

**Федеральное государственное образовательное
бюджетное учреждение
высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ
ФЕДЕРАЦИИ»**

(Финансовый университет)

**Факультет
информационных технологий и анализа больших данных
Кафедра «Бизнес-информатика»**

Домашнее задание № 6

«Решение задач паретооптимальным методом.»

Студенты группы БИ20-8:

Луканина Полина

Аверкин Никита

Филимонова Арина

Совин Владимир

Горшков Георгий

Киселева Евгения

Руководитель:

Аксенов Дмитрий Андреевич

Москва 2022

Оглавление

Оглавление	2
1. Постановка задачи (физическая модель)	4
2. Математическая модель	5
2.1. Поиск Пареттооптимального решения.....	5
2.2. Линейная свертка критериев.....	5
2.3. Метод идеальной точки.....	6
3. Алгоритмы решения задачи.....	7
3.1. Описание входных данных.	7
3.2. Описание алгоритма решения.....	8
3.3. Описание выходных данных.....	9
4. Вариант использования.....	11
5. Архитектура решения	13
5.1 Функции считывания информации	13
5.2 Функции обработки информации	20
5.3 Функции вывода информации	27
6. Тестирование	28
6.1. Тестирование Датасета №1:.....	28
6.1.1 Метод Python:.....	28
6.1.2 Метод Excel:.....	32
6.2. Тестирование Датасета №2:	33
6.2.1 Метод Python:.....	33
6.1.2 Метод Excel:.....	37
6.3. Тестирование Датасета №3:	38
6.3.1 Метод Python:.....	38

6.3.2 Метод Excel:.....	42
6.4. Тестирование Датасета №4:	43
6.4.1 Метод Python:.....	43
6.4.2 Метод Excel:.....	47
6.5. Тестирование Датасета №5:	48
6.5.1 Метод Python:.....	48
6.5.2 Метод Excel:.....	52
7. Заключение	53

1. Постановка задачи (физическая модель)

Сеть кофеен ООО «ВКУСНОЕ КОФЕ» столкнулось с экономическими трудностями после введения множества санкций на Россию. Одна из главных трудностей – это логистическая проблема.

Дело в том, что зерна поставляли из Испании, а одноразовые стаканчики и крышки из Нидерландов, но из-за закрытия воздушного пространства и наземных границ ЕС, поставлять эти материалы стало крайне затруднительно и не выгодно.

Наша задача найти оптимальный способ доставки этих ресурсов с двумя важными условиями, которые критичны для заказчика: срок доставки не более 7 дней (как это было до введенных санкций) и сохранить закупочную стоимость на уровне 5000 \$ за партию. Остальные параметры, такие как страна производства, производитель, качество и т.д. не критичны и могут быть изменены.

Для предоставления заказчику качественного решения мы провели исследование рынка и составили сводную таблицу по поставщикам:

Таблица 1. Результаты исследования

	Качество	Страна	Производительность в неделю	Стоимость в \$	Способ доставки	Срок доставки
Поставщик №1	Низкое	Индия	8 партий	4 000	Корабль	7 дней
Поставщик №2	Среднее	Китай	10 партий	3 500	Поезд	4 дня
Поставщик №3	Высокое	Бразилия	6 партий	4 900	Корабль	6 дней

Поставщи к №4	Среднее	Аргентин а	6 партий	4 600	Корабль	5 дней
Поставщи к №5	Высоко е	Турция	4 партии	5 000	Самолет	2 дня

2. Математическая модель

2.1. Поиск Пареттооптимального решения

Найти x , доставляющий экстремум критериям:

$$W_1 = f_1(x) \rightarrow \text{extr}$$

...

$$W_n = f_n(x) \rightarrow \text{extr}$$

при выполнении ограничений:

$$\varphi(x) \leq 0, x \geq 0$$

Соответственно, если достигается минимум — точка экстремума называется точкой минимума, а если максимум — точкой максимума.

Если $f(x) \rightarrow \max$, то критерий называется позитивным, в противном случае — негативным.

Решением задачи является паретооптимальное множество $x: \{x_i, \dots, x_n\}$, доставляющее необходимый экстремум критериям W

2.2. Линейная свертка критериев

Суть линейной свертки заключается в том, чтобы перейти от нескольких критериев к одному путем введения соответствующих коэффициентов к каждому такому критерию, за счет которых возможно будет «свернуть» все в один критерий.

В общем виде линейная свертка будет выглядеть так:

$$W = \sum_{i=1}^n a_i f_i(x) = a_1 f_1(x) + \dots + a_n f_n(x) \rightarrow \max$$

Где a_i — коэффициент значимости конкретного критерия

При условии, что все критерии $f_i(x)$ стремятся к максимуму

$$f_i(x) \rightarrow \max, \forall i = 1 \dots n$$

Если же некоторые критерии стремятся к минимуму, например, если $f_1(x) \rightarrow \max$, а $f_2(x) \rightarrow \min$, то сверстка видоизменяется и приобретает следующий вид:

$$W = \frac{f_1(x)}{f_2(x)} \rightarrow \max$$

$$W = a_1 f_1(x) - a_2 f_2(x) \rightarrow \max$$

2.3. Метод идеальной точки

Алгоритмом решения задачи методом идеальной точки будет являться следующий ход действий:

Выбирается точка, которая лучше остальных по всем критериям, причем она должна быть относительно приближена к ним. Затем из каждой точки строится расстояние до идеальной, соответственно поиск будет сводиться к нахождению гипотенузы через построения проекции для прямоугольного треугольника. Такой алгоритм используется для поиска сбалансированного решения, наиболее близкого к идеальному.

Математически интерпретироваться такая задача будет так:

Пусть имеется n -критериальная задача:

$$W_i = f_i(x) \rightarrow \max, \quad i = \overline{1, n}$$

Выберем «идеальную точку» - наилучшие значения каждого критерия:

$$(f_1^*, \dots, f_n^*)$$

Далее необходимо будет найти наименьшее расстояние до такой точки

Для этого введем глобальный критерий W , который минимизирует нормированное расстояние до идеальной точки:

$$W = \sqrt{\sum_{i=1}^n \left(\frac{f_i^* - f_i(x)}{\max f_i(x)} \right)^2} \rightarrow \min$$

Решая задачу, находим такое W , при котором все функции $f_i(x)$ ближе всего к своим идеальным значениям f_i^{\max}

Преимущество метода идеальной точки заключается в том, что присутствие эксперта, который анализирует какие-либо параметры, не требуется, поэтому данный способ может быть полностью автоматизирован.

3. Алгоритмы решения задачи

Алгоритмы решения реализованы с помощью программного кода в Python.

3.1. Описание входных данных.

Формат входных данных определяется тем, что программа принимает только CSV файл, который содержит в себе массивы данных.

	A	B	C	D	E	F	G	
1	№	цена	площадь	до метро	год	нравится	этаж	
2	1	270000	148	5	2004	7	2	
3	2	270000	150	5	2004	7	2	
4	3	200000	120	4	2001	8	2	
5	4	100000	56	1	1929	5	3	
6	5	270000	147	7	2004	7	2	
7	6	160000	55	12	2019	6	2	
8	7	120000	54	5	1932	4	4	
9	8	250000	65	1	1929	5	5	
10								

Рисунок 1. Пример входных данных в CSV файле

Введите путь к файлу:
 /Users/evgeniakiseleva/Desktop/5.3.csv

Рисунок 2. Пример заполнения пути к файлу

Вы импортировали:							
№	№	цена	площадь	до метро	год	нравится	этаж
1	1.0	270000.0	148.0	5.0	2004.0	7.0	2.0
2	2.0	270000.0	150.0	5.0	2004.0	7.0	2.0
3	3.0	200000.0	120.0	4.0	2001.0	8.0	2.0
4	4.0	100000.0	56.0	1.0	1929.0	5.0	3.0
5	5.0	270000.0	147.0	7.0	2004.0	7.0	2.0
6	6.0	160000.0	55.0	12.0	2019.0	6.0	2.0
7	7.0	120000.0	54.0	5.0	1932.0	4.0	4.0
8	8.0	250000.0	65.0	1.0	1929.0	5.0	5.0

Рисунок 3. Пример импортированных данных в программу

3.2. Описание алгоритма решения

После того как данные введены, программе необходимо преобразовать данные для дальнейшего использования.

Шаг 1: создаем список критериев, по которым осуществляется оптимизация.

Шаг 2: заполнение массива значений идеальной точки.

Шаг 3: рассчитываем дисперсию по выборке.

Шаг 4: нахождение набора точек поверхности Парето.

Шаг 5: построение точечной диаграммы для двухкритериальной оптимизации по Парето.

Шаг 6: построение лепестковой диаграммы для многокритериальной задачи.

Шаг 7: линейная свертка критериев, при помощи возвращения множителя, формирующий элемент линейной свертки.

Шаг 8: построение диаграммы линейной свертки критериев.

Шаг 9: линейная свертка для более двух критериев.

Шаг 10: расчет оптимального значения методом идеальной точки.

Шаг 11: построение лепестковой диаграммы.

3.3. Описание выходных данных

В конце программа выдает оптимальные значения по Паретто, по методу идеальной точки и линейной свертки.

Введите путь к файлу:

/Users/evgeniakiseleva/Desktop/tt6_4.csv

Вы импортировали:

	№	цена	площадь	до метро	год	нравится	этаж
№							
1	1.0	270000.0	148.0	5.0	2004.0	7.0	2.0
2	2.0	270000.0	150.0	5.0	2004.0	7.0	2.0
3	3.0	200000.0	120.0	4.0	2001.0	8.0	2.0
4	4.0	100000.0	56.0	1.0	1929.0	5.0	3.0
5	5.0	270000.0	147.0	7.0	2004.0	7.0	2.0
6	6.0	160000.0	55.0	12.0	2019.0	6.0	2.0
7	7.0	120000.0	54.0	5.0	1932.0	4.0	4.0
8	8.0	250000.0	65.0	1.0	1929.0	5.0	5.0

Сколько критериев оптимизировать? 6

Название критерия: цена

Направление ("макс" или "мин"): мин

Важность (от 1 до 10): 9

Название критерия: площадь

Направление ("макс" или "мин"): макс

Важность (от 1 до 10): 7

Название критерия: до метро

Направление ("макс" или "мин"): мин

Важность (от 1 до 10): 6

Название критерия: год

Направление ("макс" или "мин"): макс



Важность (от 1 до 10): 2

Название критерия: нравится

Направление ("макс" или "мин"): макс

Важность (от 1 до 10): 10

Рисунок 4. Пример выходных данных

 Название критерия: этаж
 Направление ("макс" или "мин"): мин
 Важность (от 1 до 10): 5

Есть идеальная точка? ('да' или 'нет'): да
 Значение по столбцу цена: 50000
 Значение по столбцу площадь: 200
 Значение по столбцу до метро: 0.5
 Значение по столбцу год: 2022
 Значение по столбцу нравится: 10
 Значение по столбцу этаж: 1

Поиск Паретооптимального решения

Точки поверхности Парето:

	№	цена	площадь	до метро	год	нравится	этаж
№							
2	2.0	270000.0	150.0	5.0	2004.0	7.0	2.0
3	3.0	200000.0	120.0	4.0	2001.0	8.0	2.0
4	4.0	100000.0	56.0	1.0	1929.0	5.0	3.0

Нормированные данные:

	№	цена	площадь	до метро	год	нравится	этаж
№							
1	0.125	1.000000	0.986667	0.416667	0.992571	0.875	0.4
2	0.250	1.000000	1.000000	0.416667	0.992571	0.875	0.4
3	0.375	0.740741	0.800000	0.333333	0.991085	1.000	0.4
4	0.500	0.370370	0.373333	0.083333	0.955423	0.625	0.6
5	0.625	1.000000	0.980000	0.583333	0.992571	0.875	0.4
6	0.750	0.592593	0.366667	1.000000	1.000000	0.750	0.4
7	0.875	0.444444	0.360000	0.416667	0.956909	0.500	0.8
8	1.000	0.925926	0.433333	0.083333	0.955423	0.625	1.0

Рисунок 5. Поиск Паретооптимального решения

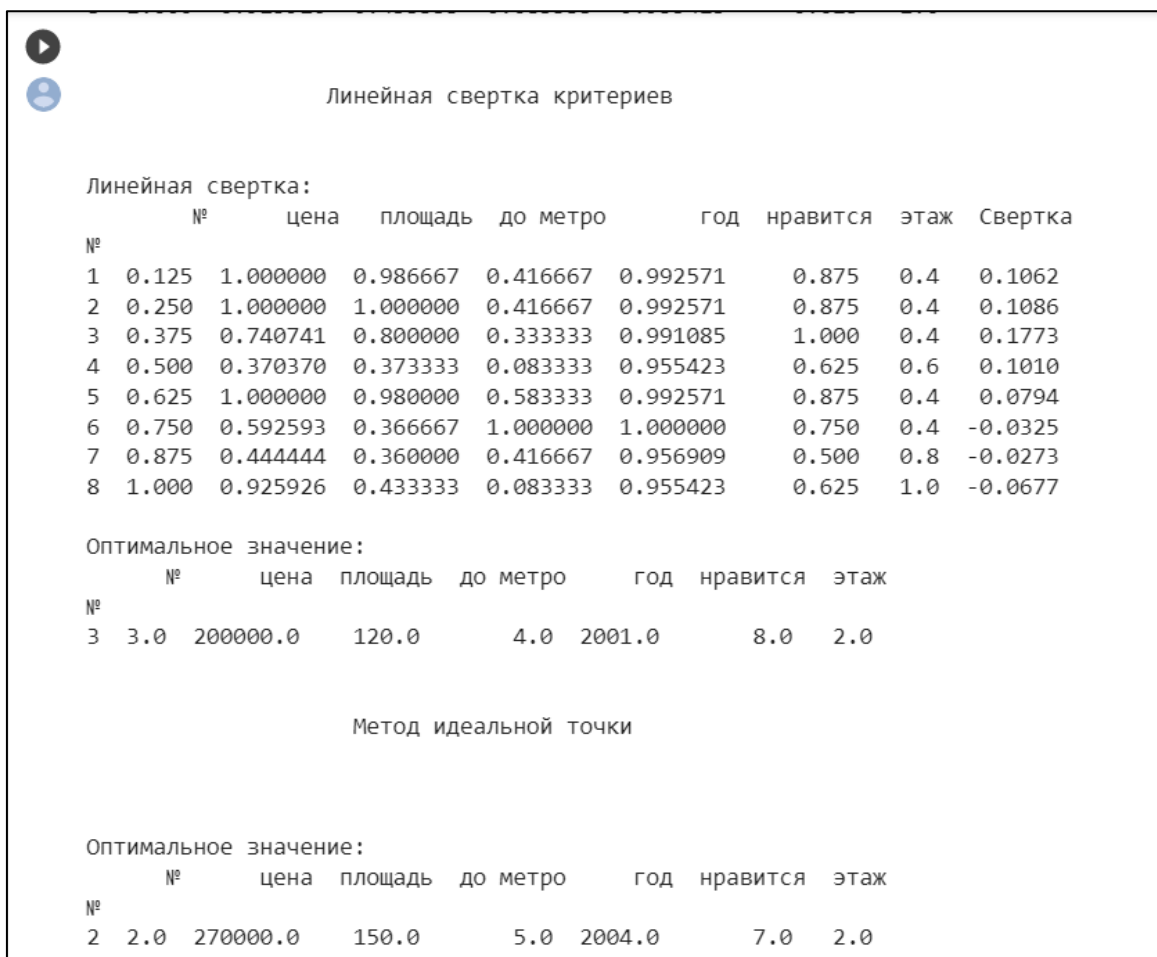


Рисунок 6. Линейная свертка критериев и метод идеальной точки

4. Вариант использования

Существует единственный вариант использования кода – это загрузка файла. Данный вариант использования включает в себя ввод данных с помощью файла csv. Для того, чтобы ввести путь к файлу необходимо запустить программу.

