

Python textbook and workbook

Ultimate

(C) 2021 by Dr Nathan Ng

*“Everybody should learn to program a computer, because it teaches you how to think.”- Steve Jobs*

# FOREWORDS

*Computer, Telecom, Artificial Intelligence is changing our lives. Some get better living, some go unemployed. The wave of change that entails every organization and person to transform digitally. Now or Never, Change or Die. That imperative trend insists every of us to be part of the global reshaping or to be completely excluded. Personal Digital Transformation opens a call for everyone to participate. For a better life for us now and our next generations.*

“

1. *“Any fool can write code that a computer can understand. Good programmers write code that humans can understand.” – Martin Fowler*
2. *“First, solve the problem. Then, write the code.” – John Johnson*
3. *“Experience is the name everyone gives to their mistakes.” – Oscar Wilde*
4. *“ In order to be irreplaceable, one must always be different” – Coco Chanel*
5. *“Java is to JavaScript what car is to Carpet.” – Chris Heilmann*
6. *“Knowledge is power.” – Francis Bacon*
7. *“Sometimes it pays to stay in bed on Monday, rather than spending the rest of the week debugging Monday’s code.” – Dan Salomon*
8. *“Perfection is achieved not when there is nothing more to add, but rather when there is nothing more to take away.” – Antoine de Saint-Exupery*
9. *“Ruby is rubbish! PHP is* phpantastic*!” – Nikita Popov*
10. *“ Code is like humor. When you have to explain it, it’s bad.” – Cory House*
11. *“Fix the cause, not the symptom.” – Steve Maguire*
12. *“Optimism is an occupational hazard of programming: feedback is the* treatment*. “ Kent Beck*
13. *“When to use iterative development? You should use iterative development only on projects that you want to succeed.” – Martin Fowler*
14. *“Simplicity is the soul of efficiency.” – Austin Freeman*
15. *“Before software can be reusable it first has to be usable.” – Ralph Johnson*
16. *“Make it work, make it right, make it fast.” – Kent Beck*

”

# Table of Content:

[**FOREWORDS**](#_heading=h.2et92p0) **[2](#_heading=h.2et92p0)**

[**Table of Content:**](#_heading=h.3dy6vkm) **[4](#_heading=h.3dy6vkm)**

[**Level 71: CSV files**](#_heading=h.4d34og8) **[5](#_heading=h.4d34og8)**

[**Level 72: Reading CSV files using pandas**](#_heading=h.2s8eyo1) **[7](#_heading=h.2s8eyo1)**

[**Level 73: Extracting data using iloc**](#_heading=h.17dp8vu) **9**

[**Level 74: Formatting data using reshape and ravel**](#_heading=h.3rdcrjn) **11**

[**Level 75. Draw a line chart using MatPlotLib**](#_heading=h.26in1rg) **13**

[**Level 76. Draw a scatter chart using MatPlotLib**](#_heading=h.lnxbz9) **15**

[**Level 77. Draw a bar chart using MatPlotLib**](#_heading=h.35nkun2) **17**

[**Level 78. Linear Regression**](#_heading=h.1ksv4uv) **19**

[**Level 79. Logistic Regression**](#_heading=h.44sinio) **23**

[**Level 80. K-mean clustering**](#_heading=h.z337ya) **26**

# 

# Level 71: CSV files

We can use a csv file to store data needed for machine learning.

For example, if we know the temperature and electricity consumption in the past we can predict the consumption in the future based on the predicted temperature.

**Temperature(oC),Energy(kWh)**

**30,200**

**31,210**

**33,240**

**34,250**

1. Build a sample file named **cloth.csv** to store temperature and money spent for buying warm clothes
2. Build a sample file named **flu.csv** to store temperature and number of cases of flu
3. Build a sample file named **dengue.csv** to store rain water and number of cases of dengue fever.
4. Build a sample file named **salary.csv** to store age and salary of each age
5. Build a sample file named **fun.csv** to store age and how much fun you have at that age.

# Level 72: Reading CSV files using pandas

Let’s say we have the following file energy.csv

**Temperature,Energy**

**30,200**

**31,210**

**33,240**

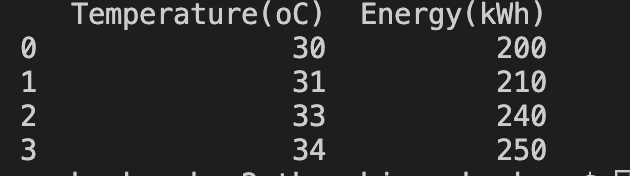
**34,250**

We can use a library called pandas to read csv files.

