Обработка данных с помощью Pandas

Преобразование DataFrame

Осмотр DataFrame

Когда у вас появляется новый DataFrame для работы, первое, что вам нужно сделать, это изучить его и посмотреть, что он содержит. Существует несколько полезных методов и атрибутов для этого.

- `head()` возвращает несколько первых строк («голову» DataFrame).
- `info()` показывает информацию по каждому столбцу, такую как тип данных и количество отсутствующих значений.
- `.shape` возвращает количество строк и столбцов DataFrame.
- `describe()` вычисляет несколько сводных статистических данных для каждого столбца.
- `homelessness` - это DataFrame, содержащий оценки бездомности в каждом штате США в 2018 году. Индивидуальный столбец - это количество бездомных людей без детей. Столбец `family_members` - количество бездомных людей с детьми. Столбец `state_pop` - общая численность населения штата.

Необходимо импортировать pandas.

```
import pandas as pd
```

Выведите первые строки DataFrame `homelessness`.

```
homelessness = pd.read_csv('datasets/homelessness.csv')
homelessness.head()
```

```
   Unnamed: 0              region       state  individuals
family_members  \
0           0  East South Central     Alabama       2570.0
864.0
1           1             Pacific      Alaska       1434.0
582.0
2           2            Mountain     Arizona       7259.0
2606.0
3           3  West South Central    Arkansas       2280.0
432.0
4           4             Pacific  California     109008.0
20964.0

   state_pop
0    4887681
1     735139
```

```
2    7158024
3    3009733
4   39461588

homelessness.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51 entries, 0 to 50
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Unnamed: 0      51 non-null     int64
 1   region          51 non-null     object
 2   state           51 non-null     object
 3   individuals     51 non-null     float64
 4   family_members  51 non-null     float64
 5   state_pop       51 non-null     int64
dtypes: float64(2), int64(2), object(2)
memory usage: 2.5+ KB

homelessness.shape

(51, 6)
```

Print some summary statistics that describe the `homelessness` DataFrame.

```
# Print a description of homelessness
print(homelessness.describe())

       Unnamed: 0    individuals  family_members     state_pop
count   51.000000      51.000000       51.000000  5.100000e+01
mean    25.000000    7225.784314     3504.882353  6.405637e+06
std     14.866069   15991.025083     7805.411811  7.327258e+06
min      0.000000     434.000000       75.000000  5.776010e+05
25%     12.500000    1446.500000      592.000000  1.777414e+06
50%     25.000000    3082.000000     1482.000000  4.461153e+06
75%     37.500000    6781.500000     3196.000000  7.340946e+06
max     50.000000  109008.000000    52070.000000  3.946159e+07
```

Части DataFrame

Для лучшего понимания объектов DataFrame полезно знать, что они состоят из трех компонентов, хранящихся как атрибуты:

- `values`: Двумерный массив значений NumPy.
- `columns`: Индекс столбцов - имена столбцов.
- `index`: Индекс строк - либо номера строк, либо имена строк.

- Обычно индексы можно представить как список строк или чисел, хотя тип данных Index в pandas позволяет использовать более сложные варианты. (Об этом будет рассказано позже в курсе.)
- `homelessness` доступен.

