

# 第8回 データの集計とグラフ作成1

# 目次

- 事例: アンケートデータの集計
- データの取り込み(テキストファイル(1))
- データの取り込み(テキストファイル(2))
- データの取り込み(Excelファイル)
- 特定の行や列を取り出す
- 質的変数の単純集計(1)
- 質的変数の単純集計(2)
- 質的変数の単純集計(3)
- 課題8
- 量的変数の単純集計
- クロス集計(1)
- クロス集計(2)
- クロス集計(3)
- 結果の保存

# 事例: アンケートデータの集計

- pandasモジュールを使った、簡単なアンケートデータ(質的変数、量的変数)の集計について試します。
- データファイルをダウンロードする
  - Python講座のサイト  
<http://pana4405.u-shizuoka-ken.ac.jp/python201803>  
から、Excel形式のサンプルファイル「python.xlsx」をダウンロードします。
  - 以下、python.xlsxがユーザのダウンロードフォルダに保存されていると仮定して演習を進めます。
  - ダウンロードしたpython.xlsxを開いて「アンケート」シートを確認します。

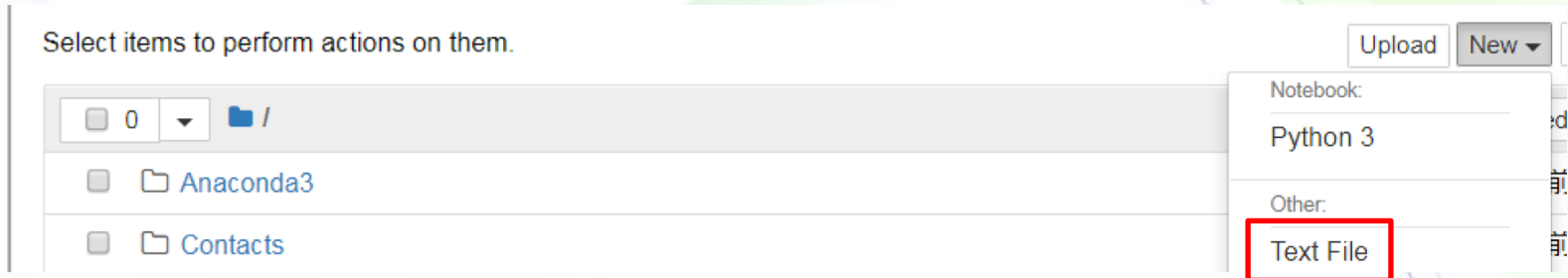
	A	B	C	D	F
1	性別	年代	職業	満足度	消費金額
2	女性	20代	学生	満足	¥7,400
3	女性	20代	学生	とても満足	¥4,000
4	男性	60代	無職	ふつう	¥9,300
5	女性	50代	会社員	やや満足	¥5,900
6	女性	20代	会社員	ふつう	¥9,400
7	女性	20代	パート・アルバイト	やや満足	¥6,900
8	女性	40代	会社員	とても満足	¥6,500

