Pandas/ Visualization

CAC 350

Catch Up/Today

* Please submit Assignment #2 if you haven't already

* Explore pandas further by analyzing a dataset - please grab it from Moodle

Warning - I may ask you to share your screen

Open Editor

GroupBy

* Used to aggregate conditionally on a label or index
* GroupBy comes from SQL
* Split, apply, combine
  * Slit: break up and group a DataFrame depending on the value of the specified key
  * Apply: Compute a function (usually an aggregate), transformation, filter
  * Combine: merge results into an output array

Aggregation	Description
count()	Total num
first(), last()	first, last item
mean(), median()	mean/ median
min(), max()	min/max
std(), var()	std/var
prod()	product
sum()	sum
mad()	mean absolute deviation

how spread out the data is

GroupBy

```
In [19]: data.groupby('Team')

Out[19]: <pandas.core.groupby.generic.DataFrameGroupBy object at 0x11637b220>
```

* Returns a groupby object, which we would need to apply some aggregate to in order to produce a result

* True of SQL GroupBy as well - it aggregates the rows in some way

jupyter

Merging and Joining Datasets

* Let's say we need to combine two dataframes...we have options!

In [7]:
```
df1 = make_df('AB', [1, 2])
df2 = make_df('AB', [3, 4])
display('df1', 'df2', 'pd.concat([df1, df2])')
```

Out[7]:

df1

	A	B
1	A1	B1
2	A2	B2

df2

	A	B
3	A3	B3
4	A4	B4

pd.concat([df1, df2])

	A	B
1	A1	B1
2	A2	B2
3	A3	B3
4	A4	B4

Merging and Joining Datasets

* Can also concatenate on columns

In [8]:
```
df3 = make_df('AB', [0, 1])
df4 = make_df('CD', [0, 1])
display('df3', 'df4', "pd.concat([df3, df4], axis='col')")
```

Out[8]:

df3

	A	B
0	A0	B0
1	A1	B1

df4

	C	D
0	C0	D0
1	C1	D1

pd.concat([df3, df4], axis='col')

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1

Merging and Joining Datasets

* Can also use append() or extend() but know that it creates a new object and does not change the original

In [16]:
```
display('df1', 'df2', 'df1.append(df2)')
```

Out[16]:

df1

	A	B
1	A1	B1
2	A2	B2

df2

	A	B
3	A3	B3
4	A4	B4

df1.append(df2)

	A	B
1	A1	B1
2	A2	B2
3	A3	B3
4	A4	B4

Joins

* Types of joins:
  * one-to-one
  * many-to-one
  * many-to-many

One-to-One

* Similar to column-wise concatenation

In [2]:
```
df1 = pd.DataFrame({'employee': ['Bob', 'Jake', 'Lisa', 'Sue'],
                    'group': ['Accounting', 'Engineering', 'Engineerin
df2 = pd.DataFrame({'employee': ['Lisa', 'Bob', 'Jake', 'Sue'],
                    'hire_date': [2004, 2008, 2012, 2014]})
display('df1', 'df2')
```

Out[2]:

df1

	employee	group
0	Bob	Accounting
1	Jake	Engineering
2	Lisa	Engineering
3	Sue	HR

df2

	employee	hire_date
0	Lisa	2004
1	Bob	2008
2	Jake	2012
3	Sue	2014

In [3]:
```
df3 = pd.merge(df1, df2)
df3
```

Out[3]:

	employee	group	hire_date
0	Bob	Accounting	2008
1	Jake	Engineering	2012
2	Lisa	Engineering	2004
3	Sue	HR	2014

Many-to-One

* One of the two key columns contains duplicate entries

In [4]:
```
df4 = pd.DataFrame({'group': ['Accounting', 'Engineering', 'HR'],
                    'supervisor': ['Carly', 'Guido', 'Steve']})
display('df3', 'df4', 'pd.merge(df3, df4)')
```

Out[4]:

df3

	employee	group	hire_date
0	Bob	Accounting	2008
1	Jake	Engineering	2012
2	Lisa	Engineering	2004
3	Sue	HR	2014

df4

	group	supervisor
0	Accounting	Carly
1	Engineering	Guido
2	HR	Steve

pd.merge(df3, df4)

	employee	group	hire_date	supervisor
0	Bob	Accounting	2008	Carly
1	Jake	Engineering	2012	Guido
2	Lisa	Engineering	2004	Guido
3	Sue	HR	2014	Steve

Many-to-Many

* The key column in both the left and right array contains duplicates

In [5]:
```
df5 = pd.DataFrame({'group': ['Accounting', 'Accounting',
                              'Engineering', 'Engineering', 'HR', 'HR'
                    'skills': ['math', 'spreadsheets', 'coding', 'linu
                               'spreadsheets', 'organization']})
display('df1', 'df5', "pd.merge(df1, df5)")
```

Out[5]:

df1

	employee	group
0	Bob	Accounting
1	Jake	Engineering
2	Lisa	Engineering
3	Sue	HR

df5

	group	skills
0	Accounting	math
1	Accounting	spreadsheets
2	Engineering	coding
3	Engineering	linux
4	HR	spreadsheets
5	HR	organization

pd.merge(df1, df5)

	employee	group	skills
0	Bob	Accounting	math
1	Bob	Accounting	spreadsheets
2	Jake	Engineering	coding
3	Jake	Engineering	linux
4	Lisa	Engineering	coding
5	Lisa	Engineering	linux
6	Sue	HR	spreadsheets
7	Sue	HR	organization

jupyter

Pivot Tables

* Multidimensional version of GroupBy

```
In [3]: titanic.groupby('sex')[['survived']].mean()
```

Out[3]:

	survived
sex	
female	0.742038
male	0.188908

```
In [4]: titanic.groupby(['sex', 'class'])['survived'].aggregate('mean').unstac
```

Out[4]:

class	First	Second	Third
sex			
female	0.968085	0.921053	0.500000
male	0.368852	0.157407	0.135447

```
In [5]: titanic.pivot_table('survived', index='sex', columns='class')
```

Out[5]:

class	First	Second	Third
sex			
female	0.968085	0.921053	0.500000
male	0.368852	0.157407	0.135447

```
In [6]: age = pd.cut(titanic['age'], [0, 18, 80])
        titanic.pivot_table('survived', ['sex', age], 'class')
```

Out[6]:

	class	First	Second	Third
sex	**age**			
female	**(0, 18]**	0.909091	1.000000	0.511628
	(18, 80]	0.972973	0.900000	0.423729
male	**(0, 18]**	0.800000	0.600000	0.215686
	(18, 80]	0.375000	0.071429	0.133663

jupyter

Next Class

* Continue skimming through Chapter Four: Visualizations will be focus
* Reading quiz
* Assignment #3