Python数据分析实战

第二十课 pandas分组和合并

本节课程目标

- pandas如何分组和合并
- pandas如何转换数据
- pandas如何过滤数据

pandas如何分组和合并

```
import pandas as pd
import numpy as np
%matplotlib inline
#分别为姓名、年份、工资、奖金（）
salaries = pd.DataFrame({
    'Name': ['laowang', 'laosong', 'laosong', 'rongmei', 'laowang', 'laowang', 'laosong', 'laowang'],
    'Year': [2016,2016,2016,2016,2017,2017,2017,2017],
    'Salary': [10000,2000,4000,5000,18000,25000,3000,4000],
    'Bonus': [3000,1000,1000,1200,4000,2300,500,1000],
    'jjj':['1','2','3','4','5','6','7','8'],
})
print(salaries)
```

```
      Name  Year  Salary  Bonus jjj
0  laowang  2016   10000   3000   1
1  laosong  2016    2000   1000   2
2  laosong  2016    4000   1000   3
3  rongmei  2016    5000   1200   4
4  laowang  2017   18000   4000   5
5  laowang  2017   25000   2300   6
6  laosong  2017    3000    500   7
7  laowang  2017    4000   1000   8
```

```
group_by_name = salaries.groupby('Name')
group_by_name
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x10b3af780>
```

```
group_by_name.sum()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Year	Salary	Bonus
Name			
laosong	6049	9000	2500
laowang	8067	57000	10300
rongmei	2016	5000	1200

```
salaries.groupby('Name',sort=False).sum()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Year	Salary	Bonus
Name			
laowang	8067	57000	10300
laosong	6049	9000	2500
rongmei	2016	5000	1200

```
group_by_name.aggregate('sum')
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Year	Salary	Bonus
Name			
laosong	6049	9000	2500
laowang	8067	57000	10300
rongmei	2016	5000	1200

```
group_by_name.aggregate(np.mean)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Year	Salary	Bonus
Name			
laosong	2016.333333	3000.0	833.333333
laowang	2016.750000	14250.0	2575.000000
rongmei	2016.000000	5000.0	1200.000000

```
group_by_name.aggregate(np.std)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Year	Salary	Bonus
Name			
laosong	0.57735	1000.000000	288.675135
laowang	0.50000	9178.779875	1260.621540
rongmei	NaN	NaN	NaN

```
group_by_name.groups
```

```
{'laosong': Int64Index([1, 2, 6], dtype='int64'),
 'laowang': Int64Index([0, 4, 5, 7], dtype='int64'),
 'rongmei': Int64Index([3], dtype='int64')}
```

```
len(group_by_name)
```

```
3
```

```
dir(group_by_name)
```

```
['Bonus',
 'Name',
 'Salary',
 'Year',
 '__bytes__',
 '__class__',
 '__delattr__',
 '__dict__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattr__',
 '__getattribute__',
 '__getitem__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__module__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
```

```
'__unicode__',
'__weakref__',
'_accessors',
'_add_numeric_operations',
'_agg_examples_doc',
'_agg_see_also_doc',
'_aggregate',
'_aggregate_generic',
'_aggregate_item_by_item',
'_aggregate_multiple_funcs',
'_apply_filter',
'_apply_to_column_groupbys',
'_apply_whitelist',
'_assure_grouper',
'_block_agg_axis',
'_bool_agg',
'_builtin_table',
'_choose_path',
'_concat_objects',
'_constructor',
'_cumcount_array',
'_cython_agg_blocks',
'_cython_agg_general',
'_cython_table',
'_cython_transform',
'_decide_output_index',
'_def_str',
'_define_paths',
'_deprecations',
'_dir_additions',
'_dir_deletions',
'_fill',
'_get_cythonized_result',
'_get_data_to_aggregate',
'_get_index',
'_get_indices',
'_gotitem',
'_group_selection',
'_insert_inaxis_grouper_inplace',
'_internal_names',
'_internal_names_set',
'_is_builtin_func',
'_is_cython_func',
'_iterate_column_groupbys',
'_iterate_slices',
'_make_wrapper',
'_obj_with_exclusions',
'_post_process_cython_aggregate',
'_python_agg_general',
'_python_apply_general',
'_reindex_output',
'_reset_cache',
'_reset_group_selection',
'_selected_obj',
'_selection',
'_selection_list',
'_selection_name',
'_set_group_selection',
'_set_result_index_ordered',
'_shallow_copy',
'_transform_fast',
'_transform_general',
'_transform_item_by_item',
'_transform_should_cast',
'_try_aggregate_string_function',
'_try_cast',
'_wrap_agged_blocks',
'_wrap_aggregated_output',
'_wrap_applied_output',
'_wrap_generic_output',
'_wrap_transformed_output',
'agg',
'aggregate',
'all',
'any',
'apply',
'backfill',
'bfill',
'boxplot',
```

