



Section (5) **Data Mining**

Nagwa El-Araby , Mohammed El-Babeer , Aml Magdi
Information System Department
2019 - 2020



Data Manipulation with pandas (Review)

- **Import Required Packages :**
 - Use required packages using **import** statement.
 - The **'as'** is used to alias package name.
 - In the syntax below, we import *numpy*

```
import numpy as np
```

```
import pandas as pd
```



Data Manipulation with pandas (Review)

- To import data from CSV file:
 - You can use **read_csv()** function from pandas package to get data into python from CSV file.
 - Example:
 - `mydata=`
`pd.read_csv("C:\\Users\\DataScience\\Documents\\file1.csv")`
 - Make sure you use **double backslash (\\)** when specifying path of CSV file.
 - Alternatively, you can use forward slash (/) to mention file path inside `read_csv()` function.

Data Manipulation with pandas (Review)

– Build DataFrame:

- Using **DataFrame()** function of pandas package.

Import pandas as pd

mydata =


{'productcode':

['AA', 'AA', 'AA', 'BB', 'BB', 'BB'],

'sales': [1010, 1025.2, 1404.2, 1251.7,
1160, 1604.8],

'cost' : [1020, 1625.2, 1204, 1003.7,
1020, 1124]}


df = pd.DataFrame(mydata)



productcode	sales	cost
AA	1010	1020
AA	1025.2	1625.2
AA	1404.2	1204
BB	1251.7	1003.7
BB	1160	1020
BB	1604.8	1124

Data Manipulation with pandas (Review)

- To see number of rows and columns:
 - You can run the command below to find out number of rows and columns.
 - `df.shape`
 - **Result** : (6, 3). It means 6 rows and 3 columns
- To view a specific number of rows:
 - The `df.head(N)` function can be used to check out first some N rows.
 - By Default, Return with 5 rows.
 - The `df.tail(N)` function can be used to check out last some N rows.
 - By Default, Return with 5 rows.



	cost	productcode	sales
0	1020.0	AA	1010.0
1	1625.2	AA	1025.2
2	1204.0	AA	1404.2

Data Manipulation with pandas

DataFrame Examples

```
>>> df = pd.DataFrame([[1, 2], [4, 5], [7, 8]],  
...                    index=['cobra', 'viper', 'sidewinder'],  
...                    columns=['max_speed', 'shield'])
```

```
>>> df
```

	max_speed	shield
cobra	1	2
viper	4	5
sidewinder	7	8



Data Manipulation with pandas

DataFrame Examples

– DataFrame.loc

- Access a group of rows and columns by label(s).
- Example – 1:

```
>>> df.loc['viper']  
max_speed      4  
shield         5  
Name: viper, dtype: int64
```



Data Manipulation with pandas

DataFrame Examples

– DataFrame.loc

- Access a group of rows and columns by label(s).
- Example – 2:

```
>>> df.loc['cobra':'viper', 'max_speed']  
cobra      1  
viper      4  
Name: max_speed, dtype: int64
```




Data Manipulation with pandas

DataFrame Examples

– Select or Drop Variable:

- To keep a single variable, you can write in any of the following three methods
 - **`df.max_speed`**
 - **`df["max_speed"]`**
 - **`df.loc[:, "max_speed"]`**

```
max_speed
cobra      1
viper      4
sidewinder 7
```



Data Manipulation with pandas DataFrame Examples

– Select or Drop Variable:

- We can keep multiple variables by specifying desired variables inside `[]`.
- Also, we can make use of `df.loc()` function.
- Example:
 - `df[["max_speed", "shield"]]`
`df.loc[: , [" max_speed ", " shield "]]`

	max_speed	shield
cobra	1	2
viper	4	5
sidewinder	7	8



Data Manipulation with pandas

DataFrame Examples

– Select or Drop Variable:

- To select variable by column position, you can use DataFrame.**iloc** function.
- In the example below, we are selecting second column.
 - **df.iloc[:, 1]**
 - Column Index starts from 0. Hence, 1 refers to second column.

	shield
cobra	2
viper	5
sidewinder	8



Data Manipulation with pandas

DataFrame Examples

– Select or Drop Variable:

- DataFrame.**drop()** :
 - Drop specified labels from rows or columns.
 - Remove rows or columns by specifying label names and corresponding axis, or by specifying directly index or column names.
 - Example:
 - **df2 = df.drop(['max_speed'], axis = 1)**
 - » Note : axis = 1 (for columns) , axis = 0 (for rows).
 - » Default (0)



Data Manipulation with pandas

DataFrame Examples

- To summarize DataFrame:
 - **df.describe()**
 - Generate descriptive statistics that summarize the central tendency, dispersion, and shape of a dataset distribution.

