In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
df= pd.read_csv('iris.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [5]:
```
df.Species.value_counts()
```

Out[5]:
```
Iris-versicolor    50
Iris-setosa        50
Iris-virginica     50
Name: Species, dtype: int64
```

In [6]:
```
df.head()
```

Out[6]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
Id               150 non-null int64
SepalLengthCm    150 non-null float64
SepalWidthCm     150 non-null float64
PetalLengthCm    150 non-null float64
PetalWidthCm     150 non-null float64
Species          150 non-null object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

In [9]:
```
df.describe() #skips the categorical features as calculations cant be done on categorical variables
```

Out[9]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

In [11]:
```
df1= pd.read_csv('iris.csv', usecols=['SepalLengthCm','PetalLengthCm'])
```

In [12]:
```
df1
```

Out[12]:

	SepalLengthCm	PetalLengthCm
0	5.1	1.4
1	4.9	1.4
2	4.7	1.3
3	4.6	1.5
4	5.0	1.4
...	...	...
145	6.7	5.2
146	6.3	5.0
147	6.5	5.2
148	6.2	5.4
149	5.9	5.1

150 rows × 2 columns

to save our df1 dataframe data to a new file:

df1.to_csv('newfilename.csv')

In [16]:
```
df1.dtypes
```

Out[16]:
```
SepalLengthCm    float64
PetalLengthCm    float64
dtype: object
```

In [17]:
```
df.dtypes
```

Out[17]:
```
Id                int64
SepalLengthCm     float64
SepalWidthCm      float64
PetalLengthCm     float64
PetalWidthCm      float64
Species            object
dtype: object
```

In [18]:
```
df.head()
```

Out[18]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [19]:
```
# we can specify any of the column as Indexing column, here we are using the 'Id' colum
n.
df4=pd.read_csv('iris.csv', index_col=0)
df4
```

Out[19]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
Id					
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
146	6.7	3.0	5.2	2.3	Iris-virginica
147	6.3	2.5	5.0	1.9	Iris-virginica
148	6.5	3.0	5.2	2.0	Iris-virginica
149	6.2	3.4	5.4	2.3	Iris-virginica
150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [21]:
```
# we use index_col= False to specify that none of our data columns need to be assumed as Indexes.
df5=pd.read_csv('iris.csv', index_col=False)
df5
```

Out[21]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

In [23]:
```
# While importing our data set we can also escape some unwanted characters in our datase t
#compare the output below with the normal df output in the next line to understand the d ifference.

df6= pd.read_csv('iris.csv', escapechar=',')
```

In [24]:
```
df6
```

Out[24]:

	IdSepalLengthCmSepalWidthCmPetalLengthCmPetalWidthCmSpecies
0	15.13.51.40.2Iris-setosa
1	24.93.01.40.2Iris-setosa
2	34.73.21.30.2Iris-setosa
3	44.63.11.50.2Iris-setosa
4	55.03.61.40.2Iris-setosa
...	...
145	1466.73.05.22.3Iris-virginica
146	1476.32.55.01.9Iris-virginica
147	1486.53.05.22.0Iris-virginica
148	1496.23.45.42.3Iris-virginica
149	1505.93.05.11.8Iris-virginica

150 rows × 1 columns

In [25]:
```
df.head()
```

Out[25]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Some useful Panda File-handling methods:

In [ ]:
```
#Reading Excel files:
df4= pd.read_excel('filename.xlsx', sheetname='sheet 3')
```

In [ ]:
```
#saving json data without confusing the default indexing with our data

df.to_json(orient="records")
```

PICKLING

to_pickle method in pandas uses Python's cPickle module to save datastructures to the disk directly.

For example a dataframe can be stored on the harddrive using the to_pickle method and if needed can be read back.

In [28]:
```
df2= pd.read_csv('iris.csv')
df2.to_pickle('df10')
```

In [29]:
```
df11=pd.read_pickle('df10')
df11.head()
```

Out[29]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

HTML file handling

In [ ]:
```
#importing html tables with table_names/table keywords
url='https://sampleurl'
df1= pd.read_html(url, match='zipcode', header= 0)
```

In [ ]:
```

```

/