

Statistical Techniques for Monitoring Industrial Processes



Lecture : Python Installation & Development Environment

Module : Python Installation and Basics

Course TOC

☐ Introduction to Statistical Process Monitoring (SPM)

☒ Python Installation and basics (optional)

- Development environment; Scientific computing packages

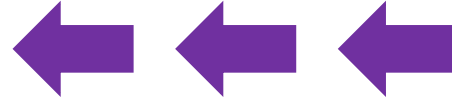
☐ Univariate SPM & Control Charts

- Shewhart Charts
- CUSUM Charts
- EWMA Charts

☐ Multivariate SPM

- Fault detection using Principal Component Analysis (PCA)
- Fault detection using Partial Least Squares (PLS) regression
- Fault diagnosis using PCA/PLS contribution charts
- Strategies for handling nonlinear, dynamic, multimode systems

☐ Deploying SPM solutions



Implemented using
Python

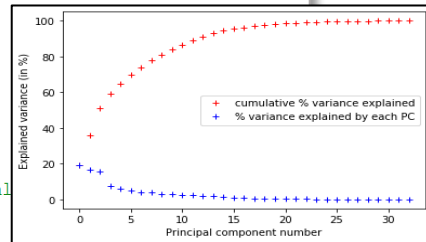
What is Python?

High-level modern
programming language

```

5  ### import required packages
6  import numpy as np
7  import pandas as pd
8  from sklearn.preprocessing import StandardScaler
9  from sklearn.decomposition import PCA
10
11  ### fetch data
12  data = pd.read_excel('procla.xls', skiprows = 1, usecols = 'C:AI')
13
14  ### separate train data
15  data_train = data.iloc[0:69,]
16
17  ### scale data
18  scaler = StandardScaler()
19  data_train_normal = scaler.fit_transform(data_train)
20
21  ### PCA
22  pca = PCA()
23  score_train = pca.fit_transform(data_train_normal)
24
25  ### confirm no correlation
26  corr_coef = np.corrcoef(score_train, rowvar = False)
27  print('Correlation matrix: \n', corr_coef[0:3,0:3]) # printing only
28
29  ### visualize explained variance
30  import matplotlib.pyplot as plt
31
32  explained_variance = 100*pca.explained_variance_ratio_ # in percentage
33  cum_explained_variance = np.cumsum(explained_variance) # cumulative % variance explained
34
35  plt.figure()
36  plt.plot(cum_explained_variance, 'r+', label = 'cumulative % variance explained')
37  plt.plot(explained_variance, 'b+', label = '% variance explained by each PC')
38  plt.ylabel('Explained variance (in %)')
39  plt.xlabel('Principal component number')
40  plt.legend()

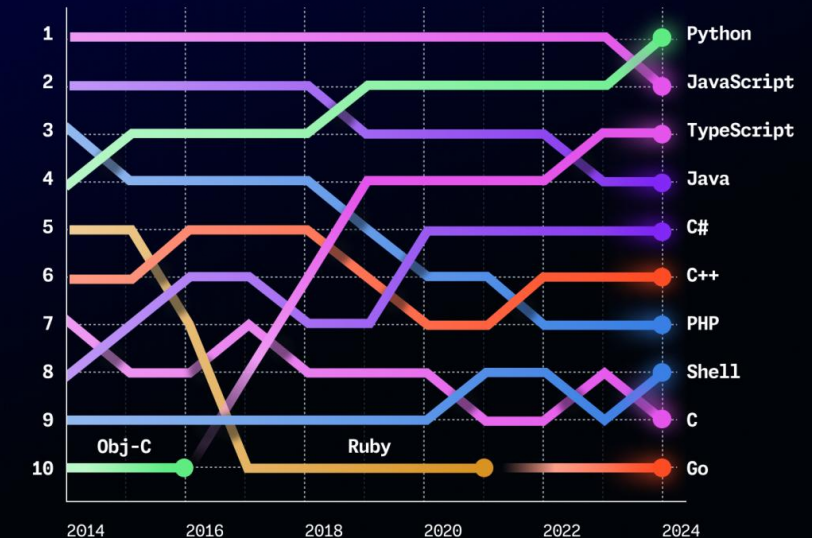
```



De-facto language of choice
for data science

Top programming languages on GitHub

RANKED BY COUNT OF DISTINCT USERS CONTRIBUTING TO PROJECTS OF EACH LANGUAGE.



Why Python is so Popular?



