypes: float64(4), int64(4), object(2)
memory usage: 123.4+ KB
```

Converting variable’s data types

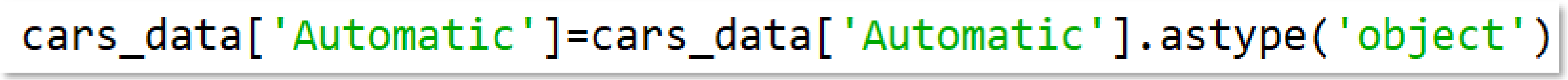
`astype()` method is used to explicitly convert data types from one to another

Syntax: `DataFrame.astype(dtype)`

Converting ‘MetColor’ , ‘Automatic’ to object data type:



```
cars_data['MetColor'] = cars_data['MetColor'].astype('object')
```



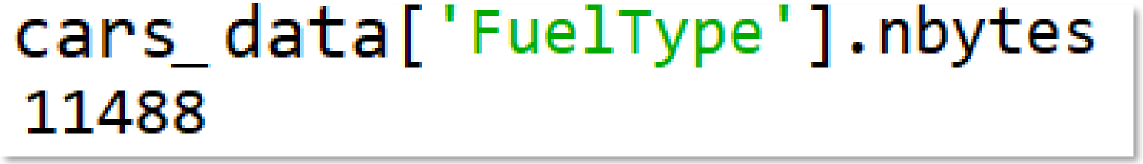
```
cars_data['Automatic']=cars_data['Automatic'].astype('object')
```

category vs object data type

nbytes() is used to get the total bytes consumed by the elements of the columns

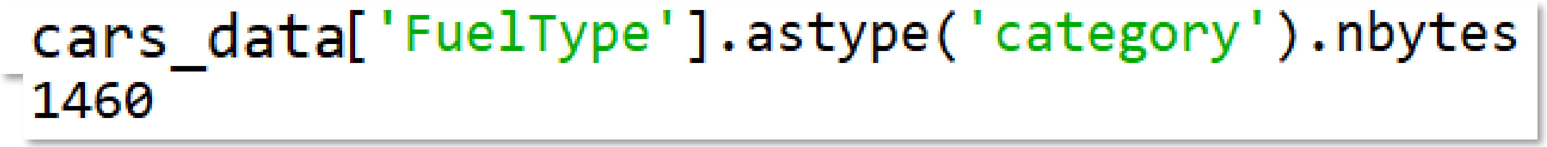
Syntax: ndarray.nbytes

If 'FuelType' is of object data type,



```
cars_data['FuelType'].nbytes
11488
```

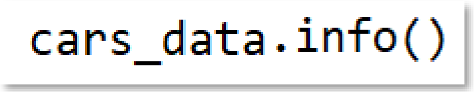
If 'FuelType' is of category data type,



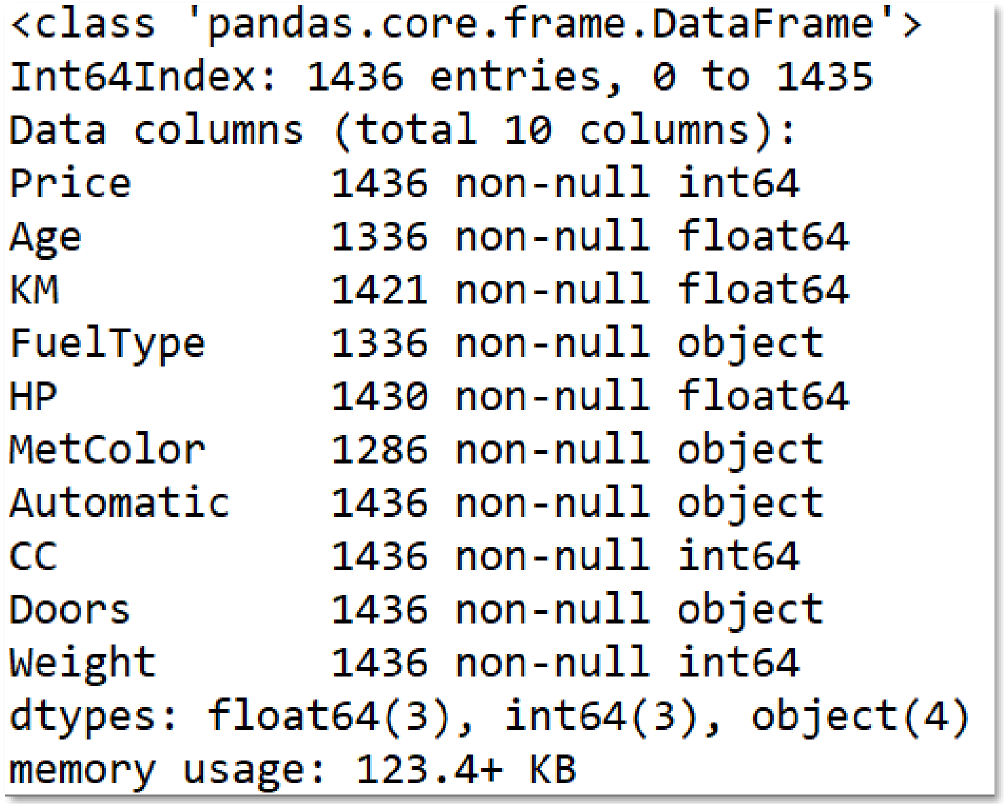
```
cars_data['FuelType'].astype('category').nbytes
1460
```

Re-checking the data type of variables

Re-checking the data type of variables after all the conversions



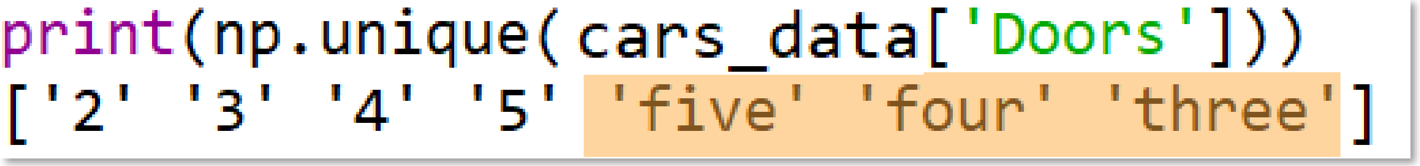
```
cars_data.info()
```



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1436 entries, 0 to 1435
Data columns (total 10 columns):
Price         1436 non-null int64
Age           1336 non-null float64
KM            1421 non-null float64
FuelType      1336 non-null object
HP            1430 non-null float64
MetColor      1286 non-null object
Automatic     1436 non-null object
CC            1436 non-null int64
Doors         1436 non-null object
Weight        1436 non-null int64
dtypes: float64(3), int64(3), object(4)
memory usage: 123.4+ KB
```