После этого появляется окно, в котором вводим путь к csv файлу. Например:

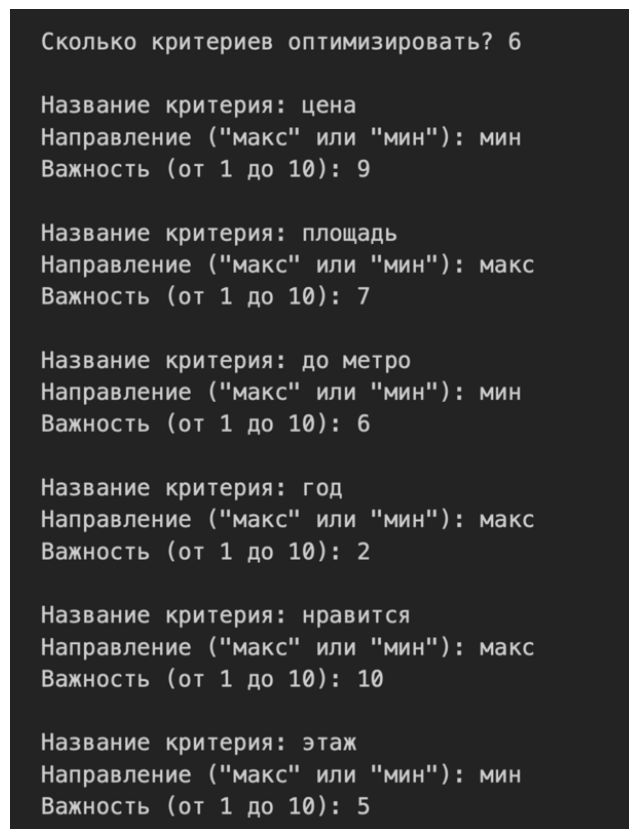
Введите путь к файлу:
/Users/evgeniakiseleva/Desktop/1dataset.csv

Рисунок 7. Выбор местонахождения csv файла

2. Определите количество критериев, по которым необходимо осуществить оптимизацию (более одного).

3. Для каждого оптимизационного критерия последовательно введите его название (им является имя столбца импортируемой таблицы) и направление оптимизации (укажите “Макс”, если следует оптимизировать критерий к максимуму, и “Мин”, если к минимуму).

4. Далее для каждого критерия будет предложено ввести весовой коэффициент (важность). Он является необязательным, поэтому его допустимо пропустить (просто нажмите enter, не заполняя ячейку).



```
Сколько критериев оптимизировать? 6

Название критерия: цена
Направление ("макс" или "мин"): мин
Важность (от 1 до 10): 9

Название критерия: площадь
Направление ("макс" или "мин"): макс
Важность (от 1 до 10): 7

Название критерия: до метро
Направление ("макс" или "мин"): мин
Важность (от 1 до 10): 6

Название критерия: год
Направление ("макс" или "мин"): макс
Важность (от 1 до 10): 2

Название критерия: нравится
Направление ("макс" или "мин"): макс
Важность (от 1 до 10): 10

Название критерия: этаж
Направление ("макс" или "мин"): мин
Важность (от 1 до 10): 5
```

Рисунок 8. Пример

5. После программа предложит ввести идеальную точку (для метода оптимизации по идеальной точке). Это также является необязательным, поэтому укажите “нет”, если вводить ее не будете, и “да”, если данные по ней имеются.

6. При вводе “нет” программа выведет результат оптимизации с имеющимися данными без идеальной точки, при “да” – пользователю следует последовательно ввести ее весовые коэффициенты для каждого критерия.

```
Есть идеальная точка? ('да' или 'нет'): да
Значение по столбцу цена: 50000
Значение по столбцу площадь: 200
Значение по столбцу до метро: 0.5
Значение по столбцу год: 2022
Значение по столбцу нравится: 10
Значение по столбцу этаж: 1
```

Рисунок 9. Пример

5. Архитектура решения

Для решения задачи использовались методы (функции), которые можно разделить на 3 принципиальных кода.

5.1 Функции считывания информации

Функция `import_data`:

Функция `import_csv`:

Входные данные:

- Нет входных данных.

Выходные данные:

- `A` – массив данных в формате `dataframe`.

Переменные, затрагиваемые в ходе работы:

- `csv` – пустой список.
- `A` – массив данных в формате `dataframe`.
- `x` – переменная, затрагиваемая в ходе работы.
- `columns` – список значений столбца.

- `index` – список значений индекса.

Функция `input_num_criteria`:

Входные данные:

- Нет входных данных.

Выходные данные:

- Нет выходных данных.

Переменные, затрагиваемые в ходе работы:

- `number` – количество критериев, которых нужно оптимизировать.
- `A` – массив данных в формате `dataframe`.

Функция `input_name_criteria`:

Входные данные:

- Нет входных данных.

Выходные данные:

- Нет выходных данных.

Переменные, затрагиваемые в ходе работы:

- `name` – название критериев.
- `A` – массив данных в формате `dataframe`.

Функция `input_direction_of_optimiz`:

Входные данные:

- Нет входных данных.

Выходные данные:

- Нет выходных данных.

Переменные, затрагиваемые в ходе работы:

- A – массив данных в формате dataframe.
- direction – значение направления.

Функция input_weight_coeff:

Входные данные:

- Нет входных данных.

Выходные данные:

- Нет выходных данных.

Переменные, затрагиваемые в ходе работы:

- weight – значение важности.
- A – массив данных в формате dataframe.

Функция optimiz_criteria:

Входные данные:

- Нет входных данных.

Выходные данные:

- Нет выходных данных.

Переменные, затрагиваемые в ходе работы:

- weight – значение важности.

- `list_optimiz_criteria` – список критериев.
- `A` – массив данных в формате `dataframe`.
- `number` – количество критериев, которых нужно оптимизировать.

Функция `input_ideal_point`:

Функция `input_value`:

Входные данные:

- Нет входных данных.

Выходные данные:

- Нет выходных данных.

Переменные, затрагиваемые в ходе работы:

- `value` – значения по столбцу массивов данных.
- `data` – переменная, используемая в функции.
- `i` – переменная, используемая в функции.
- `inpt` – присутствие идеальной точки.
- `ideal_point` – список значений идеальной точки.


```

def input_data():

    """Формирование исходных данных
    Функция считывает csv файл и формирует массивы с введенными пользователем данными,
    включающими имена критериев, по которым осуществляется оптимизация, направления оптимизации
    для каждого критерия и весовые коэффициенты."""

def import_csv():
    """Метод считывает данные из CSV файла и преобразует их в DataFrame."""
    try:
        with open(input('Введите путь к файлу: \n'), encoding='utf-8-sig') as data_file:
            csv = []
            A = []
            for line in data_file:
                csv = line.strip().split(';')
                for n, x in enumerate(csv):
                    if ',' in x:
                        csv = list(map(lambda x: float(x.replace(',', '.'))))
                    else:
                        try:
                            csv[n] = float(x)
                        except:
                            pass

                A.append(csv)
            A = pd.DataFrame(A[1:],
                              columns=A[0],
                              index=[i for i in range(1, len(A))])
            A.index.name = '№'
            print('\nВы импортировали: \n', A)
        return A
    except FileNotFoundError:
        print('Искомый файл не найден! Попробуйте еще раз: \n')
        return import_csv()

```

Рисунок 10. Часть программного кода, отвечающего за функцию считывания информации

```

def input_num_criteria(A):
    """Ввод пользователем количества критериев, по которым осуществляется оптимизация."""
    try:
        number = int(input('\nСколько критериев оптимизировать? '))
        if number < 2:
            print('Количество критериев должно быть не менее 2-х!')
            return input_num_criteria(A)
        else:
            return number
    except:
        print('Количество должно быть числом!')
        return input_num_criteria(A)

def input_name_criteria(A):
    """Ввод пользователем имен критериев, по которым осуществляется оптимизация."""
    name = input('\nНазвание критерия: ')
    if name not in A.columns:
        print('Такого критерия нет!\nПроверьте правильность написания и попробуйте еще раз')
        return input_name_criteria(A)
    return name

def input_direction_of_optimiz(A):
    """Ввод пользователем направлений оптимизации для каждого критерия."""
    direction = input('Направление ("макс" или "мин"): ')
    direction = direction.replace(' ', '')
    direction = direction.upper()
    if direction not in ('МАКС', 'МИН'):
        print('Ошибка ввода!\nВведите значение "макс" или "мин"')
        return input_direction_of_optimiz(A)
    return direction

```

Рисунок 11. Часть программного кода, отвечающего за функцию считывания информации

```

def input_weight_coeff(A):
    """Ввод пользователем весовых коэффициентов характеристик."""
    try:
        weight = input('Важность (от 1 до 10): ')
        if weight == '':
            return None
        else:
            if ',' in weight:
                weight = weight.replace(',', '.')
            return float(weight)
    except:
        print('Вес должен быть числом! Попробуйте еще раз')
        return input_weight_coeff(A)

def optimiz_criteria(A):
    """Создание списка критериев, по которым осуществляется оптимизация."""
    list_optimiz_criteria = []
    number = input_num_criteria(A)
    lin_conv = True
    for i in range(number):
        name_criteria = input_name_criteria(A)
        direction_of_optimiz = input_direction_of_optimiz(A)
        weight_coeff = input_weight_coeff(A)
        list_optimiz_criteria.append([name_criteria, direction_of_optimiz, weight_coeff])
        if weight_coeff is None:
            lin_conv = None
    return number, list_optimiz_criteria, lin_conv

```

Рисунок 12. Часть программного кода, отвечающего за функцию считывания информации

```

def input_ideal_point(data):
    """Ввод пользователем идеальной точки."""

    def input_value(data, i):
        """Функция для заполнения массива значений идеальной точки."""
        try:
            value = float(input(f'Значение по столбцу {i}: '))
            return value
        except:
            print('Значение должно быть числом! Попробуйте еще раз')
            return input_value(data, i)

    inpt = input("\nЕсть идеальная точка? ('да' или 'нет'): ")
    inpt = inpt.replace(' ', '')
    inpt = inpt.upper()
    if inpt not in ('ДА', 'НЕТ'):
        print('Введите "да" или "нет"!')
        return input_ideal_point()
    else:
        if inpt == 'ДА':
            ideal_point = []
            for i in data:
                if i != '№':
                    value = input_value(data, i)
                    ideal_point.append(value)
            else:
                ideal_point = None
        return ideal_point

data = import_csv()
number_optimize_criteria, list_optimiz_criteria, lin_conv = optimiz_criteria(data)
ideal_point = input_ideal_point(data)
return data, number_optimize_criteria, list_optimiz_criteria, ideal_point, lin_conv

```

Рисунок 13. Часть программного кода, отвечающего за функцию считывания информации

5.2 Функции обработки информации

После того, как вы введете все необходимые данные, программа их получит и начнет первичную обработку.

Поскольку ввод данных осуществляется с помощью файла csv, то для начала файл необходимо открыть и прочитать программе, а затем разделить данные и также занести их в пустой словарь. Более того, программа образует список с названиями колонок массива данных, а также список с названиями строк.

Функция `graph_pareto` строит точечную диаграмму для двухкритериальной оптимизации по Парето и лепестковую диаграмму для многокритериальной задачи, более того, она включает функцию `opt`, `pareto_surface`, `graph_for_two_criteria`. В функции `opt` происходит проверка точки на вхождение в набор оптимальных значений по Парето. В `pareto_surface` находятся наборы точек поверхности Парето, а в функции `graph_for_two_criteria` строятся точечная диаграмма для двухкритериальной оптимизации по Парето.

В функции `data_normalization` происходит нормирование исходных данных с помощью цикла `for in i`.

Функция `linear_convolution` реализует оптимальное значение и набор точек поверхности Парето с помощью линейной свертки, а также строит диаграмму линейной свертки критериев. Также она включает в себя следующие функции: `func`, которая возвращает множитель, формирующий элемент линейной свертки; `convolution_matrix`, которая возвращает таблицу линейной свертки для двух оптимизационных критериев; `output`, которая выводит оптимальные значения на экран (для оптимизации двух критериев); `graph_linear_convolution`, которая строит диаграмму линейной свертки критериев (для оптимизации двух критериев); `convolution_more_2_cr`, которая возвращает таблицу линейной свертки и оптимальное значение для более двух оптимизационных критериев.

В функции `meth_ideal_point` рассчитывается оптимальное значение методом идеальной точки и формирует лепестковую диаграмму указанием Паретооптимального множества решений.

Функция `graph_pareto`:

Входные данные:

- Нет входных данных.

Выходные данные:

- data – переменная, затрагиваемая в функции.
- criteria – значение критерия.
- Pareto_surface – точки поверхности Парето.

Переменные, затрагиваемые в ходе работы:

- data - переменная, затрагиваемая в функции.
- criteria – значение критерия.
- i – переменная, затрагиваемая в функции.
- j - переменная, затрагиваемая в функции.
- Pareto_surface – точки поверхности Парето.
- optimal_point_x – список набора точек по x.
- optimal_point_y – список набора точек по y.
- x – значение критерия по x.
- y - значение критерия по y.

Функция `linear_convolution`:

Входные данные:

- Нет входных данных.

Выходные данные:

- Нет выходных данных.

Переменные, затрагиваемые в ходе работы:

- data - переменная, принимающая значения в функции.
- criteria – значение критерия.

- `i_data` – переменная, принимающая значения в функции.
- `number` – количество критериев.
- `cr` – значение критерия.
- `name_columns` – название столбцов.
- `row` – список значений.
- `lin_conv` – список значений.

Функция `meth_ideal_point`:

Входные данные:

- Нет входных данных.

Выходные данные:

- Нет выходных данных.

