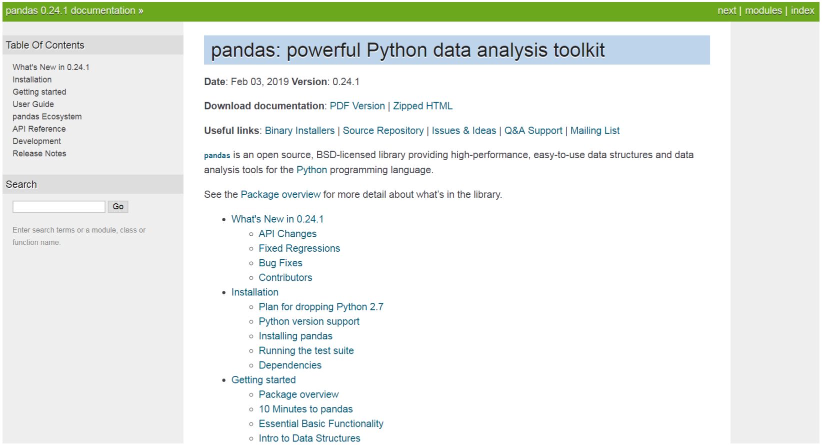
Pandas

pandas: a library for data manipulation

- Documentation: http://pandas.pydata.org/pandas-docs/stable/



pandas 0.24.1 documentation » next | modules | index

Table Of Contents

Search

Go

Enter search terms or a module, class or function name.

pandas: powerful Python data analysis toolkit

Date: Feb 03, 2019 **Version**: 0.24.1

Download documentation: PDF Version | Zipped HTML

Useful links: Binary Installers | Source Repository | Issues & Ideas | Q&A Support | Mailing List

`pandas` is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

See the Package overview for more detail about what's in the library.

Reading in any textfile

```python
import pandas as pd
FNAME = "http://www.stat.ucla.edu/projects/datasets/twins.dat"

df = pd.read_csv(FNAME, sep=',')
```

Inspecting the DataFrame

```python
df.shape  # Dimensions
df.head()  # First 5 lines
df.tail()  # Last 5 lines
df.columns  # List of variables as list
df.describe()  # Summary statistics
```

Inspecting the DataFrame

```python
df.shape  # Dimensions
df.head()  # First 5 lines
df.tail()  # Last 5 lines
df.columns  # List of variables as list
df.describe()  # Summary statistics
```

1. How many observations do you have?
2. How many variables do you have?
3. Which variables are numeric?
4. What is the mean of variable `"DEDUC1"`?

Understanding dtypes

```
df.info()
```

Understanding dtypes

```
df.info()
```

Pandas	Python	Purpose
object	unicode	Text
int64	int	Integers
float64	float	Floating numbers
bool	bool	True and False values
datetime64		Date and time values
timedelta[ns]		Differences between two datetimes
category		Finite list of text values

Slicing the DataFrame

```python
df["DEDUC"]  # Column by column name
df.loc[0]  # Row by index name
df.iloc[0]  # Row by row number
df[["AGE", "HRWAGEH"]]  # Columns by list of names
df.loc["AGE":"HRWAGEH"]  # Column range by column names (will be empty)
df.iloc[:,5:7]  # Column range by column indices
df.iloc[1, 4]  # Value by row and column index
df.loc[18, "WHITEH"]  # Value by row and column name
df["WHITEH"].iloc[18]  # Same as above
```

Slicing the DataFrame

```python
df["DEDUC"]  # Column by column name
df.loc[0]  # Row by index name
df.iloc[0]  # Row by row number
df[["AGE", "HRWAGEH"]]  # Columns by list of names
df.loc["AGE":"HRWAGEH"]  # Column range by column names (will be empty)
df.iloc[:,5:7]  # Column range by column indices
df.iloc[1, 4]  # Value by row and column index
df.loc[18, "WHITEH"]  # Value by row and column name
df["WHITEH"].iloc[18]  # Same as above
```

1. What is the 6th entry of the 5th column?
2. What is the 5th entry of column `"DTEN"`?
3. What is the last entry of column `"HRWAGEL"`?