```
 'corr',
 'corrwith',
 'count',
 'cov',
 'cumcount',
 'cummax',
 'cummin',
 'cumprod',
 'cumsum',
 'describe',
 'diff',
 'dtypes',
 'expanding',
 'ffill',
 'fillna',
 'filter',
 'first',
 'get_group',
 'groups',
 'head',
 'hist',
 'idxmax',
 'idxmin',
 'indices',
 'jjj',
 'last',
 'mad',
 'max',
 'mean',
 'median',
 'min',
 'ndim',
 'ngroup',
 'ngroups',
 'nth',
 'nunique',
 'ohlc',
 'pad',
 'pct_change',
 'pipe',
 'plot',
 'prod',
 'quantile',
 'rank',
 'resample',
 'rolling',
 'sem',
 'shift',
 'size',
 'skew',
 'std',
 'sum',
 'tail',
 'take',
 'transform',
 'tshift',
 'var']
```

```
group_by_name_year = salaries.groupby(['Name','Year'])
group_by_name_year.sum()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

		Salary	Bonus
Name	**Year**		
laosong	**2016**	6000	2000
	2017	3000	500
laowang	**2016**	10000	3000
	2017	47000	7300
rongmei	**2016**	5000	1200

```
group_by_name.describe()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead tr th {
    text-align: left;
}

.dataframe thead tr:last-of-type th {
    text-align: right;
}
```

	Year								Salary					Bonus	
	count	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**	**count**	**mean**	**...**	**75%**	**max**	**count**	**mean**
Name															
laosong	3.0	2016.333333	0.57735	2016.0	2016.00	2016.0	2016.5	2017.0	3.0	3000.0	...	3500.0	4000.0	3.0	833.33
laowang	4.0	2016.750000	0.50000	2016.0	2016.75	2017.0	2017.0	2017.0	4.0	14250.0	...	19750.0	25000.0	4.0	2575.0
rongmei	1.0	2016.000000	NaN	2016.0	2016.00	2016.0	2016.0	2016.0	1.0	5000.0	...	5000.0	5000.0	1.0	1200.0

3 rows × 24 columns

```
group_by_name
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x10b3af780>
```

```
for name,group in group_by_name:
    print(name)
    print(group)
```

```
laosong
      Name  Year  Salary  Bonus jjj
1  laosong  2016    2000   1000   2
2  laosong  2016    4000   1000   3
6  laosong  2017    3000    500   7
laowang
      Name  Year  Salary  Bonus jjj
0  laowang  2016   10000   3000   1
4  laowang  2017   18000   4000   5
5  laowang  2017   25000   2300   6
7  laowang  2017    4000   1000   8
rongmei
      Name  Year  Salary  Bonus jjj
3  rongmei  2016    5000   1200   4
```

```
type(group_by_name.get_group('laowang'))
```

```
pandas.core.frame.DataFrame
```

```
#agg
```

```
group_by_name.agg(sum)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Year	Salary	Bonus
Name			
laosong	6049	9000	2500
laowang	8067	57000	10300
rongmei	2016	5000	1200