Инструкции

- Вывести двумерный массив значений в `homelessness`.
- Вывести имена столбцов в `homelessness`.
- Вывести индекс в `homelessness`.

```
# Вывести значения DataFrame homelessness
homelessness.values
```

```
array([[0, 'East South Central', 'Alabama', 2570.0, 864.0, 4887681],
       [1, 'Pacific', 'Alaska', 1434.0, 582.0, 735139],
       [2, 'Mountain', 'Arizona', 7259.0, 2606.0, 7158024],
       [3, 'West South Central', 'Arkansas', 2280.0, 432.0, 3009733],
       [4, 'Pacific', 'California', 109008.0, 20964.0, 39461588],
       [5, 'Mountain', 'Colorado', 7607.0, 3250.0, 5691287],
       [6, 'New England', 'Connecticut', 2280.0, 1696.0, 3571520],
       [7, 'South Atlantic', 'Delaware', 708.0, 374.0, 965479],
       [8, 'South Atlantic', 'District of Columbia', 3770.0, 3134.0,
        701547],
       [9, 'South Atlantic', 'Florida', 21443.0, 9587.0, 21244317],
       [10, 'South Atlantic', 'Georgia', 6943.0, 2556.0, 10511131],
       [11, 'Pacific', 'Hawaii', 4131.0, 2399.0, 1420593],
       [12, 'Mountain', 'Idaho', 1297.0, 715.0, 1750536],
       [13, 'East North Central', 'Illinois', 6752.0, 3891.0,
12723071],
       [14, 'East North Central', 'Indiana', 3776.0, 1482.0, 6695497],
       [15, 'West North Central', 'Iowa', 1711.0, 1038.0, 3148618],
       [16, 'West North Central', 'Kansas', 1443.0, 773.0, 2911359],
       [17, 'East South Central', 'Kentucky', 2735.0, 953.0, 4461153],
       [18, 'West South Central', 'Louisiana', 2540.0, 519.0,
4659690],
       [19, 'New England', 'Maine', 1450.0, 1066.0, 1339057],
       [20, 'South Atlantic', 'Maryland', 4914.0, 2230.0, 6035802],
       [21, 'New England', 'Massachusetts', 6811.0, 13257.0, 6882635],
       [22, 'East North Central', 'Michigan', 5209.0, 3142.0,
9984072],
       [23, 'West North Central', 'Minnesota', 3993.0, 3250.0,
5606249],
       [24, 'East South Central', 'Mississippi', 1024.0, 328.0,
2981020],
       [25, 'West North Central', 'Missouri', 3776.0, 2107.0,
6121623],
       [26, 'Mountain', 'Montana', 983.0, 422.0, 1060665],
       [27, 'West North Central', 'Nebraska', 1745.0, 676.0, 1925614],
       [28, 'Mountain', 'Nevada', 7058.0, 486.0, 3027341],
```

```
       [29, 'New England', 'New Hampshire', 835.0, 615.0, 1353465],
       [30, 'Mid-Atlantic', 'New Jersey', 6048.0, 3350.0, 8886025],
       [31, 'Mountain', 'New Mexico', 1949.0, 602.0, 2092741],
       [32, 'Mid-Atlantic', 'New York', 39827.0, 52070.0, 19530351],
       [33, 'South Atlantic', 'North Carolina', 6451.0, 2817.0,
10381615],
       [34, 'West North Central', 'North Dakota', 467.0, 75.0,
758080],
       [35, 'East North Central', 'Ohio', 6929.0, 3320.0, 11676341],
       [36, 'West South Central', 'Oklahoma', 2823.0, 1048.0,
3940235],
       [37, 'Pacific', 'Oregon', 11139.0, 3337.0, 4181886],
       [38, 'Mid-Atlantic', 'Pennsylvania', 8163.0, 5349.0, 12800922],
       [39, 'New England', 'Rhode Island', 747.0, 354.0, 1058287],
       [40, 'South Atlantic', 'South Carolina', 3082.0, 851.0,
5084156],
       [41, 'West North Central', 'South Dakota', 836.0, 323.0,
878698],
       [42, 'East South Central', 'Tennessee', 6139.0, 1744.0,
6771631],
       [43, 'West South Central', 'Texas', 19199.0, 6111.0, 28628666],
       [44, 'Mountain', 'Utah', 1904.0, 972.0, 3153550],
       [45, 'New England', 'Vermont', 780.0, 511.0, 624358],
       [46, 'South Atlantic', 'Virginia', 3928.0, 2047.0, 8501286],
       [47, 'Pacific', 'Washington', 16424.0, 5880.0, 7523869],
       [48, 'South Atlantic', 'West Virginia', 1021.0, 222.0,
1804291],
       [49, 'East North Central', 'Wisconsin', 2740.0, 2167.0,
5807406],
       [50, 'Mountain', 'Wyoming', 434.0, 205.0, 577601]],
dtype=object)
```

```
# Вывести индекс столбцов DataFrame homelessness
homelessness.columns
```

```
Index(['Unnamed: 0', 'region', 'state', 'individuals',
'family_members',
       'state_pop'],
      dtype='object')
```

```
homelessness.index
```

```
RangeIndex(start=0, stop=51, step=1)
```