Boolean indexing

```
df[df["AGE"] > 20]
df[(df["AGE"] > 20) & (df["WHITEL"] == 1)]
df[~df["AGE"] > 20]
df[df["AGE"].isin((20, 21, 22, 23))]
```

Boolean indexing

```python
df[df["AGE"] > 20]
df[(df["AGE"] > 20) & (df["WHITEL"] == 1)]
df[~df["AGE"] > 20]
df[df["AGE"].isin((20, 21, 22, 23))]
```

1. How many observations have `"WHITEL"` equal to 0?
2. How many observations have `"WHITEH"` equal to 1 and `"DEDUC1` unequal to 0?
3. What is the mean age of married twins whose L-parent is a non-white male? (Use `"DMARRIED"`, `"WHITEL"` and `"MALEL"`)
4. In how many rows do the values for `"WHITEH"` and `"WHITEL"` differ?

Aggregate data

```
df["WHITEL"].value_counts()
pd.crosstab(df["WHITEH"], df["WHITEL"])
df[["DEDUC2", "EDUCL"]].corr()
```

Aggregate data

```
df["WHITEL"].value_counts()
pd.crosstab(df["WHITEH"], df["WHITEL"])
df[["DEDUC2", "EDUCL"]].corr()
```

1. What is the most common value in `"EDUCL"`?
2. What is the most common combination of `"MALEH"` and `"MALEL"`?
3. What is the Spearman correlation between `"EDUCH"` and `"EDUCL"`? What is the Pearson correlation? (Check documentation!)

Changing dtypes

```python
df["DMARRIED"] = df["DMARRIED"].astype(bool)
df["WHITEH"] = df["WHITEH"].astype("category")
df["HRWAGEH"] = pd.to_numeric(df["HRWAGEH"], errors="coerce")
```

Manipulation

```python
df = df.sort_values(by='HRWAGEH')  # Sort by column
df = df[sorted(df.columns)]  # Re-order columns alphabetically
df['new'] = 9  # Add new column
df['AGETR'] = df['AGE']**3
df['combined'] = df['MALEH'] + df['EDUCH']
df["HRWAGEH_new"] = df["HRWAGEH"].fillna(0)  # Fill missings with 0
df = df.dropna(subset=["HRWAGEH"])  # Drop rows missing in "HRWAGEH"
```

Grouping

```python
grouped = df.groupby(['MALEH'])
print(grouped['AGE'].mean())
print(grouped['EDUCH'].agg(['mean', 'sum']))
print(grouped[['EDUCH', 'AGE']].agg(['mean', 'std']))
```

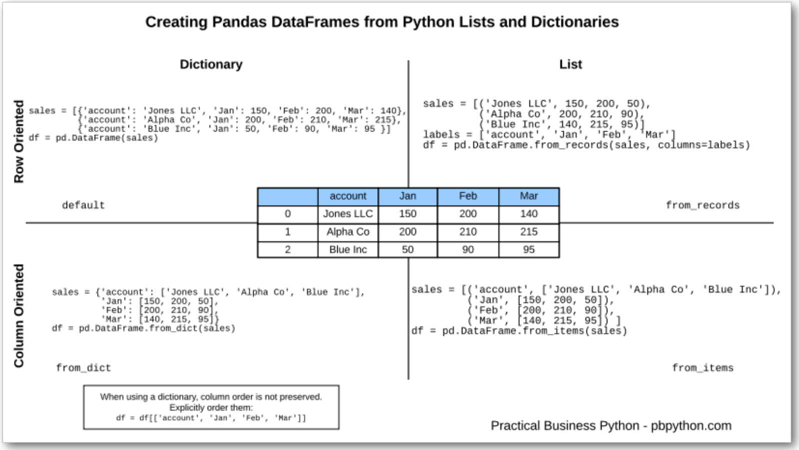
Grouping

```python
grouped = df.groupby(['MALEH'])
print(grouped['AGE'].mean())
print(grouped['EDUCH'].agg(['mean', 'sum']))
print(grouped[['EDUCH', 'AGE']].agg(['mean', 'std']))
```

→ Full list at `http://pandas.pydata.org/pandas-docs/stable/getting_started/basics.html#descriptive-statistics`

- What is the `"AGE"` variance for `"MALEL" == 0` individuals?
- What are the second and the third quartile of `"EUDCL"` for `"MALEL" == 0` individuals?
- What is the average age for individuals with `"MALEL" == 0` and `"MALEH" = 0`?