```

```

pandas如何转换数据

```
nvda = pd.read_csv("./datas/NVDA.csv", index_col=0, parse_dates=[0])
nvda.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Open	High	Low	Close	Adj Close	Volume
Date						
1999-01-22	1.750000	1.953125	1.552083	1.640625	1.523430	67867200
1999-01-25	1.770833	1.833333	1.640625	1.812500	1.683028	12762000
1999-01-26	1.833333	1.869792	1.645833	1.671875	1.552448	8580000
1999-01-27	1.677083	1.718750	1.583333	1.666667	1.547611	6109200
1999-01-28	1.666667	1.677083	1.651042	1.661458	1.542776	5688000

```
nvda.index.year
```

```
Int64Index([1999, 1999, 1999, 1999, 1999, 1999, 1999, 1999, 1999, 1999,
            ...
            2017, 2017, 2017, 2017, 2017, 2017, 2017, 2017, 2017, 2017],
           dtype='int64', name='Date', length=4654)
```

```
nvda.groupby(nvda.index.year).agg(np.mean)
```

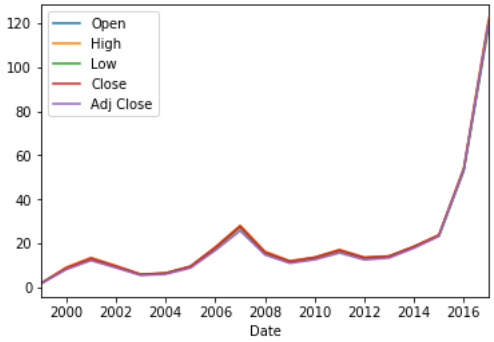
```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Open	High	Low	Close	Adj Close	Volume
Date						
1999	1.950782	2.007317	1.883559	1.947230	1.808134	6.433220e+06
2000	8.781084	9.222697	8.360522	8.778826	8.151729	1.104182e+07
2001	13.091254	13.600750	12.680548	13.181552	12.239956	2.782387e+07
2002	9.690344	9.955093	9.344391	9.614749	8.927940	3.168655e+07
2003	5.902434	6.042659	5.764960	5.900344	5.478865	2.430220e+07
2004	6.484735	6.608810	6.353558	6.465913	6.004034	1.706331e+07
2005	9.512381	9.659656	9.353175	9.513823	8.834223	1.542825e+07
2006	18.057902	18.425126	17.720279	18.095963	16.803316	1.534446e+07
2007	27.762045	28.251673	27.206056	27.724542	25.744098	1.514562e+07
2008	16.004308	16.426245	15.521462	15.945613	14.806572	2.022721e+07
2009	11.825119	12.114762	11.565952	11.850873	11.004331	1.919821e+07
2010	13.576349	13.802659	13.318532	13.563175	12.594318	1.853295e+07
2011	16.912540	17.267540	16.512143	16.887540	15.681214	2.289352e+07
2012	13.526200	13.717400	13.319800	13.507880	12.551166	1.207757e+07
2013	14.173571	14.329802	14.035278	14.189127	13.412278	8.843986e+06
2014	18.543056	18.745476	18.348214	18.547064	17.875053	7.098902e+06
2015	23.680595	23.979524	23.411071	23.718254	23.262283	7.756520e+06
2016	53.630833	54.415397	52.895119	53.761190	53.475737	1.107062e+07
2017	120.481305	122.300725	118.402754	120.547971	120.436863	1.907742e+07

```
nvda.groupby(nvda.index.year).mean().iloc[:,:-1].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x11c132978>
```