Сортировка строк

Найти интересные фрагменты данных в DataFrame часто легче, если вы измените порядок строк. Вы можете отсортировать строки, передав имя столбца в `.sort_values()`.

В случаях, когда строки имеют одинаковое значение (это часто бывает при сортировке по категориальной переменной), вы можете разрешить ситуацию, отсортировав по другому столбцу. Вы можете сортировать по нескольким столбцам, передав список имен столбцов.

Сортировать по	Синтаксис
one column	df.sort_values("breed")
multiple columns	df.sort_values(["breed", "weight_kg"])

Инструкции 1/2

- Совместив `.sort_values()` с `.head()`, вы можете ответить на вопросы в форме "Какие самые верхние случаи, где...?".
- Доступен `homelessness`, и `pandas` загружен как `pd`.
- Отсортируйте `homelessness` по количеству бездомных людей, от наименьшего к наибольшему, и сохраните это как `homelessness_ind`.
- Выведите первые строки отсортированного DataFrame.

```
# Отсортировать homelessness по числу бездомных
homelessness_ind = homelessness.sort_values("individuals")

# Вывести несколько первых строк
print(homelessness_ind.head())
```

```
    Unnamed: 0              region         state  individuals
family_members  \
50          50            Mountain       Wyoming        434.0
205.0
34          34  West North Central  North Dakota        467.0
75.0
7            7      South Atlantic      Delaware        708.0
374.0
39          39         New England  Rhode Island        747.0
354.0
45          45         New England       Vermont        780.0
511.0

    state_pop
50     577601
34     758080
7      965479
39    1058287
45     624358
```

```
# Sort homelessness by descending family members
homelessness_fam =
homelessness.sort_values("family_members",ascending=False)
```

```
# Print the top few rows
print(homelessness_fam.head())
```

```
    Unnamed: 0              region          state  individuals  \
32          32        Mid-Atlantic       New York      39827.0
4            4             Pacific     California     109008.0
21          21         New England  Massachusetts       6811.0
9            9      South Atlantic        Florida      21443.0
43          43  West South Central          Texas      19199.0

    family_members  state_pop
32         52070.0   19530351
4          20964.0   39461588
21         13257.0    6882635
9           9587.0   21244317
43          6111.0   28628666
```

Инструкции 2/2

- Отсортируйте `homelessness` сначала по региону (по возрастанию), а затем по количеству членов семьи (по убыванию). Сохраните это как `homelessness_reg_fam`.
- Выведите первые строки отсортированного DataFrame.

```
# Отсортировать homelessness по региону, затем по убыванию числа членов семьи
homelessness_reg_fam =
homelessness.sort_values(["region","family_members"],
ascending=[True,False])

# Вывести несколько первых строк
print(homelessness_reg_fam.head())
```

```
    Unnamed: 0              region      state  individuals
family_members  \
13          13  East North Central   Illinois       6752.0
3891.0
35          35  East North Central       Ohio       6929.0
3320.0
22          22  East North Central   Michigan       5209.0
3142.0
49          49  East North Central  Wisconsin       2740.0
2167.0
14          14  East North Central    Indiana       3776.0
1482.0

    state_pop
13   12723071
35   11676341
22    9984072
```

```
49    5807406
14    6695497
```

Выбор столбцов

При работе с данными вам может не понадобиться все переменные в вашем наборе данных. Квадратные скобки ([]) можно использовать для выбора только тех столбцов, которые вам интересны, в порядке, который имеет для вас смысл. Чтобы выбрать только "col_a" DataFrame df, используйте

```
df["col_a"]
```

Чтобы выбрать `"col_a"` и `"col_b"` в `df`, используйте

```
df[["col_a", "col_b"]]
```

`homelessness` доступен, и pandas загружен как pd.