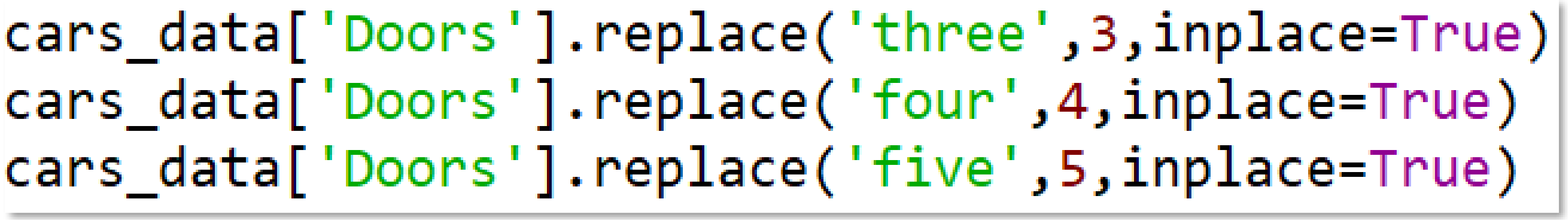
Cleaning column ‘Doors’

Checking unique values of variable ‘Doors’ :



```
print(np.unique(cars_data['Doors']))
['2' '3' '4' '5' 'five' 'four' 'three']
```

- replace() is used to replace a value with the desired value
- Syntax: DataFrame.replace([to_replace, value, …])



```
cars_data['Doors'].replace('three',3,inplace=True)
cars_data['Doors'].replace('four',4,inplace=True)
cars_data['Doors'].replace('five',5,inplace=True)
```

Try out !

numpy.where()

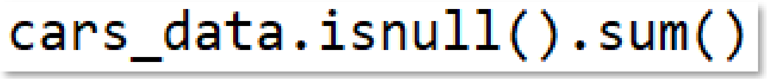
Converting ‘Doors’ data type

Converting ‘Doors’ to int64:

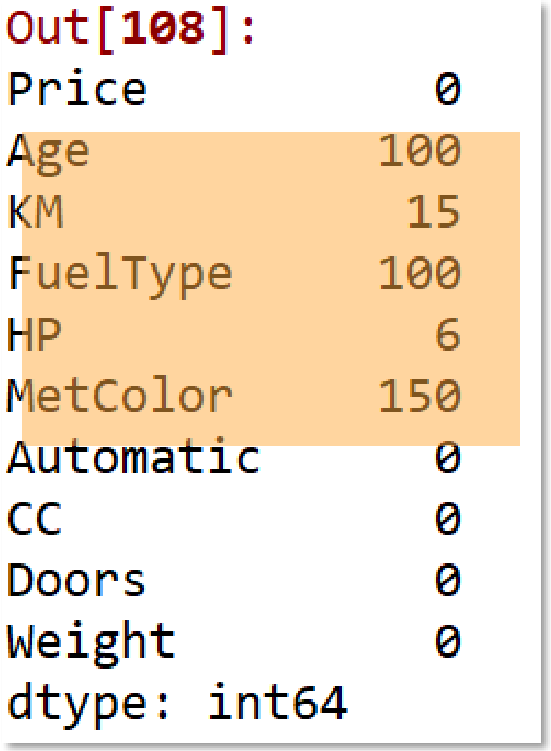
```
cars_data['Doors']=cars_data['Doors'].astype('int64')
```

To detect missing values

To check the count of missing values present in each column `Dataframe.isnull.sum()` is used



```
cars_data.isnull().sum()
```



```
Out[108]:
Price          0
Age          100
KM            15
FuelType     100
HP             6
MetColor     150
Automatic      0
CC             0
Doors          0
Weight         0
dtype: int64
```

Summary

- Imported data
- Concise summary of dataframe
- Converted variable’s data types
- Category vs Object data type
- Cleaned column ‘Doors
- Got count of missing values

```
rror_object
operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
 _operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
 _operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True

selection at the end -add
r_ob.select= 1
ier_ob.select=1
ntext.scene.objects.active
("Selected" + str(modifier
mirror_ob.select = 0
 = bpy.context.selected_ob
data.objects[one.name].se

print("please select exactly
```

THANK YOU