Foundations of Data Science & Analytics: Python Programming

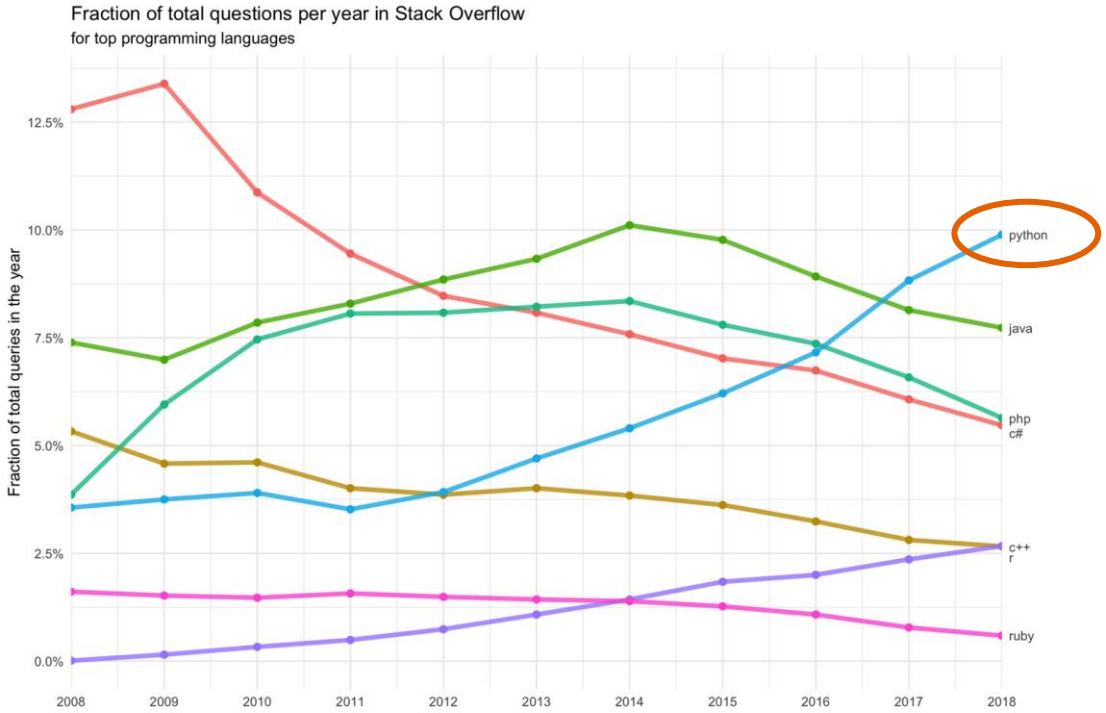
Ezgi Siir Kibris

Introduction to Data Mining, 2nd Edition
by
Tan, Steinbach, Karpatne, Kumar

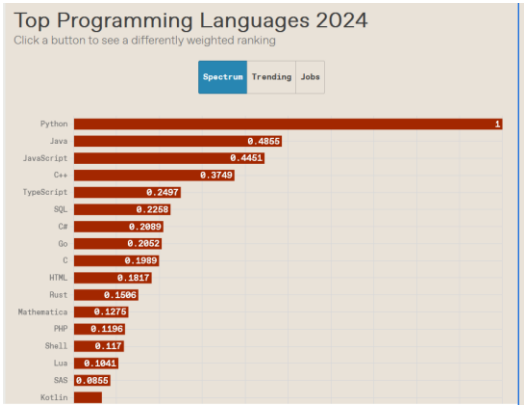
Why Python?

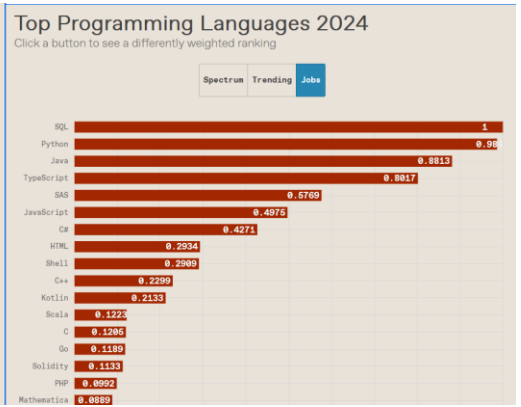
- Easy to code
  - Designed to minimize the time programmers spend programing.
- Can still be efficient
  - Most high computational packages are written in C.
- Large community support
  - Almost all data mining and machine learning tools/algorithms have Python implementations.
  - What else can we do when all other data science and machine learning researchers code in Python?

