Lecture 1 Basic Operations on Pandas DataFrame

@Yuanyuan Tang

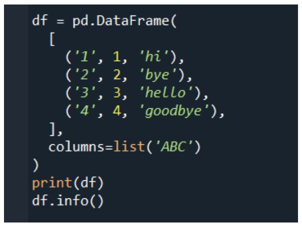
1 Definition:

- Two-dimensional, size-mutable, potentially heterogeneous tabular data.
- A dict-like container for Series objects, where each column is a series.

2 Create a DataFrame

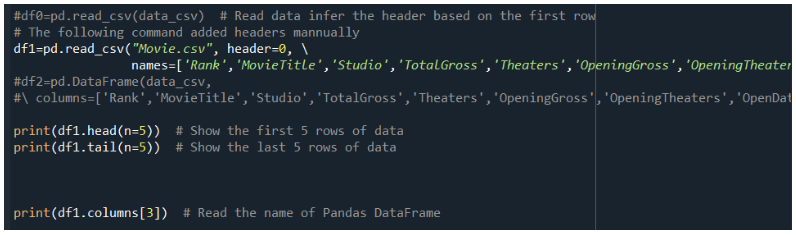
pandas.DataFrame — pandas 1.5.1 documentation (pydata.org)

- By the function pd.DataFrame(), import data, column names, and even data types



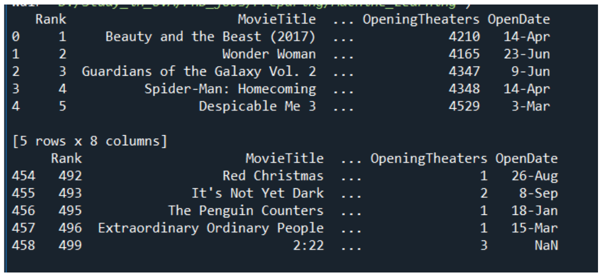
```
df = pd.DataFrame(
  [
    ('1', 1, 'hi'),
    ('2', 2, 'bye'),
    ('3', 3, 'hello'),
    ('4', 4, 'goodbye'),
  ],
  columns=list('ABC')
)
print(df)
df.info()
```

- Read from CSV by the function pd.read_csv. Note that if we want to add names for columns, we need to set *header=0*.



```
#df0=pd.read_csv(data_csv)  # Read data infer the header based on the first row
# The following command added headers mannually
df1=pd.read_csv("Movie.csv", header=0, \
                names=['Rank','MovieTitle','Studio','TotalGross','Theaters','OpeningGross','OpeningTheater
#df2=pd.DataFrame(data_csv,
#\ columns=['Rank','MovieTitle','Studio','TotalGross','Theaters','OpeningGross','OpeningTheaters','OpenDat

print(df1.head(n=5))  # Show the first 5 rows of data
print(df1.tail(n=5))  # Show the last 5 rows of data



print(df1.columns[3])  # Read the name of Pandas DataFrame
```

- After loading the DataFrame, we can display the first n rows by *df.head(n)*.

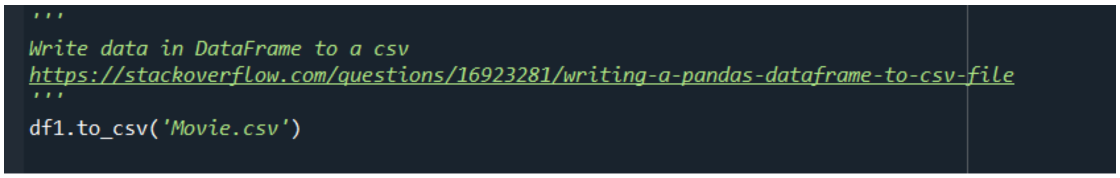


```
   Rank                   MovieTitle  ... OpeningTheaters OpenDate
0     1     Beauty and the Beast (2017)  ...            4210   14-Apr
1     2                  Wonder Woman  ...            4165   23-Jun
2     3  Guardians of the Galaxy Vol. 2  ...            4347    9-Jun
3     4          Spider-Man: Homecoming  ...            4348   14-Apr
4     5               Despicable Me 3  ...            4529    3-Mar

[5 rows x 8 columns]
     Rank                   MovieTitle  ... OpeningTheaters OpenDate
454   492               Red Christmas  ...               1   26-Aug
455   493            It's Not Yet Dark  ...               2    8-Sep
456   495         The Penguin Counters  ...               1   18-Jan
457   496  Extraordinary Ordinary People  ...               1   15-Mar
458   499                         2:22  ...               3      NaN
```