	cost	sales
count	6.000000	6.000000
mean	1166.150000	1242.650000
std	237.926793	230.466669
min	1003.700000	1010.000000
25%	1020.000000	1058.900000
50%	1072.000000	1205.850000
75%	1184.000000	1366.075000
max	1625.200000	1604.800000



Data Manipulation with pandas DataFrame Examples

- To summarize DataFrame:
 - **include** :
 - 'all', list-like of dtypes or None (default), optional
 - A white list of data types to include in the result.
 - None (default) : The result will include all numeric columns.
 - To summaries all the **character variables**, you can use the following script.
 - **df.describe(include=['O'])**
 - Similarly, you can use to view summary of all the numeric variables with decimals
 - **df.describe(include=['float64'])**



Data Manipulation with pandas

DataFrame Examples

– To summarize DataFrame:

- To select only a particular variable, you can write the following code:

- **`df.productcode.describe()`**

OR

- **`df["productcode"].describe()`**

```
count      6
unique      2
top         BB
freq        3
Name: productcode, dtype: object
```



Data Manipulation with pandas DataFrame Examples

- To calculate summary statistics:
 - We can manually find out summary statistics such as count, mean, median by using commands below:
 - **df.sales.mean()**
 - **df.sales.median()**
 - **df.sales.count()**
 - **df.sales.min()**
 - **df.sales.max()**



Data Manipulation with pandas DataFrame Examples

– Filter Data:

- Suppose you are asked to apply condition - productcode is equal to "AA" and sales greater than or equal to 1250.

– **df1 = df[(df.productcode == "AA") & (df.sales >= 1250)]**

It can also be written like :

– **df1 = df.query('(productcode == "AA") & (sales >= 1250)')**

» we do not need to specify DataFrame along with variable name.



Data Manipulation with pandas DataFrame Examples

– Sort Data:

- In the code below, we are arrange data in ascending order by sales.
 - **df.sort_values(['sales'])**
 - **ascending** : bool parameter , default True
 - For **DESC** ordering : **ascending=False**



Data Manipulation with pandas DataFrame Examples

– Group By :

- Summary by Grouping Variable
- Like SQL GROUP BY, you want to summarize continuous variable by classification variable.
- In this case, we are calculating average sale and cost by product code.
 - **`df.groupby(df.productcode).mean()`**



Data Manipulation with pandas DataFrame Examples

– Group By :

- Summary by Grouping Variable
- Instead of summarizing for multiple variable, you can run it for a single variable i.e. sales. Submit the following script.

– **`df["sales"].groupby(df.productcode).mean()`**

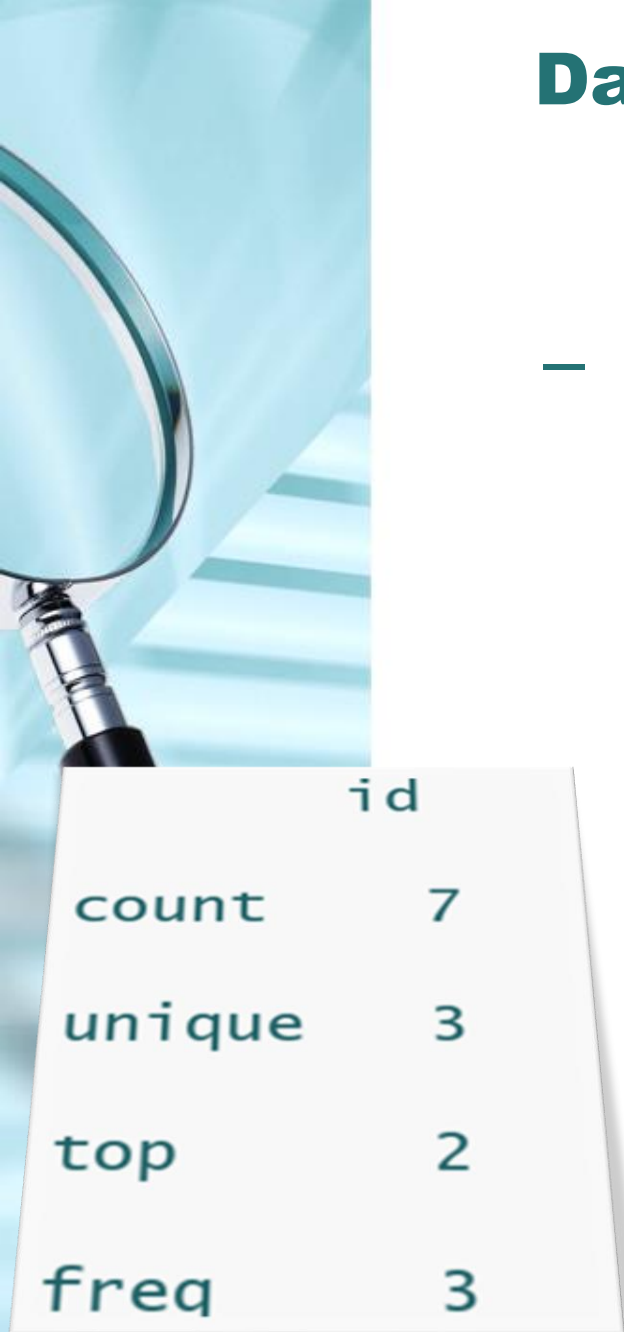
	cost	sales
productcode		
AA	1283.066667	1146.466667
BB	1049.233333	1338.833333