Переменные, затрагиваемые в ходе работы:

- `data` – переменная, принимающая значения в функции.
- `criteria` – значение критерия.
- `i_data` – переменная, принимающая значения в функции.
- `ideal_point` – значение идеальной точки.
- `row` – список значений.
- `norm_meth_ideal_point` – список значений.
- `distance` – список значений.

```
def graph_pareto(data, number_optimize_criteria, criteria):

    """Построение графиков

    Метод строит точечную диаграмму для двухкритериальной оптимизации по Парето
    и липестковую диаграмму для многокритериальной задачи.

    """

    def opt(i, j, criteria):
        """Метод проверяет точку на вхождение в набор оптимальных по Парето."""
        if criteria == 'МИН':
            if j <= i:
                return False
            else:
                return True
        elif criteria == 'МАКС':
            if j >= i:
                return False
            else:
                return True

    def pareto_surface(data, criteria):
        """Метод находит набор точек поверхности Парето."""
        optimal_point_x = []
        optimal_point_y = []
        index_optimal_point = []
        k = True
        for n, i in enumerate(data[criteria[0][0]], start=1):
            for num in range(1, len(data[criteria[0][0]]) + 1):
                if (i == data[criteria[0][0]][num]) and (data[criteria[1][0]][n] == data[criteria[1][0]][num]):
                    continue
                optimal_x = opt(i, data[criteria[0][0]][num], criteria[0][1])
```

Рисунок 14. Часть кода, отвечающая за функцию

```
                optimal_x = opt(i, data[criteria[0][0]][num], criteria[0][1])
                optimal_y = opt(data[criteria[1][0]][n], data[criteria[1][0]][num], criteria[1][1])
                if (optimal_x == False) and (optimal_y == False):
                    k = False
                    break
            if k == True:
                optimal_point_x.append(i)
                optimal_point_y.append(data[criteria[1][0]][n])
                index_optimal_point.append(n)
            k = True
        row = list(set([i for i in range(1, len(data) + 1)] ^ set(index_optimal_point)))
        return data.drop(row, axis=0)

def graph_for_two_criteria(data, criteria, Pareto_surface):
    """Метод строит точечную диаграмму для двухкритериальной оптимизации по Парето."""
    plt.figure(figsize=(14, 8))
    x = data[criteria[0][0]]
    y = data[criteria[1][0]]
    plt.plot(x, y, 'o',
             color='#1e5356',
             label="Позиция")

    for i in data[data.columns[0]]:
        plt.annotate(int(i), (data[criteria[0][0]][i], data[criteria[1][0]][i]), fontsize=12)

    Pareto_surface = Pareto_surface.sort_values(by=criteria[0][0])
    plt.plot(Pareto_surface[criteria[0][0]],
             Pareto_surface[criteria[1][0]],
             'o-',
             color='#F97306',
             label="Поверхность Парето")
```

Рисунок 15. Часть кода, отвечающая за функцию


```

plt.title('Оптимизация по Парето')
plt.xlabel(criteria[0][0])
plt.ylabel(criteria[1][0])
plt.grid()
plt.legend(fontsize=12,
           bbox_to_anchor=(1, 1))
plt.show()

Pareto_surface = pareto_surface(data, criteria)
print('\n\nТочки поверхности Парето:\n', Pareto_surface)

if number_optimize_criteria == 2:
    graph_for_two_criteria(data, criteria, Pareto_surface)

def data_normalization(data):
    """Функция нормирует исходные данные."""
    for col in data[1:]:
        mean = data[col].max()
        for i in data.index:
            data[col][i] = float(data[col][i] / mean)
    return data

def linear_convolution(i_data, data, number, criteria):
    """линейная свертки критериев

    Функция, используя линейную свертку, возвращает оптимальное значение,
    диаграмму линейной свертки критериев и набор точек поверхности Парето.

    """

```

Рисунок 16. Часть кода, отвечающая за функцию

```

def fun(cr):
    """Функция возвращает множитель, формирующий элемент линейной свертки."""
    if cr == 'МИН':
        return -1
    elif cr == 'МАКС':
        return 1

def convolution_matrix(data, criteria):
    """Функция возвращает таблицу линейной свертки для двух оптимизационных критериев."""
    x = [i/10 for i in range(0, 11)]
    y = [i/10 for i in range(10, -1, -1)]
    name_columns = ['/'.join(map(str, lst)) for lst in list(zip(x, y))]

    row = []
    lin_conv = []
    for i in data.index:
        for n in range(len(x)):
            value = fun(criteria[0][1]) * x[n] * data[criteria[0][0]][i] + fun(criteria[1][1]) * y[n] * data[criteria[1][0]][i]
            row.append(value)
        lin_conv.append(row)
        row = []

    lin_conv = pd.DataFrame(lin_conv,
                           columns=name_columns,
                           index=[i for i in range(1, len(lin_conv) + 1)])
    return lin_conv, name_columns

def output(data, lin_conv):
    """Функция выводит оптимальные значения на экран (для оптимизации двух критериев)."""
    print('\nОптимальные значения:\n')
    for i in lin_conv:

```

Рисунок 17. Часть кода, отвечающая за функцию

```

        for n, value in enumerate(lin_conv[i], start=1):
            if value == lin_conv[i].max():
                print(f'При важности {i} оптимальный выбор под номером - {n}')

def graph_linear_convolution(lin_conv, name_columns):
    """Функция строит диаграмму линейной свертки критериев (для оптимизации двух критериев)."""
    lin_conv = lin_conv.T
    lin_conv['name'] = name_columns

    lin_conv.plot(x='name',
                  kind='bar',
                  stacked=False,
                  title="Линейная свертка критериев",
                  width=0.7,
                  xlabel='Весовые коэффициенты',
                  ylabel='Значение',
                  figsize=(15,8)
                  )

    del lin_conv['name']
    lin_conv = lin_conv.T
    x = np.arange(0, 11)
    answr = [max(lin_conv[i]) for i in lin_conv]
    plt.plot(x, answr, label = 'Оптимальные значения', c = 'm')

    plt.legend(fontsize=10,
               ncol=1,
               bbox_to_anchor=(1, 1)
               )

    plt.grid()

def convolution_more_2_cr(i_data, data, criteria):

```

Рисунок 18. Часть кода, отвечающая за функцию

```

    """Линейная свертка (более 2-х критериев)

    Функция возвращает таблицу линейной свертки
    и оптимальное значение для более двух оптимизационных критериев.

    """

    criteria_list = []
    for i in range(len(criteria)):
        criteria_list.append(criteria[i][2])

    for i in range(len(criteria)):
        value = fun(criteria[i][1]) * criteria[i][2] / sum(criteria_list)
        criteria[i].append(value)

    norm_criteria_list = []
    for i in data.index:
        value = 0
        for j in range(len(criteria)):
            value += criteria[j][3] * data[criteria[j][0]][i]
        norm_criteria_list.append(value)

    data['Свертка'] = list(map(lambda x: round(x, 4), norm_criteria_list))
    print('Линейная свертка:\n', data)
    print('\nОптимальное значение:\n', i_data.iloc[[norm_criteria_list.index(max(norm_criteria_list))]])

    if number == 2:
        lin_conv, name_columns = convolution_matrix(data, criteria)
        print('Линейная свертка: \n', lin_conv)
        output(data, lin_conv)
        graph_linear_convolution(lin_conv, name_columns)
    else:
        convolution_more_2_cr(i_data, data, criteria)

```

Рисунок 19. Часть кода, отвечающая за функцию

```

def meth_ideal_point(i_data, data, criteria, ideal_point):

    """Метод идеальной точки

    Функция рассчитывает оптимальное значение методом идеальной точки
    и формирует лепестковую диаграмму указанием Паретооптимального множества решений.

    """

    norm_meth_ideal_point = []
    row = []
    for i in i_data.index:
        if i != '№':
            for num, j in enumerate(i_data, start=0):
                if j != '№':
                    if criteria[num-1][1] == 'МИН':
                        value = 1 / ((ideal_point[num-1] - i_data[j][i]) / ideal_point[num-1])
                    else:
                        value = (ideal_point[num - 1] - i_data[j][i]) / ideal_point[num - 1]
                    row.append(value)
            norm_meth_ideal_point.append(row)
            row = []
    distance = []
    for row in norm_meth_ideal_point:
        value = 0
        for i in row:
            value += i**2
        distance.append(value)

    print('Оптимальное значение: \n', i_data.iloc[[distance.index(min(distance))]])

data, number_optimize_criteria, list_optimiz_criteria, ideal_point, lin_conv = input_data()

```

Рисунок 20. Часть кода, отвечающая за функцию

5.3 Функции вывода информации

Метод вывода информации

Что делает: осуществляет вывод необходимой информации

В данном методе осуществляется непосредственно вызов функций с помощью метода print ().

```
# Парето
print('\n\n', 'Поиск Паретооптимального решения'.center(60))
graph_pareto(data, number_optimize_criteria, list_optimiz_criteria)

# Нормирование данных
data_norm = data_normalization(data.copy())
print('\nНормированные данные: \n', data_norm)

# Линейная оптимизация
print('\n\n', 'Линейная свертка критериев'.center(60), '\n\n')
if lin_conv is None:
    print('Линейная оптимизация недоступна, поскольку не были введены весовые коэффициенты или данных недостаточно.')
else:
    linear_convolution(data, data_norm, number_optimize_criteria, list_optimiz_criteria)

# Метод идеальной точки
print('\n\n', 'Метод идеальной точки'.center(60), '\n\n')
if ideal_point is not None:
    meth_ideal_point(data, data_norm, list_optimiz_criteria, ideal_point)
else:
    print('Оптимизация методом идеальной точки недоступна, поскольку идеальная точка не была введена.')
```

Рисунок 21. Часть кода, отвечающая за функцию

6. Тестирование

Проведём тестирование нашей программы и сравним полученные показатели,

чтобы сделать вывод о предпочтительном варианте использования нашей программы

или Excel под условия заказчика.

6.1. Тестирование Датасета №1:

6.1.1 Метод Python:

Импортируем датасет в Python, выбираем количество оптимизируемых критериев, названия критериев, направление и важность их оптимизации. Уточняем наличие идеальной точки, вводим идеальные параметры.

Введите путь к файлу:

/Users/aniki/Desktop/1dataset.csv

Вы импортировали:

	№	цена	площадь	до метро	год	нравится	этаж
№							
1	1.0	270000.0	148.0	5.0	2004.0	7.0	2.0
2	2.0	270000.0	213.0	5.0	2004.0	7.0	1.0
3	3.0	678000.0	120.0	12.0	2001.0	8.0	2.0
4	4.0	200000.0	12.0	1.0	1953.0	5.0	3.0
5	5.0	270000.0	147.0	7.0	2001.0	6.0	2.0
6	6.0	160000.0	32.0	12.0	2019.0	6.0	3.0
7	7.0	123000.0	423.0	23.0	1932.0	4.0	4.0
8	8.0	250000.0	76.0	1.0	1929.0	5.0	5.0

Сколько критериев оптимизировать? 2

Название критерия: цена

Направление ("макс" или "мин"): мин

Важность (от 1 до 10): 5

Название критерия: площадь

Направление ("макс" или "мин"): макс

Важность (от 1 до 10): 4

Есть идеальная точка? ('да' или 'нет'): да

Значение по столбцу цена: 100000

Значение по столбцу площадь: 100

Значение по столбцу до метро: 3

Значение по столбцу год: 2000

Значение по столбцу нравится: 6

Значение по столбцу этаж: 2

Рисунок 22. импорт

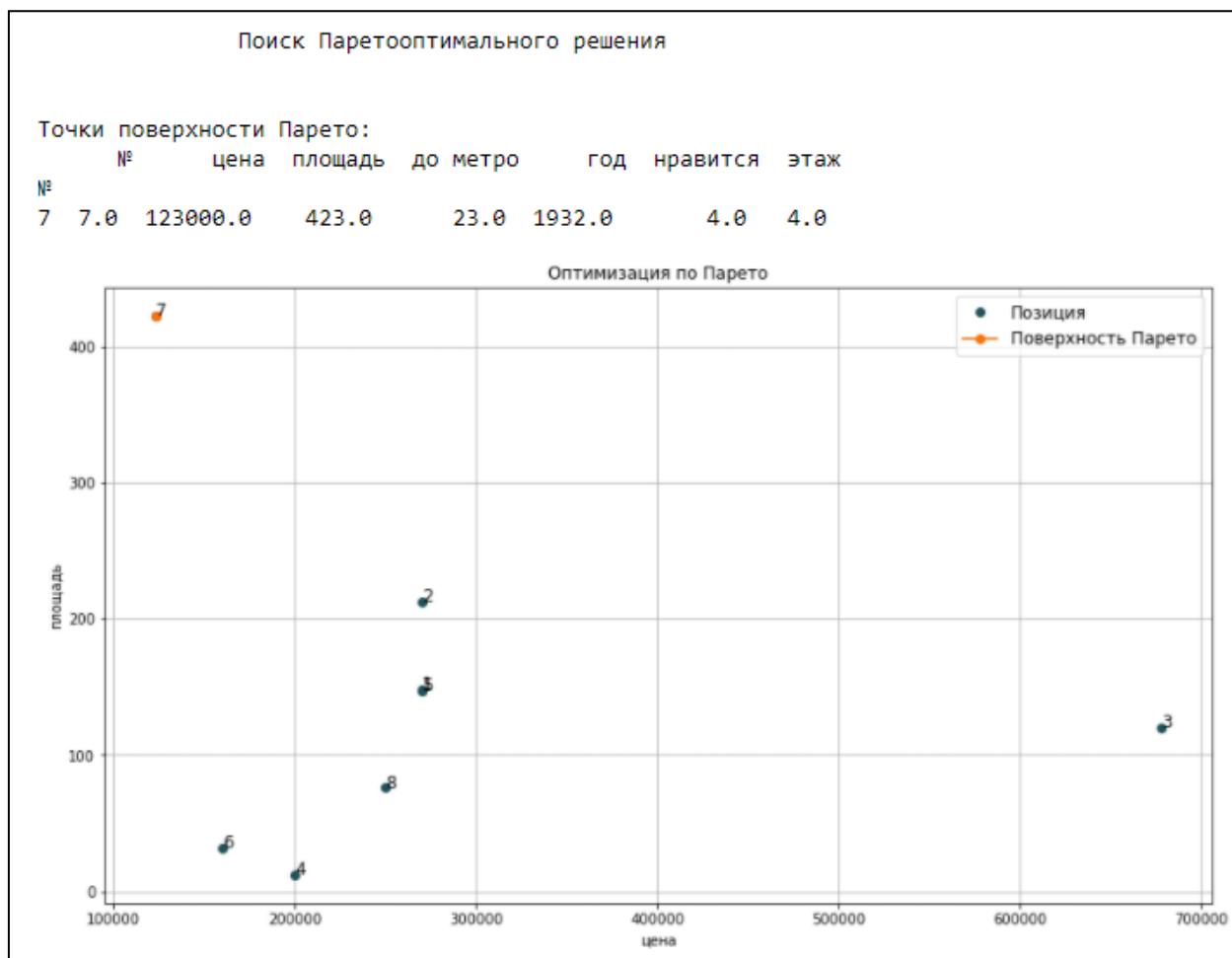


Рисунок 23. паретооптимальное решение

Нормированные данные:

№	цена	площадь	до метро	год	нравится	этаж
1	0.125	0.398230	0.349882	0.217391	0.992571	0.875
2	0.250	0.398230	0.503546	0.217391	0.992571	0.875
3	0.375	1.000000	0.283688	0.521739	0.991085	1.000
4	0.500	0.294985	0.028369	0.043478	0.967311	0.625
5	0.625	0.398230	0.347518	0.304348	0.991085	0.750
6	0.750	0.235988	0.075650	0.521739	1.000000	0.750
7	0.875	0.181416	1.000000	1.000000	0.956909	0.500
8	1.000	0.368732	0.179669	0.043478	0.955423	0.625

