Adult Dataset

"Census Income" dataset.

Number of Instances: 48842 Number of Attributes: 14 Date Donated: 1996-05-01 Missing Values?: Yes

Attributes:

Number of Attributes: 6 continuous, 8 nominal attributes

- age: continuous.
- workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
- fnlwgt: continuous.
- education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
- education-num: continuous.
- marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
- occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.
- relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.
- race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
- sex: Female, Male.
- capital-gain: continuous.
- capital-loss: continuous.
- hours-per-week: continuous.
- native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.
- class: >50K, <=50K

In [18]:

```
import pandas as pd

adult_df = pd.read_csv('adult.csv')
adult_df.head()
```

Out[18]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	F

In [19]:

```
type(adult_df.age)
```

Out[19]:

```
pandas.core.series.Series
```

In [20]:

```
type(adult_df)
```

Out[20]:

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

In [21]:

```
adult_df.loc[0].index
```

Out[21]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'education-num',
       'marital-status', 'occupation', 'relationship', 'race', 'se
x',
       'capitalGain', 'capitalLoss', 'hoursPerWeek', 'nativeCountr
y',
       'income'],
      dtype='object')
```

In [22]:

```
adult_df.age.index
```

Out[22]:

```
RangeIndex(start=0, stop=32561, step=1)
```

In [23]:

```
adult_df.set_index(np.arange(10000,42561),inplace=True)
```

In [24]:

```
adult_df.set_index(np.arange(10000,42561))
```

Out[24]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	ra
10000	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	Wh
10001	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	Wh
10002	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	Wh
10003	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Bla
10004	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Bla
...	...	...	...	...	...	...	...	...	
42556	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	Wh
42557	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	Wh
42558	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried	Wh
42559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	Wh
42560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	Wh

32561 rows × 15 columns

In [26]:

```
adult_df.iloc[2].loc['education']
```

Out[26]:

```
'HS-grad'
```

In [27]:

```
adult_df.education.loc[10002]
```

Out[27]:

```
'HS-grad'
```

In [28]:

```
adult_df['education'].iloc[2]
```

Out[28]:

```
'HS-grad'
```

In [29]:

```
adult_df.at[10002,'education']
```

Out[29]:

```
'HS-grad'
```

In [30]:

```
row_series = adult_df.loc[10002]
print(row_series.loc['education'])
print(row_series.iloc[3])
print(row_series['education'])
print(row_series.education)
```

```
HS-grad
HS-grad
HS-grad
HS-grad
```

In [31]:

```
columns_series = adult_df.education
print(columns_series.loc[10002])
print(columns_series.iloc[2])
print(columns_series[10002])
# print(row_series.10002)  This will give syntax error!
```

```
HS-grad
HS-grad
HS-grad
```

Slicing

In [32]:

```
my_array = np.array([[2,3,5,7],[11,13,17,19],
                     [23,29,31,37,], [41,43,47,49]])
my_array
```

Out[32]:

```
array([[ 2,  3,  5,  7],
       [11, 13, 17, 19],
       [23, 29, 31, 37],
       [41, 43, 47, 49]])
```

In [33]:

```
my_array[1,1]
```

Out[33]:

```
13
```

In [34]:

```
my_array[1,:]
```

Out[34]:

```
array([11, 13, 17, 19])
```

In [35]:

```
my_array[:,1]
```

Out[35]:

```
array([ 3, 13, 29, 43])
```

In [36]:

```
my_array
```

Out[36]:

```
array([[ 2,  3,  5,  7],
       [11, 13, 17, 19],
       [23, 29, 31, 37],
       [41, 43, 47, 49]])
```

In [37]:

```
my_array[1:3,:]
```

Out[37]:

```
array([[11, 13, 17, 19],
       [23, 29, 31, 37]])
```

In [38]:

```
my_array[1:3,0:2]
```

Out[38]:

```
array([[11, 13],
       [23, 29]])
```

In [39]:

```
my_array[1:3,[0,2]]
```

Out[39]:

```
array([[11, 17],
       [23, 31]])
```

In [40]:

```
adult_df.loc[:,'education':'occupation']
```

Out[40]:

	education	education-num	marital-status	occupation
10000	Bachelors	13	Never-married	Adm-clerical
10001	Bachelors	13	Married-civ-spouse	Exec-managerial
10002	HS-grad	9	Divorced	Handlers-cleaners
10003	11th	7	Married-civ-spouse	Handlers-cleaners
10004	Bachelors	13	Married-civ-spouse	Prof-specialty
...	...	...	...	...
42556	Assoc-acdm	12	Married-civ-spouse	Tech-support
42557	HS-grad	9	Married-civ-spouse	Machine-op-inspct
42558	HS-grad	9	Widowed	Adm-clerical
42559	HS-grad	9	Never-married	Adm-clerical
42560	HS-grad	9	Married-civ-spouse	Exec-managerial

32561 rows × 4 columns

In [41]:

```
adult_df.sort_values('education-num').reset_index().iloc[1:32561:3617]
```

Out[41]:

	index	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationsh
1	23248	68	Private	168794	Preschool	1	Never-married	Machine-op-inspct	Not-i fam
3618	19607	25	Private	251854	11th	7	Never-married	Adm-clerical	Own-ch
7235	38845	31	Private	272856	HS-grad	9	Never-married	Craft-repair	Own-ch
10852	32759	56	Private	182273	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husbai
14469	10419	34	State-gov	240283	HS-grad	9	Divorced	Transport-moving	Unmarri
18086	31532	25	Self-emp-inc	98756	Some-college	10	Divorced	Adm-clerical	Own-ch
21703	17245	37	Federal-gov	40955	Some-college	10	Never-married	Other-service	Own-ch
25320	40595	43	Private	342567	Bachelors	13	Married-spouse-absent	Adm-clerical	Unmarri
28937	15200	43	Federal-gov	144778	Bachelors	13	Never-married	Exec-managerial	Not-i fam
32554	27308	55	Self-emp-not-inc	53566	Doctorate	16	Divorced	Exec-managerial	Not-i fam

In [42]:

```
twopowers_sr = pd.Series([1,2,4,8,16,32,64,128,256,512,1024])
BM = [False,False,False,True,False,False,False,True,True,True,True]
twopowers_sr[BM]
```

Out[42]:

```
3        8
7      128
8      256
9      512
10    1024
dtype: int64
```

In [43]:

```
twopowers_sr >=500
```

Out[43]:

```
0     False
1     False
2     False
3     False
4     False
5     False
6     False
7     False
8     False
9      True
10     True
dtype: bool
```

In [44]:

```
BM = twopowers_sr >=500
twopowers_sr[BM]
```

Out[44]:

```
9      512
10    1024
dtype: int64
```

In [45]:

```
twopowers_sr[twopowers_sr >=500]
```

Out[45]:

```
9      512
10    1024
dtype: int64
```

In [46]:

```
BM = adult_df.education == 'Preschool'
print('Mean: {}'.format(np.mean(adult_df[BM].age)))
print('Median: {}'.format(np.median(adult_df[BM].age)))
```

```
Mean: 42.76470588235294
Median: 41.0
```

In [47]:

```
BM1 = adult_df['education-num'] > 10
BM2 = adult_df['education-num'] < 10

print('More than 10 years of education - Capital Gain: {}'
      .format(np.mean(adult_df[BM1].capitalGain)))
print('Less than 10 years of education - Capital Gain: {}'
      .format(np.mean(adult_df[BM2].capitalGain)))
```

```
More than 10 years of education - Capital Gain: 2230.9397109166985
Less than 10 years of education - Capital Gain: 492.25532059102613
```

In [48]:

```
adult_df.shape
```

Out[48]:

```
(32561, 15)
```

In [49]:

```
adult_df.columns
```

Out[49]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'education-num',
       'marital-status', 'occupation', 'relationship', 'race', 'se
x',
       'capitalGain', 'capitalLoss', 'hoursPerWeek', 'nativeCountr
y',
       'income'],
      dtype='object')
```

In [50]:

```
adult_df.columns = ['age', 'workclass', 'fnlwgt', 'education',
                    'education_num', 'marital_status', 'occupation',
                    'relationship', 'race', 'sex', 'capitalGain',
                    'capitalLoss', 'hoursPerWeek', 'nativeCountry',
                    'income']
```

In [51]:

```
adult_df.describe()
```

Out[51]:

