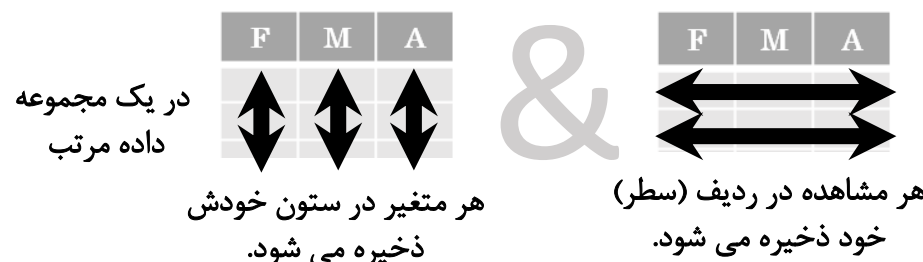
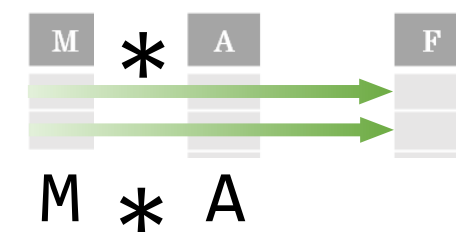


داده های مرتب (Tidy Data) - مباحث پایه ای داده آرایی در پانداس



داده های مرتب، عملیات برداری پانداس را تکمیل می کند. پانداس به طور خودکار مشاهدات را در حین دستکاری متغیرها حفظ می کند. هیچ فرمت دیگری به طور مستقیم با پانداس کار نمی کند.



Data Wrangling

with pandas Cheat Sheet
<http://pandas.pydata.org>

Pandas [API Reference](#) Pandas [User Guide](#)

ساخت DataFrames

	a	b	c
1	4	7	10
2	5	8	11
3	6	9	12

```
df = pd.DataFrame(
    {"a": [4, 5, 6],
     "b": [7, 8, 9],
     "c": [10, 11, 12]},
    index = [1, 2, 3])
```

مشخص کردن مقادیر هر ستون

```
df = pd.DataFrame(
    [[4, 7, 10],
     [5, 8, 11],
     [6, 9, 12]],
    index=[1, 2, 3],
    columns=['a', 'b', 'c'])
```

مشخص کردن مقادیر هر ردیف (سطر)

		a	b	c
N	v			
D	1	4	7	10
	2	5	8	11
e	2	6	9	12

```
df = pd.DataFrame(
    {"a": [4, 5, 6],
     "b": [7, 8, 9],
     "c": [10, 11, 12]},
    index = pd.MultiIndex.from_tuples(
        [('d', 1), ('d', 2),
         ('e', 2)], names=['n', 'v']))
```

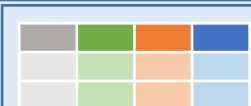
ایجاد یک DataFrame با استفاده از MultiIndex

روش زنجیره ای (Method Chaining)

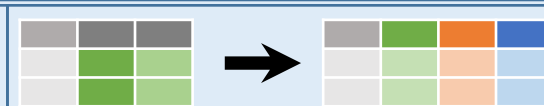
اکثر متدهای پانداس یک DataFrame برمی گردانند تا متد پانداس دیگری را بتوان در نتیجه اعمال کرد. این امر خوانایی کد را بهبود می بخشد.

```
df = (pd.melt(df)
      .rename(columns={
          'variable': 'var',
          'value': 'val'})
      .query('val >= 200'))
```

تغییر شکل داده ها - تغییر طرح، مرتب سازی، فهرست بندی مجدد، تغییر نام



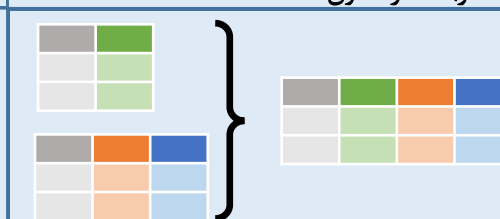
`pd.melt(df)`
جمع آوری ستون ها در ردیف (سطر) ها



`df.pivot(columns='var', values='val')`
گسترش ردیف (سطر) ها در ستون ها



`pd.concat([df1, df2])`
اضافه کردن ردیف (سطر) های دو DataFrame به هم



`pd.concat([df1, df2], axis=1)`
اضافه کردن ستون های دو DataFrame به هم

`df.sort_values('mpg')`
مرتب سازی ردیف (سطر) ها بر اساس مقادیر یک ستون (کم به زیاد).