Data Manipulation with pandas

DataFrame Examples

– Define Categorical Variable:

- Let's create a classification variable - id which contains only 3 unique values - 1/2/3.
- Example:
 - **`df0 = pd.DataFrame({'id': [1, 1, 2, 3, 1, 2, 2]})`**
- Let's define as a categorical variable.
We can use `astype()` function to make id as a categorical variable.
 - **`df0["id"].astype('category')`**
- Summarize this classification variable to check descriptive statistics:
 - **`df0["id"].describe()`**



	id
count	7
unique	3
top	2
freq	3



Data Manipulation with pandas DataFrame Examples

– Frequency Distribution:

- You can calculate frequency distribution of a categorical variable.
- It is one of the method to explore a categorical variable.
- **`df['productcode'].value_counts()`**

BB	3
AA	3



Data Manipulation with pandas

DataFrame Examples

– Generate Histogram:


- Histogram is one of the method to check distribution of a continuous variable.
- In the figure shown below, there are two values for variable 'sales' in range 1000-1100.
- In the remaining intervals, there is only a single value.
- In this case, there are only 5 values.
- If you have a large dataset, you can plot histogram to identify outliers in a continuous variable.

– **`df['sales'].hist()`**

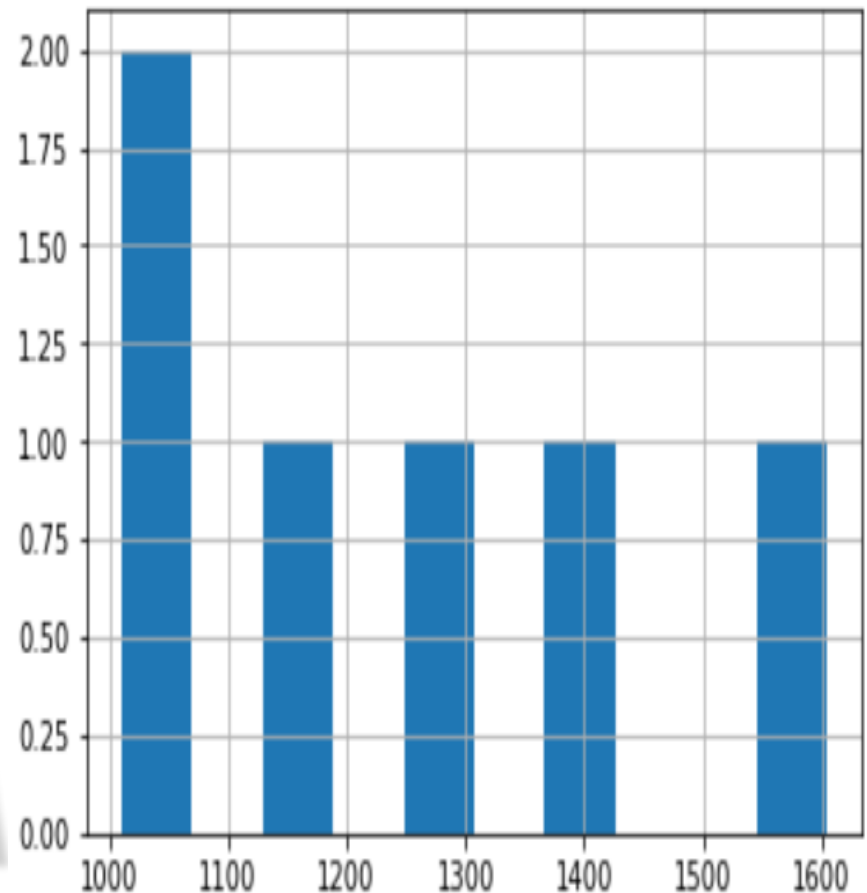
Data Manipulation with pandas

DataFrame Examples

– Generate Histogram:



<i>productcode</i>	<i>sales</i>	<i>cost</i>
AA	1010	1020
AA	1025.2	1625.2
AA	1404.2	1204
BB	1251.7	1003.7
BB	1160	1020
BB	1604.8	1124



Data Manipulation with pandas

DataFrame Examples

– BoxPlot:

- Boxplot is a method to visualize continuous or numeric variable.
- It shows minimum, Q1, Q2, Q3, IQR, maximum value in a single graph.

– **`df.boxplot(column='sales')`**