質的変数

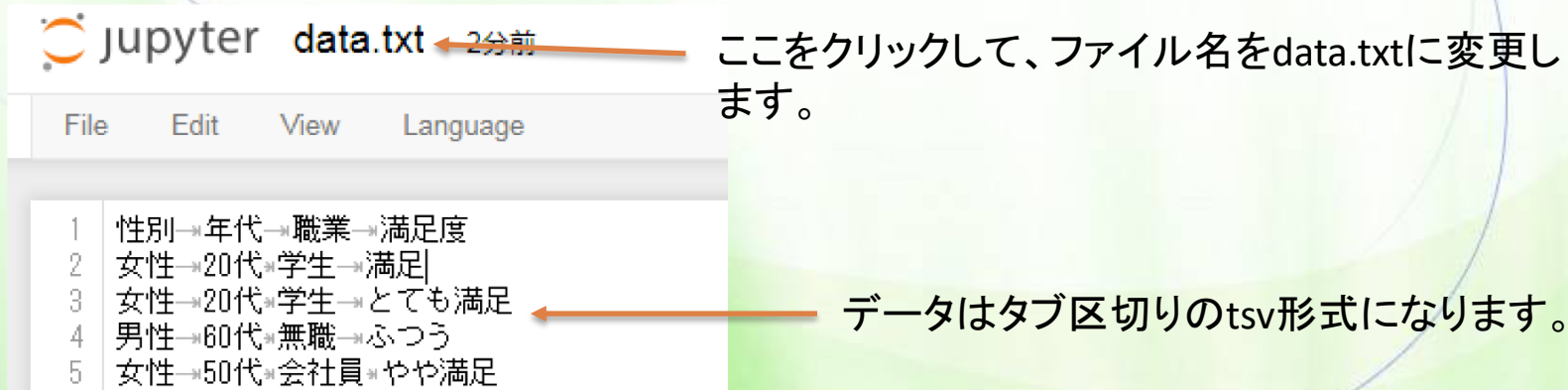
量的変数

# データの取り込み(テキストファイル(1))

- データをテキストファイル(csv, tsv)から取り込む方法について試します。
- 手順
  - Jupyterを起動し、HomeにText形式のファイルを新規作成します。



- 新規作成したTextファイルのファイル名を「data.txt」に変更します。
- ExcelのアンケートシートのA1からD242を範囲指定してコピーし、Jupyterのdata.txtに貼り付けます。[File]-[Save]で保存します。

A screenshot of the JupyterLab editor interface. The top bar shows 'jupyter data.txt' with a timestamp '2分前'. Below the menu bar (File, Edit, View, Language), the text content of the file is displayed in a table format with tab-separated columns. An orange arrow points from the text 'ここをクリックして、ファイル名をdata.txtに変更します。' to the 'data.txt' label in the top bar. Another orange arrow points from the text 'データはタブ区切りのtsv形式になります。' to the first row of data in the table.

1	性別	年代	職業	満足度
2	女性	20代	学生	満足
3	女性	20代	学生	とても満足
4	男性	60代	無職	ふつう
5	女性	50代	会社員	やや満足

# データの取り込み(テキストファイル(2))

- Jupyterで新たなNotebookを[New]-[Python 3]として新規作成します。
- pandasによってテキストファイルを取り込むには、read\_csv関数を使います。その際、区切り文字を引数sepで指定します。
  - csv(カンマ区切り)ならば","
  - tsv(タブ区切り)ならば"¥t"
- ファイル名にURLを記述すれば、ネット上のファイルも取り込めます。
- 取り込んだデータは、DataFrame型という行列形式のデータ型として管理されます。また、1行目は自動的にラベルとして認識されます。

DataFrame型

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams["font.family"] = "IPAexGothic"

data = pd.read_csv("data.txt", sep="¥t")
print(data)
```

pandasのインポート

# データの取り込み (Excelファイル)

- pandasでは、テキストファイルだけでなくExcelファイルを直接読み込むことができます。
  - ExcelFile関数でExcelファイルを開く。
  - 開いたファイルのparseメソッドでシート名を指定して取り込む。

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams["font.family"] = "IPAexGothic"

file = pd.ExcelFile("Downloads¥python.xlsx")
data = file.parse("アンケート")
print(data)
```

ダウンロードフォルダ内に  
保存されていると仮定

「アンケート」シートの取り込み

取り込んだデータには自動  
的に連番のindexが割り当  
てられます。

	性別	年代	職業	満足度	消費金額
0	女性	20代	学生	満足	7400
1	女性	20代	学生	とても満足	4000
2	男性	60代	無職	ふつう	9300
3	女性	50代	会社員	やや満足	5900

# 特定の行や列を取り出す

- pandasのDataFrameでは特定の行や列を取り出すことができます。

		性別	年代	職業	満足度	消費金額	← 列名
0行目	→ 0	女性	20代	学生	満足	7400	
1行目	→ 1	女性	20代	学生	とても満足	4000	

- 特定の列を取り出す
  - データフレーム.列名
- インデックス名やラベル名を指定して取り出す
  - データフレーム.loc[行インデックス名, 列ラベル名]
- 番号を指定して取り出す
  - データフレーム.iloc[行番号, 列番号]

```
#性別の列を取り出し
print(data.性別)
#0行目の"性別"を取り出し
print(data.loc[0, "性別"])
#0行目の性別と満足度を取り出し (リストで指定)
print(data.loc[0, ["性別", "満足度"]])
#0行0列を取り出し
print(data.iloc[0, 0])
#0行の0列から3列までを取り出し (スライスで指定)
print(data.iloc[0, 0:4])
#0行の1列と3列を取り出し (リストで指定)
print(data.iloc[0, [1,3]])
```