Инструкции

- Создайте DataFrame с именем `individuals`, который содержит только столбец `individuals` из `homelessness`.
- Выведите несколько первых строк результата.

```
# Выбрать столбец individuals
individuals = homelessness["individuals"]

# Вывести несколько первых строк результата
print(individuals.head())
```

```
0      2570.0
1      1434.0
2      7259.0
3      2280.0
4    109008.0
Name: individuals, dtype: float64
```

1. Создайте DataFrame с именем `state_fam`, который содержит только столбцы `state` и `family_members` из `homelessness`, в указанном порядке.
- Выведите несколько первых строк результата.

```
# Выбрать столбцы state и family_members
state_fam = homelessness[["state" ,"family_members"]]

# Вывести несколько первых строк результата
print(state_fam.head())
```

```
        state  family_members
0     Alabama           864.0
1      Alaska           582.0
```

```
2     Arizona          2606.0
3    Arkansas           432.0
4  California         20964.0
```

1. Создайте DataFrame с именем `ind_state`, который содержит только столбцы `individuals` и `state` из `homelessness`, в указанном порядке. Выведите несколько первых строк результата.

```
# Выбрать только столбцы individuals и state, в указанном порядке
ind_state = homelessness[["individuals","state"]]

# Вывести несколько первых строк результата
print(ind_state.head())
```

```
   individuals       state
0       2570.0     Alabama
1       1434.0      Alaska
2       7259.0     Arizona
3       2280.0    Arkansas
4     109008.0  California
```

Выбор строк

Большая часть работы в области анализа данных заключается в поиске интересных фрагментов вашего набора данных. Одним из самых простых методов для этого является поиск подмножества строк, которые соответствуют определенным критериям. Это иногда известно как фильтрация строк или выбор строк.

Существует много способов выбора подмножества DataFrame, одним из наиболее распространенных является использование операторов отношений для возврата True или False для каждой строки, а затем передача этого в квадратных скобках.

```
dogs[dogs["height_cm"] > 60]
dogs[dogs["color"] == "tan"]
```

Вы можете фильтровать по нескольким условиям сразу, используя оператор "логическое и", &.

```
dogs[(dogs["height_cm"] > 60) & (dogs["color"] == "tan")]
```

`homelessness` доступен, а pandas загружен как pd.

Инструкции

Отфильтруйте `homelessness` для случаев, где количество `individuals` больше десяти тысяч, присвоив это `ind_gt_10k`. Посмотрите на результат вывода.

```
# Отфильтровать строки, где individuals больше 10000
ind_gt_10k = homelessness[homelessness["individuals"] > 10000]
```

```
# Посмотреть результат
print(ind_gt_10k)
```

```
    Unnamed: 0              region       state  individuals
family_members  \
4            4             Pacific  California     109008.0
20964.0
9            9      South Atlantic     Florida      21443.0
9587.0
32          32        Mid-Atlantic    New York      39827.0
52070.0
37          37             Pacific      Oregon      11139.0
3337.0
43          43  West South Central       Texas      19199.0
6111.0
47          47             Pacific  Washington      16424.0
5880.0

    state_pop
4    39461588
9    21244317
32   19530351
37    4181886
43   28628666
47    7523869
```