`df.sort_values('mpg', ascending=False)`
مرتب سازی ردیف (سطر) ها بر اساس مقادیر یک ستون (زیاد به کم).

`df.rename(columns = {'y': 'year'})`
تغییر نام ستون های یک DataFrame

`df.sort_index()`
مرتب سازی ایندکس یک DataFrame

`df.reset_index()`
بازنشانی ایندکس یک DataFrame به شماره ردیف (سطر) ها و انتقال ایندکس به ستون ها.

`df.drop(columns=['Length', 'Height'])`
حذف ستون ها از یک DataFrame

مشاهدات زیر مجموعه - ردیف (سطر)



`df[df.Length > 7]`
استخراج ردیف (سطر) هایی که معیارهای منطقی را دارند.

`df.drop_duplicates()`
حذف ردیف (سطر) های تکراری (فقط ستون ها را در نظر میگیرد)

`df.sample(frac=0.5)`
انتخاب بخشی از ردیف (سطر) ها به صورت تصادفی

`df.sample(n=10)`
انتخاب n ردیف (سطر) به صورت تصادفی

`df.nlargest(n, 'value')`
انتخاب و مرتب سازی n ورودی بالایی

`df.nsmallest(n, 'value')`
انتخاب و مرتب سازی n ورودی پایینی

`df.head(n)`
انتخاب n ردیف (سطر) اول

`df.tail(n)`
انتخاب n ردیف (سطر) آخر

متغیرهای زیرمجموعه - ستون ها



`df[['width', 'length', 'species']]`
انتخاب چندین ستون با اسامی مشخص

`df['width']` or `df.width`
انتخاب یک ستون با نام مشخص

`df.filter(regex='regex')`
انتخاب ستون هایی که با عبارت منظم regex مطابقت دارند

استفاده از query

`query()` با استفاده از عبارات بولی توانایی فیلتر کردن ردیف (سطر) ها را ایجاد می کند

```
df.query('Length > 7')
df.query('Length > 7 and Width < 8')
df.query('Name.str.startswith("abc")',
        engine="python")
```

زیر مجموعه ها - ردیف ها و ستون ها

از `df.loc[]` و `df.iloc[]` برای انتخاب فقط سطر، فقط ستون یا هر دو استفاده کنید.

از `df.at[]` و `df.iat[]` برای دسترسی به یک مقدار واحد توسط ردیف (سطر) و ستون استفاده کنید.

ایندکس اول ردیف (سطر) ها را انتخاب می کند، ایندکس دوم ستون ها را

`df.iloc[10:20]`
ردیف (سطر) ۱۰-۲۰ را انتخاب می کند

`df.iloc[:, [1, 2, 5]]`
ستون ها را در موقعیت های ۱، ۲ و ۵ انتخاب می کند (ستون اول است).

`df.loc[:, 'x2': 'x4']`
تمام ستون های بین x2 و x4 را انتخاب می کند.

`df.loc[df['a'] > 10, ['a', 'c']]`
ردیف (سطر) هایی که شرط منطقی را دارند و فقط ستون های مشخص شده را انتخاب می کند.

`df.iat[1, 2]`
دسترسی به مقدار واحد براساس ایندکس

`df.at[4, 'A']`
دسترسی به مقدار واحد توسط برچسب

مثال های (عبارات منظم) regex

'\.'	رشته های حاوی نقطه "." را مطابقت می دهد.
'Length\$'	رشته هایی که با کلمه "Length" ختم می شوند مطابقت دارد
'^Sepal'	رشته هایی را که با کلمه "Sepal" شروع می شوند مطابقت می دهد.
'^x[1-5]\$'	رشته هایی که با 'x' شروع می شوند و با ۱، ۲، ۳، ۴، ۵ ختم می شوند، مطابقت می دهد
'^(?!Species\$).*'	رشته ها را به جز رشته «Species» مطابقت می کند