Рисунок 24. нормированные данные

Линейная свертка критериев

Линейная свертка:

	0.0/1.0	0.1/0.9	0.2/0.8	0.3/0.7	0.4/0.6	0.5/0.5	0.6/0.4	\
1	0.349882	0.275071	0.200259	0.125448	0.050637	-0.024174	-0.098985	
2	0.503546	0.413368	0.323191	0.233013	0.142836	0.052658	-0.037520	
3	0.283688	0.155319	0.026950	-0.101418	-0.229787	-0.358156	-0.486525	
4	0.028369	-0.003967	-0.036302	-0.068637	-0.100973	-0.133308	-0.165644	
5	0.347518	0.272943	0.198368	0.123793	0.049219	-0.025356	-0.099931	
6	0.075650	0.044486	0.013322	-0.017841	-0.049005	-0.080169	-0.111333	
7	1.000000	0.881858	0.763717	0.645575	0.527434	0.409292	0.291150	
8	0.179669	0.124829	0.069989	0.015149	-0.039691	-0.094531	-0.149371	

	0.7/0.3	0.8/0.2	0.9/0.1	1.0/0.0
1	-0.173797	-0.248608	-0.323419	-0.398230
2	-0.127697	-0.217875	-0.308052	-0.398230
3	-0.614894	-0.743262	-0.871631	-1.000000
4	-0.197979	-0.230314	-0.262650	-0.294985
5	-0.174506	-0.249081	-0.323655	-0.398230
6	-0.142497	-0.173661	-0.204824	-0.235988
7	0.173009	0.054867	-0.063274	-0.181416
8	-0.204211	-0.259051	-0.313892	-0.368732

Оптимальные значения:

При важности 0.0/1.0 оптимальный выбор под номером - 7
 При важности 0.1/0.9 оптимальный выбор под номером - 7
 При важности 0.2/0.8 оптимальный выбор под номером - 7
 При важности 0.3/0.7 оптимальный выбор под номером - 7
 При важности 0.4/0.6 оптимальный выбор под номером - 7
 При важности 0.5/0.5 оптимальный выбор под номером - 7
 При важности 0.6/0.4 оптимальный выбор под номером - 7
 При важности 0.7/0.3 оптимальный выбор под номером - 7
 При важности 0.8/0.2 оптимальный выбор под номером - 7
 При важности 0.9/0.1 оптимальный выбор под номером - 7
 При важности 1.0/0.0 оптимальный выбор под номером - 7

Рисунок 25. линейная свёрстка

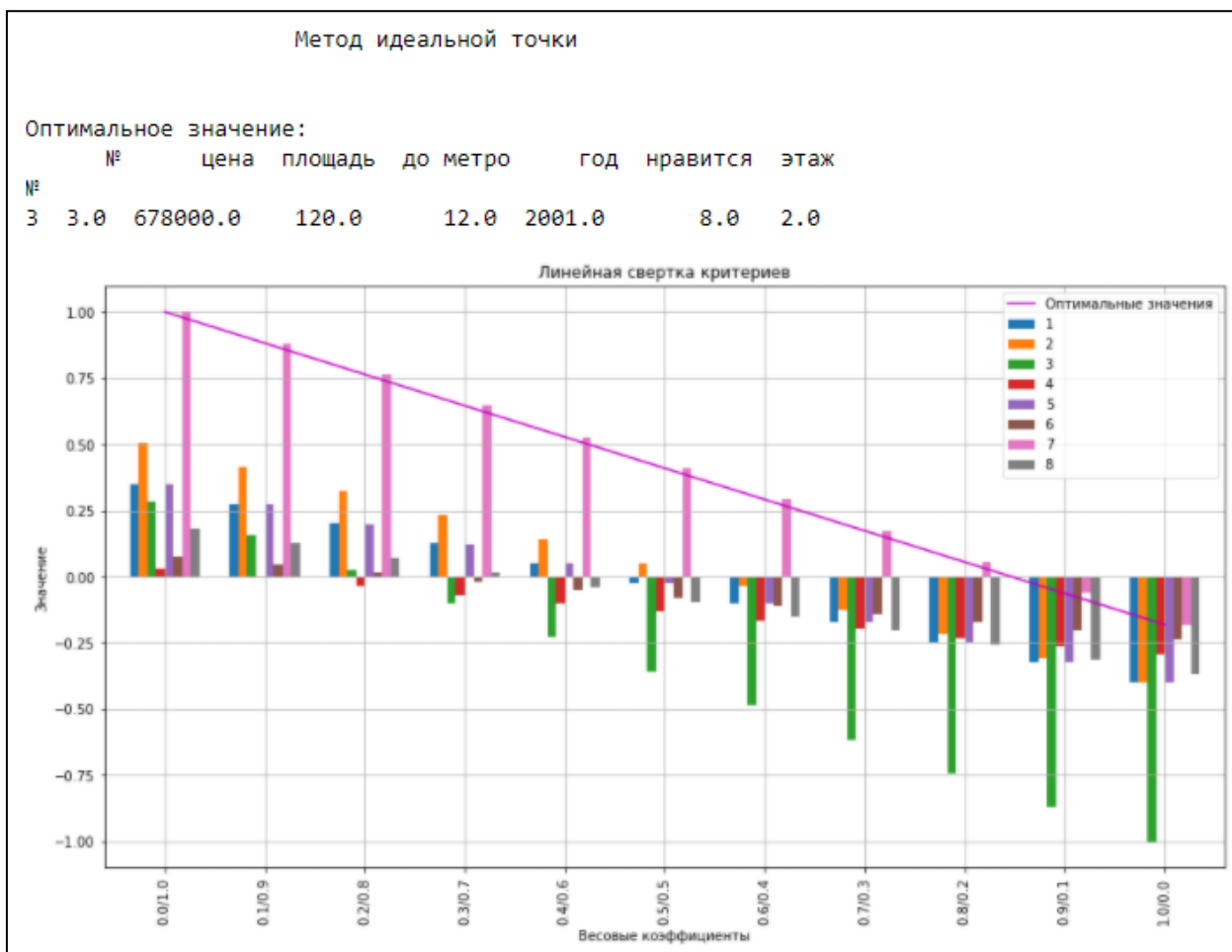


Рисунок 26. идеальная точка

6.1.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

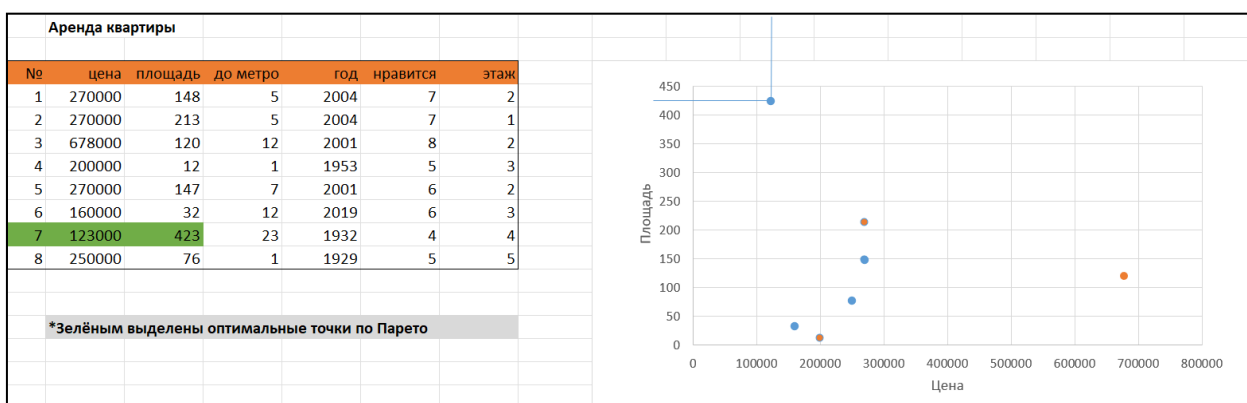


Рисунок 27. паретто

	min	max							Нормированные данные одного порядка								
№	Цена	Площадь	До метро	Год	Нравится	Этаж	Свёртка		№	Цена	Площадь	До метро	Год	Нравится	Этаж	Свёртка	
1	270000	148	5	2004	7	2	-758		1	0,39823	0,34988	0,21739	0,99257	0,875	0,4	1,39754	
2	270000	213	5	2004	7	1	-498		2	0,39823	0,50355	0,21739	0,99257	0,875	0,2	2,01219	
3	678000	120	12	2001	8	2	-2910		3	1	0,28369	0,52174	0,99108	1	0,4	1,12975	
4	200000	12	1	1953	5	3	-952		4	0,29499	0,02837	0,04348	0,96731	0,625	0,6	0,112	
5	270000	147	7	2001	6	2	-762		5	0,39823	0,34752	0,30435	0,99108	0,75	0,4	1,38808	
6	160000	32	12	2019	6	3	-672		6	0,23599	0,07565	0,52174	1	0,75	0,6	0,30142	
7	123000	423	23	1932	4	4	1077		7	0,18142	1	1	0,95691	0,5	0,8	3,99909	
8	250000	76	1	1929	5	5	-946		8	0,36873	0,17967	0,04348	0,95542	0,625	1	0,71683	
МАКС=		678000	423	23	2019	8	5	1077	=Данные№1-8Столбика/МАКССтолбика							МАКС= 3,99909	
Значим.		5	4														
Значим.		0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1					
		1	0,9	0,8	0,7	0,6	0,5	0,4	0,3	0,2	0,1	0					
	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка					
1	0,34988	0,31485	0,27983	0,2448	0,20977	0,17474	0,13971	0,10469	0,06966	0,03463	-0,0004						
2	0,50355	0,45315	0,40276	0,35236	0,30197	0,25157	0,20118	0,15079	0,10039	0,05	-0,0004						
3	0,28369	0,25522	0,22675	0,19828	0,16981	0,14134	0,11288	0,08441	0,05594	0,02747	-0,001						
4	0,02837	0,0255	0,02264	0,01977	0,0169	0,01404	0,01117	0,0083	0,00544	0,00257	-0,00029						
5	0,34752	0,31273	0,27793	0,24314	0,20835	0,17356	0,13877	0,10398	0,06918	0,03439	-0,0004						
6	0,07565	0,06806	0,06047	0,05288	0,0453	0,03771	0,03012	0,02253	0,01494	0,00735	-0,00024						
7	1	0,89998	0,79996	0,69995	0,59993	0,49991	0,39989	0,29987	0,19985	0,09984	-0,00018						
8	0,17967	0,16167	0,14366	0,12566	0,10765	0,08965	0,07165	0,05364	0,03564	0,01764	-0,00037						
max		1	0,89998	0,79996	0,69995	0,59993	0,49991	0,39989	0,29987	0,19985	0,09984	-0,00018					

Рисунок 28. линейная свёртка

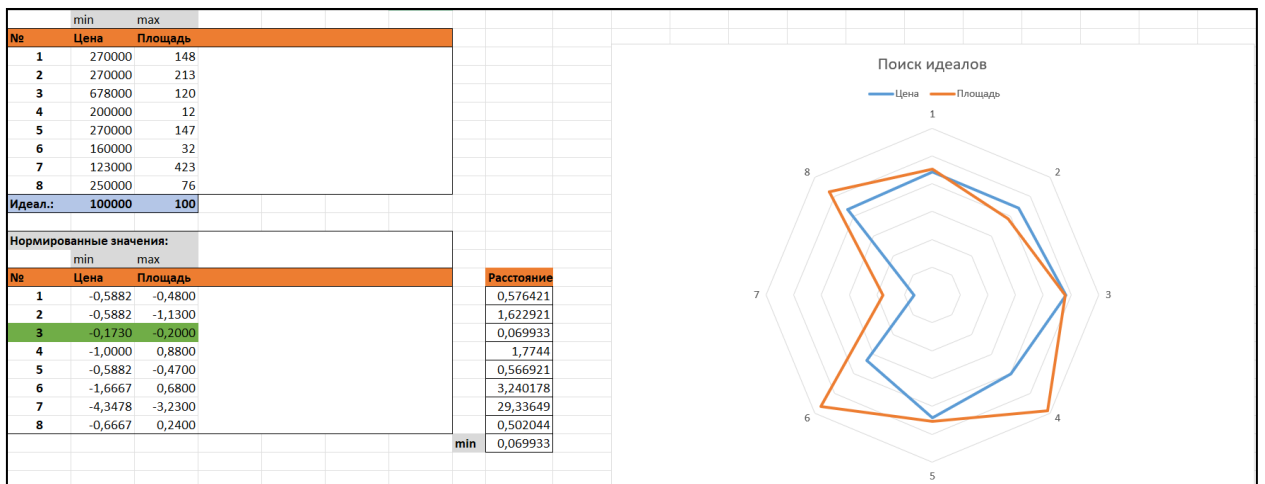


Рисунок 29. Идеальная точка

6.2. Тестирование Датасета №2:

6.2.1 Метод Python:

Импортируем датасет в Python, выбираем количество оптимизируемых критериев, названия критериев, направление и важность их оптимизации. Уточняем наличие идеальной точки, вводим идеальные параметры.