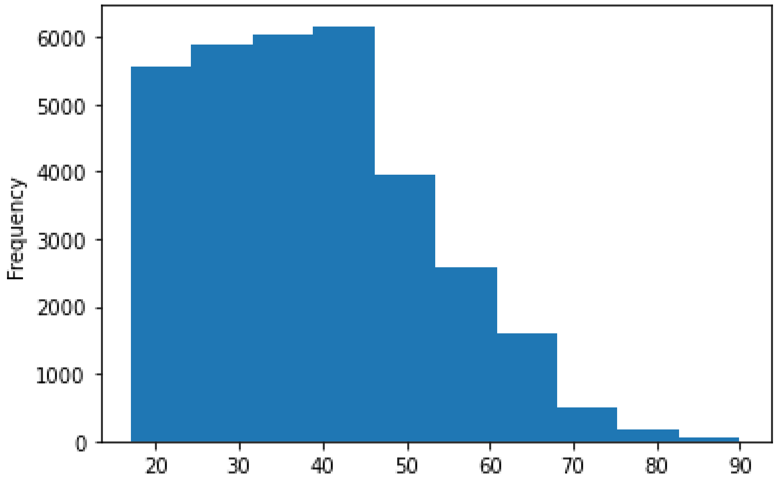
	age	fnlwgt	education_num	capitalGain	capitalLoss	hoursPerWeel
count	32561.000000	3.256100e+04	32561.000000	32561.000000	32561.000000	32561.00000(
mean	38.581647	1.897784e+05	10.080679	1077.648844	87.303830	40.43745(
std	13.640433	1.055500e+05	2.572720	7385.292085	402.960219	12.34742(
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.00000(
25%	28.000000	1.178270e+05	9.000000	0.000000	0.000000	40.00000(
50%	37.000000	1.783560e+05	10.000000	0.000000	0.000000	40.00000(
75%	48.000000	2.370510e+05	12.000000	0.000000	0.000000	45.00000(
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	99.00000(

In [52]:

```
adult_df.age.plot.hist()
```

Out[52]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1ae2c81d850>
```



In [53]:

```
adult_df.relationship.unique()
```

Out[53]:

```
array(['Not-in-family', 'Husband', 'Wife', 'Own-child', 'Unmarried',
       'Other-relative'], dtype=object)
```

In [54]:

```
adult_df.relationship.value_counts()
```

Out[54]:

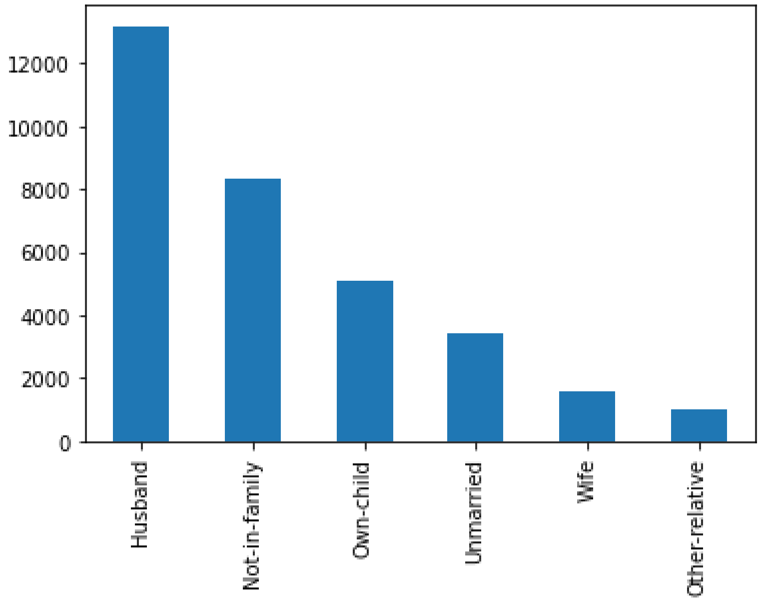
```
Husband           13193
Not-in-family      8305
Own-child          5068
Unmarried          3446
Wife               1568
Other-relative      981
Name: relationship, dtype: int64
```

In [55]:

```
adult_df.relationship.value_counts().plot.bar()
```

Out[55]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1ae2d043820>
```



Appy a function

In [56]:

```
def MultiplyBy2(n):
    return n*2

adult_df.age.apply(MultiplyBy2)
```

Out[56]:

```
10000     78
10001    100
10002     76
10003    106
10004     56
        ...
42556     54
42557     80
42558    116
42559     44
42560    104
Name: age, Length: 32561, dtype: int64
```

Applying a Function - Analytic Example 1

Divide every value in column fnlwgt by the sum of all its values.