**import pandas as pd**

**data = pd.read\_csv("energy.csv")**

**print(data)**



To get only 1 column:

**print(data['Temperature'])**

Or

**print(data['Energy'])**

1. Write code to read and print data from **cloth.csv**
2. Write code to read and print data from **dengue.csv**
3. Write code to read and print data from **salary.csv**
4. Write code to read and print data from **fun.csv**
5. Write code to read and print data from **flu.csv**

# Level 73: Extracting data using iloc

We use **iloc** (index location) to get some columns or some rows from a table.

Let take an example: buycar.csv file

Ages,Income,BuyACar

30,100,0

33,150,0

40,120,1

45,200,1

48,400,1

We use pandas to read data from a csv file:

**import pandas as pd**

**data = pd.read\_csv("data.csv")**

To get all rows of the first column

**rows = data.iloc[:, 0:1]**

To get all rows of the first and second columns

**rows = data.iloc[:, 0:2]**

To get all rows of the second column

**rows = data.iloc[:, 1:2]**

To get all rows of the third column

**rows = data.iloc[:, 2:3]**

1. Get all rows of second and third columns
2. Get all rows of first, second and third columns
3. Get all rows of first column
4. Get all rows of first and second columns
5. Get all rows of third column

# Level 74: Formatting data using reshape and ravel

We use **reshape** and **ravel** to format data for training.

**reshape** (-1, 1) will create a 2 dimension array with 1 column

**ravel** will create 1D array

Let take an example: data.csv file

Ages,Income,BuyACar

30,100,0

33,150,0

40,120,1

45,200,1

48,400,1

We use pandas to read data from a csv file:

**import pandas as pd**

**data = pd.read\_csv("data.csv")**

To get all rows of the third column

**rows = data.iloc[:, 2:3]**

Format data using reshape:

**new\_rows = rows.values.reshape(-1, 1)**

Format data using ravel:

**new\_rows = rows.values.ravel()**

1. Get all rows of the first column and format it using reshape
2. Get all rows of the first column and format it using ravel
3. Get all rows of the second column and format it using reshape
4. Get all rows of the second column and format it using ravel
5. Get all rows of the third column and format it using reshape and ravel

# Level 75. Draw a line chart using **MatPlotLib**

We can open a csv file, read data and draw a chart to visual our data.

**import pandas as pd**

**import matplotlib.pyplot as plt**

**data = pd.read\_csv("energy.csv")**

**plt.plot(data['Temperature'], data['Energy'])**

1. Write code to read and draw a line chart from **cloth.csv**
2. Write code to read and draw a line chart from **dengue.csv**
3. Write code to read and draw a line chart from **salary.csv**
4. Write code to read and draw a line chart from **fun.csv**
5. Write code to read and draw a line chart from **flu.csv**

# Level 76. Draw a scatter chart using **MatPlotLib**

We can open a csv file, read data and draw a chart to visualize our data.

**import pandas as pd**

**import matplotlib.pyplot as plt**

**data = pd.read\_csv("energy.csv")**

**plt.scatter(data['Temperature'], data['Energy'], color='blue', label = 'Energy vs Temperature')**

1. Write code to read and draw a scatter chart from **cloth.csv**
2. Write code to read and draw a scatter chart from **dengue.csv**
3. Write code to read and draw a scatter chart from **salary.csv**
4. Write code to read and draw a scatter chart from **fun.csv**
5. Write code to read and draw a scatter chart from **flu.csv**

# Level 77. Draw a bar chart using **MatPlotLib**

We can open a csv file, read data and draw a chart to visual our data.

**import pandas as pd**

**import matplotlib.pyplot as plt**

**data = pd.read\_csv("data.csv")**

**plt.bar(data["Temp"], data['Energy'], color='red', label = 'Temp and Energy')**

**plt.xlabel('Temperature')**

**plt.ylabel('Energy')**

**plt.legend()**

**plt.grid(True)**

1. Write code to read and draw a bar chart from **cloth.csv**
2. Write code to read and draw a bar chart from **dengue.csv**
3. Write code to read and draw a bar chart from **salary.csv**
4. Write code to read and draw a bar chart from **fun.csv**
5. Write code to read and draw a bar chart from **flu.csv**

# Level 78. Linear Regression

Computers can learn the way humans learn.

You give them some data, they can learn and answer your question.

We call this process machine learning. This is a branch of AI - Artificial Intelligence.

With AI, we can predict the future based on past data.