Введите путь к файлу:
/Users/aniki/Desktop/2dataset.csv

Вы импортировали:

	№	цена	площадь	до метро	год	нравится	этаж
№							
1	1.0	122000.0	148.0	5.0	2004.0	7.0	2.0
2	2.0	321000.0	150.0	5.0	2004.0	8.0	2.0
3	3.0	654000.0	120.0	4.0	2001.0	8.0	5.0
4	4.0	100000.0	56.0	1.0	1943.0	9.0	8.0
5	5.0	543000.0	147.0	7.0	2004.0	4.0	2.0
6	6.0	160000.0	55.0	12.0	2020.0	6.0	8.0
7	7.0	784000.0	54.0	5.0	1965.0	1.0	4.0
8	8.0	250000.0	65.0	1.0	1973.0	5.0	5.0

Сколько критериев оптимизировать? 2

Название критерия: цена

Направление ("макс" или "мин"): мин

Важность (от 1 до 10): 2

Название критерия: площадь

Направление ("макс" или "мин"): макс

Важность (от 1 до 10): 5

Есть идеальная точка? ('да' или 'нет'): да

Значение по столбцу цена: 200000

Значение по столбцу площадь: 150

Значение по столбцу до метро: 3

Значение по столбцу год: 2002

Значение по столбцу нравится: 8

Значение по столбцу этаж: 3

Рисунок 30. Импорт

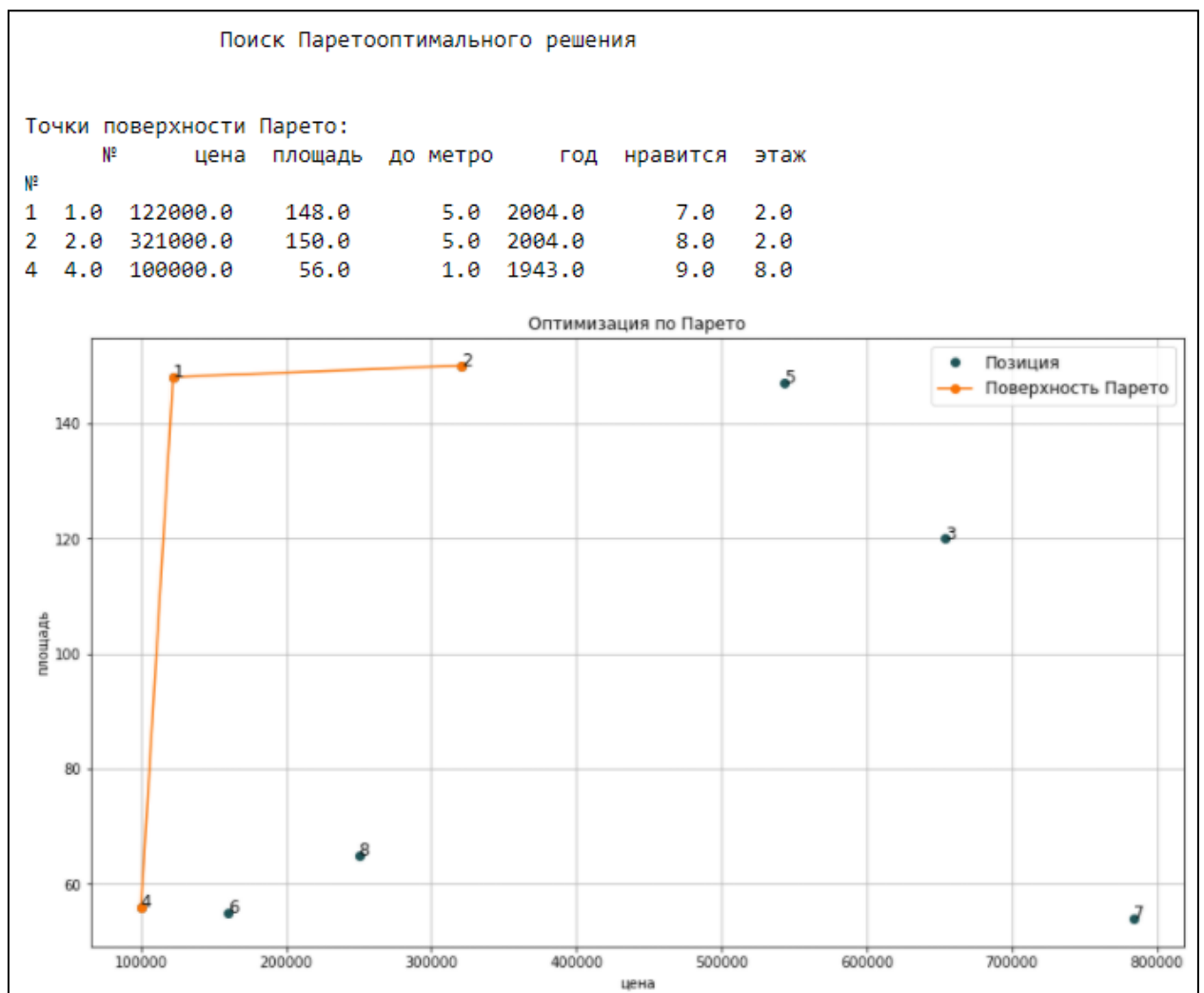


Рисунок 31. паретто

Нормированные данные:

№	цена	площадь	до метро	год	нравится	этаж	
1	0.125	0.155612	0.986667	0.416667	0.992079	0.777778	0.250
2	0.250	0.409439	1.000000	0.416667	0.992079	0.888889	0.250
3	0.375	0.834184	0.800000	0.333333	0.990594	0.888889	0.625
4	0.500	0.127551	0.373333	0.083333	0.961881	1.000000	1.000
5	0.625	0.692602	0.980000	0.583333	0.992079	0.444444	0.250
6	0.750	0.204082	0.366667	1.000000	1.000000	0.666667	1.000
7	0.875	1.000000	0.360000	0.416667	0.972772	0.111111	0.500
8	1.000	0.318878	0.433333	0.083333	0.976733	0.555556	0.625

Рисунок 32. нормированные данные

Линейная свертка критериев

Линейная свертка:

	0.0/1.0	0.1/0.9	0.2/0.8	0.3/0.7	0.4/0.6	0.5/0.5	0.6/0.4
1	0.986667	0.872439	0.758211	0.643983	0.529755	0.415527	0.301299
2	1.000000	0.859056	0.718112	0.577168	0.436224	0.295281	0.154337
3	0.800000	0.636582	0.473163	0.309745	0.146327	-0.017092	-0.180510
4	0.373333	0.323245	0.273156	0.223068	0.172980	0.122891	0.072803
5	0.980000	0.812740	0.645480	0.478219	0.310959	0.143699	-0.023561
6	0.366667	0.309592	0.252517	0.195442	0.138367	0.081293	0.024218
7	0.360000	0.224000	0.088000	-0.048000	-0.184000	-0.320000	-0.456000
8	0.433333	0.358112	0.282891	0.207670	0.132449	0.057228	-0.017993

	0.7/0.3	0.8/0.2	0.9/0.1	1.0/0.0
1	0.187071	0.072844	-0.041384	-0.155612
2	0.013393	-0.127551	-0.268495	-0.409439
3	-0.343929	-0.507347	-0.670765	-0.834184
4	0.022714	-0.027374	-0.077463	-0.127551
5	-0.190821	-0.358082	-0.525342	-0.692602
6	-0.032857	-0.089932	-0.147007	-0.204082
7	-0.592000	-0.728000	-0.864000	-1.000000
8	-0.093214	-0.168435	-0.243656	-0.318878

Оптимальные значения:

При важности 0.0/1.0 оптимальный выбор под номером - 2
 При важности 0.1/0.9 оптимальный выбор под номером - 1
 При важности 0.2/0.8 оптимальный выбор под номером - 1
 При важности 0.3/0.7 оптимальный выбор под номером - 1
 При важности 0.4/0.6 оптимальный выбор под номером - 1
 При важности 0.5/0.5 оптимальный выбор под номером - 1
 При важности 0.6/0.4 оптимальный выбор под номером - 1
 При важности 0.7/0.3 оптимальный выбор под номером - 1
 При важности 0.8/0.2 оптимальный выбор под номером - 1
 При важности 0.9/0.1 оптимальный выбор под номером - 1
 При важности 1.0/0.0 оптимальный выбор под номером - 4

Рисунок 33. линейная свёрстка

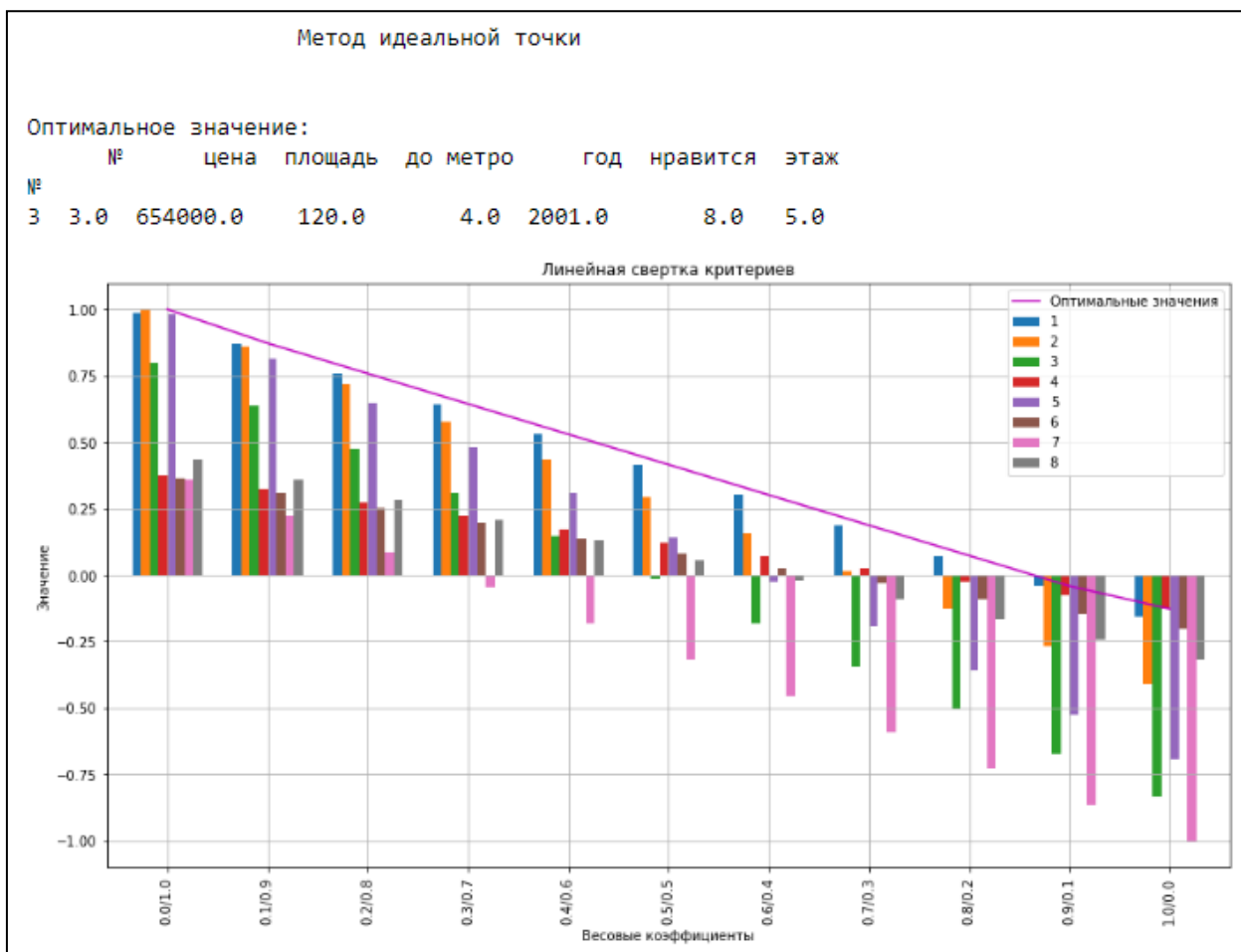


Рисунок 34. идеальная точка

6.1.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

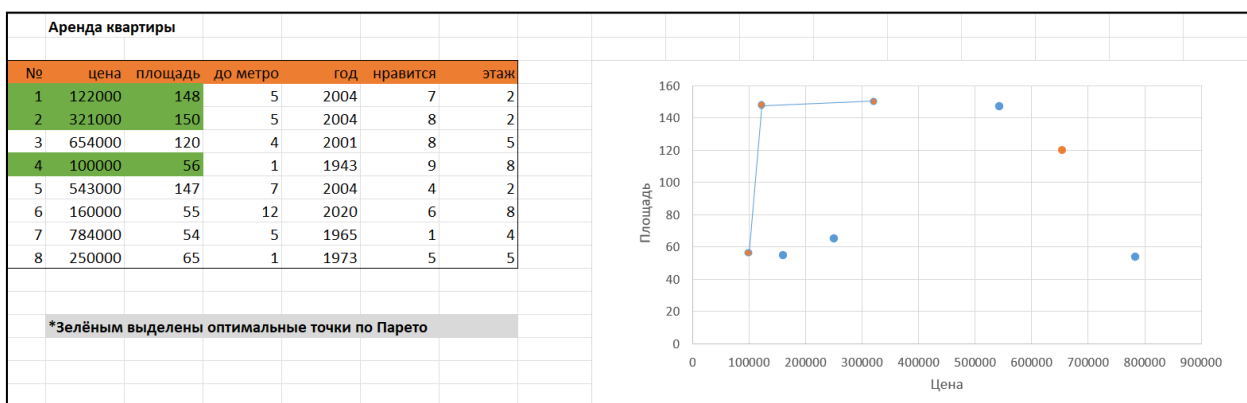


Рисунок 35. Паретооптимальное

	min	max							Нормированные данные одного порядка											
№	Цена	Площадь	До метро	Год		Нравится	Этаж	Свёртка		№	Цена	Площадь	До метро	Год	Нравится	Этаж	Свёртка			
1	122000	148	5	2004		7	2	496		1	0,15561	0,98667	0,41667	0,99208	0,77778	0,25	4,93302			
2	321000	150	5	2004		8	2	108		2	0,40944	1	0,41667	0,99208	0,88889	0,25	4,99918			
3	654000	120	4	2001		8	5	-708		3	0,83418	0,8	0,33333	0,99059	0,88889	0,625	3,99833			
4	100000	56	1	1943		9	8	80		4	0,12755	0,37333	0,08333	0,96188	1	1	1,86641			
5	543000	147	7	2004		4	2	-351		5	0,6926	0,98	0,58333	0,99208	0,44444	0,25	4,89861			
6	160000	55	12	2020		6	8	-45		6	0,20408	0,36667	1	1	0,66667	1	1,83293			
7	784000	54	5	1965		1	4	-1298		7	1	0,36	0,41667	0,97277	0,11111	0,5	1,798			
8	250000	65	1	1973		5	5	-175		8	0,31888	0,43333	0,08333	0,97673	0,55556	0,625	2,16603			
МАКС=		784000	150	12	2020	9	8	496		=Данные№1-8Столбика/МАКССтолбика								МАКС=		4,99918
Значим.		2	5																	
Значим.		0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1								
		1	0,9	0,8	0,7	0,6	0,5	0,4	0,3	0,2	0,1	0								
	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка								
1	0,98667	0,88798	0,7893	0,69062	0,59194	0,49326	0,39457	0,29589	0,19721	0,09853	-0,00016									
2	1	0,89996	0,79992	0,69988	0,59984	0,4998	0,39975	0,29971	0,19967	0,09963	-0,00041									
3	0,8	0,71992	0,63983	0,55975	0,47967	0,39958	0,3195	0,23942	0,15933	0,07925	-0,00083									
4	0,37333	0,33599	0,29864	0,2613	0,22395	0,1866	0,14926	0,11191	0,07456	0,03722	-0,00013									
5	0,98	0,88193	0,78386	0,68579	0,58772	0,48965	0,39158	0,29352	0,19545	0,09738	-0,00069									
6	0,36667	0,32998	0,29329	0,25661	0,21992	0,18323	0,14654	0,10986	0,07317	0,03648	-0,0002									
7	0,36	0,3239	0,2878	0,2517	0,2156	0,1795	0,1434	0,1073	0,0712	0,0351	-0,001									
8	0,43333	0,38997	0,3466	0,30324	0,25987	0,21651	0,17314	0,12978	0,08641	0,04305	-0,00032									
max		1	0,89996	0,79992	0,69988	0,59984	0,4998	0,39975	0,29971	0,19967	0,09963	-0,00013								

Рисунок 36. линейная свёртка

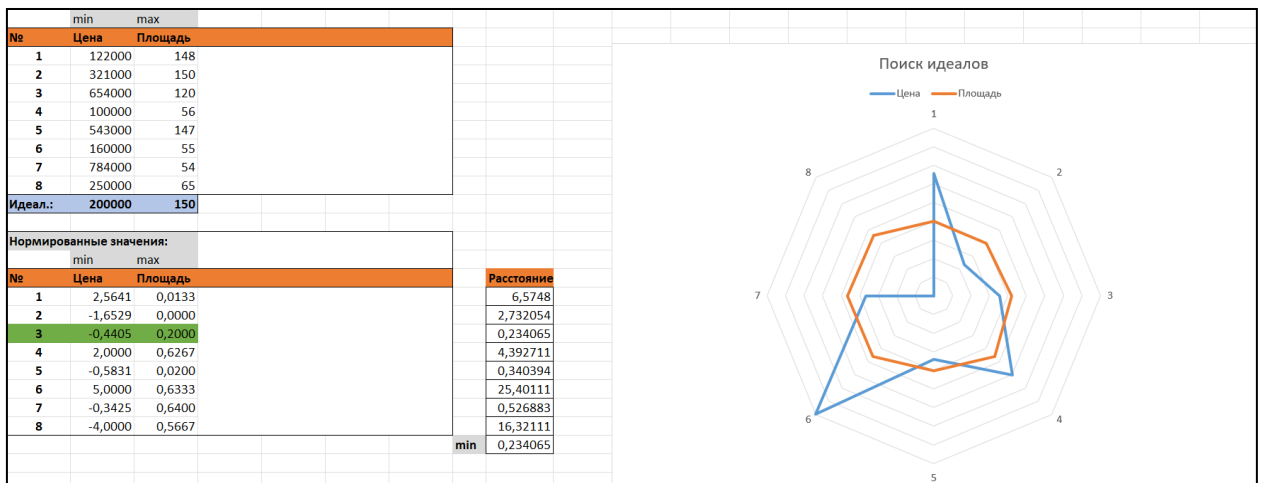


Рисунок 37. идеальная точка

6.3. Тестирование Датасета №3:

6.3.1 Метод Python:

Импортируем датасет в Python, выбираем количество оптимизируемых критериев, названия критериев, направление и важность их оптимизации. Уточняем наличие идеальной точки, вводим идеальные параметры.