行指定の0を「:」とすると、全行を取り出せます。

# 質的変数の単純集計(1)

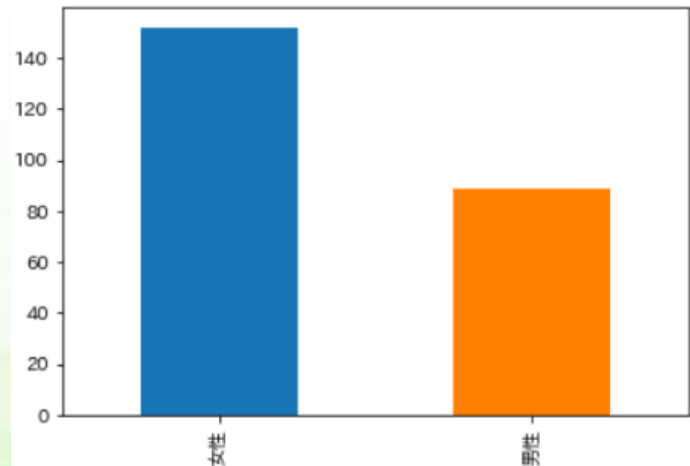
- value\_counts関数により、質的変数の項目ごとに単純集計を実行できます。
  - 構文 データフレーム.value\_counts()
- 集計により新たなDataFrameが生成され、そのDataFrameを元に縦棒グラフを描画します。
  - 構文 データフレーム.plot.bar()

```
sei = data.性別.value_counts()  
print(sei)  
sei.plot.bar()
```

集計結果は新たなデータフレームとなる

女性	152
男性	89

pandasでは内部的にmatplotlibを利用してグラフが描画できる

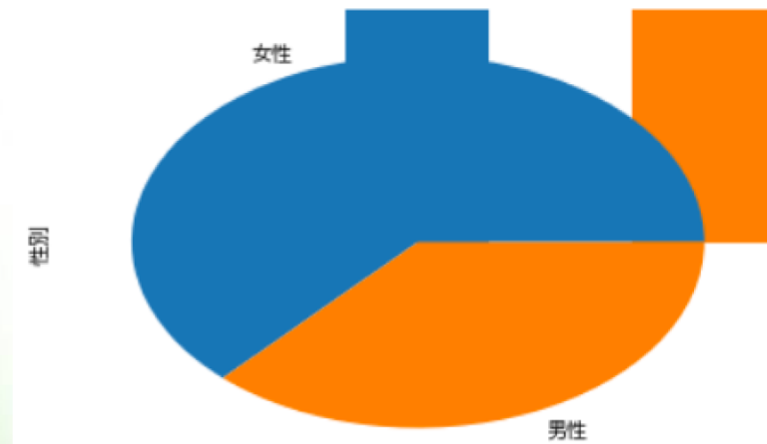




# 質的変数の単純集計(2)

- 棒グラフbar以外にも、円グラフpieなど様々な種類のグラフを描くことができます。
  - bar(縦棒グラフ)、barh(横棒グラフ)、pie(円グラフ)、scatter(散布図)、line(折れ線グラフ)など
- 複数のグラフを一度に描きたい場合は、単純にplotメソッドを並べただけでは、グラフが重なって描かれてしまいます。

```
sei.plot.bar()  
sei.plot.pie()
```



# 質的変数の単純集計(3)

- 複数のグラフを並べたい場合は、あらかじめmatplotlibのpyplotで、subplotsメソッドを使いグラフをいくつ並べるか設定しておき、軸(axes)を指定してグラフを描きます。

```
sei = data.性別.value_counts()
print(sei)
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(10, 5))
sei.plot.bar(ax=axes[0])
sei.plot.pie(ax=axes[1], autopct="%.1f%%")
```