Programing Languages



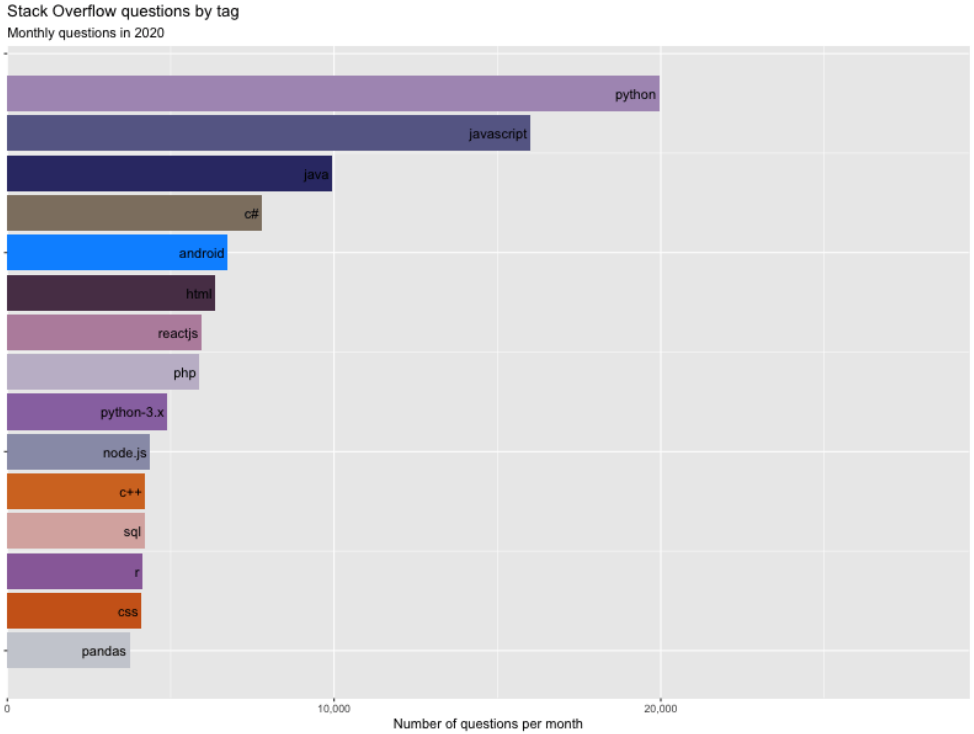
The Top Programming Languages 2024





https://shorturl.at/qQS2Y

Programing Languages



Easy to code

- Interpreted language
  - No explicit compiling or linking the code.
- Dynamically typed language
  - No explicit declaration of the type of a variable before using it.
- Less typing
  - Uses whitespace indentation to group together related statements in loops or other control-flow statements.
  - No terminating characters (e.g. semicolon in C/C++/Java).

Other difference

- Comment
  - # This is a comment
- Null object
  - a = None
- Access another Python program file
  - import numpy as np
  - from numpy import random
  - from numpy import *

A difference that leads to many errors

- Pass by reference (not value)

```
>>> a = [1, 2]
>>> b = a
>>> b[0] = 2
>>> print(b)
[2, 2]
>>> print(a)
[2, 2]
```

```
def my_function(input):
        input[0] = 2

>>> a = [1, 2]
>>> my_function(a)
>>> print(a)
[2, 2]
```

A difference that leads to many errors

- How to pass by value?

```
>>> a = [1, 2]
>>> b = a[:]
>>> b[0] = 2
>>> print(b)
[2, 2]
>>> print(a)
[1, 2]
```

```
# Some classes have deep copy func
import pandas as pd
>>> a = pd.DataFrame({“x”:[1, 2]})
>>> b = a.copy(deep = True)
>>> b[“x”][0] = 2
>>> print(a[“x”][0])
1
```

Data Type

```
x = 4               # integer
print(x, type(x))
```

4 <class 'int'>

```
y = True            # boolean (True, False)
print(y, type(y))
```

True <class 'bool'>

```
z = 3.7             # floating point
print(z, type(z))
```

3.7 <class 'float'>

```
s = "This is a string"    # string
print(s, type(s))
```

This is a string <class 'str'>

Assignment 1

Mac Users

- Homebrew:
  - https://brew.sh/
  - https://docs.brew.sh/Homebrew-and-Python