Введите путь к файлу:

/Users/aniki/Desktop/3dataset.csv

Вы импортировали:

	№	цена	площадь	до метро	год	нравится	этаж
№							
1	1.0	270000.0	148.0	5.0	2004.0	7.0	2.0
2	2.0	400000.0	140.0	3.0	2001.0	5.0	1.0
3	3.0	200000.0	120.0	4.0	2002.0	8.0	2.0
4	4.0	123000.0	56.0	1.0	1929.0	5.0	3.0
5	5.0	270000.0	147.0	7.0	2004.0	7.0	2.0
6	6.0	380000.0	55.0	12.0	2019.0	6.0	2.0
7	7.0	120000.0	54.0	5.0	1932.0	4.0	4.0
8	8.0	250000.0	65.0	1.0	1929.0	5.0	5.0

Сколько критериев оптимизировать? 2

Название критерия: цена

Направление ("макс" или "мин"): мин

Важность (от 1 до 10): 10

Название критерия: площадь

Направление ("макс" или "мин"): макс

Важность (от 1 до 10): 5

Есть идеальная точка? ('да' или 'нет'): да

Значение по столбцу цена: 233000

Значение по столбцу площадь: 147

Значение по столбцу до метро: 4

Значение по столбцу год: 2005

Значение по столбцу нравится: 7

Значение по столбцу этаж: 2

Рисунок 38. импорт

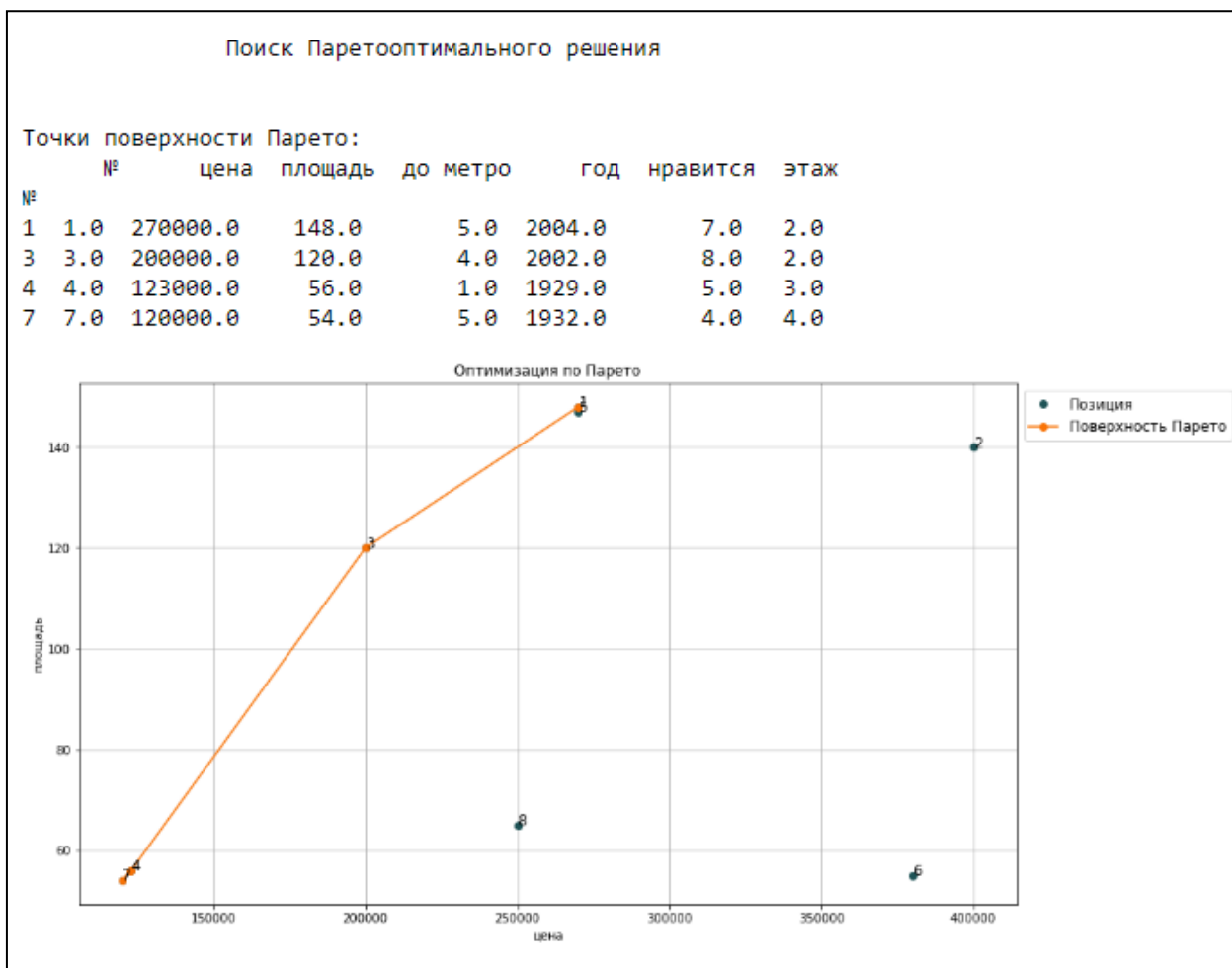


Рисунок 39. Паретооптимальное

Нормированные данные:							
	№	цена	площадь	до метро	год	нравится	этаж
№							
1	0.125	0.6750	1.000000	0.416667	0.992571	0.875	0.4
2	0.250	1.0000	0.945946	0.250000	0.991085	0.625	0.2
3	0.375	0.5000	0.810811	0.333333	0.991580	1.000	0.4
4	0.500	0.3075	0.378378	0.083333	0.955423	0.625	0.6
5	0.625	0.6750	0.993243	0.583333	0.992571	0.875	0.4
6	0.750	0.9500	0.371622	1.000000	1.000000	0.750	0.4
7	0.875	0.3000	0.364865	0.416667	0.956909	0.500	0.8
8	1.000	0.6250	0.439189	0.083333	0.955423	0.625	1.0

Рисунок 40. нормированные данные

Линейная свертка критериев

Линейная свертка:

	0.0/1.0	0.1/0.9	0.2/0.8	0.3/0.7	0.4/0.6	0.5/0.5	0.6/0.4
1	1.000000	0.832500	0.665000	0.497500	0.330000	0.162500	-0.005000
2	0.945946	0.751351	0.556757	0.362162	0.167568	-0.027027	-0.221622
3	0.810811	0.679730	0.548649	0.417568	0.286486	0.155405	0.024324
4	0.378378	0.309791	0.241203	0.172615	0.104027	0.035439	-0.033149
5	0.993243	0.826419	0.659595	0.492770	0.325946	0.159122	-0.007703
6	0.371622	0.239459	0.107297	-0.024865	-0.157027	-0.289189	-0.421351
7	0.364865	0.298378	0.231892	0.165405	0.098919	0.032432	-0.034054
8	0.439189	0.332770	0.226351	0.119932	0.013514	-0.092905	-0.199324

	0.7/0.3	0.8/0.2	0.9/0.1	1.0/0.0
1	-0.172500	-0.340000	-0.507500	-0.6750
2	-0.416216	-0.610811	-0.805405	-1.0000
3	-0.106757	-0.237838	-0.368919	-0.5000
4	-0.101736	-0.170324	-0.238912	-0.3075
5	-0.174527	-0.341351	-0.508176	-0.6750
6	-0.553514	-0.685676	-0.817838	-0.9500
7	-0.100541	-0.167027	-0.233514	-0.3000
8	-0.305743	-0.412162	-0.518581	-0.6250

Оптимальные значения:

При важности 0.0/1.0 оптимальный выбор под номером - 1
 При важности 0.1/0.9 оптимальный выбор под номером - 1
 При важности 0.2/0.8 оптимальный выбор под номером - 1
 При важности 0.3/0.7 оптимальный выбор под номером - 1
 При важности 0.4/0.6 оптимальный выбор под номером - 1
 При важности 0.5/0.5 оптимальный выбор под номером - 1
 При важности 0.6/0.4 оптимальный выбор под номером - 3
 При важности 0.7/0.3 оптимальный выбор под номером - 7
 При важности 0.8/0.2 оптимальный выбор под номером - 7
 При важности 0.9/0.1 оптимальный выбор под номером - 7
 При важности 1.0/0.0 оптимальный выбор под номером - 7