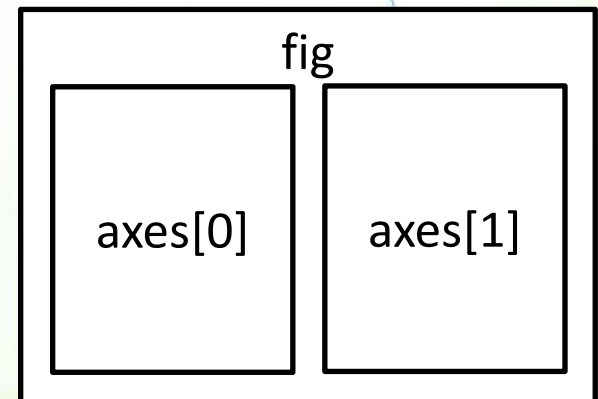
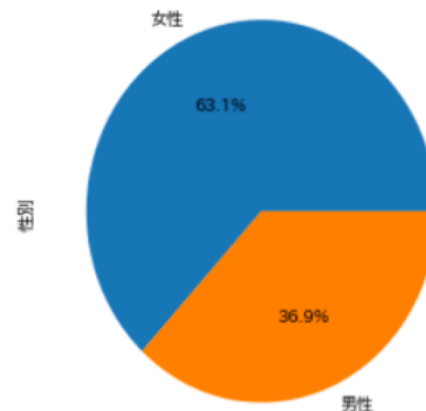
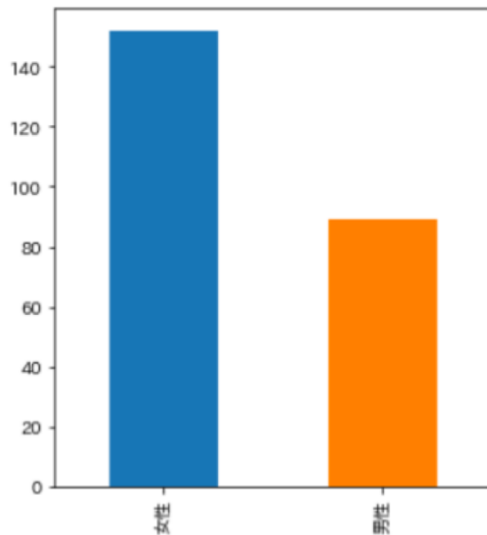
グラフの行数

グラフの列数

グラフ全体のサイズ(インチ)

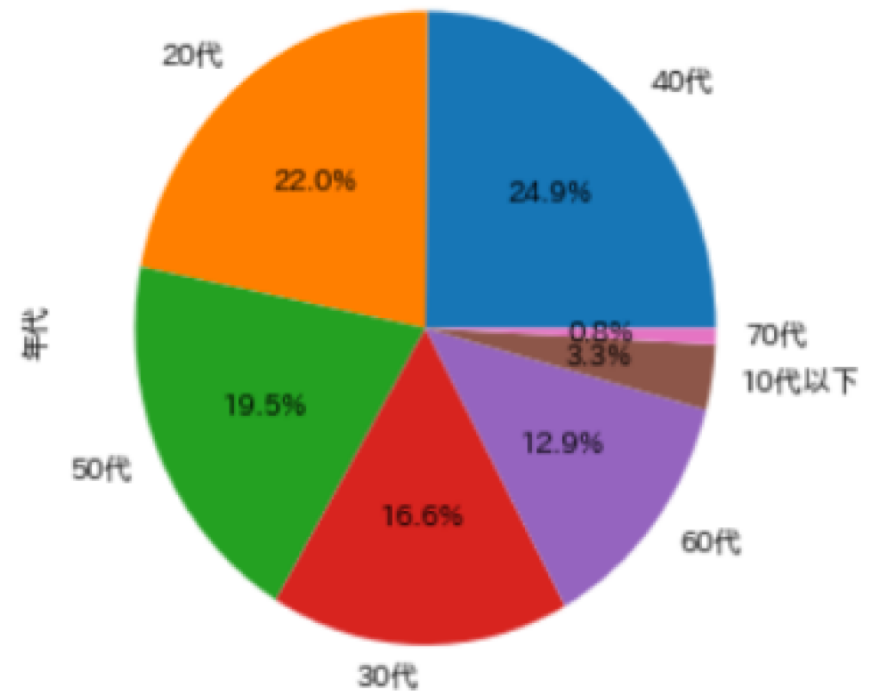
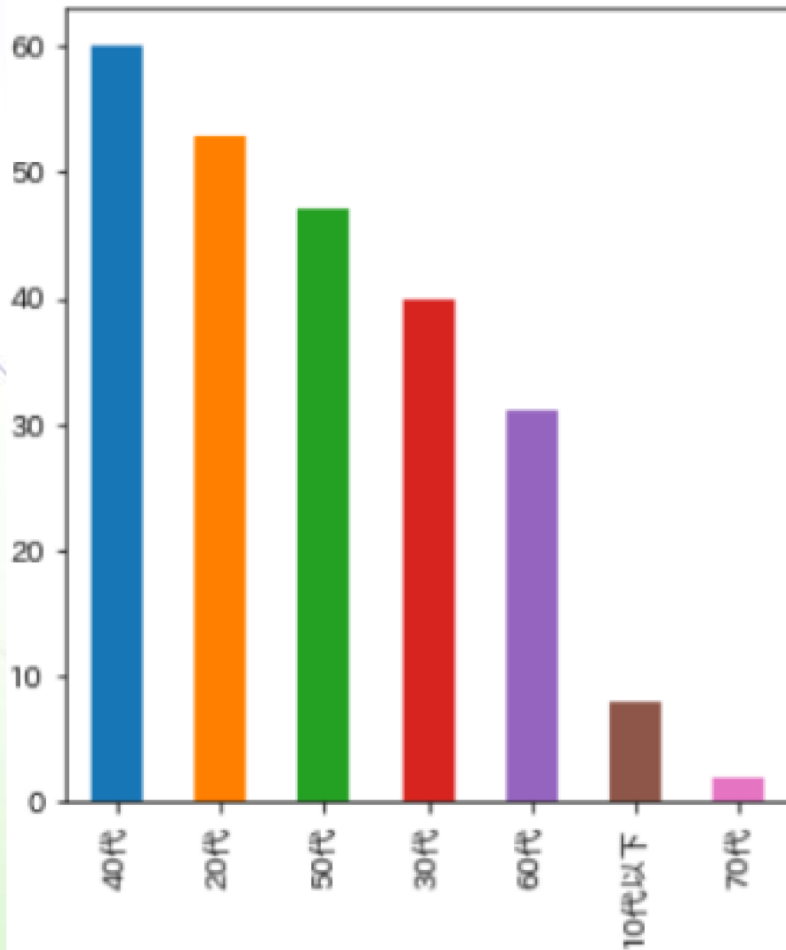
軸の指定