3 Save DataFrame to csv

https://stackoverflow.com/questions/16923281/writing-a-pandas-dataframe-to-csv-file

- Applied the function *df.to_csv('name.csv')*



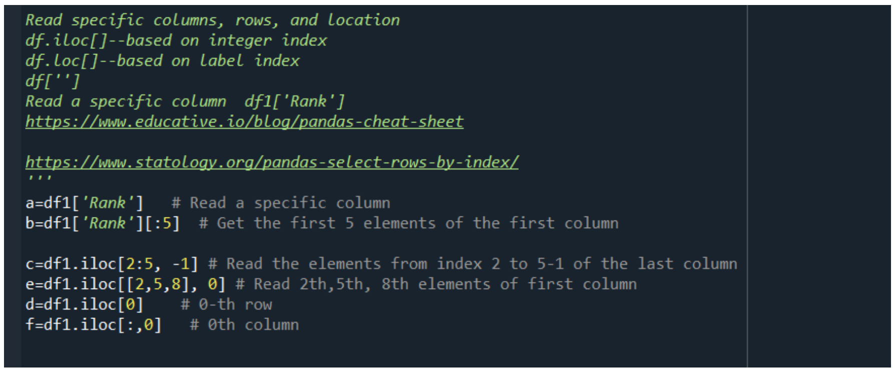
```
'''
Write data in DataFrame to a csv
https://stackoverflow.com/questions/16923281/writing-a-pandas-dataframe-to-csv-file
'''
df1.to_csv('Movie.csv')
```

4 Read specific columns, rows, and location

https://www.educative.io/blog/pandas-cheat-sheet

https://www.statology.org/pandas-select-rows-by-index/

- 1: read a column by df['column_name'],
- 2: Use df.iloc [a,b] for integers a,b
- 3: Use df.loc[a,b] for labels a,b

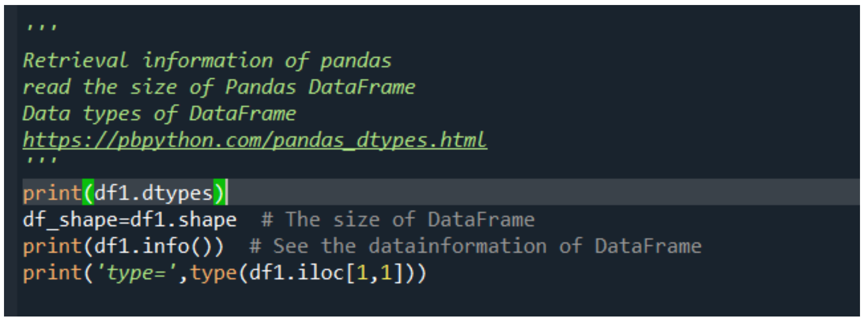


```
Read specific columns, rows, and location
df.iloc[]--based on integer index
df.loc[]--based on label index
df['']
Read a specific column  df1['Rank']
https://www.educative.io/blog/pandas-cheat-sheet

https://www.statology.org/pandas-select-rows-by-index/
'''
a=df1['Rank']    # Read a specific column
b=df1['Rank'][:5]  # Get the first 5 elements of the first column

c=df1.iloc[2:5, -1] # Read the elements from index 2 to 5-1 of the last column
e=df1.iloc[[2,5,8], 0] # Read 2th,5th, 8th elements of first column
d=df1.iloc[0]     # 0-th row
f=df1.iloc[:,0]    # 0th column
```