1. Отфильтруйте `homelessness` для случаев, где регион переписи США - `Mountain`, присвоив это `mountain_reg`. Посмотрите на результат вывода.

```
# Отфильтровать строки, где region равен Mountain
mountain_reg = homelessness[homelessness["region"] == "Mountain"]

# Посмотреть результат
print(mountain_reg)
```

```
    Unnamed: 0    region       state  individuals  family_members
state_pop
2            2  Mountain     Arizona       7259.0          2606.0
7158024
5            5  Mountain    Colorado       7607.0          3250.0
5691287
12          12  Mountain       Idaho       1297.0           715.0
1750536
26          26  Mountain     Montana        983.0           422.0
1060665
28          28  Mountain      Nevada       7058.0           486.0
3027341
31          31  Mountain  New Mexico       1949.0           602.0
2092741
```

```
44          44  Mountain        Utah       1904.0           972.0
3153550
50          50  Mountain     Wyoming        434.0           205.0
577601
```

1. Отфильтруйте `homelessness` для случаев, где количество `family_members` меньше тысячи и регион - "Pacific", присвоив это `fam_lt_1k_pac`. Посмотрите на результат вывода.

```
# Отфильтровать строки, где family_members меньше 1000
# и region равен Pacific
fam_lt_1k_pac = homelessness[(homelessness["family_members"] < 1000) &
(homelessness["region"] == "Pacific")]

# Посмотреть результат
print(fam_lt_1k_pac)
```

```
   Unnamed: 0   region   state  individuals  family_members  state_pop
1           1  Pacific  Alaska       1434.0           582.0     735139
```

Выбор подмножества строк по категориальным переменным

Выбор данных на основе категориальной переменной часто включает использование оператора "или" (|) для выбора строк из нескольких категорий. Это может быть утомительным, когда вам нужны все штаты в одном из трех разных регионов, например. Вместо этого используйте метод .isin(), который позволит решить эту проблему, написав одно условие вместо трех отдельных.

`colors = ["brown", "black", "tan"]` `condition = dogs["color"].isin(colors)` `dogs[condition]` `homelessness` is available and pandas is loaded as pd.

Инструкции

1. Отфильтруйте `homelessness` для случаев, где регион переписи США - "South Atlantic" или "Mid-Atlantic", присвоив это `south_mid_atlantic`. Посмотрите на результат вывода.

```
# Выбор строк в регионах South Atlantic или Mid-Atlantic
south_mid_atlantic = homelessness[homelessness["region"].isin(["South
Atlantic", "Mid-Atlantic"])]

# Посмотреть результат
print(south_mid_atlantic)
```

```
    Unnamed: 0          region                 state  individuals  \
7            7  South Atlantic              Delaware        708.0
8            8  South Atlantic  District of Columbia       3770.0
9            9  South Atlantic               Florida      21443.0
10          10  South Atlantic               Georgia       6943.0
20          20  South Atlantic              Maryland       4914.0
```

```
30          30    Mid-Atlantic            New Jersey       6048.0
32          32    Mid-Atlantic              New York      39827.0
33          33  South Atlantic        North Carolina       6451.0
38          38    Mid-Atlantic          Pennsylvania       8163.0
40          40  South Atlantic        South Carolina       3082.0
46          46  South Atlantic              Virginia       3928.0
48          48  South Atlantic         West Virginia       1021.0

    family_members  state_pop
7            374.0     965479
8           3134.0     701547
9           9587.0   21244317
10          2556.0   10511131
20          2230.0    6035802
30          3350.0    8886025
32         52070.0   19530351
33          2817.0   10381615
38          5349.0   12800922
40           851.0    5084156
46          2047.0    8501286
48           222.0    1804291
```

1. Отфильтруйте `homelessness` для случаев, где перепись США государство находится в списке штатов Мохаве, присвоив это `mojave_homelessness`. Посмотрите на результат вывода.

```
# Штаты Мохаве-дезерта
canu = ["California", "Arizona", "Nevada", "Utah"]

# Отфильтровать строки в штатах Мохаве-дезерта
mojave_homelessness = homelessness[homelessness["state"].isin(canu)]

# Посмотреть результат
print(mojave_homelessness)
```

```
    Unnamed: 0    region       state  individuals  family_members
state_pop
2            2  Mountain     Arizona       7259.0          2606.0
7158024
4            4   Pacific  California     109008.0         20964.0
39461588
28          28  Mountain      Nevada       7058.0           486.0
3027341
44          44  Mountain        Utah       1904.0           972.0
3153550
```

Добавление новых столбцов

Вы не ограничены только данными, которые у вас есть. Вместо этого вы можете добавлять новые столбцы в DataFrame. Это имеет много названий, таких как преобразование, мутация и инженерия признаков.

