# レシピ60 減量の勝負でtransform（P167）

- このレシピでは、シミュレーションデータを使い、２人が４ヶ月間減量したパーセントを追跡する
- 月末に、その月の減量パーセントの高い人が勝ち
- 減量の追跡のため、データを月と人でグループ分けし、`transform`メソッドを使い、月初から毎週の減量パーセントを計算する

---

（1）AmyとBobのデータの最初の月を調べる。毎月４回測定がある

In [98]:
```
weight_loss = pd.read_csv('weight_loss.csv')
print(weight_loss.shape)
weight_loss.head(3).append(weight_loss.tail(3))
```

(32, 4)

Out[98]:

| | Name | Month | Week | Weight |
|---|---|---|---|---|
| 0 | Bob | Jan | Week 1 | 291 |
| 1 | Amy | Jan | Week 1 | 197 |
| 2 | Bob | Jan | Week 2 | 288 |
| 29 | Amy | Apr | Week 3 | 164 |
| 30 | Bob | Apr | Week 4 | 250 |
| 31 | Amy | Apr | Week 4 | 161 |

In [99]:
```
weight_loss.query('Month == "Jan"')
```

Out[99]:

| | Name | Month | Week | Weight |
|---|---|---|---|---|
| 0 | Bob | Jan | Week 1 | 291 |
| 1 | Amy | Jan | Week 1 | 197 |
| 2 | Bob | Jan | Week 2 | 288 |
| 3 | Amy | Jan | Week 2 | 189 |
| 4 | Bob | Jan | Week 3 | 283 |
| 5 | Amy | Jan | Week 3 | 189 |
| 6 | Bob | Jan | Week 4 | 283 |
| 7 | Amy | Jan | Week 4 | 190 |

---

（2）

- 各月の勝者を決定するため、各月のW1とW4の減量幅を調べればよい
- しかし週ごとに更新が入る場合もあるため、各月でW1から今週までの減量幅も計算できるような毎週更新する関数をつくる

In [100…
```
def find_perc_loss(s):
    return (s - s.iloc[0]) / s.iloc[0]

# sはSeriesが引数になるのか。　そのiloc[0]（先頭）
```

---

（3）この関数を1月のBobでテストしてみる

In [101…
```
bob_jan = weight_loss.query('Name == "Bob" and Month == "Jan"')
bob_jan
```

Out[101…

| | Name | Month | Week | Weight |
|---|---|---|---|---|
| 0 | Bob | Jan | Week 1 | 291 |
| 2 | Bob | Jan | Week 2 | 288 |
| 4 | Bob | Jan | Week 3 | 283 |
| 6 | Bob | Jan | Week 4 | 283 |

In [102…
```
# ユーザー関数の引数として、「Weight」のSeriesを渡す
find_perc_loss(bob_jan['Weight'])
```

Out[102…
```
0    0.000000
2   -0.010309
4   -0.027491
6   -0.027491
Name: Weight, dtype: float64
```

---

（4）人と月でグループ化、第1週と比較した減量率を取得したい。

- 集約関数として、`transform`を使っている（引数：ユーザー定義関数pcnt_loss）
- `transform`は同じ長さの値を戻す

### `tranceform`

- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.transform.html

In [103…
```
# 減量率
pcnt_loss = weight_loss.groupby(['Name', 'Month'])['Weight']\
                       .transform(find_perc_loss)

pcnt_loss.head(8)
```

Out[103…
```
0    0.000000
1    0.000000
2   -0.010309
3   -0.040609
4   -0.027491
5   -0.040609
6   -0.027491
7   -0.035533
Name: Weight, dtype: float64
```

---

（5）

- `transform`は呼び出したDataFrameと同じ行数のオブジェクトを返す
- 結果を新たなカラムとして追加する、出力はBobの最初の2ヶ月のデータを使う

In [104…
```
weight_loss['Perc Weight Loss'] = pcnt_loss.round(3)

weight_loss.query('Name=="Bob" and Month in ["Jan", "Feb"]')
```

Out[104…

| | Name | Month | Week | Weight | Perc Weight Loss |
|---|---|---|---|---|---|
| 0 | Bob | Jan | Week 1 | 291 | 0.000 |
| 2 | Bob | Jan | Week 2 | 288 | -0.010 |
| 4 | Bob | Jan | Week 3 | 283 | -0.027 |
| 6 | Bob | Jan | Week 4 | 283 | -0.027 |
| 8 | Bob | Feb | Week 1 | 283 | 0.000 |
| 10 | Bob | Feb | Week 2 | 275 | -0.028 |
| 12 | Bob | Feb | Week 3 | 268 | -0.053 |
| 14 | Bob | Feb | Week 4 | 268 | -0.053 |