5 Read information of DataFrame

https://pbpython.com/pandas_dtypes.html

- Using function *df1.dtypes*, where df1 is a DataFrame
- Using df.info()
- The function type(element)



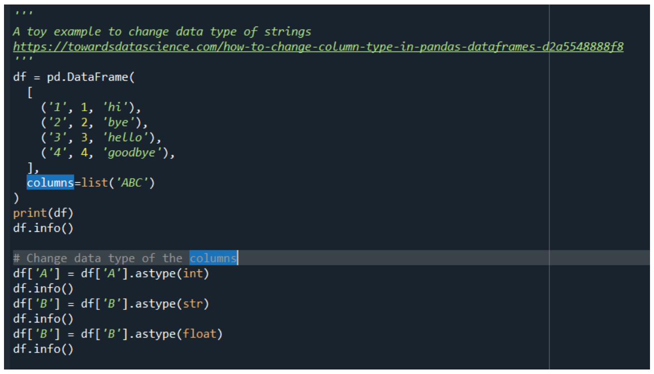
```
'''
Retrieval information of pandas
read the size of Pandas DataFrame
Data types of DataFrame
https://pbpython.com/pandas_dtypes.html
'''
print(df1.dtypes)
df_shape=df1.shape  # The size of DataFrame
print(df1.info())  # See the datainformation of DataFrame
print('type=',type(df1.iloc[1,1]))
```

6 Change data type

https://towardsdatascience.com/how-to-change-column-type-in-pandas-dataframes-d2a5548888f8

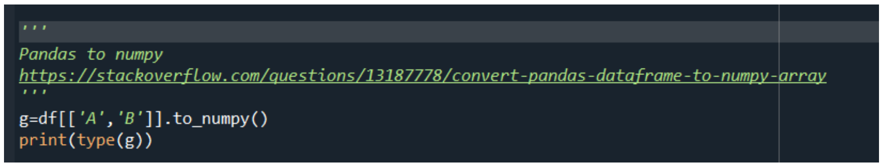
https://stackoverflow.com/questions/13187778/convert-pandas-dataframe-to-numpy-array

- Use function df['a'].astype(int): change the data in column "a" as int



```
'''
A toy example to change data type of strings
https://towardsdatascience.com/how-to-change-column-type-in-pandas-dataframes-d2a5548888f8
'''
df = pd.DataFrame(
  [
    ('1', 1, 'hi'),
    ('2', 2, 'bye'),
    ('3', 3, 'hello'),
    ('4', 4, 'goodbye'),
  ],
  columns=list('ABC')
)
print(df)
df.info()

# Change data type of the columns
df['A'] = df['A'].astype(int)
df.info()
df['B'] = df['B'].astype(str)
df.info()
df['B'] = df['B'].astype(float)
df.info()
```

- Read a specific column and use specific function int() to change the data type, and then replace the original column.
- Change DataFrame to numpy matrix by using df.to_numpy()

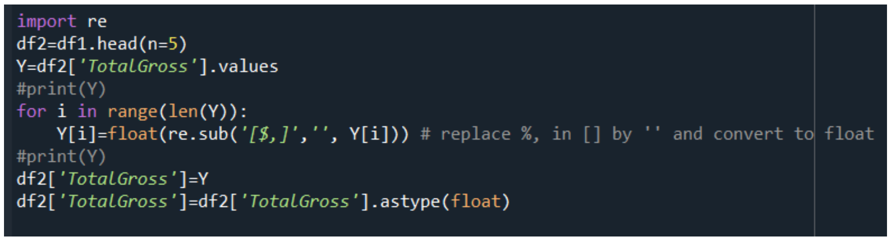


```
'''
Pandas to numpy
https://stackoverflow.com/questions/13187778/convert-pandas-dataframe-to-numpy-array
'''
g=df[['A','B']].to_numpy()
print(type(g))
```