Рисунок 41. линейная свёрстка

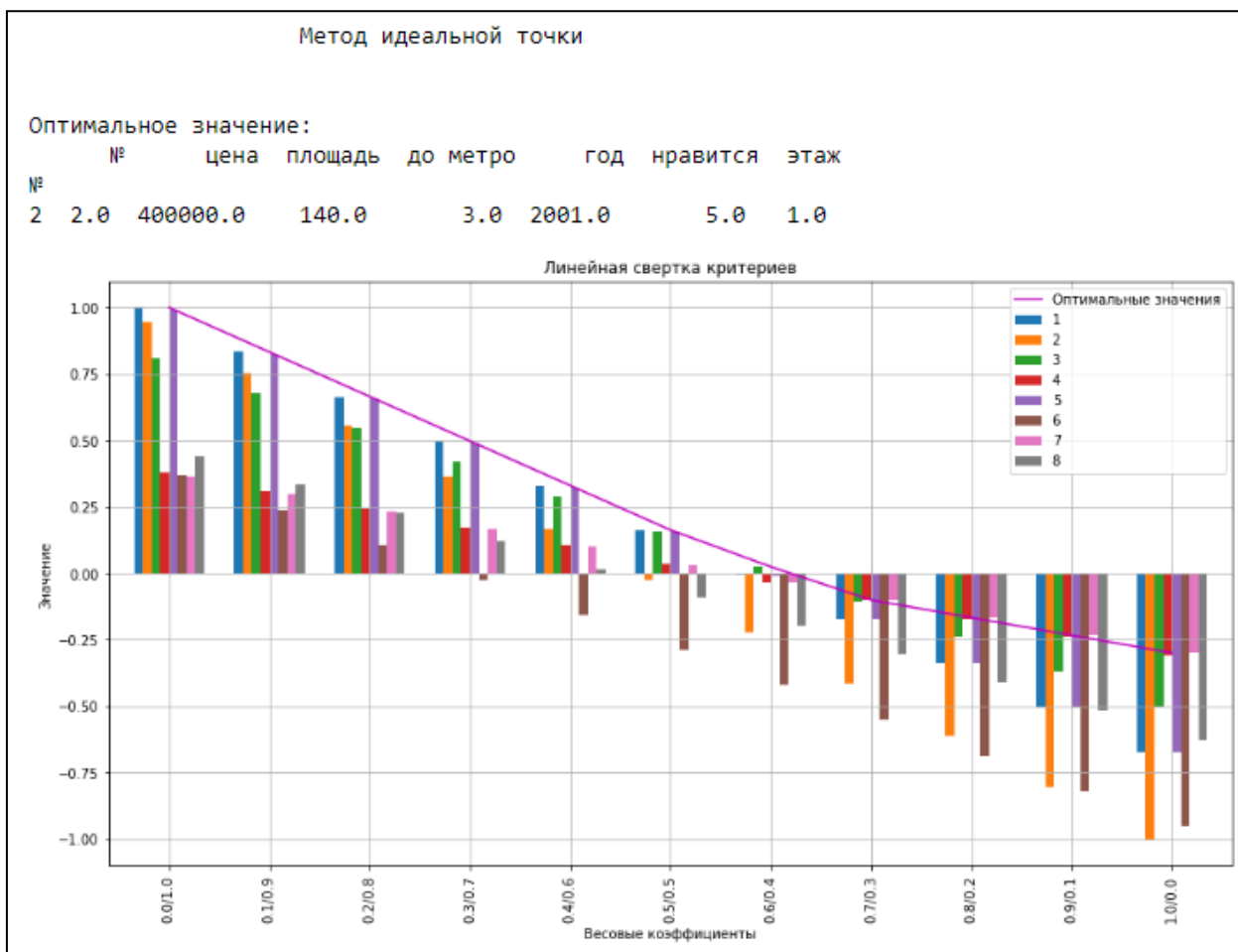


Рисунок 42. идеальная точка

6.3.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

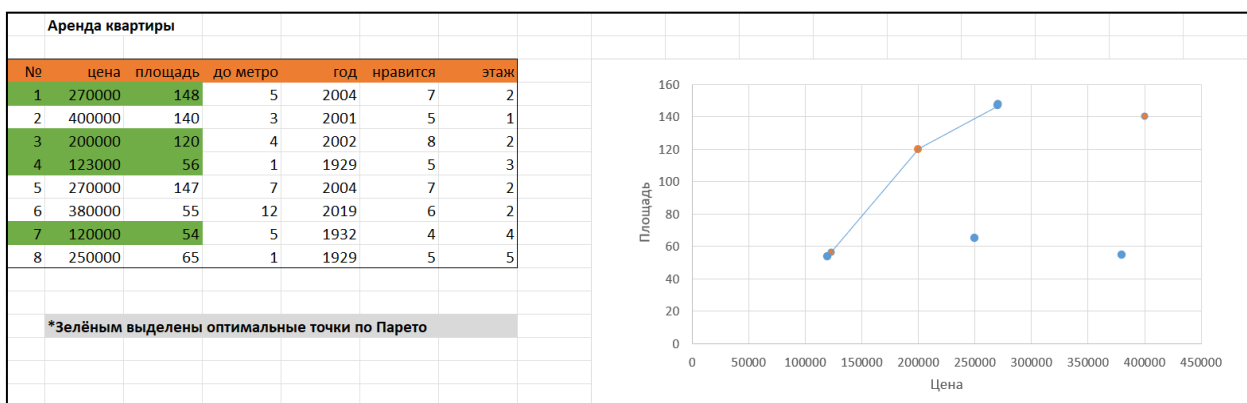


Рисунок 43. Паретооптимальное

min max								Нормированные данные одного порядка							
№	Цена	Площадь	До метро	Год	Нравится	Этаж	Свёртка	№	Цена	Площадь	До метро	Год	Нравится	Этаж	Свёртка
1	270000	148	5	2004	7	2	-1960	1	0,675	1	0,41667	0,99257	0,875	0,4	4,99325
2	400000	140	3	2001	5	1	-3300	2	1	0,94595	0,25	0,99108	0,625	0,2	4,71973
3	200000	120	4	2002	8	2	-1400	3	0,5	0,81081	0,33333	0,99158	1	0,4	4,04905
4	123000	56	1	1929	5	3	-950	4	0,3075	0,37838	0,08333	0,95542	0,625	0,6	1,88882
5	270000	147	7	2004	7	2	-1965	5	0,675	0,99324	0,58333	0,99257	0,875	0,4	4,95947
6	380000	55	12	2019	6	2	-3525	6	0,95	0,37162	1	1	0,75	0,4	1,84861
7	120000	54	5	1932	4	4	-930	7	0,3	0,36486	0,41667	0,95691	0,5	0,8	1,82132
8	250000	65	1	1929	5	5	-2175	8	0,625	0,43919	0,08333	0,95542	0,625	1	2,1897
МАКС=	400000	148	12	2019	8	5	-930	^Данные№1-8Столбика/МАКССтолбика							
Значим.	10	5													МАКС= 4,99325
Значим.															
	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1				
	1	0,9	0,8	0,7	0,6	0,5	0,4	0,3	0,2	0,1	0				
	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка				
1	1	0,89993	0,79987	0,6998	0,59973	0,49966	0,3996	0,29953	0,19946	0,09939	-0,00068				
2	0,94595	0,85125	0,75656	0,66186	0,56717	0,47247	0,37778	0,28308	0,18839	0,09369	-0,001				
3	0,81081	0,72968	0,64855	0,56742	0,48629	0,40516	0,32402	0,24289	0,16176	0,08063	-0,0005				
4	0,37838	0,34051	0,30264	0,26477	0,2269	0,18904	0,15117	0,1133	0,07543	0,03756	-0,00031				
5	0,99324	0,89385	0,79446	0,69507	0,59568	0,49628	0,39689	0,2975	0,19811	0,09872	-0,00068				
6	0,37162	0,33436	0,29711	0,25985	0,22259	0,18534	0,14808	0,11082	0,07356	0,03631	-0,00095				
7	0,36486	0,32835	0,29183	0,25532	0,2188	0,18228	0,14577	0,10925	0,07273	0,03622	-0,0003				
8	0,43919	0,39521	0,35123	0,30724	0,26326	0,21928	0,1753	0,13132	0,08734	0,04336	-0,00063				
max	1	0,89993	0,79987	0,6998	0,59973	0,49966	0,3996	0,29953	0,19946	0,09939	-0,0003				

Рисунок 44. линейная свёртка

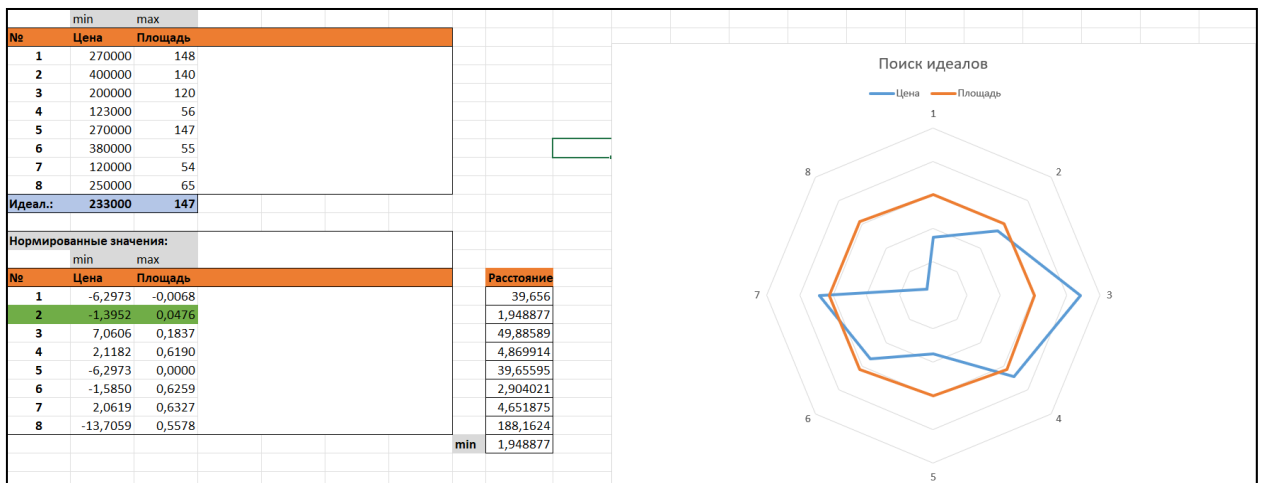


Рисунок 45. идеальная точка

6.4. Тестирование Датасета №4:

6.4.1 Метод Python:

Импортируем датасет в Python, выбираем количество оптимизируемых критериев, названия критериев, направление и важность их оптимизации. Уточняем наличие идеальной точки, вводим идеальные параметры.

Введите путь к файлу:
/Users/aniki/Desktop/4dataset.csv

Вы импортировали:

	№	цена	площадь	до метро	год	нравится	этаж
№							
1	1.0	175000.0	129.0	7.0	2020.0	10.0	4.0
2	2.0	563000.0	100.0	4.0	2007.0	7.0	3.0
3	3.0	555000.0	123.0	21.0	2001.0	8.0	2.0
4	4.0	101000.0	64.0	9.0	1965.0	5.0	8.0
5	5.0	277000.0	123.0	4.0	2004.0	4.0	2.0
6	6.0	160000.0	94.0	12.0	2019.0	6.0	2.0
7	7.0	121000.0	44.0	5.0	1932.0	4.0	6.0
8	8.0	276000.0	88.0	4.0	1976.0	5.0	5.0

Сколько критериев оптимизировать? 2

Название критерия: цена

Направление ("макс" или "мин"): мин

Важность (от 1 до 10): 3

Название критерия: площадь

Направление ("макс" или "мин"): макс

Важность (от 1 до 10): 1

Есть идеальная точка? ('да' или 'нет'): да

Значение по столбцу цена: 199000

Значение по столбцу площадь: 100

Значение по столбцу до метро: 2

Значение по столбцу год: 2009

Значение по столбцу нравится: 9

Значение по столбцу этаж: 3

Рисунок 46. Импорт

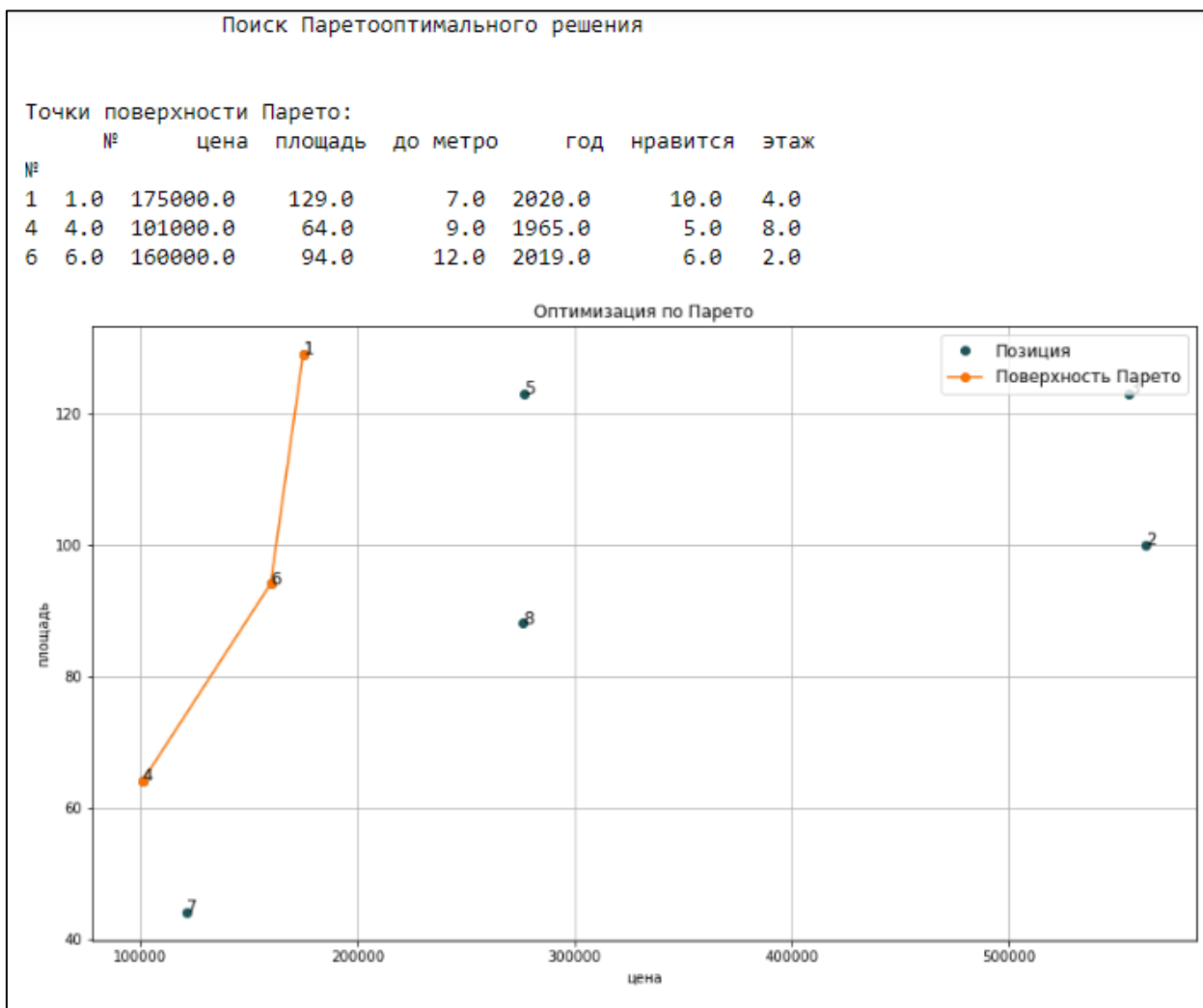


Рисунок 47. Паретооптимальное

Нормированные данные:							
	№	цена	площадь	до метро	год	нравится	этаж
№							
1	0.125	0.310835	1.000000	0.333333	1.000000	1.0	0.500
2	0.250	1.000000	0.775194	0.190476	0.993564	0.7	0.375
3	0.375	0.985790	0.953488	1.000000	0.990594	0.8	0.250
4	0.500	0.179396	0.496124	0.428571	0.972772	0.5	1.000
5	0.625	0.492007	0.953488	0.190476	0.992079	0.4	0.250
6	0.750	0.284192	0.728682	0.571429	0.999505	0.6	0.250
7	0.875	0.214920	0.341085	0.238095	0.956436	0.4	0.750
8	1.000	0.490231	0.682171	0.190476	0.978218	0.5	0.625

Рисунок 48. нормированные данные

Линейная свертка критериев

Линейная свертка:

	0.0/1.0	0.1/0.9	0.2/0.8	0.3/0.7	0.4/0.6	0.5/0.5	0.6/0.4
1	1.000000	0.868917	0.737833	0.606750	0.475666	0.344583	0.213499
2	0.775194	0.597674	0.420155	0.242636	0.065116	-0.112403	-0.289922
3	0.953488	0.759560	0.565633	0.371705	0.177777	-0.016151	-0.210079
4	0.496124	0.428572	0.361020	0.293468	0.225916	0.158364	0.090812
5	0.953488	0.808939	0.664389	0.519840	0.375290	0.230741	0.086191
6	0.728682	0.627395	0.526107	0.424820	0.323533	0.222245	0.120958
7	0.341085	0.285485	0.229884	0.174284	0.118683	0.063083	0.007482
8	0.682171	0.564930	0.447690	0.330450	0.213210	0.095970	-0.021270

	0.7/0.3	0.8/0.2	0.9/0.1	1.0/0.0
1	0.082416	-0.048668	-0.179751	-0.310835
2	-0.467442	-0.644961	-0.822481	-1.000000
3	-0.404007	-0.597935	-0.791863	-0.985790
4	0.023260	-0.044292	-0.111844	-0.179396
5	-0.058358	-0.202908	-0.347458	-0.492007
6	0.019670	-0.081617	-0.182904	-0.284192
7	-0.048118	-0.103719	-0.159320	-0.214920
8	-0.138510	-0.255751	-0.372991	-0.490231

Оптимальные значения:

При важности 0.0/1.0 оптимальный выбор под номером - 1
 При важности 0.1/0.9 оптимальный выбор под номером - 1
 При важности 0.2/0.8 оптимальный выбор под номером - 1
 При важности 0.3/0.7 оптимальный выбор под номером - 1
 При важности 0.4/0.6 оптимальный выбор под номером - 1
 При важности 0.5/0.5 оптимальный выбор под номером - 1
 При важности 0.6/0.4 оптимальный выбор под номером - 1
 При важности 0.7/0.3 оптимальный выбор под номером - 1
 При важности 0.8/0.2 оптимальный выбор под номером - 4
 При важности 0.9/0.1 оптимальный выбор под номером - 4
 При важности 1.0/0.0 оптимальный выбор под номером - 4