円グラフに割合を表示(小数点以下1位まで%つきで表示)



# 課題8

- 年代、職業、満足度についてもコードを修正してグラフを描画してください。



# 量的変数の単純集計

- 量的変数については、和(sum)、平均値(mean)、最小値(min)、最大値(max)などの集計が可能です。
- 基本統計量の一括計算がdescribeメソッドで可能です。
- 量的変数に対してヒストグラムの描画がplot.histメソッドで可能です。

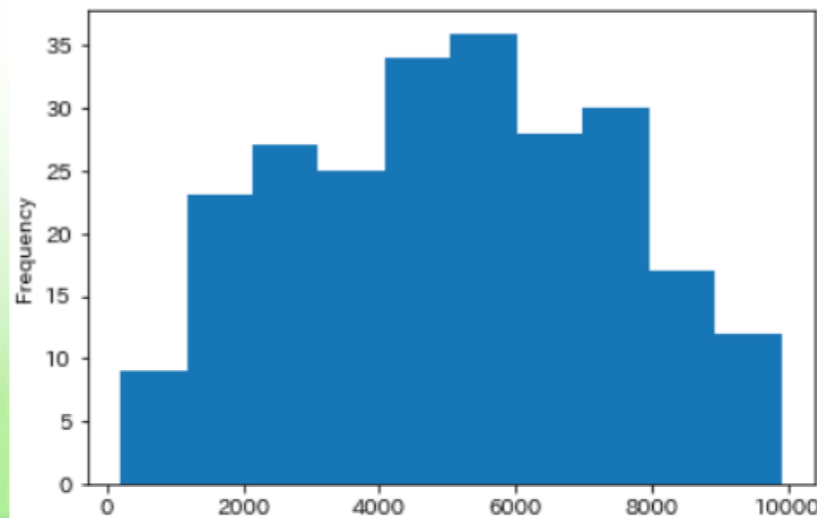
```
print(data.消費金額.sum())  
print(data.消費金額.mean())  
print(data.消費金額.min())  
print(data.消費金額.max())  
print(data.消費金額.describe())  
data.消費金額.plot.hist(bins=10)
```

和や平均値の計算

基本統計量

binsでヒストグラムの区分数を指定

```
1211530  
5027.095435684647  
200  
9900  
count      241.000000  
mean       5027.095436  
std        2351.373328  
min        200.000000  
25%       3200.000000  
50%       5100.000000  
75%       6900.000000  
max       9900.000000
```



