Лабораторная работа №29 (2 часа)

Тема работы: «Разработка, отладка и испытание программ создания мультииндексов»

1 Цель работы

Закрепить навык создания мультииндексов.

2 Задание

Примените мультииндексирование к объекту Series из лабораторной работы № 27

3 Оснащение работы

Задание по варианту, ЭВМ, среда разработки **Python 3.7, IDLE**.

4 Основные теоретические сведения

Иерархическое индексирование — это важная особенность pandas, поскольку она позволяет иметь несколько уровней индексов в одной оси. С ее помощью можно работать с данными в большом количестве измерений, по-прежнему используя для этого структуру данных из двух измерений.

Начнем с простого примера, создав Series с двумя массивами индексов — структуру с двумя уровнями.

```
>>> mser = pd.Series(np.random.rand(8),
...          index=[['white','white','white','blue','blue','red','red',
 'red'],
...                 ['up','down','right','up','down','up','down','left']])
>>> mser
white  up      0.661039
       down    0.512268
       right   0.639885
blue   up      0.081480
       down    0.408367
red    up      0.465264
       down    0.374153
       left    0.325975
dtype: float64
>>> mser.index
MultiIndex(levels=[['blue', 'red', 'white'], ['down', 'left', 'right', 'up']],
           labels=[[2, 2, 2, 0, 0, 1, 1, 1], [3, 0, 2, 3, 0, 3, 0, 1]])
```

За счет спецификации иерархического индексирования, выбор подмножеств значений в таком случае заметно упрощен. Можно выбрать значения для определенного значения первого индекса стандартным способом:

```
>>> mser['white']
up      0.661039
```

down 0.512268
right 0.639885
dtype: float64
Или же значения для конкретного значения во втором индекса — таким:
>>> mser[:,'up']
white 0.661039
blue 0.081480
red 0.465264
dtype: float64

Если необходимо конкретное значение, просто указываются оба индекса.
>>> mser['white','up']
0.66103875558038194
Иерархическое индексирование играет важную роль в изменении формы данных и групповых операциях, таких как сводные таблицы. Например, данные могут быть перестроены и использованы в объекте Dataframe с помощью функции unstack(). Она конвертирует Series с иерархическими индексами в простой Dataframe, где второй набор индексов превращается в новые колонки.
>>> mser.unstack()

	down	left	right	up
blue	0.408367	NaN	NaN	0.081480
red	0.374153	0.325975	NaN	0.465264
white	0.512268	NaN	0.639885	0.661039

Если необходимо выполнить обратную операцию — превратить Dataframe в Series, — используется функция stack().
>>> frame

	ball	pen	pencil	paper
red	0	1	2	3
blue	4	5	6	7
yellow	8	9	10	11
white	12	13	14	15

```
>>> frame.stack()
red    ball      0
       pen       1
       pencil    2
       paper     3
blue   ball      4
       pen       5
       pencil    6
       paper     7
yellow ball      8
       pen       9
       pencil    10
       paper     11
white  ball      12
       pen       13
       pencil    14
       paper     15
dtype: int32
```

В Dataframe можно определить иерархическое индексирование для строк и колонок. Для этого необходимо определить массив массивов для параметров index и columns.

```
>>> mframe = pd.DataFrame(np.random.randn(16).reshape(4,4),
...                       index=[['white','white','red','red'], ['up','down','up','down']],
...                       columns=[['pen','pen','paper','paper'],[1,2,1,2]])
>>> mframe
```

		pen		paper	
		1	2	1	2
white	up	1.562883	0.919727	-0.397509	-0.314159
	down	0.580848	1.124744	0.741454	-0.035455
red	up	-1.721348	0.989703	-1.454304	-0.249718
	down	-0.113246	-0.441528	-0.105028	0.285786

5 Порядок выполнения работы

1. Выделить ключевые моменты задачи.
2. Построить алгоритм решения задачи.
3. Запрограммировать полученный алгоритм.
4. Провести тестирование полученной программы.

6 Форма отчета о работе

Лабораторная работа № ___
Номер учебной группы _____
Фамилия, инициалы учащегося: _____
Дата выполнения работы: _____
Тема работы: _____
Цель работы: _____
Оснащение работы: _____
Результат выполнения работы: _____
_____

7. Контрольные вопросы и задания

1. Для чего используется мультииндексирование?
2. Опишите преимущества применения нескольких индексов на одной оси
3. Что такое иерархическое индексирование.

8 Рекомендуемая литература

Плас, Дж. В. Python для сложных задач. Наука о данных и машинное обучение / Дж.В. Плас. – СПб: Питер, 2018.

Прохоренок, Н.А. Python 3. Самое необходимое / Н.А Прохоренок, В.А. Дронов – СПб.: БВХ-Петербург, 2016.