k-Means and Python Programming

Pandas DataFrame and k-Means hands-on

ABDUL HAFEEZ
PHD, VIRGINIA TECH, VA USA
POSTDOC, GEORGIA TECH, ATLANTA GA, USA

Pandas DataFrame

Generally, it’s a two dimensional data structure

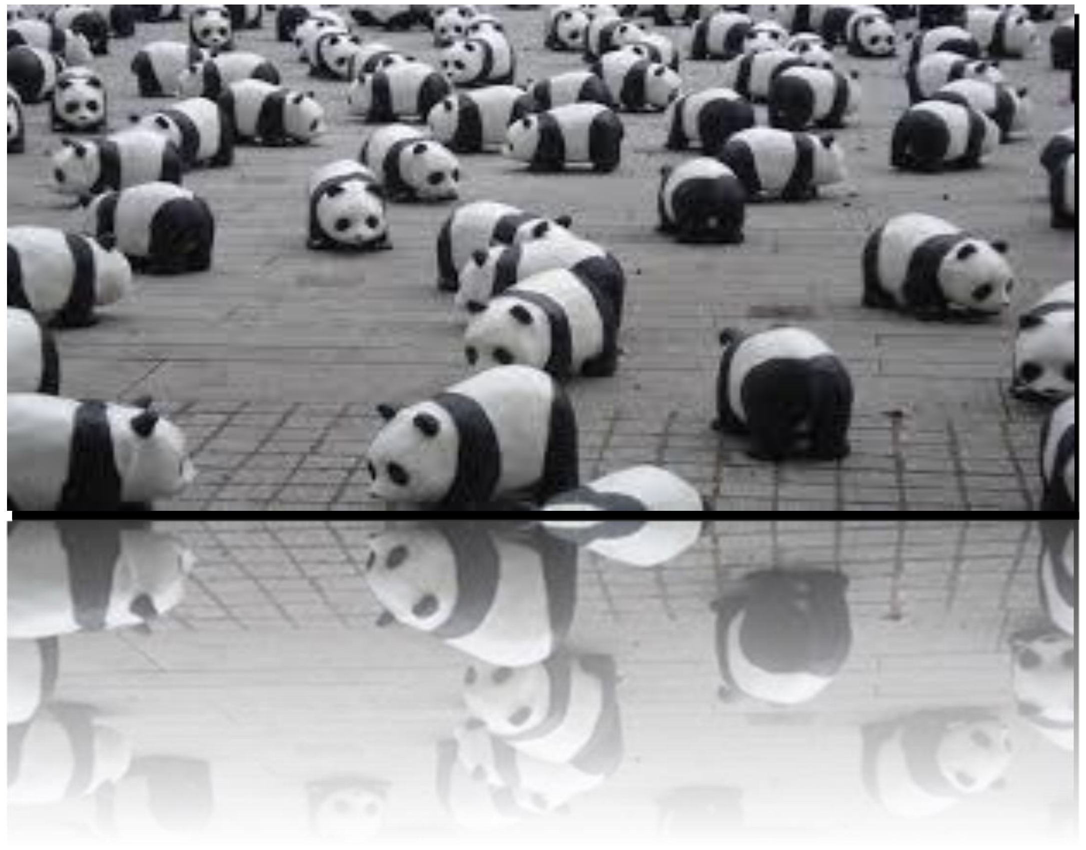
Contains labels for rows and columns

Can contain heterogeneous data

Basically, has three parts:

the data,
rows, and

columns



Pandas DataFrame

Columns

Rows

	Name	Company	Position	Age
0	Mati	Google	Analyst	30
1	Sal	Facebook	Developer	29
2	Aly	Microsoft	Team Lead	31
3	Aun	Amazon	Manager	33

What can we do to a Data Frame?

How to create a DataFrame?

How to manipulate Rows and Columns of a DataFrame?

How to index and select data?

How to address the issue of missing Data?

Can we iterate over individual rows and columns?

Create a DataFrame from a list

```
#import pandas as pd
Import pandas as pd

# list of strings
myList = [‘Truthfulness’, ‘Honesty’, ‘Sincerity’, ‘are’, ‘important’, ‘for’, ‘success’]

# Create a dataframe
df = pd.DataFrame(myList)

#display the dataframe
print(df)
```

Create a DataFrame from a list

Output

```
             0
0    Truthfulness
1    Honesty
2    Sincerity
3    are
4    paramounts
5    of
6    success
```

Create DataFrame from dict of lists

```
#Demonstrate how to create a dataframe
#from a dictionary of lists
import pandas as pd
#initialize data of lists
# initialize data for two lists
myList = { 'Name': ['Aly', 'Adi', 'Sal', 'Kashi', 'ryan'],  'Age': [19, 22, 17, 23, 25] }

# Create a dataframe
df = pd.DataFrame(myList)

print(df)
```

Create DataFrame from dict of lists

OUPUT:

```
  Name  Age
0   Aly   19
1   Adi   22
2   Sal   17
3 Kashi   23
4  ryan   25
```

Manipulating Rows and Columns

```
#Demonstrate how to create a dataframe
#from a dictionary of lists

#import pandas as pd
import pandas as pd

# Initialize a dictionary for employee data
myList = { 'Name': ['Aly', 'Adi', 'Sal', 'Kashi', 'Ryan'],
           'Age': [19, 22, 17, 23, 25],
           'Address': ['Atlanta', 'FairFax', 'SilverSprings', 'Youngstown', 'Chicago'],
           'Education': ['UG', 'PhD', 'HighSchool', 'MSc', 'UG']}
```

Manipulating Rows and Columns

```
# Convert the dictionary into a DataFrame
df = pd.DataFrame(myList)

# Select any two columns of your choice
print(df[['Address','Education']])
```