7 Remove characters in specific elements

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.str.replace.html

- Use the function re.sub() from the library re



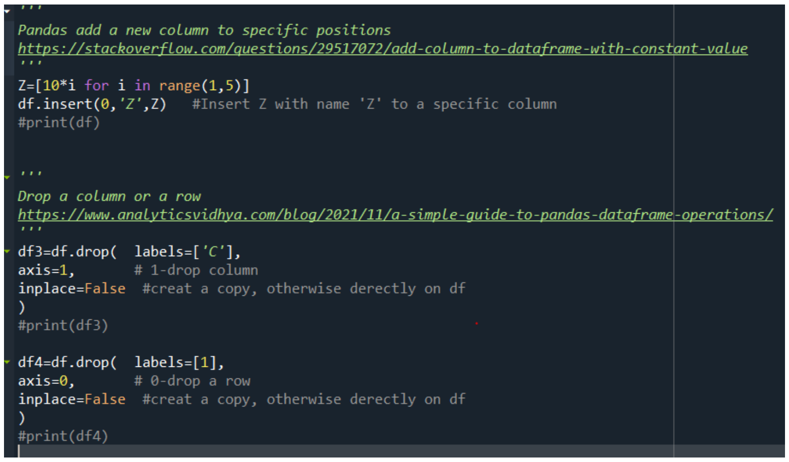
```
import re
df2=df1.head(n=5)
Y=df2['TotalGross'].values
#print(Y)
for i in range(len(Y)):
    Y[i]=float(re.sub('[$,]','', Y[i])) # replace %, in [] by '' and convert to float
#print(Y)
df2['TotalGross']=Y
df2['TotalGross']=df2['TotalGross'].astype(float)
```

8 Insert and Drop a column

https://stackoverflow.com/questions/29517072/add-column-to-dataframe-with-constant-value

https://www.analyticsvidhya.com/blog/2021/11/a-simple-guide-to-pandas-dataframe-operations/

- Using the function df.insert() to insert a column to a specific position
- Using df.drop() to delete a specific column or row

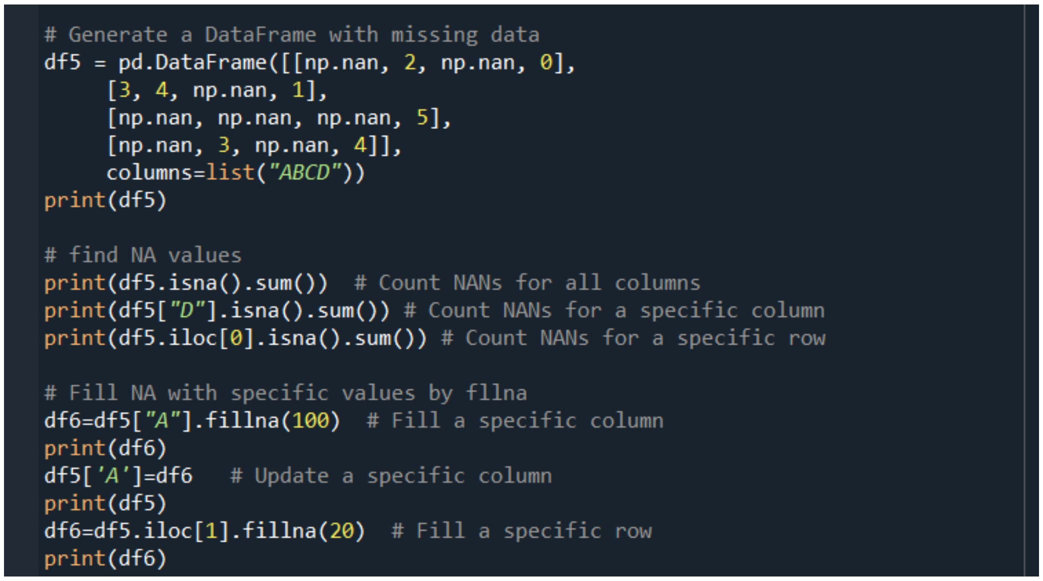


```
'''
Pandas add a new column to specific positions
https://stackoverflow.com/questions/29517072/add-column-to-dataframe-with-constant-value
'''
Z=[10*i for i in range(1,5)]
df.insert(0,'Z',Z)   #Insert Z with name 'Z' to a specific column
#print(df)


'''
Drop a column or a row
https://www.analyticsvidhya.com/blog/2021/11/a-simple-guide-to-pandas-dataframe-operations/
'''
df3=df.drop(  labels=['C'],
axis=1,       # 1-drop column
inplace=False  #creat a copy, otherwise derectly on df
)
#print(df3)

df4=df.drop(  labels=[1],
axis=0,       # 0-drop a row
inplace=False  #creat a copy, otherwise derectly on df
)
#print(df4)
```