عملگر های منطقی در پایتون (و پانداس)			
<	کمتر از	!=	نامبرابر
>	بزرگتر از	<code>df.column.isin(values)</code>	عضویت در گروه
==	برابر	<code>pd.isnull(obj)</code>	است NaN
<=	کمتر مساوی	<code>pd.notnull(obj)</code>	نیست NaN
>=	بیشتر مساوی	<code>&, , ~, ^, df.any(), df.all()</code>	and, or, not, xor, any, all

خلاصه سازی داده

`df['w'].value_counts()`

تعداد ردیف (سطر) ها را با هر مقدار منحصر به فرد متغیر بشمارید

`len(df)`

تعداد ردیف (سطر) های DataFrame

`df.shape`

تاپلی از تعداد ردیف (سطر) ها و تعداد ستون ها در DataFrame

`df['w'].nunique()`

تعداد مقادیر متمایز در یک ستون

`df.describe()`

توصیف و آمار پایه برای هر ستون (یا GroupBy)



پانداس مجموعه بزرگی از توابع خلاصه را ارائه می دهد که بر روی انواع مختلف اشیاء

پانداس (ستون های DataFrame, Series, GroupBy, Expanding, Rolling و) عمل می کند و مقادیر واحدی را برای هر یک از گروه ها تولید می کند. هنگامی

که به یک DataFrame اعمال می شود، نتیجه به عنوان یک سری پانداس برای هر

ستون برگردانده می شود. مثال ها:

`sum()`

مجموع مقادیر هر آبجکت

`min()`

حداقل مقدار هر آبجکت

`count()`

تعداد مقادیر غیر non-NA/null هر آبجکت

`max()`

حداکثر مقدار هر آبجکت

`median()`

مقدار متوسط هر آبجکت

`mean()`

مقدار میانگین هر آبجکت

`quantile([0.25,0.75])`

چندک هر آبجکت

`var()`

واریانس هر آبجکت

`apply(function)`

اعمال تابع برای هر آبجکت

`std()`

انحراف استاندارد هر آبجکت

داده های گروهی (Group Data)



`df.groupby(by="col")`

یک شی GroupBy را که بر اساس مقادیر در ستونی به نام "col" گروه بندی شده است، برمی گرداند.

`df.groupby(level="ind")`

یک شی GroupBy را که بر اساس مقادیر در سطح ایندکس به نام "ind" گروه بندی شده است، برمی گرداند.

همه توابع خلاصه ذکر شده در بالا را می توان برای یک گروه اعمال کرد. توابع اضافی GroupBy:

`size()`

اندازه ی هر گروه

`agg(function)`

جمع کردن گروه با استفاده از تابع

Windows

`df.expanding()`

یک شی در حال گسترش را برمی گرداند که اجازه می دهد توابع خلاصه به صورت تجمعی اعمال شوند.

`df.rolling(n)`

یک شی Rolling را برمی گرداند که اجازه می دهد توابع خلاصه در پنجره هایی با طول n اعمال شوند.

رسیدگی داده های از دست رفته

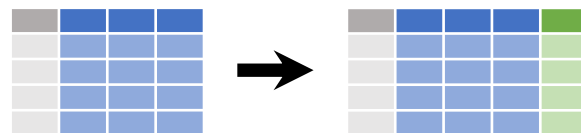
`df.dropna()`

حذف ردیف (سطر) هایی با هر ستون دارای مقادیر NA/null

`df.fillna(value)`

جایگزینی تمامی مقادیر NA/null با مقدار

ساخت ستون های جدید



`df.assign(Area=lambda df: df.Length*df.Height)`

محاسبه و اضافه کردن یک یا چند ستون جدید

`df['Volume'] = df.Length*df.Height*df.Depth`

اضافه کردن یک ستون

`pd.qcut(df.col, n, labels=False)`

قرار دادن ستون در n سطل



پانداس مجموعه بزرگی از توابع برداری را ارائه می دهد که بر روی تمام ستون های یک

DataFrame یا یک ستون منفرد انتخاب شده (یک سری پانداس) عمل می کنند. این

توابع بردارهایی از مقادیر را برای هر یک از ستون ها یا یک سری واحد برای هر سری

جداگانه تولید می کنند. مثال ها:

`max(axis=1)`

حداکثر عنصر

`min(axis=1)`

حداقل عنصر

`clip(lower=-10,upper=10)`

برش مقادیر در آستانه ورودی

`abs()`

مقدار مطلق

مثال های زیر را می توان برای گروه ها نیز اعمال کرد. در این مورد، تابع بر اساس هر گروه اعمال می شود و بردارهای برگشتی به طول DataFrame اصلی هستند.

`shift(1)`

کپی با مقادیر با تقدم 1

`shift(-1)`

کپی با مقادیر با تاخیر 1

`rank(method='dense')`

رتبه بندی بدون فاصله بین رتبه ها.

`cumsum()`

جمع تجمعی

`rank(method='min')`

رتبه بندی مقادیر مساوی کمترین رتبه را دریافت می کنند.

`cummax()`

حداکثر تجمعی

`rank(pct=True)`

رتبه بندی با مقیاس بندی مجدد به بازه [0, 1].

`cummin()`

حداقل تجمعی

`rank(method='first')`

رتبه بندی مقادیر مساوی رتبه اولین مقدار را دریافت می کنند.

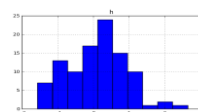
`cumprod()`

محصول تجمعی

Plotting

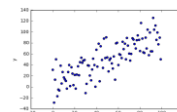
`df.plot.hist()`

هیستوگرام برای هر ستون



`df.plot.scatter(x='w',y='h')`

نمودار پراکندگی با استفاده از جفت نقطه



ترکیب مجموعه داده ها

adf

x1	x2
A	1
B	2
C	3

bdf

x1	x3
A	T
B	F
D	T



Standard Joins

x1	x2	x3
A	1	T
B	2	F
C	3	NaN

`pd.merge(adf, bdf, how='left', on='x1')`

ادغام ردیف (سطر) های مشابه از DataFrame دوم (bdf) به DataFrame اول (adf)

x1	x2	x3
A	1.0	T
B	2.0	F
D	NaN	T

`pd.merge(adf, bdf, how='right', on='x1')`

ادغام ردیف (سطر) های مشابه از DataFrame اول (adf) به DataFrame دوم (bdf)

x1	x2	x3
A	1	T
B	2	F

`pd.merge(adf, bdf, how='inner', on='x1')`

ادغام داده ها. فقط ردیف (سطر) های مشترک در هر دو مجموعه حفظ شوند.

x1	x2	x3
A	1	T
B	2	F
C	3	NaN
D	NaN	T

`pd.merge(adf, bdf, how='outer', on='x1')`

ادغام داده ها. تمام مقادیر و تمام ردیف (سطر) ها حفظ شوند.

Filtering Joins

x1	x2
A	1
B	2

`adf[adf.x1.isin(bdf.x1)]`

تمام ردیف (سطر) های موجود در adf که در bdf تطابق دارند.

x1	x2
C	3

`adf[~adf.x1.isin(bdf.x1)]`

تمام ردیف (سطر) های موجود در adf که در bdf تطابق ندارند.

ydf

x1	x2
A	1
B	2
C	3

zdf

x1	x2
B	2
C	3
D	4



Set-like Operations

x1	x2
B	2
C	3

`pd.merge(ydf, zdf)`

ردیف (سطر) هایی که هم در ydf و هم در zdf وجود دارند (اشتراک).

x1	x2
A	1
B	2
C	3
D	4

`pd.merge(ydf, zdf, how='outer')`

ردیف (سطر) هایی که در ydf یا zdf یا هر دو وجود دارند (اجتماع).

x1	x2
A	1

`pd.merge(ydf, zdf, how='outer', indicator=True)`

`.query('_merge == "left_only"')`

`.drop(columns=['_merge'])`

ردیف (سطر) هایی که در ydf ظاهر می شوند اما zdf نیستند (Setdiff).