Manipulating Rows and Columns

OUTPUT:

```
      Address     Education
0     Atlanta       UG
1     FairFax       PhD
2   SilverSprings   HighSchool
3   Youngstown     MSc
4     Chicago      UG
```

Select a single column

```
# create a data frame from a csv file
myDF = pd.read_csv('nba.csv', index_col = "Name")

# retrieve columns through indexing
aCol = myDF["Age"]

print(aCol)
```

Select a single column

```
print("\n\n")


#retrieve another column thorugh indexing
anotherCol = myDF["College"]


print(anotherCol)
```

OUPUT:

```
Name
Avery Bradley    25
Jae Crowder      25
John Holland     27
....

Name
Avery Bradley              Texas
Jae Crowder             Marquette
John Holland     Boston University
....
```

Index a DataFrame using .loc[]

```
# retrieve row by loc method
first = myDF.loc["Avery Bradley"]
another = myDF.loc["R.J. Hunter"]

print(first, "\n\n\n", another)
```

Index a DataFrame using .loc[]

OUTPUT:

```
Team        Boston Celtics
Number                   0
Position                PG
Age                     25
…
Team        Boston Celtics
Number                  28
Position                SG
Age                     22
```

Index a DataFrame using .iloc[]

```
# retrieve rows by iloc method
sample_row = myDF.iloc[4]



print(sample_row)
```

Index a DataFrame using .iloc[]

OUTPUT:

```
Team        Boston Celtics
Number                   8
Position                PF
…
Name: Jonas Jerebko, dtype: object
```

Addressing Missing Data: isnull() notnull()

Missing data occurs when the information is missing for an item or so

It appears as NA values in pandas dataframes

isnull() and notnull() is used to check missing values

Addressing Missing Data : isnull() notnull()

```
# import pandas as pd
import pandas as pd

#import numpy as np
import numpy as np

#dictionary of lists
myDict = {'First Score': [100, 90, np.nan, 95],
        'Second Score': [30, 45, 56, np.nan],
        'Third Score': [np.nan, 40, 80, 98]}
```

Addressing Missing Data : isnull() notnull()

```
#convert the list into a dataframe
myDF = pd.DataFrame(myDict)

#check for null values
print(myDF.isnull())\

#alternatively, check for null values
print(myDF.notnull())
```

Addressing Missing Data : isnull() notnull()

```
     First Score  Second Score  Third Score
0       False        False            True
1       False        False             False
2        True        False            False
3        False         True           False

     First Score  Second Score  Third Score
0          True          True        False
1          True          True         True
2         False           True        True
3          True         False         True
```

Addressing Missing Data:
fillna(), replace() and interpolate()

In order to replace NaN with some reasonable value

```
#fill the missing value with fillna(0)
print("fillna()")
print(myDF.fillna(0), "\n")

#alternatively, fill missing values with replace()
print("replace()")
print(myDF.replace(), "\n")

#futhremore, fill missing values with interpolate()
print("interpolate()")
print(myDF.interpolate())
```

Addressing Missing Data:
fillna(), replace() and interpolate()

OUTPUT:

fillna()

```
   First Score  Second Score  Third Score
0       100.0          30.0          0.0
1        90.0          45.0         40.0
2         0.0          56.0         80.0
3        95.0           0.0         98.0
```

Addressing Missing Data:
fillna(), replace() and interpolate()

OUTPUT:

replace()

```
   First Score  Second Score  Third Score
0        100.0          30.0          NaN
1         90.0         45.0         40.0
2         90.0         56.0         80.0
3         95.0         56.0         98.0
   First Score  Second Score  Third Score
0        100.0          30.0          NaN
1         90.0         45.0         40.0
2         92.5         56.0         80.0
3         95.0         56.0         98.0
```

Drop Missing values: dropna()

```
#Drop rows with at least one NaN value
myDF.dropna()
```

Iterate over Rows and Columns

```
#dictionary of lists
myDict = { 'Name': ['Aly', 'Adi', 'Sal', 'Kashi', 'Ryan'],
           'Age': [19, 22, 17, 23, 25],
           'Address': ['Atlanta', 'FairFax', 'SilverSprings', 'Youngstown', 'Chicago'],
           'Education': ['UG', 'PhD', 'HighSchool', 'MSc', 'UG']}

#convert the list into a dataframe
myDF = pd.DataFrame(myDict)

print(myDF)
```

Iterate over Rows

```
#iterate over rows using iterrows() function

for i, j in myDF.iterrows():
    print(i, j)
    print()
```

Iterate over Rows

OUTPUT:

```
Name  Age      Address   Education
0    Aly   19       Atlanta         UG
1    Adi   22       FairFax        PhD
2    Sal   17  SilverSprings  HighSchool
3  Kashi   23    Youngstown        MSc
4   Ryan   25       Chicago        UG
```

Iterate over Rows

OUTPUT:

```
0 Name          Aly
Age             19
Address      Atlanta
Education       UG
Name: 0, dtype: object

1 Name          Adi
Age             22
Address      FairFax
Education      PhD
Name: 1, dtype: object
```

Iterate over Rows

OUTPUT:

```
2 Name               Sal
Age                 17
Address      SilverSprings
Education       HighSchool
Name: 2, dtype: object

3 Name             Kashi
Age               23
Address      Youngstown
Education           MSc
Name: 3, dtype: object
```

Iterate over a Column

```
#convert the list into a dataframe
myDF = pd.DataFrame(myDict)

#create a list of dataframe columns
cols = list(myDF)

#iterate over the column
for c in cols:
    print (myDF[c][2])
```

Iterate over a column

OUTPUT:

```
Sal
17
SilverSprings
HighSchool
```