In [57]:

```python
total_fnlwgt = adult_df.fnlwgt.sum()

def CalculatePercentage(v):
    return v/total_fnlwgt*100

adult_df.fnlwgt = adult_df.fnlwgt.apply(CalculatePercentage)
adult_df
```

Out[57]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relati
10000	39	State-gov	0.001254	Bachelors	13	Never-married	Adm-clerical	I
10001	50	Self-emp-not-inc	0.001348	Bachelors	13	Married-civ-spouse	Exec-managerial	Hu
10002	38	Private	0.003490	HS-grad	9	Divorced	Handlers-cleaners	I
10003	53	Private	0.003798	11th	7	Married-civ-spouse	Handlers-cleaners	Hu
10004	28	Private	0.005476	Bachelors	13	Married-civ-spouse	Prof-specialty	
...	...	...	...	...	...	...	...	
42556	27	Private	0.004164	Assoc-acdm	12	Married-civ-spouse	Tech-support	
42557	40	Private	0.002498	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Hu
42558	58	Private	0.002458	HS-grad	9	Widowed	Adm-clerical	Unr
42559	22	Private	0.003261	HS-grad	9	Never-married	Adm-clerical	Ow
42560	52	Self-emp-inc	0.004659	HS-grad	9	Married-civ-spouse	Exec-managerial	

32561 rows × 15 columns

In [58]:

```python
total_fnlwgt = adult_df.fnlwgt.sum()

adult_df.fnlwgt = adult_df.fnlwgt.apply(lambda v: v/total_fnlwgt*100)
adult_df
```

Out[58]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relati
10000	39	State-gov	0.001254	Bachelors	13	Never-married	Adm-clerical	I
10001	50	Self-emp-not-inc	0.001348	Bachelors	13	Married-civ-spouse	Exec-managerial	Hu
10002	38	Private	0.003490	HS-grad	9	Divorced	Handlers-cleaners	I
10003	53	Private	0.003798	11th	7	Married-civ-spouse	Handlers-cleaners	Hu
10004	28	Private	0.005476	Bachelors	13	Married-civ-spouse	Prof-specialty	
...	...	...	...	...	...	...	...	
42556	27	Private	0.004164	Assoc-acdm	12	Married-civ-spouse	Tech-support	
42557	40	Private	0.002498	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Hu
42558	58	Private	0.002458	HS-grad	9	Widowed	Adm-clerical	Unr
42559	22	Private	0.003261	HS-grad	9	Never-married	Adm-clerical	Ow
42560	52	Self-emp-inc	0.004659	HS-grad	9	Married-civ-spouse	Exec-managerial	

32561 rows × 15 columns

In [59]:

```
def CalcLifeNoEd(row):
    return row.age - row.education_num

adult_df.apply(CalcLifeNoEd,axis=1)
```

Out[59]:

```
10000    26
10001    37
10002    29
10003    46
10004    15
         ..
42556    15
42557    31
42558    49
42559    13
42560    43
Length: 32561, dtype: int64
```

In [60]:

```
adult_df.apply(lambda r: r.age-r.education_num,axis=1)
```

Out[60]:

```
10000    26
10001    37
10002    29
10003    46
10004    15
         ..
42556    15
42557    31
42558    49
42559    13
42560    43
Length: 32561, dtype: int64
```

In [61]:

```
adult_df['lifeNoEd'] = adult_df.apply(
    lambda r: r.age-r.education_num,axis=1)

adult_df['capitalNet'] = adult_df.apply(
    lambda r: r.capitalGain - r.capitalLoss,axis=1)

adult_df[['education_num','lifeNoEd','capitalNet']].corr()
```

Out[61]:

	education_num	lifeNoEd	capitalNet
education_num	1.000000	-0.150452	0.117891
lifeNoEd	-0.150452	1.000000	0.051490
capitalNet	0.117891	0.051490	1.000000

Groupby

In [62]:

```
adult_df.groupby(['marital_status','sex']).age.median()
```

Out[62]:

```
marital_status         sex
Divorced               Female    43.0
                       Male      42.0
Married-AF-spouse      Female    31.0
                       Male      29.0
Married-civ-spouse     Female    38.0
                       Male      43.0
Married-spouse-absent  Female    39.0
                       Male      41.0
Never-married          Female    25.0
                       Male      25.0
Separated              Female    39.0
                       Male      38.0
Widowed                Female    60.0
                       Male      62.5
Name: age, dtype: float64
```