9 Handle missing values—NAN

https://www.analyticsvidhya.com/blog/2021/11/a-simple-guide-to-pandas-dataframe-operations/

- Create a DataFrame with NAN
- Using function *df.isna()* to detect nan elements
- Using function *df.fillna()* to fill NAN with specific values or mean by df.count().mean



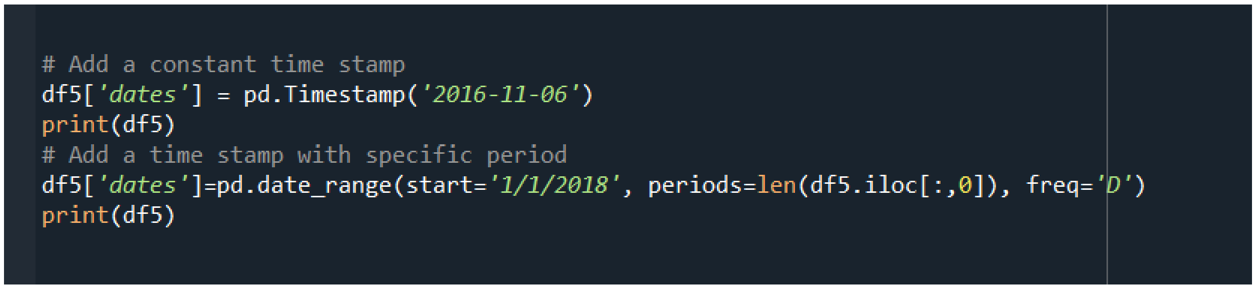
```
# Generate a DataFrame with missing data
df5 = pd.DataFrame([[np.nan, 2, np.nan, 0],
     [3, 4, np.nan, 1],
     [np.nan, np.nan, np.nan, 5],
     [np.nan, 3, np.nan, 4]],
     columns=list("ABCD"))
print(df5)

# find NA values
print(df5.isna().sum())  # Count NANs for all columns
print(df5["D"].isna().sum()) # Count NANs for a specific column
print(df5.iloc[0].isna().sum()) # Count NANs for a specific row

# Fill NA with specific values by fllna
df6=df5["A"].fillna(100)  # Fill a specific column
print(df6)
df5['A']=df6   # Update a specific column
print(df5)
df6=df5.iloc[1].fillna(20)  # Fill a specific row
print(df6)
```

10 Add date stamp

https://stackoverflow.com/questions/40858880/add-a-date-column-in-pandas-df-using-constant-value-in-str

https://pandas.pydata.org/docs/reference/api/pandas.date_range.html

- Add a constant time by df.Timestamp()
- Add a range of dates by pd.date_range(start='1/1/2018', periods=len(df5.iloc[:,0]), freq='D')

