8 نکته کلیدی و مهم پایتون که خیلی از توسعه‌دهندگان اطلاعی از آن ندارند

پایتون شرایطی را فراهم کرده که دیگر برنامه‌نویسی برای علاقه‌مندان، یک آرزو نباشد و بسیاری از آنان که به کدنویسی نیاز دارند، استفاده از این زبان برنامه‌نویسی را یک مسیر خوب و مستقیم برای دستیابی به اهدافشان می‌دانند. از جمله کاربردهای پایتون، استفاده گسترده از آن در حوزه تحلیل داده‌ها یا به‌طور دقیق‌تر علم داده است. در زبان برنامه‌نویسی نکته‌هایی وجود دارد که آشنایی با آن‌ها کار ما را در کدنویسی برای تحلیل داده‌ها بسیار ساده‌تر خواهد کرد. در این مقاله چند نمونه از این نکته‌ها را که برگرفته از نوشته Conor Dewey است و در در وب‌سایت towardsdatascience.com منتشر شده، مرور می‌کنیم.

تهیه خلاصه فهرست با یک خط کد

نوشتن حلقه‌های For برای تعریف فهرست، کاری خسته‌کننده است. پایتون قابلیتی دارد که با کمک آن می‌توان این مشقت را با نوشتن یک خط کد به پایان رساند. این روش در ابتدا کمی گیج‌کننده به نظر می‌رسد، اما پس از آشنایی با آن متوجه خواهید شد که چقدر کار شما را در کدنویسی ساده‌تر می‌کند. در مثال زیر دو روش برای به توان دو رساندن اعضای یک فهرست استفاده‌شده و می‌توانید روش معمول استفاده از حلقه For را برای تولید خلاصه فهرست (List Comprehension) با روش تک‌خطی و بدون نیاز به نوشتن حلقه For مقایسه کنید. (شکل ۱)

```
x = [1,2,3,4]
out = []
for item in x:
    out.append(item**2)
print(out)

[1, 4, 9, 16]

# vs.

x = [1,2,3,4]
out = [item**2 for item in x]
print(out)

[1, 4, 9, 16]
```

توابع Lambda

شاید بارها اتفاق افتاده که مجبور شده‌اید برای برنامه خود و انجام کارهای ساده، توابع متعددی بنویسید. توابع lambda در چنین مواردی می‌توانند به کمک شما بیایند. می‌توان از lambda برای ساخت توابع جمع‌وجور، یک‌بارمصرف و بی‌نام در پایتون کمک گرفت، بدون این‌که مجبور باشید از روش معمول ساخت تابع در پایتون (کلمات کلیدی def و return) استفاده کنید. صورت کلی استفاده از این روش به قرار زیر است:

lambda arguments: expression

توابع lambda همان قابلیت‌های توابع معمول پایتون را دارند؛ با این تفاوت که فقط از یک عبارت می‌توان استفاده کرد.
برای درک بهتر توانایی‌های توابع lambda به مثال زیر توجه کنید:

```
double = lambda x: x * 2
print(double(5))
10
```

در شکل ۲، تابعی برای یافتن بزرگ‌ترین عدد نوشته‌شده است. (شکل2- الف. نوشتن چنین تابعی را به روش معمول نشان می‌دهد. شکل2- ب. از روش lambda استفاده‌شده است.)

```
#Max of x,y
def mx(x,y):
    if x>y:
        return x
    else:
        return y
print(mx(15,40))
```

```python
#Lambda / Max of x,y
mx=lambda x,y:x if x>y else y
print(mx(15,40))


```

توابع Map و Filter

ترکیب توابع lambda با دو تابع Map و Filter قابلیت‌های قدرتمندی را در اختیار شما قرار می‌دهد.
Map، یک فهرست را به‌عنوان ورودی دریافت کرده و با اجرای عملیاتی روی تک‌به‌تک اعضای آن فهرست، فهرستی دیگر تحویل می‌دهد. در این مثال تابع list، خروجی را در قالب «فهرست» تحویل می‌دهد.

Map#
seq = [1, 2, 3, 4, 5]
result = list(map(lambda var: var*2, seq))
print(result)
[[2, 4, 6, 8, 10

تابع filter یک فهرست را دریافت می‌کند و بر اساس قانونی که تعریف کرده‌ایم، عناصری از فهرست ورودی را انتخاب و نتیجه را در قالب یک فهرست دیگر که زیرمجموعه‌ای از ورودی است، تحویل می‌دهد.

Filter #
seq = [1, 2, 3, 4, 5]
result = list(filter(lambda x: x > 2, seq))
(print(result

[5 ,4 ,3]

توابع Arange و Linspace

برای تولید ساده و سریع آرایه‌های Numpy به گزینه‌ای جز این دو فکر نکنید. هر یک از این دو تابع کاربردهای خاص خودشان را دارند، ولی در اینجا می‌خواهیم به جای استفاده از Range از این دو استفاده کنیم. خروجی به‌صورت آرایه‌های Numpy خواهد بود که استفاده از آن‌ها در کاربردهایی نظیر علم داده ساده‌تر است. Arange مقادیری از یک بازه را برمی‌گرداند که فاصله‌شان از یکدیگر به یک اندازه است. در صورت نیاز می‌توانید علاوه بر نقطه شروع و پایان، اندازه هر گام (فاصله بین اعداد) و نوع داده را تعریف کنید. لازم به یادآوری است که نقطه پایان تعیین‌شده، در آرایه خروجی ظاهر نخواهد شد.