Creating DataFrames from other objects



Creating Pandas DataFrames from Python Lists and Dictionaries

Dictionary

List

Row Oriented

```
sales = [{'account': 'Jones LLC', 'Jan': 150, 'Feb': 200, 'Mar': 140},
         {'account': 'Alpha Co',  'Jan': 200, 'Feb': 210, 'Mar': 215},
         {'account': 'Blue Inc',  'Jan': 50,  'Feb': 90,  'Mar': 95 }]
df = pd.DataFrame(sales)
```

`default`

```
sales = [('Jones LLC', 150, 200, 50),
         ('Alpha Co', 200, 210, 90),
         ('Blue Inc', 140, 215, 95)]
labels = ['account', 'Jan', 'Feb', 'Mar']
df = pd.DataFrame.from_records(sales, columns=labels)
```

`from_records`

	account	Jan	Feb	Mar
0	Jones LLC	150	200	140
1	Alpha Co	200	210	215
2	Blue Inc	50	90	95

Column Oriented

```
sales = {'account': ['Jones LLC', 'Alpha Co', 'Blue Inc'],
         'Jan': [150, 200, 50],
         'Feb': [200, 210, 90],
         'Mar': [140, 215, 95]}
df = pd.DataFrame.from_dict(sales)
```

`from_dict`

```
sales = [('account', ['Jones LLC', 'Alpha Co', 'Blue Inc']),
         ('Jan', [150, 200, 50]),
         ('Feb', [200, 210, 90]),
         ('Mar', [140, 215, 95]) ]
df = pd.DataFrame.from_items(sales)
```

`from_items`

When using a dictionary, column order is not preserved.
Explicitly order them:
`df = df[['account', 'Jan', 'Feb', 'Mar']]`

Practical Business Python - pbpython.com

Creating DataFrames from other objects, cont.

```python
d = {'employee': ['Hannes', 'Fabiana', 'George', 'Olga'],
     'group': ['Accounting', 'Engineering', 'Engineering', 'HR']}
df1 = pd.DataFrame.from_dict(d)
t = [('Hannes', 2004), ('Fabiana', 2008), ('George', 2012), ('Olga', 2014)]
df2 = pd.DataFrame.from_records(t, columns=["employee", "hire_date"])
```

Appending, Concatening and Merging

```python
df3 = df1.append(df2)
df4 = pd.concat([df1, df2])

df5 = pd.concat([df1, df2], axis=1)
df6 = df1.merge(df2, left_on="employee", right_on="employee")
```

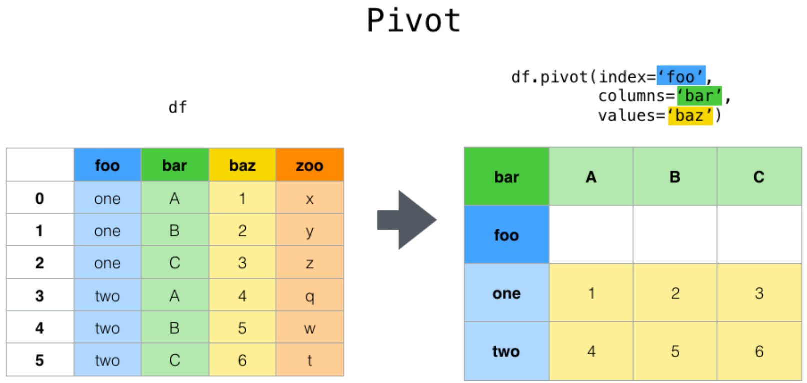
Appending, Concatening and Merging

```python
df3 = df1.append(df2)
df4 = pd.concat([df1, df2])

df5 = pd.concat([df1, df2], axis=1)
df6 = df1.merge(df2, left_on="employee", right_on="employee")
```

- How do objects df3 and df4 differ?
- How do objects df5 and df6 differ?

What is pivot?



Pivot

df

	foo	bar	baz	zoo
0	one	A	1	x
1	one	B	2	y
2	one	C	3	z
3	two	A	4	q
4	two	B	5	w
5	two	C	6	t

```
df.pivot(index='foo',
         columns='bar',
         values='baz')
```

bar	**A**	**B**	**C**
foo			
one	1	2	3
two	4	5	6

from: “Reshaping and Pivot Tables”

Pivoting and melting

```python
pivoted = df6.pivot(index='employee', columns='group', values='hire_date')
reverse = (pivoted.reset_index()
                  .melt(id_vars="employee", value_name="hire_date")
```

Pivoting and melting

```
pivoted = df6.pivot(index='employee', columns='group', values='hire_date')
reverse = (pivoted.reset_index()
                  .melt(id_vars="employee", value_name="hire_date")
```

- How do you make `reverse` look like `df6` again?

Output

- https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html

Output

- https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html

```
df.to_csv(FNAME, sep=";")
df.to_html(FNAME, decimal=",", justify="center")
df.to_stata(FNAME, write_index=False)
```