In [63]:

```
adult_df.groupby(['race','sex']).capitalNet.mean()
```

Out[63]:

```
race                sex
Amer-Indian-Eskimo  Female     530.142857
                    Male       628.864583
Asian-Pac-Islander  Female     727.583815
                    Male      1707.440115
Black               Female     471.142765
                    Male       627.268324
Other               Female     218.385321
                    Male      1314.438272
White               Female     508.219857
                    Male      1266.413112
Name: capitalNet, dtype: float64
```

In [64]:

```
grb_result =adult_df.groupby(['race','sex']).capitalNet.mean()

print(grb_result.index)
```

```
MultiIndex([('Amer-Indian-Eskimo', 'Female'),
            ('Amer-Indian-Eskimo',   'Male'),
            ('Asian-Pac-Islander', 'Female'),
            ('Asian-Pac-Islander',   'Male'),
            (             'Black', 'Female'),
            (             'Black',   'Male'),
            (             'Other', 'Female'),
            (             'Other',   'Male'),
            (             'White', 'Female'),
            (             'White',   'Male')],
           names=['race', 'sex'])
```

In [65]:

```
grb_result =adult_df.groupby(['race','sex']).capitalNet.mean()
grb_result
```

Out[65]:

```
race                sex
Amer-Indian-Eskimo  Female     530.142857
                    Male       628.864583
Asian-Pac-Islander  Female     727.583815
                    Male      1707.440115
Black               Female     471.142765
                    Male       627.268324
Other               Female     218.385321
                    Male      1314.438272
White               Female     508.219857
                    Male      1266.413112
Name: capitalNet, dtype: float64
```

In [66]:

```
grb_result.unstack()
```

Out[66]:

sex	Female	Male
race		
Amer-Indian-Eskimo	530.142857	628.864583
Asian-Pac-Islander	727.583815	1707.440115
Black	471.142765	627.268324
Other	218.385321	1314.438272
White	508.219857	1266.413112

In [67]:

```
mlt_seris =adult_df.groupby(['race','sex','income']).fnlwgt.mean()
mlt_seris
```

Out[67]:

```
race                sex     income
Amer-Indian-Eskimo  Female  <=50K     0.001764
                            >50K      0.002395
                    Male    <=50K     0.002046
                            >50K      0.001954
Asian-Pac-Islander  Female  <=50K     0.002398
                            >50K      0.002305
                    Male    <=50K     0.002652
                            >50K      0.002762
Black               Female  <=50K     0.003454
                            >50K      0.003331
                    Male    <=50K     0.003922
                            >50K      0.003971
Other               Female  <=50K     0.002803
                            >50K      0.002593
                    Male    <=50K     0.003478
                            >50K      0.003310
White               Female  <=50K     0.002969
                            >50K      0.002978
                    Male    <=50K     0.003074
                            >50K      0.003025
Name: fnlwgt, dtype: float64
```

In [68]:

```
mlt_seris.unstack()
```

Out[68]:

	income	**<=50K**	**>50K**
race	**sex**		
Amer-Indian-Eskimo	**Female**	0.001764	0.002395
	Male	0.002046	0.001954
Asian-Pac-Islander	**Female**	0.002398	0.002305
	Male	0.002652	0.002762
Black	**Female**	0.003454	0.003331
	Male	0.003922	0.003971
Other	**Female**	0.002803	0.002593
	Male	0.003478	0.003310
White	**Female**	0.002969	0.002978
	Male	0.003074	0.003025

In [69]:

```
mlt_seris.unstack().unstack()
```

Out[69]:

income	<=50K		>50K	
sex	Female	Male	Female	Male
race				
Amer-Indian-Eskimo	0.001764	0.002046	0.002395	0.001954
Asian-Pac-Islander	0.002398	0.002652	0.002305	0.002762
Black	0.003454	0.003922	0.003331	0.003971
Other	0.002803	0.003478	0.002593	0.003310
White	0.002969	0.003074	0.002978	0.003025

In [70]:

```
mlt_df= mlt_seris.unstack().unstack()
mlt_df.columns
```

Out[70]:

```
MultiIndex([('<=50K', 'Female'),
            ('<=50K',   'Male'),
            ( '>50K', 'Female'),
            ( '>50K',   'Male')],
           names=['income', 'sex'])
```