np.arange(start, stop, step) #
np.arange(3, 7, 2)
([array([3, 5

تابع Linspace نیز مشابه Arange است، اما با یک تفاوت جزیی. Linspace اعدادی از یک بازه تعیین‌شده را که با هم فاصله یکسانی دارند، برمی‌گرداند. با تعیین کردن نقاط شروع و پایان و «تعداد اعداد»، تابع linspace اعداد را با فاصله یکسان در آرایه NumPy می‌چیند. این تابع در کاربردهایی نظیر مصورسازی داده‌ها (Data Visualization) و تعریف محورهای مختصات برای رسم مفید خواهد بود.

np.linspace(start, stop, num) #
np.linspace(2.0, 3.0, num=5)
([array([2.0, 2.25, 2.5, 2.75, 3.0

Axis

به‌منظور کار روی داده‌های موجود در یک دیتافریم، لازم است به‌طور روشن به پایتون اعلام کنیم که قرار است در چه جهتی حرکت کنیم. فرض کنید دیتافریمی متشکل از چهارستون و سه سطر داریم و می‌خواهیم با استفاده از Drop ستون آخر را جدا کنیم. برای این کار با عبارت axis=1 اعلام می‌کنیم که تصمیم داریم روی ستون کار کنیم.

(df.drop('Column A', axis=1

اما اگر بخواهیم Drop را روی یک ردیف از دیتافریم اعمال کنیم، axis را صفر می‌دهیم:

(df.drop('Row A', axis=0

Concat و Merge و Join

اگر با SQL آشنایی داشته باشید، درک عملکرد این سه تابع برای شما ساده‌تر خواهد بود. با کمک این سه تابع و تعریف محور، می‌توانیم دیتافریم‌ها را آن‌گونه که می‌خواهیم با یکدیگر ترکیب کنیم. با استفاده از Concat می‌توان دیتافریم‌ها را بر اساس محوری که تعیین می‌کنید، برای مثال ستونی یا به ردیف، به هم الحاق کنید.(شکل ۳ـ الف) Merge چندین دیتافریم را بر اساس ستون‌های تعیین‌شده و مشترک به هم می‌چسباند (ترکیب می‌کند). (شکل ۳ـب) تابع Join نظیر Merge دو دیتافریم را ترکیب می‌کند. البته در اینجا دیتافریم‌ها بر اساس اندیس‌ها ترکیب می‌شوند، نه ستون‌های مشخص. (شکل ۳ـ ج)

left

	v
K0	1
K1	2
K2	3

right

	v
K0	4
K0	5
K3	6

right2

	v
K1	7
K1	8
K2	9

Result

	v_x	v_y	v
K0	1.0	4.0	NaN
K0	1.0	5.0	NaN
K1	2.0	NaN	7.0
K1	2.0	NaN	8.0
K2	3.0	NaN	9.0
K3	NaN	6.0	NaN

Apply

Apply شبیه تابع Map است، اما برای دیتافریم‌های Pandas و به‌طور دقیق‌تر، برای استفاده در سری‌ها (Series) مناسب است. سری‌ها از بسیاری جهات مشابه آرایه‌های NumPy هستند. Apply یک تابع را بر تک‌تک عناصر ستون یا سطری که تعیین کرده‌اید، اعمال می‌کند. واضح است که چنین قابلیتی تا چه میزان در کاربردهایی نظیر قالب‌بندی (Formatting) و دستکاری مقادیر موجود در یک ستون دیتافریم مفید است، بدون این‌که مجبور باشید از حلقه برای این کار استفاده کنید.

Pivot table

اگر تجربه کار با اکسل مایکروسافت را داشته باشید، احتمالا با جدول‌های محوری (Pivot Tables) آشنایی دارید. تابع Pivot_Table در Pandas امکان تولید جدول محوری به سبک صفحه گسترده و در قالب دیتافریم را فراهم می‌سازد. نکته این‌که levels های جدول در آبجکت‌های MultiIndex روی اندیس (index) و ستون‌های دیتافریم حاصل ذخیره می‌شوند.

تاریخ انتشار:
06 اردیبهشت 1398

نشانی منبع:

https://www.shabakeh-mag.com/workshop/programming/14707/8-%D9%86%DA%A9%D8%AA%D9%87-%DA%A9%D9%84%DB%8C%D8%AF%DB%8C-%D9%88-%D9%85%D9%87%D9%85-%D9%BE%D8%A7%DB%8C%D8%AA%D9%88%D9%86-%DA%A9%D9%87-%D8%AE%DB%8C%D9%84%DB%8C-%D8%A7%D8%B2-%D8%AA%D9%88%D8%B3%D8%B9%D9%87%E2%80%8C%D8%AF%D9%87%D9%86%D8%AF%DA%AF%D8%A7%D9%86-%D8%A7%D8%B7%D9%84%D8%A7%D8%B9%DB%8C-%D8%A7%D8%B2-%D8%A2%D9%86-%D9%86%D8%AF%D8%A7%D8%B1%D9%86%D8%AF