Easy & Simple

- Gentle learning curve
- Easy to read; Low maintenance



Versatile

- ML, DL, scientific computing
- Web development
- GUI programming



Extensive Libraries

- Large standard library set for common tasks
- Ever-growing collection from Python community



Community Support

- Vast community contributes to open-source knowledgebase
- Easy to find timely support



Documentation

- Well-written documentation
- Plenty of resources (guides, tutorials) for all level (beginner, expert)



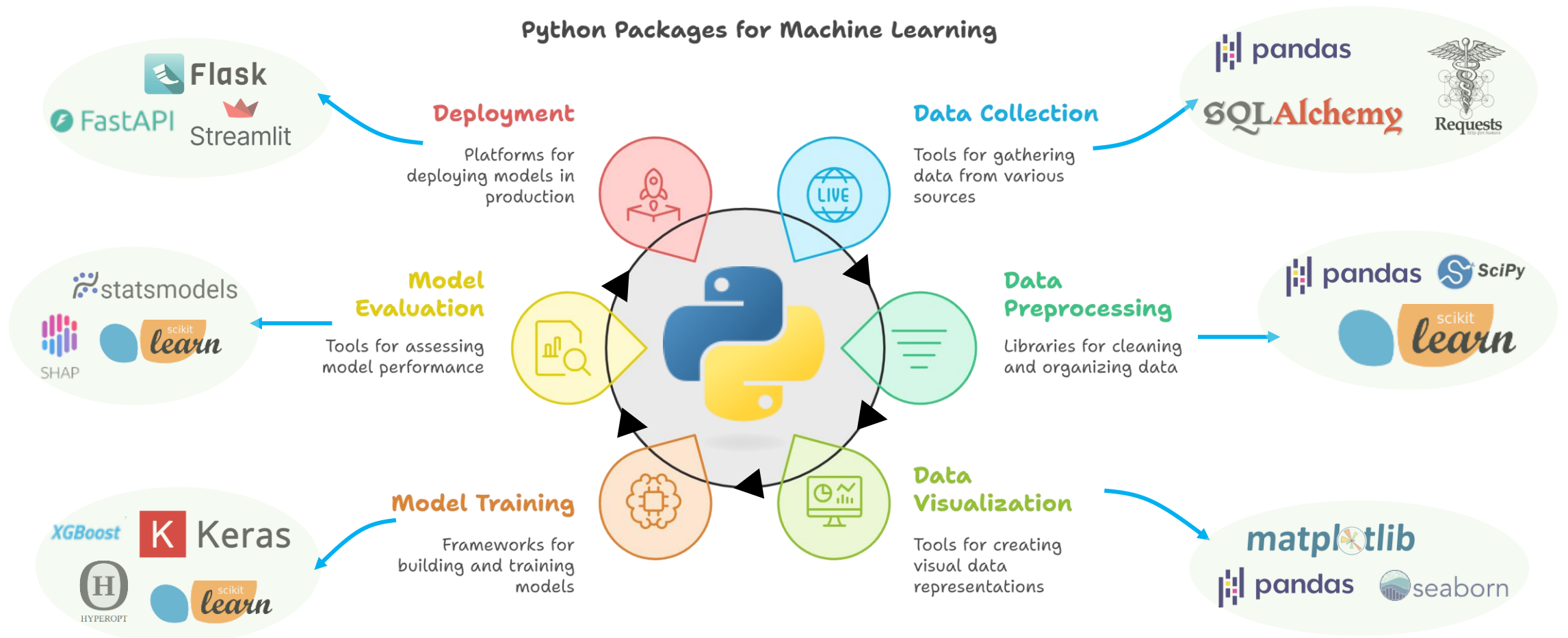
Portable / Platform-independent

- Code written on Windows PC can be easily executed on a Linux/Mac/UNIX machine

Why Python is so Popular?

Python provides several tools to conveniently perform all steps of a data science project

Python Packages for Machine Learning



Python Installation

Popular and convenient way

www.anaconda.com/download

ANACONDA. Products Solutions Resources Partners Company Sign Up


Download Now

For installation assistance, refer to [Troubleshooting](#).

Download Anaconda Distribution or [Miniconda](#) by choosing the proper installer for your machine. Learn the difference from our [Documentation](#).

Anaconda Installers


Download



Windows

Python 3.12

64-Bit Graphical Installer (912.3M)



Mac


Python 3.12

64-Bit (Apple silicon) Graphical Installer (704.7M)

64-Bit (Apple silicon) Command Line Installer (707.3M)

64-Bit (Intel chip) Graphical Installer (734.7M)

64-Bit (Intel chip) Command Line Installer (731.2M)



Linux

Python 3.12

64-Bit (x86) Installer (1007.9M)

64-Bit (AWS Graviton2 / ARM64) Installer (800.6M)

64-bit (Linux on IBM Z & LinuxONE) Installer (425.8M)

- When you install Anaconda, several commonly used packages are installed along with Python

Using base Python installer

www.python.org/downloads

Python PSF Docs PyPI Jobs Community

python™ Donate Search GO Socialize

About Downloads Documentation Community Success Stories News Events

Download the latest version for Windows

Download Python 3.13.0

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [macOS](#), [Other](#)

Want to help test development versions of Python 3.14? [Pre-releases](#), [Docker images](#)

Join the official Python Developers Survey 2024 and have a chance to win a prize Take the 2024 survey!