For example, we can predict the energy consumption if we know the temperature.

| **from sklearn.linear\_model import LinearRegression**  import pandas as pd  import matplotlib.pyplot as plt  data = pd.read\_csv("energy.csv")  **lr = LinearRegression()**  temp\_x = data['Temperature'].values.reshape(-1, 1)  energy\_y = data['Energy'].values.reshape(-1, 1)  **lr.fit(temp\_x, energy\_y)**  **predicted\_energy = lr.predict(temp\_x)**  plt.plot(data['Temperature'], predicted\_energy) |
| --- |

1. Using Linear Regression, predict and visualize data from **cloth.csv**
2. Using Linear Regression, predict and visualize data from **dengue.csv**
3. Using Linear Regression, predict and visualize data from **salary.csv**
4. Using Linear Regression, predict and visualize data from **fun.csv**
5. Using Linear Regression, predict and visualize data from **flu.csv**

# Level 79. Logistic Regression

Similar to Linear Regression, Logistic Regression helps us answer questions. The difference is Logistic Regression focuses on answering Yes/No question.

Sometimes, we want to find answers for yes-no question such as:

* Will someone get an HD for a course given the number of hours he/she would spend to learn the course?
* Will someone buy a house when he is at a specific age and with specific income?
* Will an email be a spam mail?
* Will a credit card transaction be fraud?
* Will a customer buy a product?

Let take an example: buycar.csv file

Ages,Income,BuyACar

30,100,0

33,150,0

40,120,1

45,200,1

48,400,1

| from sklearn.linear\_model import LogisticRegression  import pandas as pd  data = pd.read\_csv("buycar.csv")  age\_income = data.iloc[:, 0:2]  buy = data.iloc[:, 2:3]  lr = LogisticRegression()  lr.fit(age\_income, buy.values.ravel())  predicted\_values = lr.predict(age\_income)  print(predicted\_values) |
| --- |

Explanation:

* iloc is to select data by index.

For example:

data.iloc[:, 0:2] means that you get all rows and the two first columns.

data.iloc[:, 2:3] means that you get all rows and the column indexed 2.

* We use the ravel() function to flatten an array to fit into the model training.
* We use reshape() function to create a new array

We use reshape(-1, 1) to reshape your data. Using array.reshape(-1, 1) if your data has a **single feature**

<https://stackoverflow.com/questions/18691084/what-does-1-mean-in-numpy-reshape>

1. Predict if someone gets an HD for a course given the number of hours he/she would spend to learn the course per week?

File hd.csv:

Hours,Year,HD

20,1,0

30,1,1

40,2,1

40,3,0

50,3,1

1. Predict if someone buys a house when he is at a specific age and with specific income?

File house.csv:

Age,Income,BuyHouse

30,20000,1

20,10000,0

35,30000,1

40,45000,1

1. Predict if someone gets a flu

Age,NumberOfPeopleMet,Flu

60,20,1

20,10,0

35,40,1

40,50,1

1. Predict if a credit card transaction is fraud?

File fraud.csv:

Country,Amount,Fraud

vn,1000,1

us,2000,0

cn,900,1

cn,1100,1

1. Predict if a customer will buy a spa service?

File spa.csv:

Age,Male,Female,Buy

30,0,1,1

20,1,0,0

35,0,1,1

40,0,1,1

# 

# Level 80A. Python Dictionary

To store many values in one variable, we can use a list, as you already know.

When we need to store a list of pairs of values, we can use a dictionary.

Like a real dictionary, each word in English will have a Vietnamese word.

For example:

I: toi

programming: lap trinh

love: thich

We can do this in Python as follows:

**words = {“I”: “toi”, “programming”: “lap trinh”, “love”: “thich”}**

If we have a list of pairs of x and y values:

**1 and 7**

**4 and 3**

**5 and 2**

**We can do this in Python as follows:**

**locations = {“x”: [1, 4, 5], “y”: [7, 3, 2]}**

1. Define a variable named **points** to store the following pairs of numbers:

(6,9), (8,2), (4,1)

1. Define a variable named **places** to store the following pairs of numbers:

(4,6), (3,4), (9,7)

1. Define a variable named **pairs** to store the following pairs of numbers:

(5,3), (7,2), (4,8)

1. Define a variable named **numbers** to store the following pairs of numbers:

(6,7), (3,3), (3,5)

1. Define a variable named **data** to store the following pairs of numbers:

(9,7), (6,8), (1,2)

# 

# 

# 

# Level 80B. Pandas DataFrame

Did you learn database tables? Data Frame is a kind of table in memory. We can use it to store data like in database tables.