```
#apply
```

```
zscore = lambda x : (x-x.mean())/x.std()
zscore
```

```
<function __main__.<lambda>(x)>
```

```
transformed = nvda.groupby(nvda.index.year).apply(zscore)
transformed
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Open	High	Low	Close	Adj Close	Volume
Date						
1999-01-22	-0.340955	-0.088217	-0.579850	-0.510124	-0.510124	7.544438
1999-01-25	-0.305578	-0.283222	-0.424964	-0.224161	-0.224161	0.777210
1999-01-26	-0.199444	-0.223871	-0.415854	-0.458130	-0.458131	0.263637
1999-01-27	-0.464778	-0.469747	-0.525185	-0.466795	-0.466798	-0.039791
1999-01-28	-0.482465	-0.537575	-0.406741	-0.475462	-0.475461	-0.091517
1999-01-29	-0.491311	-0.554531	-0.525185	-0.605445	-0.605445	-0.040823
1999-02-01	-0.623978	-0.622359	-0.525185	-0.553452	-0.553451	-0.315073
1999-02-02	-0.623978	-0.622359	-0.771180	-0.761424	-0.761424	0.020776
1999-02-03	-0.818555	-0.758014	-0.743847	-0.709431	-0.709430	-0.559407
1999-02-04	-0.694732	-0.588446	-0.634516	-0.570782	-0.570781	-0.231516
1999-02-05	-0.544378	-0.554531	-0.516073	-0.492792	-0.492793	-0.369893
1999-02-08	-0.491311	-0.554531	-0.506962	-0.588113	-0.588113	-0.316988
1999-02-09	-0.553221	-0.605402	-0.652737	-0.692099	-0.692100	-0.523007
1999-02-10	-0.712421	-0.707143	-0.689181	-0.718096	-0.718095	-0.334967
1999-02-11	-0.730111	-0.486705	-0.634516	-0.501459	-0.501458	-0.384040
1999-02-12	-0.482465	-0.418876	-0.379409	-0.345479	-0.345479	-0.453155
1999-02-16	-0.305578	-0.266264	-0.543405	-0.328148	-0.328149	-0.142211
1999-02-17	-0.411711	-0.452790	-0.452297	-0.484127	-0.484128	-0.582101
1999-02-18	-0.411711	-0.452790	-0.434074	-0.440799	-0.440799	-0.572964
1999-02-19	-0.482465	-0.384963	-0.415854	-0.345479	-0.345479	-0.558670
1999-02-22	-0.305578	-0.351048	-0.397631	-0.328148	-0.328149	-0.159895
1999-02-23	-0.270199	-0.223871	-0.342966	-0.189500	-0.189499	-0.366061
1999-02-24	0.260468	0.293314	0.085249	0.053136	0.053135	1.091248
1999-02-25	0.189712	0.191572	0.003250	-0.050850	-0.050850	-0.332167
1999-02-26	-0.022555	-0.011911	-0.124303	-0.198165	-0.198164	-0.260105
1999-03-01	-0.128688	-0.147565	-0.233634	-0.180833	-0.180834	-0.507091
1999-03-02	-0.199444	-0.266264	-0.160746	-0.206830	-0.206829	-0.620416
1999-03-03	-0.199444	-0.283222	-0.342966	-0.414802	-0.414802	-0.601554
1999-03-04	-0.287888	-0.351048	-0.415854	-0.475462	-0.475461	-0.613932
1999-03-05	-0.464778	-0.401919	-0.361188	-0.319483	-0.319482	-0.548207