Active Python Releases

For more information visit the [Python Developer's Guide](#).

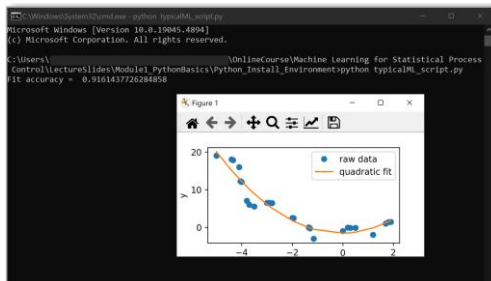
Python version	Maintenance status	First released	End of support	Release schedule
3.14	pre-release	2025-10-01 (planned)	2030-10	PEP 745
3.13	bugfix	2024-10-07	2029-10	PEP 719
3.12	bugfix	2023-10-02	2028-10	PEP 693
3.11	security	2022-10-24	2027-10	PEP 664
3.10	security	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596
3.8	end of life, last release was 3.8.20	2019-10-14	2024-10-07	PEP 569

Looking for a specific release?

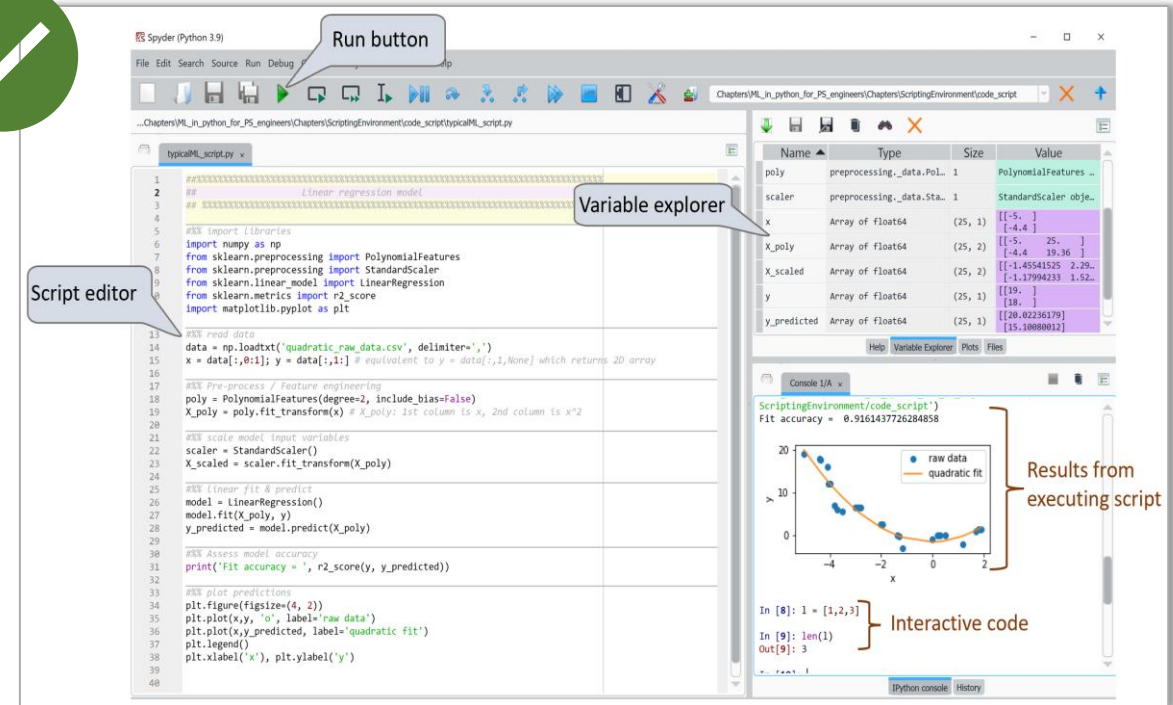
Where to Write Python Code?

