In [1]:
```
#This is a supplementary material to the lecture "Pandas" to quickly revise, whenever needed
```

In [2]:
```
#Pandas
#pandas is a package in python to manipulate the data
```

In [3]:
```
#to use it, let's first import it
import pandas as pd
```

In [4]:
```
#to read the data (csv file into the dataframe)
iris = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data')
#inside parentheses, filepath containing the data to be specified, it can be a url or local path on yo
ur system
#there are other functions too for various formats like read_excel(), read_json and read_html()
```

In [5]:
```
type(iris)
```

Out[5]:
```
pandas.core.frame.DataFrame
```

In [6]:
```
#to look at the first few rows of the data
iris.head(10)     #returns first 10 rows of the dataframe-----------------by default it treats the fi
rst row as column header, we can change that
```

Out[6]:

	5.1	3.5	1.4	0.2	Iris-setosa
0	4.9	3.0	1.4	0.2	Iris-setosa
1	4.7	3.2	1.3	0.2	Iris-setosa
2	4.6	3.1	1.5	0.2	Iris-setosa
3	5.0	3.6	1.4	0.2	Iris-setosa
4	5.4	3.9	1.7	0.4	Iris-setosa
5	4.6	3.4	1.4	0.3	Iris-setosa
6	5.0	3.4	1.5	0.2	Iris-setosa
7	4.4	2.9	1.4	0.2	Iris-setosa
8	4.9	3.1	1.5	0.1	Iris-setosa
9	5.4	3.7	1.5	0.2	Iris-setosa

In [7]:
```
#loading the data with column names pre decided
iris = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data',
                   header = None, names=['sl', 'sw', 'pl', 'pw', 'species'])
```

In [8]:
```
iris.head(10)
```

Out[8]:

	sl	sw	pl	pw	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
5	5.4	3.9	1.7	0.4	Iris-setosa
6	4.6	3.4	1.4	0.3	Iris-setosa
7	5.0	3.4	1.5	0.2	Iris-setosa
8	4.4	2.9	1.4	0.2	Iris-setosa
9	4.9	3.1	1.5	0.1	Iris-setosa

In [9]:
```
#to get the basic details about the data (especially numeric columns)
iris.describe()
```

Out[9]:

	sl	sw	pl	pw
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [10]:
```
#to aceess a particular column
iris.sl
```

Out[10]:
```
0      5.1
1      4.9
2      4.7
3      4.6
4      5.0
      ...
145    6.7
146    6.3
147    6.5
148    6.2
149    5.9
Name: sl, Length: 150, dtype: float64
```

In [11]:
```
#alternatively
iris['sl']
```

Out[11]:
```
0      5.1
1      4.9
2      4.7
3      4.6
4      5.0
      ...
145    6.7
146    6.3
147    6.5
148    6.2
149    5.9
Name: sl, Length: 150, dtype: float64
```

In [12]:
```
#to see the total number of null entries in each column
iris.isnull().sum()
```

Out[12]:
```
sl         0
sw         0
pl         0
pw         0
species    0
dtype: int64
```

In [13]:
```
#to access some part of the dataframe
iris.iloc[1:5, 1:3]
```

Out[13]:

	sw	pl
1	3.0	1.4
2	3.2	1.3
3	3.1	1.5
4	3.6	1.4

In [14]:
```
#selecting data based on some condition applied on feature values in columns
#say, we want to select only those rows, where sl > 6 and pl > 5
iris[(iris.sl > 6) & (iris.pl > 5)]
```

Out[14]:

	sl	sw	pl	pw	species
100	6.3	3.3	6.0	2.5	Iris-virginica
102	7.1	3.0	5.9	2.1	Iris-virginica
103	6.3	2.9	5.6	1.8	Iris-virginica
104	6.5	3.0	5.8	2.2	Iris-virginica
105	7.6	3.0	6.6	2.1	Iris-virginica
107	7.3	2.9	6.3	1.8	Iris-virginica
108	6.7	2.5	5.8	1.8	Iris-virginica
109	7.2	3.6	6.1	2.5	Iris-virginica
110	6.5	3.2	5.1	2.0	Iris-virginica
111	6.4	2.7	5.3	1.9	Iris-virginica
112	6.8	3.0	5.5	2.1	Iris-virginica
115	6.4	3.2	5.3	2.3	Iris-virginica
116	6.5	3.0	5.5	1.8	Iris-virginica
117	7.7	3.8	6.7	2.2	Iris-virginica
118	7.7	2.6	6.9	2.3	Iris-virginica
120	6.9	3.2	5.7	2.3	Iris-virginica
122	7.7	2.8	6.7	2.0	Iris-virginica
124	6.7	3.3	5.7	2.1	Iris-virginica
125	7.2	3.2	6.0	1.8	Iris-virginica
128	6.4	2.8	5.6	2.1	Iris-virginica
129	7.2	3.0	5.8	1.6	Iris-virginica
130	7.4	2.8	6.1	1.9	Iris-virginica
131	7.9	3.8	6.4	2.0	Iris-virginica
132	6.4	2.8	5.6	2.2	Iris-virginica
133	6.3	2.8	5.1	1.5	Iris-virginica
134	6.1	2.6	5.6	1.4	Iris-virginica
135	7.7	3.0	6.1	2.3	Iris-virginica
136	6.3	3.4	5.6	2.4	Iris-virginica
137	6.4	3.1	5.5	1.8	Iris-virginica
139	6.9	3.1	5.4	2.1	Iris-virginica
140	6.7	3.1	5.6	2.4	Iris-virginica
141	6.9	3.1	5.1	2.3	Iris-virginica
143	6.8	3.2	5.9	2.3	Iris-virginica
144	6.7	3.3	5.7	2.5	Iris-virginica
145	6.7	3.0	5.2	2.3	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica

In [15]:
```
#grouby function is pandas library to group values based on categorical variables
iris.groupby('species').mean()['pl']
```

Out[15]:
```
species
Iris-setosa        1.464
Iris-versicolor    4.260
Iris-virginica     5.552
Name: pl, dtype: float64
```

In [16]:
```
iris.groupby('species').mean()['pw']
```

Out[16]:
```
species
Iris-setosa        0.244
Iris-versicolor    1.326
Iris-virginica     2.026
Name: pw, dtype: float64
```

In [17]:
```
iris.groupby('species').mean()['sw']
```

Out[17]:
```
species
Iris-setosa        3.418
Iris-versicolor    2.770
Iris-virginica     2.974
Name: sw, dtype: float64
```

In [18]:
```
#deleting rows
df_new = iris.drop(0)          #it will return new dataframe, without row labelled 0 in original dataf
rame
df_new.head()
```

Out[18]:

	sl	sw	pl	pw	species
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
5	5.4	3.9	1.7	0.4	Iris-setosa

In [19]:
```
#if we want to change in the original dataframe itself
iris.drop(0, inplace = True)
iris.head()
```

Out[19]:

	sl	sw	pl	pw	species
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
5	5.4	3.9	1.7	0.4	Iris-setosa

In [20]:
```
#to drop column
iris.drop('sl', axis = 1, inplace=True)
iris.head()
```

Out[20]:

	sw	pl	pw	species
1	3.0	1.4	0.2	Iris-setosa
2	3.2	1.3	0.2	Iris-setosa
3	3.1	1.5	0.2	Iris-setosa
4	3.6	1.4	0.2	Iris-setosa
5	3.9	1.7	0.4	Iris-setosa

In [21]:
```
#alternatively to delete column inplace
del iris['sw']
iris.head()
```

Out[21]:

	pl	pw	species
1	1.4	0.2	Iris-setosa
2	1.3	0.2	Iris-setosa
3	1.5	0.2	Iris-setosa
4	1.4	0.2	Iris-setosa
5	1.7	0.4	Iris-setosa

In [22]:
```
#handling nan values in dataframe
#we can either drop the nan entries or we can fill some values in those places
#there are various approaches, we can think of, to fill some value in nan, like filling the average va
lue o the most occuring value
#to drop nan values
iris.dropna(inplace = True)
#to fill, let's say, we want to fill all the nan values in column 'pl' (if there are) with the mean of
the column
iris.pl.fillna(iris.pl.mean(), inplace = True)
#although, here in this dataset, we don't have any nan values
```

In [23]:
```
#handling string data
#most of the ML algorithms work very well with the numeric data
#so, if we have any string data in the dataframe, we can think of a some way to convert that to the nu
meric data
#for example, here, in the column 'species', we have 3 different types of string values, let's try to
 assign 0, 1  and 2 to thise categories
#let's first write a function, which will do this for us
def getNumber(s):
  if s == 'Iris-setosa':
    return 0
  elif s == 'Iris-versicolor':
    return 1
  else:
    return 2

iris['category'] = iris.species.apply(getNumber)
del iris['species']
iris.head()
```

Out[23]:

	pl	pw	category
1	1.4	0.2	0
2	1.3	0.2	0
3	1.5	0.2	0
4	1.4	0.2	0
5	1.7	0.4	0

In [24]:
```
iris.groupby('category').count()['pl']
```

Out[24]:
```
category
0    49
1    50
2    50
Name: pl, dtype: int64
```

In [25]:
```
#Thanks, Happy Coding!
```

In [ ]:
```
#To download .ipynb notebook, right click the following link and click save as
https://ninjasfiles.s3.amazonaws.com/0000000000003221.ipynb
```