...	...	...	...	...	...	...
2017-06-08	1.497139	1.674761	1.568812	1.791207	1.791015	0.928518
2017-06-09	2.009218	2.052367	1.144037	1.321034	1.322214	6.824084
2017-06-12	1.153028	1.306040	1.113965	1.337858	1.338989	2.176461
2017-06-13	1.539811	1.442422	1.280303	1.402881	1.403823	2.118167
2017-06-14	1.409068	1.410881	1.414220	1.417432	1.418332	0.981847
2017-06-15	1.202057	1.390446	1.320244	1.446989	1.447802	0.467528
2017-06-16	1.465360	1.439312	1.495980	1.412885	1.413798	0.377007
2017-06-19	1.494869	1.565033	1.637885	1.672072	1.672228	0.035122
2017-06-20	1.750000	1.752059	1.809862	1.661613	1.661800	0.774094
2017-06-21	1.712775	1.657879	1.752537	1.769835	1.769706	-0.187370
2017-06-22	1.784956	1.689865	1.832417	1.719816	1.719833	-0.684695
2017-06-23	1.734111	1.644553	1.636006	1.513377	1.513996	0.758125
2017-06-26	1.574314	1.523719	1.406233	1.436985	1.437827	0.700763
2017-06-27	1.405436	1.310038	1.313196	1.183710	1.185292	0.550605
2017-06-28	1.309195	1.316702	1.285002	1.418797	1.419692	0.540022
2017-06-29	1.367303	1.262504	1.206532	1.188257	1.189825	0.701843
2017-06-30	1.221124	1.138560	1.179279	1.091858	1.093708	-0.081430
2017-07-03	1.115349	1.037273	0.948096	0.854043	0.856587	-0.125833
2017-07-05	0.972347	0.973747	1.067916	1.023196	1.025247	0.132975
2017-07-06	0.970985	1.025279	1.003542	1.042749	1.044742	-0.039160
2017-07-07	1.148488	1.119458	1.242713	1.191895	1.193452	-0.251841
2017-07-10	1.328261	1.408216	1.422678	1.507465	1.508102	0.455109
2017-07-11	1.514843	1.505505	1.585728	1.606593	1.606940	-0.011974
2017-07-12	1.716861	1.808033	1.792947	1.908067	1.907534	0.890003
2017-07-13	1.930227	1.954634	1.895851	1.822582	1.822298	1.411561
2017-07-14	1.852598	1.897326	2.002045	2.019017	2.018161	0.416575
2017-07-17	2.081400	2.007942	2.036816	1.987188	1.986424	0.390591
2017-07-18	1.874842	1.965740	2.015672	2.064944	2.063953	0.031544
2017-07-19	2.081400	2.003500	2.171203	2.025839	2.024962	-0.177140
2017-07-20	2.063240	2.008386	2.138311	2.134969	2.133774	-0.159699

4654 rows × 6 columns

```
nvda.groupby(nvda.index.year).transform('max').head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Open	High	Low	Close	Adj Close	Volume
Date						
1999-01-22	3.947917	3.953125	3.84375	3.911458	3.632052	67867200
1999-01-25	3.947917	3.953125	3.84375	3.911458	3.632052	67867200
1999-01-26	3.947917	3.953125	3.84375	3.911458	3.632052	67867200
1999-01-27	3.947917	3.953125	3.84375	3.911458	3.632052	67867200
1999-01-28	3.947917	3.953125	3.84375	3.911458	3.632052	67867200

```
nvda.groupby(nvda.index.year).transform('min').head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Open	High	Low	Close	Adj Close	Volume
Date						
1999-01-22	1.395833	1.421875	1.333333	1.364583	1.267107	492000
1999-01-25	1.395833	1.421875	1.333333	1.364583	1.267107	492000
1999-01-26	1.395833	1.421875	1.333333	1.364583	1.267107	492000
1999-01-27	1.395833	1.421875	1.333333	1.364583	1.267107	492000
1999-01-28	1.395833	1.421875	1.333333	1.364583	1.267107	492000

pandas如何过滤数据

```
#filter
```

```
s = pd.Series([1,1,2,2,3,4,5,66,7])
s.groupby(s).sum()
```

```
1      2
2      4
3      3
4      4
5      5
7      7
66    66
dtype: int64
```

```
s.groupby(s).filter(lambda x : x.sum() > 4)
```

```
6     5
7    66
8     7
dtype: int64
```

```
df = pd.DataFrame({'A':np.arange(11),'B':list('aaabbbccddd')})
df
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	A	B
0	0	a
1	1	a
2	2	a
3	3	b
4	4	b
5	5	b
6	6	c
7	7	c
8	8	d
9	9	d
10	10	d

```
df.groupby('B').filter(lambda x : len(x) > 2)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	A	B
0	0	a
1	1	a
2	2	a
3	3	b
4	4	b
5	5	b
8	8	d
9	9	d
10	10	d

```
nvda.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Open	High	Low	Close	Adj Close	Volume
Date						
1999-01-22	1.750000	1.953125	1.552083	1.640625	1.523430	67867200
1999-01-25	1.770833	1.833333	1.640625	1.812500	1.683028	12762000
1999-01-26	1.833333	1.869792	1.645833	1.671875	1.552448	8580000
1999-01-27	1.677083	1.718750	1.583333	1.666667	1.547611	6109200
1999-01-28	1.666667	1.677083	1.651042	1.661458	1.542776	5688000