In [71]:

```
mlt_df.stack()
```

Out[71]:

	income	<=50K	>50K
race	sex		
Amer-Indian-Eskimo	Female	0.001764	0.002395
	Male	0.002046	0.001954
Asian-Pac-Islander	Female	0.002398	0.002305
	Male	0.002652	0.002762
Black	Female	0.003454	0.003331
	Male	0.003922	0.003971
Other	Female	0.002803	0.002593
	Male	0.003478	0.003310
White	Female	0.002969	0.002978
	Male	0.003074	0.003025

In [72]:

```
mlt_df.stack().stack()
```

Out[72]:

```
race                 sex     income
Amer-Indian-Eskimo   Female  <=50K     0.001764
                             >50K      0.002395
                     Male    <=50K     0.002046
                             >50K      0.001954
Asian-Pac-Islander   Female  <=50K     0.002398
                             >50K      0.002305
                     Male    <=50K     0.002652
                             >50K      0.002762
Black                Female  <=50K     0.003454
                             >50K      0.003331
                     Male    <=50K     0.003922
                             >50K      0.003971
Other                Female  <=50K     0.002803
                             >50K      0.002593
                     Male    <=50K     0.003478
                             >50K      0.003310
White                Female  <=50K     0.002969
                             >50K      0.002978
                     Male    <=50K     0.003074
                             >50K      0.003025
dtype: float64
```

Pivot & Melt

In [73]:

```
wide_df = pd.read_csv('wide.csv')
wide_df
```

Out[73]:

	ReadingDateTime	NO	NO2	NOX	PM10	PM2.5
0	01/01/2017 00:00	3.5	30.8	36.2	35.7	31.0
1	01/01/2017 01:00	3.6	31.5	37.0	28.5	31.0
2	01/01/2017 02:00	2.2	27.3	30.7	22.7	31.0

In [74]:

```
wide_df.melt(id_vars='ReadingDateTime',
             value_vars=['NO','NO2','NOX','PM10','PM2.5'],
             var_name='Species',
             value_name='Value')
```

Out[74]:

	ReadingDateTime	Species	Value
0	01/01/2017 00:00	NO	3.5
1	01/01/2017 01:00	NO	3.6
2	01/01/2017 02:00	NO	2.2
3	01/01/2017 00:00	NO2	30.8
4	01/01/2017 01:00	NO2	31.5
5	01/01/2017 02:00	NO2	27.3
6	01/01/2017 00:00	NOX	36.2
7	01/01/2017 01:00	NOX	37.0
8	01/01/2017 02:00	NOX	30.7
9	01/01/2017 00:00	PM10	35.7
10	01/01/2017 01:00	PM10	28.5
11	01/01/2017 02:00	PM10	22.7
12	01/01/2017 00:00	PM2.5	31.0
13	01/01/2017 01:00	PM2.5	31.0
14	01/01/2017 02:00	PM2.5	31.0

In [75]:

```
long_df = pd.read_csv('long.csv')
long_df
```

Out[75]:

	ReadingDateTime	Species	Value
0	01/01/2017 00:00	NO	3.5
1	01/01/2017 01:00	NO	3.6
2	01/01/2017 02:00	NO	2.2
3	01/01/2017 00:00	NO2	30.8
4	01/01/2017 01:00	NO2	31.5
5	01/01/2017 02:00	NO2	27.3
6	01/01/2017 00:00	NOX	36.2
7	01/01/2017 01:00	NOX	37.0
8	01/01/2017 02:00	NOX	30.7
9	01/01/2017 00:00	PM10	35.7
10	01/01/2017 01:00	PM10	28.5
11	01/01/2017 02:00	PM10	22.7
12	01/01/2017 00:00	PM2.5	31.0
13	01/01/2017 01:00	PM2.5	31.0
14	01/01/2017 02:00	PM2.5	31.0

In [76]:

```
long_df.pivot(index='ReadingDateTime',
              columns='Species',
              values='Value')
```

Out[76]:

Species	NO	NO2	NOX	PM10	PM2.5
ReadingDateTime					
01/01/2017 00:00	3.5	30.8	36.2	35.7	31.0
01/01/2017 01:00	3.6	31.5	37.0	28.5	31.0
01/01/2017 02:00	2.2	27.3	30.7	22.7	31.0