For example: store a table of 2 columns x and y:

**x | y**

1 | 4

3 | 5

2 | 6

7 | 8

We first define a **dictionary** like in the previous level, then do as follows:

**import pandas as pd**

**df = pd.DataFrame({"x": [1, 3, 2, 7], "y": [4, 5, 6, 8]})**

1. Define a Data Frame named **df** to store the following pairs of numbers:

(6,9), (8,2), (4,1)

1. Define a Data Frame named **df** to store the following pairs of numbers:

(4,6), (3,4), (9,7)

1. Define a Data Frame named **df** to store the following pairs of numbers:

(5,3), (7,2), (4,8)

1. Define a Data Frame named **df** to store the following pairs of numbers:

(6,7), (3,3), (3,5)

1. Define a Data Frame named **df** to store the following pairs of numbers:

(9,7), (6,8), (1,2)

# 

# Level 80C. K-mean clustering

We already know that computers can learn - Machine Learning.

You give them some data, they can learn and answer your question.

They can answer 2 types of questions:

* Yes/No question: Will it rain tomorrow? Logistic Regression.
* Specific question: How much energy a household consumes next month?

Even more, Machine Learning can help us to group items into a group. That process is called clustering.

For example, we have 50 different places, how can we group nearby places into 3 groups?

We use a method called K-mean clustering to do that job.

| import pandas as pd  import matplotlib.pyplot as plt  df = pd.DataFrame({  'x': [12, 20, 28, 18, 29, 33, 24, 45, 45, 52, 51, 52, 55, 53, 55, 61, 64, 69, 72],  'y': [39, 36, 30, 52, 54, 46, 55, 59, 63, 70, 66, 63, 58, 23, 14, 8, 19, 7, 24]  })  from sklearn.cluster import KMeans  kmeans = KMeans(n\_clusters=3)  kmeans.fit(df)  centroids = kmeans.cluster\_centers\_  plt.scatter(df['x'], df['y'])  centroids\_x = centroids[:,0]  centroids\_y = centroids[:,1]  plt.scatter(centroids\_x,centroids\_y, c=['green', 'red','blue'])  plt.show() |
| --- |

1. Given the following DataFrame, perform k-mean clustering with 4 centroids

| import pandas as pd  import matplotlib.pyplot as plt  df = pd.DataFrame({  'x': [12, 20, 28, 18, 29, 33, 24, 45, 45, 52, 51, 52, 55, 53, 55, 61, 64, 69, 72],  'y': [39, 36, 30, 52, 54, 46, 55, 59, 63, 70, 66, 63, 58, 23, 14, 8, 19, 7, 24]  }) |
| --- |

1. Given the following DataFrame, perform k-mean clustering with 2 centroids

| import pandas as pd  import matplotlib.pyplot as plt  df = pd.DataFrame({  'x': [12, 20, 28, 18, 29, 33, 24, 45, 45, 52, 51, 52, 55, 53, 55, 61, 64, 69, 72],  'y': [39, 36, 30, 52, 54, 46, 55, 59, 63, 70, 66, 63, 58, 23, 14, 8, 19, 7, 24]  }) |
| --- |

1. Given the following DataFrame, perform k-mean clustering with 5 centroids

| import pandas as pd  import matplotlib.pyplot as plt  df = pd.DataFrame({  'x': [12, 20, 28, 18, 29, 33, 24, 45, 45, 52, 51, 52, 55, 53, 55, 61, 64, 69, 72],  'y': [39, 36, 30, 52, 54, 46, 55, 59, 63, 70, 66, 63, 58, 23, 14, 8, 19, 7, 24]  }) |
| --- |

1. Given the following DataFrame, perform k-mean clustering with 6 centroids

| import pandas as pd  import matplotlib.pyplot as plt  df = pd.DataFrame({  'x': [12, 20, 28, 18, 29, 33, 24, 45, 45, 52, 51, 52, 55, 53, 55, 61, 64, 69, 72],  'y': [39, 36, 30, 52, 54, 46, 55, 59, 63, 70, 66, 63, 58, 23, 14, 8, 19, 7, 24]  }) |
| --- |

1. Given the following DataFrame, perform k-mean clustering with 3 centroids

| import pandas as pd  import matplotlib.pyplot as plt  df = pd.DataFrame({  'x': [10, 20, 28, 10, 29, 33, 30, 45, 45, 34, 51, 12, 55, 53, 55, 76, 64, 69, 72],  'y': [39, 36, 30, 52, 10, 46, 55, 59, 63, 70, 66, 63, 58, 23, 14, 8, 19, 7, 24]  }) |
| --- |