Any text editor + command terminal execution

```
1 #!/usr/bin/env python
2 # Linear regression model
3 #
4 #
5 # Import libraries
6 import numpy as np
7 from sklearn.preprocessing import PolynomialFeatures
8 from sklearn.preprocessing import StandardScaler
9 from sklearn.linear_model import LinearRegression
10 from sklearn.metrics import r2_score
11 import matplotlib.pyplot as plt
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```



Using IDE (Integrated Development Environment)

Run button

Script editor

Variable explorer

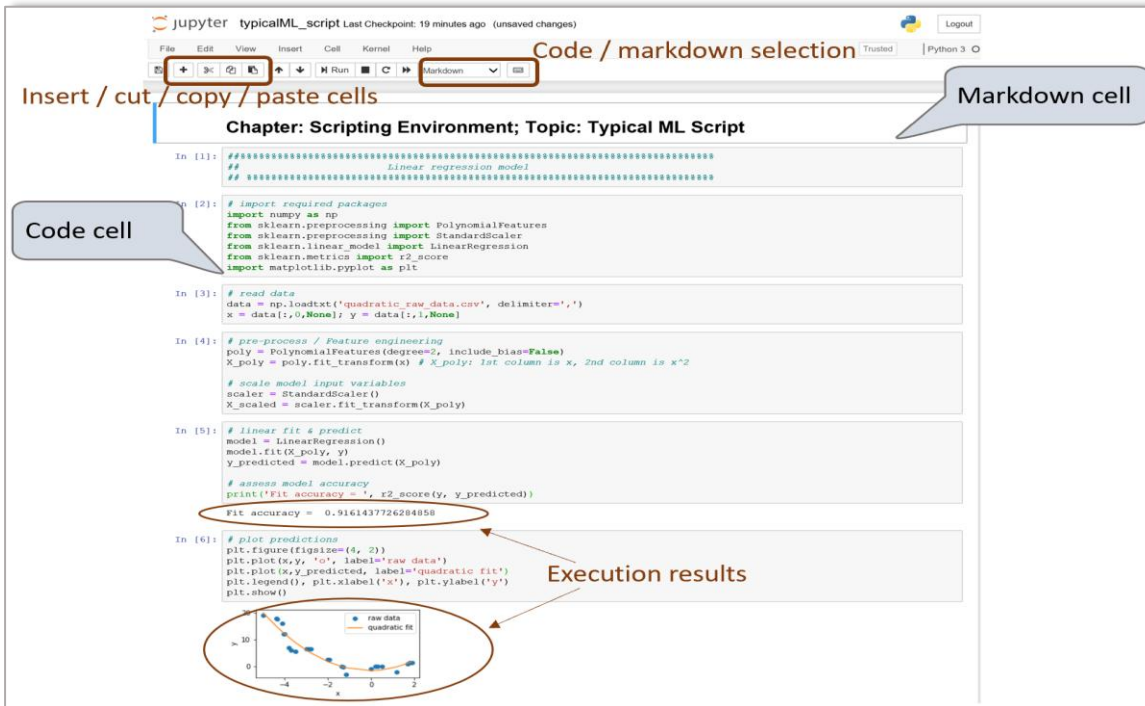
Results from executing script

Interactive code

- Allows you to write and execute your scripts, and see the results in the same environment
- Popular choices: PyCharm, Spyder, Visual Studio Code

Where to Write Python Code?

Using Jupyter Notebook



The screenshot shows a Jupyter Notebook titled 'typicalML_script'. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Help) and a toolbar with icons for inserting, cutting, copying, and pasting cells. A dropdown menu is open, showing 'Code / markdown selection'. The notebook contains several cells:

- Code cell:** Contains Python code for loading data, preprocessing, fitting a linear regression model, and plotting the results. The code is as follows:


```
##### Linear regression model #####
# import required packages
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
import matplotlib.pyplot as plt

# read data
data = np.loadtxt('quadratic_raw_data.csv', delimiter=',')
x = data[:,0,None]; y = data[:,1,None]

# pre-process / Feature engineering
poly = PolynomialFeatures(degree=2, include_bias=False)
X_poly = poly.fit_transform(x) # X_poly: 1st column is x, 2nd column is x^2

# scale model input variables
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_poly)

# linear fit & predict
model = LinearRegression()
model.fit(X_scaled, y)
y_predicted = model.predict(X_scaled)

# assess model accuracy
print('Fit accuracy = ', r2_score(y, y_predicted))
Fit accuracy = 0.916143772628856

# plot predictions
plt.figure(figsize=(4, 2))
plt.plot(x,y, 'o', label='raw data')
plt.plot(x,y_predicted, label='quadratic fit')
plt.legend(), plt.xlabel('x'), plt.ylabel('y')
plt.show()
```
- Markdown cell:** Contains the text 'Chapter: Scripting Environment; Topic: Typical ML Script'.
- Execution results:** A plot showing 'raw data' (blue dots) and 'quadratic fit' (orange line). The plot is titled 'quadratic fit' and has axes labeled 'x' and 'y'.

Very popular among data scientists for interactive data exploration and modeling experimentations

- Allows you to put code, results, plots, formatted text in the same document

Statistical Techniques for Monitoring Industrial Processes



Next Lecture : Python Language Basics

Module : Python Installation and Basics