Вы можете создавать новые столбцы с нуля, но также обычно производить их из других столбцов, например, путем сложения столбцов или изменения их единиц измерения.

homelessness доступен, и pandas загружен как pd.

Инструкции

Добавьте новый столбец в `homelessness` с именем `total`, содержащий сумму столбцов `individuals` и `family_members`.

Добавьте еще один столбец в `homelessness` с именем `p_individuals`, содержащий долю бездомных людей в каждом штате, которые являются одиночками.

```
text{p_individuals} = \frac{\text{individuals}}{\text{individuals} + \
text{family_members}}

# Добавить столбец total как сумму individuals и family_members
homelessness["total"] = homelessness["individuals"] +
homelessness["family_members"]

# Добавить столбец p_individuals как долю от total, которые являются
individuals
homelessness['p_individuals'] = homelessness['individuals'] /
(homelessness['individuals'] + homelessness['family_members'])

# Посмотреть результат
print(homelessness.head())
```

```
   Unnamed: 0              region       state  individuals
family_members  \
0           0  East South Central     Alabama       2570.0
864.0
1           1             Pacific      Alaska       1434.0
582.0
2           2            Mountain     Arizona       7259.0
2606.0
3           3  West South Central    Arkansas       2280.0
432.0
4           4             Pacific  California     109008.0
20964.0

   state_pop     total  p_individuals
0    4887681    3434.0       0.748398
1     735139    2016.0       0.711310
2    7158024    9865.0       0.735834
```

```
3    3009733    2712.0       0.840708
4   39461588  129972.0       0.838704
```

Комбо-атака!

Вы видели четыре наиболее распространенных типа манипуляций с данными: сортировка строк, выбор столбцов, выбор подмножества строк и добавление новых столбцов. В реальном анализе данных вы можете смешивать и сочетать эти четыре манипуляции, чтобы ответить на множество вопросов.

В этом упражнении вы ответите на вопрос: "Какой штат имеет наибольшее количество бездомных людей на 10 000 человек в штате?" Совместите свои новые навыки pandas, чтобы выяснить это.

Инструкции

- Добавьте столбец в `homelessness`, `indiv_per_10k`, содержащий количество бездомных людей на десять тысяч человек в каждом штате.
- Выберите строки, где `indiv_per_10k` выше 20, присвоив это `high_homelessness`.
- Отсортируйте `high_homelessness` по убыванию `indiv_per_10k`, присвоив это `high_homelessness_srt`.
- Выберите только столбцы `state` и `indiv_per_10k` из `high_homelessness_srt` и сохраните результат. Посмотрите на результат.

```
# Создать столбец indiv_per_10k как бездомные лица на 10 000 человек в штате
homelessness["indiv_per_10k"] = 10000 * (homelessness["individuals"] / homelessness["state_pop"])

# Выбрать строки для indiv_per_10k больше 20
high_homelessness = homelessness[homelessness["indiv_per_10k"] > 20]

# Отсортировать high_homelessness по убыванию indiv_per_10k
high_homelessness_srt = high_homelessness.sort_values('indiv_per_10k', ascending=False)

# Из high_homelessness_srt выбрать столбцы state и indiv_per_10k
result = high_homelessness_srt[["state","indiv_per_10k"]]

# Посмотреть результат
print(result)
```

```
                  state  indiv_per_10k
8   District of Columbia      53.738381
11                Hawaii      29.079406
4             California      27.623825
37                Oregon      26.636307
28                Nevada      23.314189
```

```
47            Washington      21.829195
32              New York      20.392363
```