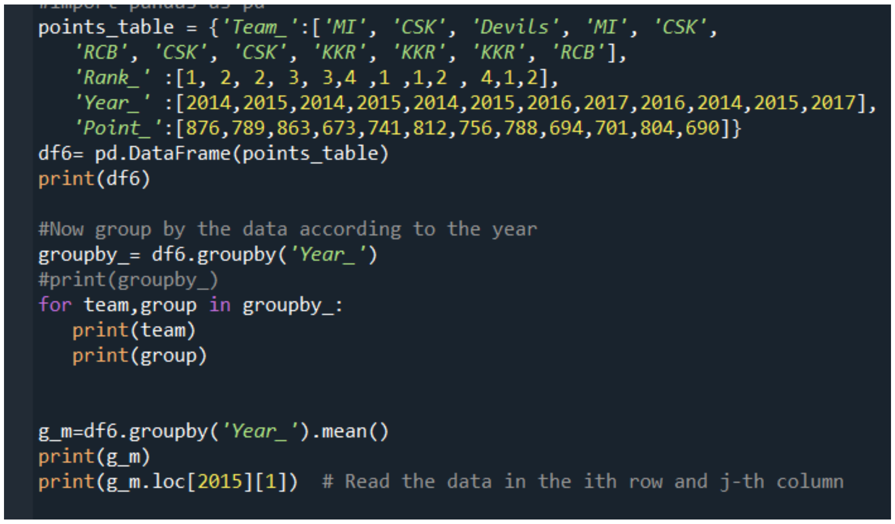


```
# Add a constant time stamp
df5['dates'] = pd.Timestamp('2016-11-06')
print(df5)
# Add a time stamp with specific period
df5['dates']=pd.date_range(start='1/1/2018', periods=len(df5.iloc[:,0]), freq='D')
print(df5)
```

11 Group data

https://stackoverflow.com/questions/63357396/calculate-mean-on-multiple-groups

- Using function *df.groupby('category_name')* to group data based on the specific category. Note that the output is still a DataFrame.

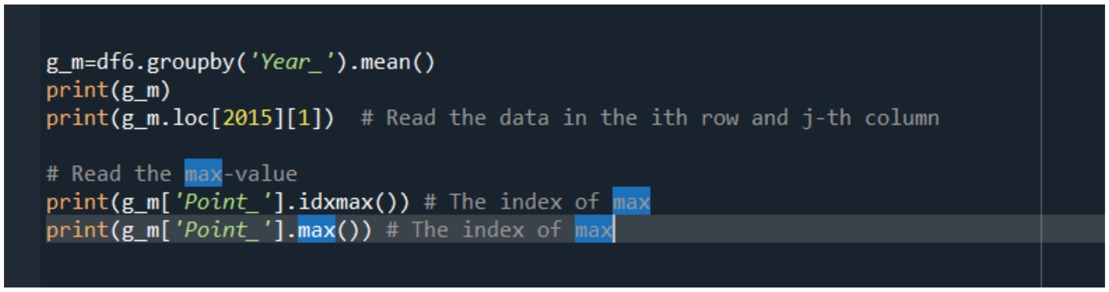


```
#import pandas as pd
points_table = {'Team_':['MI', 'CSK', 'Devils', 'MI', 'CSK',
   'RCB', 'CSK', 'CSK', 'KKR', 'KKR', 'KKR', 'RCB'],
   'Rank_' :[1, 2, 2, 3, 3,4 ,1 ,1,2 , 4,1,2],
   'Year_' :[2014,2015,2014,2015,2014,2015,2016,2017,2016,2014,2015,2017],
   'Point_':[876,789,863,673,741,812,756,788,694,701,804,690]}
df6= pd.DataFrame(points_table)
print(df6)

#Now group by the data according to the year
groupby_= df6.groupby('Year_')
#print(groupby_)
for team,group in groupby_:
   print(team)
   print(group)


g_m=df6.groupby('Year_').mean()
print(g_m)
print(g_m.loc[2015][1])  # Read the data in the ith row and j-th column
```

12 Find max/min and their indices idxmax/idxmin

https://www.analyticsvidhya.com/blog/2021/11/a-simple-guide-to-pandas-dataframe-operations/

- Using the function g_m['Point_'].idxmax() to show the index of the max value in the "Point_" column
- Using the function g_m['Point_'].max() to find the max value.



```
g_m=df6.groupby('Year_').mean()
print(g_m)
print(g_m.loc[2015][1])  # Read the data in the ith row and j-th column

# Read the max-value
print(g_m['Point_'].idxmax()) # The index of max
print(g_m['Point_'].max()) # The index of max
```