---

（6）

- 減量%は、月が替わるとリセットされるのに注意
  - **なぜなら、NameとMonthでグループ化したものを計算対象としているため**
  - このあたりが感覚的にわかりづらいので注意
- 次で自動的に勝者がわかるようにするため、最終週だけが問題なので第4週を選択する

In [105…
```
week4 = weight_loss.query('Week == "Week 4"')
week4
```

Out[105…

| | Name | Month | Week | Weight | Perc Weight Loss |
|---|---|---|---|---|---|
| 6 | Bob | Jan | Week 4 | 283 | -0.027 |
| 7 | Amy | Jan | Week 4 | 190 | -0.036 |
| 14 | Bob | Feb | Week 4 | 268 | -0.053 |
| 15 | Amy | Feb | Week 4 | 173 | -0.089 |
| 22 | Bob | Mar | Week 4 | 261 | -0.026 |
| 23 | Amy | Mar | Week 4 | 170 | -0.017 |
| 30 | Bob | Apr | Week 4 | 250 | -0.042 |
| 31 | Amy | Apr | Week 4 | 161 | -0.053 |

---

（7）データの形式を`pivot`メソッドで変形する。BobとAmyの減量率を各月で比較できるように見せる

### `pivot`

- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.pivot.html?highlight=pivot#pandas.DataFrame.pivot
- pivotはindexとcolumnsがユニークのときだけ使える。重複があるとエラーになる（その場合はpivot_tableを利用）

In [106…
```
winner = week4.pivot(index='Month', columns='Name',
                     values='Perc Weight Loss')
winner
```

Out[106…

| Name | Amy | Bob |
|---|---|---|
| Month | | |
| Apr | -0.053 | -0.042 |
| Feb | -0.089 | -0.053 |
| Jan | -0.036 | -0.027 |
| Mar | -0.017 | -0.026 |

---

（8）上記の出力をより効果的に見せる

- **Numpyには`where`というベクトル化`if-then-else`関数がある**（←Excelのifと同じ、いろいろ使えそう）★
- これは、**BooleanのSeriesや配列を他の値にマップできる**（列をつくるならSeriesがよい、DFでも可能）
- 勝者のカラムを作り、各月の勝ったパーセントをハイライトする

### `s.where`

- https://pandas.pydata.org/docs/reference/api/pandas.Series.where.html?highlight=where#pandas.Series.where

In [107…
```
winner['Winner'] = np.where(winner['Amy'] < winner['Bob'], 'Amy', 'Bob')
winner
```

Out[107…

| Name | Amy | Bob | Winner |
|---|---|---|---|
| Month | | | |
| Apr | -0.053 | -0.042 | Amy |
| Feb | -0.089 | -0.053 | Amy |
| Jan | -0.036 | -0.027 | Amy |
| Mar | -0.017 | -0.026 | Bob |

### `Buitin styles`

- https://pandas.pydata.org/docs/reference/style.html#builtin-styles

In [108…
```
winner.style.highlight_min(axis=1)
```

Out[108…

| Name | Amy | Bob | Winner |
|---|---|---|---|
| Month | | | |
| Apr | -0.053000 | -0.042000 | Amy |
| Feb | -0.089000 | -0.053000 | Amy |
| Jan | -0.036000 | -0.027000 | Amy |
| Mar | -0.017000 | -0.026000 | Bob |

---

（9）values_countsメソッドを使って、最終的に勝った月の回数を返す

In [109…
```
winner.Winner.value_counts()
```

Out[109…
```
Amy    3
Bob    1
Name: Winner, dtype: int64
```

---

（補足）

- pandasではMonthの並びは「文字順」になってしまう（現在はオブジェクト型になっている）
- 対策としては、Monthのデータ型をカテゴリ変数に変換すれば解決できる

In [110…
```
week4a = week4.copy()
month_chron = week4a['Month'].unique()
month_chron
```

Out[110…
```
array(['Jan', 'Feb', 'Mar', 'Apr'], dtype=object)
```

In [111…
```
week4a['Month'] = pd.Categorical(week4a['Month'],
                                 categories=month_chron,
                                 ordered=True)
week4a.pivot(index='Month', columns='Name',
             values='Perc Weight Loss')
```

Out[111…

| Name | Amy | Bob |
|---|---|---|
| Month | | |
| Jan | -0.036 | -0.027 |
| Feb | -0.089 | -0.053 |
| Mar | -0.017 | -0.026 |
| Apr | -0.053 | -0.042 |

---