# クロス集計(1)

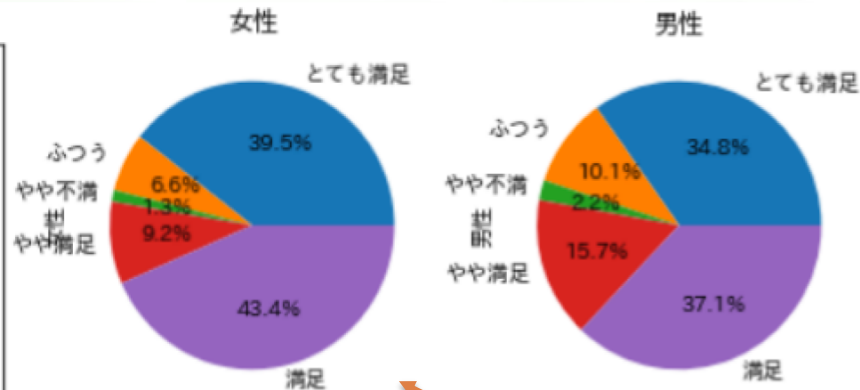
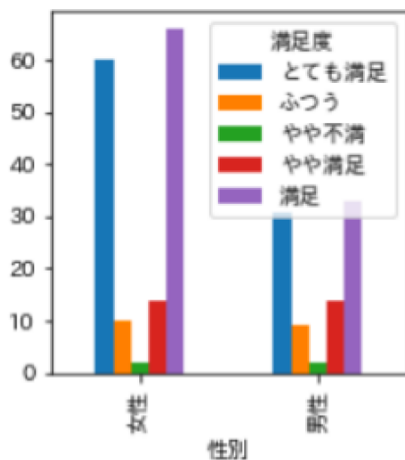
- 2つの項目を行と列に割り当てて集計するクロス集計は、pandasのcrosstabメソッドで簡単に実行できます。

```
cross = pd.crosstab(data.性別, data.満足度)
print(cross)
fig, axes = plt.subplots(nrows=1, ncols=3, figsize=(10, 3))
cross.plot.bar(ax=axes[0])
cross.loc["女性"].plot.pie(ax=axes[1], autopct="%.1f%%", title="女性")
cross.loc["男性"].plot.pie(ax=axes[2], autopct="%.1f%%", title="男性")
```

集計したい列を指定  
stacked=Trueを追記すると積み上げ型に

満足度

性別	とても満足	ふつう	やや不満	やや満足	満足
女性	60	10	2	14	66
男性	31	9	2	14	33



クロス集計することで、女性の方が満足度が高いことがわかる。 - 13 -

# クロス集計(2)

- pandasのcrosstabメソッド以外に、DataFrameのpivot\_tableメソッドでクロス集計することもできます。下記の例では、消費金額の合計値を集計しています。

```
cross = data.pivot_table(values="消費金額", index="性別", columns="年代",  
                          aggfunc="sum")  
print(cross)
```

← 集計項目      ← 行      ← 列

← 集計方法(その他にmean, countなど)

年代	10代以下	20代	30代	40代	50代	60代	70代
性別							
女性	27300.0	174910.0	125400.0	220300.0	148000.0	54500.0	NaN
男性	14000.0	81600.0	78700.0	72500.0	94100.0	116220.0	4000.0

- 値が存在しない項目にはNaNが表示されますが、fill\_value=0を設定することで、強制的に0で埋めることができます。

```
cross = data.pivot_table(values="消費金額", index="性別", columns="年代",  
                          aggfunc="sum", fill_value=0)  
print(cross)
```

← NaNを0で埋める

年代	10代以下	20代	30代	40代	50代	60代	70代
性別							
女性	27300	174910	125400	220300	148000	54500	0
男性	14000	81600	78700	72500	94100	116220	4000

# クロス集計(3)

- crosstabやpivot\_table以外に、同じ値を持つ行をグループ化するgroupbyメソッドによっても項目ごとの集計が可能です。
  - groupbyの場合には、列に並ぶ項目は無く、すべて行方向に並ぶ形になります。

性別と年代でグループ化

```
print(data.groupby(["性別", "年代"]).消費金額.sum())
```

消費金額の和を計算



下記のpivot\_tableでも同じ結果が得られます。

```
print(data.pivot_table(values="消費金額",  
index=["性別", "年代"], aggfunc="sum", fill_value=0))
```

列ではなく全て行に項目を指定

性別	年代	
女性	10代以下	27300
	20代	174910
	30代	125400
	40代	220300
	50代	148000
	60代	54500
男性	10代以下	14000
	20代	81600
	30代	78700
	40代	72500
	50代	94100
	60代	116220
	70代	4000

# 結果の保存

- matplotlibで描いた図は、plt.savefig関数で画像として保存できます。
- pandasのDataFrameは、to\_csv関数でcsvファイルとして保存できます。

```
plt.savefig("アンケート.png")  
cross.to_csv("結果.txt")
```