Рисунок 49. линейная свёрстка критериев

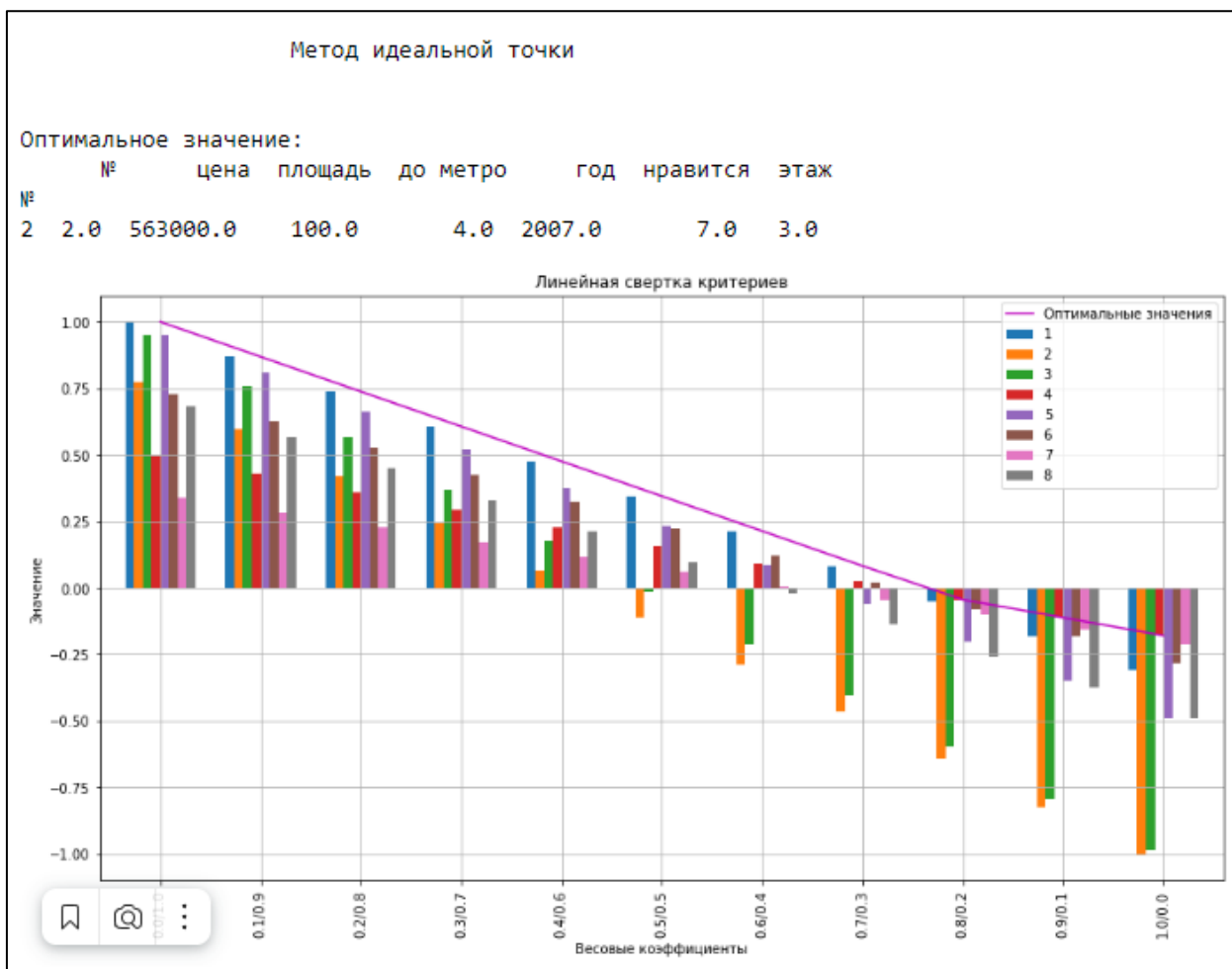


Рисунок 50. идеальная точка

6.4.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

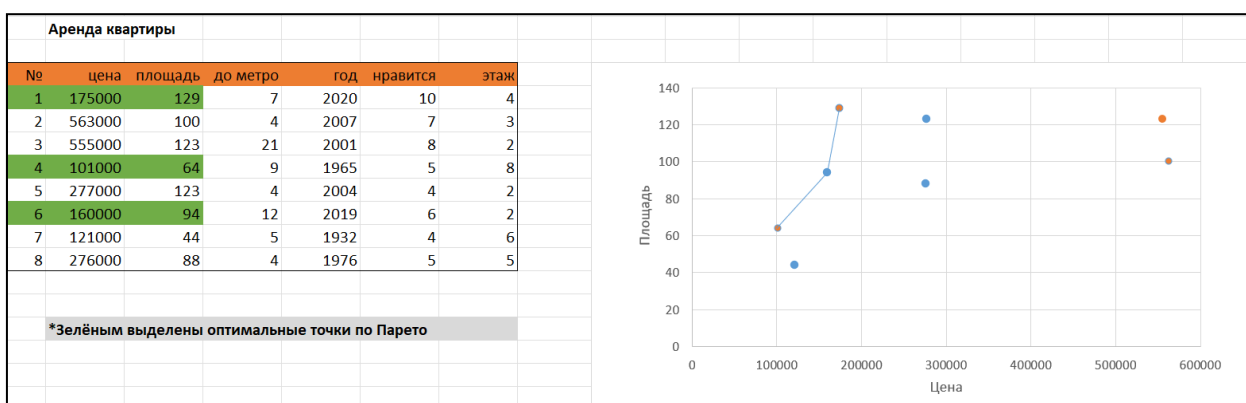


Рисунок 51. Паретооптимальное

	min	max							Нормированные данные одного порядка											
№	Цена	Площадь	До метро	Год	Нравится	Этаж	Свёртка		№	Цена	Площадь	До метро	Год	Нравится	Этаж	Свёртка				
1	175000	129	7	2020	10	4	-396		1	0,31083	1	0,33333	1	1	0,5	0,99907				
2	563000	100	4	2007	7	3	-1589		2	1	0,77519	0,19048	0,99356	0,7	0,375	0,77219				
3	555000	123	21	2001	8	2	-1542		3	0,98579	0,95349	1	0,99059	0,8	0,25	0,95053				
4	101000	64	9	1965	5	8	-239		4	0,1794	0,49612	0,42857	0,97277	0,5	1	0,49559				
5	277000	123	4	2004	4	2	-708		5	0,49201	0,95349	0,19048	0,99208	0,4	0,25	0,95201				
6	160000	94	12	2019	6	2	-386		6	0,28419	0,72868	0,57143	0,9995	0,6	0,25	0,72783				
7	121000	44	5	1932	4	6	-319		7	0,21492	0,34109	0,2381	0,95644	0,4	0,75	0,34044				
8	276000	88	4	1976	5	5	-740		8	0,49023	0,68217	0,19048	0,97822	0,5	0,625	0,6807				
МАКС=	563000	129	21	2020	10	8	-239		=Данные№1-8Столбика/МАКССтолбика										МАКС=	0,99907
Значим.	3	1																		
Значим.	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1									
	1	0,9	0,8	0,7	0,6	0,5	0,4	0,3	0,2	0,1	0									
	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка	Свёртка									
1	1	0,89997	0,79994	0,69991	0,59988	0,49984	0,39981	0,29978	0,19975	0,09972	-0,00031									
2	0,77519	0,69757	0,61996	0,54234	0,46472	0,3871	0,30948	0,23186	0,15424	0,07662	-0,001									
3	0,95349	0,85804	0,76259	0,66715	0,5717	0,47625	0,3808	0,28536	0,18991	0,09446	-0,00099									
4	0,49612	0,44649	0,39686	0,34723	0,2976	0,24797	0,19834	0,14871	0,09908	0,04945	-0,00018									
5	0,95349	0,85809	0,76269	0,66729	0,5719	0,4765	0,3811	0,2857	0,1903	0,09491	-0,00049									
6	0,72868	0,65579	0,58289	0,50999	0,4371	0,3642	0,2913	0,21841	0,14551	0,07261	-0,00028									
7	0,34109	0,30696	0,27283	0,2387	0,20457	0,17044	0,13631	0,10218	0,06805	0,03392	-0,00021									
8	0,68217	0,6139	0,54564	0,47737	0,40911	0,34084	0,27257	0,20431	0,13604	0,06778	-0,00049									
max	1	0,89997	0,79994	0,69991	0,59988	0,49984	0,39981	0,29978	0,19975	0,09972	-0,00018									

Рисунок 52. линейная свёрстка

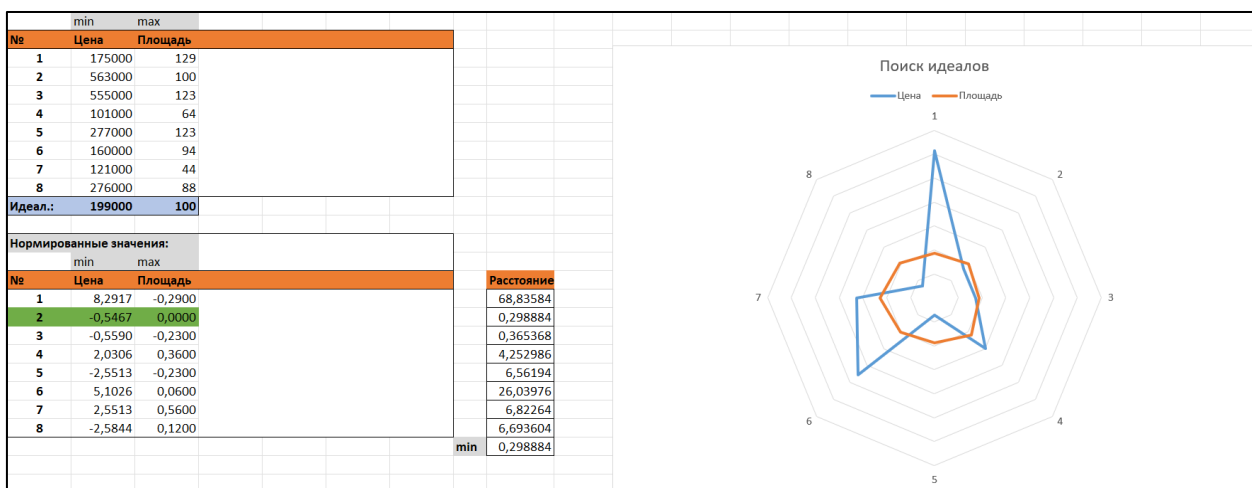


Рисунок 53. идеальная точка

6.5. Тестирование Датасета №5:

6.5.1 Метод Python:

Импортируем датасет в Python, выбираем количество оптимизируемых критериев, названия критериев, направление и важность их оптимизации. Уточняем наличие идеальной точки, вводим идеальные параметры.

Введите путь к файлу:

/Users/aniki/Desktop/5dataset.csv

Вы импортировали:

	№	цена	площадь	до метро	год	нравится	этаж
№							
1	1.0	645000.0	148.0	5.0	2000.0	8.0	2.0
2	2.0	222000.0	23.0	10.0	2004.0	7.0	1.0
3	3.0	200000.0	120.0	4.0	2021.0	6.0	3.0
4	4.0	321000.0	56.0	5.0	1954.0	5.0	4.0
5	5.0	270000.0	214.0	7.0	2006.0	5.0	2.0
6	6.0	463000.0	43.0	12.0	2018.0	6.0	5.0
7	7.0	863000.0	76.0	5.0	1932.0	2.0	4.0
8	8.0	222000.0	100.0	10.0	1929.0	5.0	7.0

Сколько критериев оптимизировать? 2

Название критерия: цена

Направление ("макс" или "мин"): мин

Важность (от 1 до 10): 10

Название критерия: площадь

Направление ("макс" или "мин"): макс

Важность (от 1 до 10): 2

Есть идеальная точка? ('да' или 'нет'): да

Значение по столбцу цена: 300000

Значение по столбцу площадь: 99

Значение по столбцу до метро: 3

Значение по столбцу год: 2008

Значение по столбцу нравится: 8

Значение по столбцу этаж: 1

Рисунок 54. Импорт

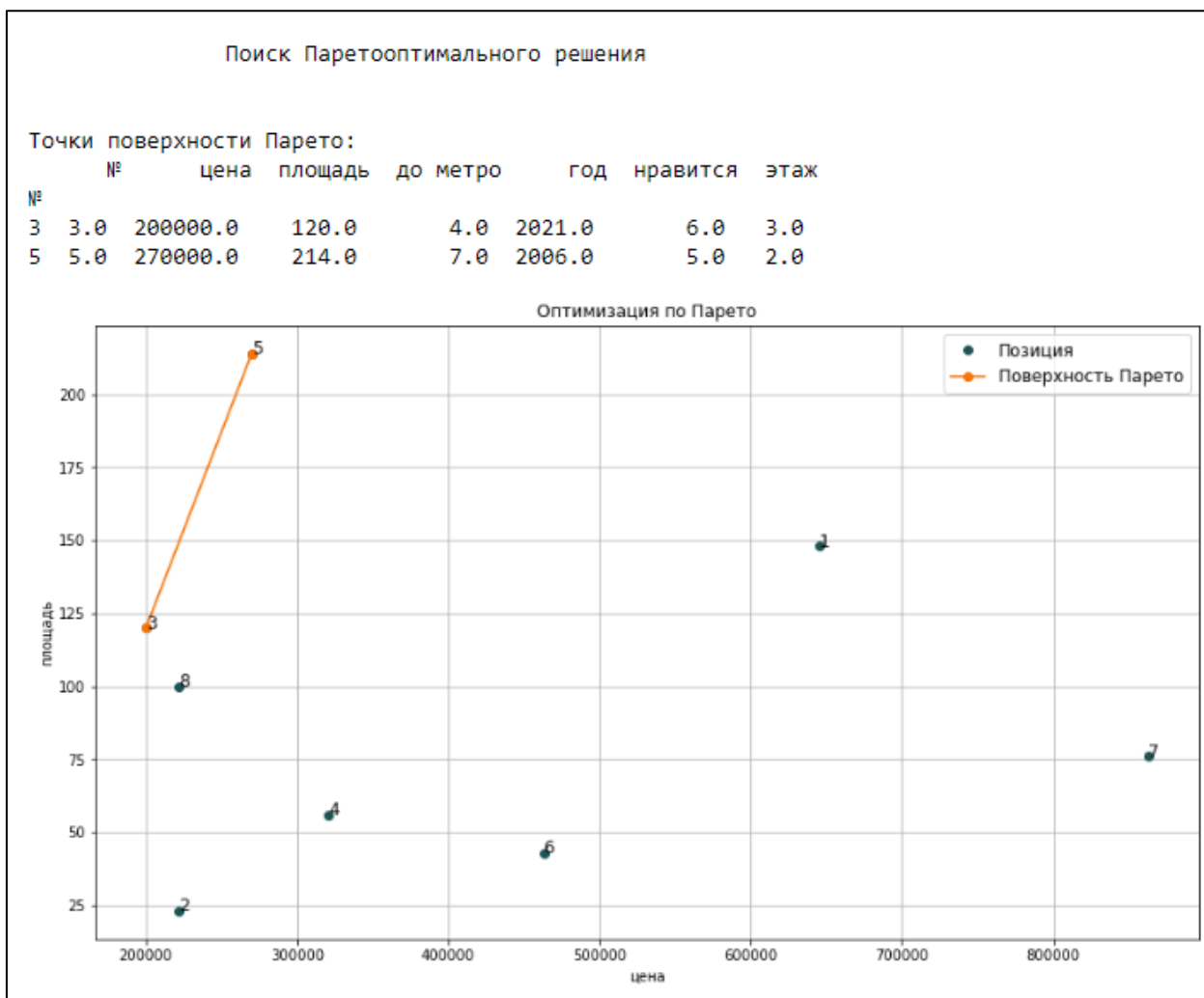


Рисунок 55. Паретооптимальное

Нормированные данные:

№	цена	площадь	до метро	год	нравится	этаж
1	0.125	0.747393	0.691589	0.416667	0.989609	1.000
2	0.250	0.257242	0.107477	0.833333	0.991588	0.875
3	0.375	0.231750	0.560748	0.333333	1.000000	0.750
4	0.500	0.371958	0.261682	0.416667	0.966848	0.625
5	0.625	0.312862	1.000000	0.583333	0.992578	0.625
6	0.750	0.536501	0.200935	1.000000	0.998516	0.750
7	0.875	1.000000	0.355140	0.416667	0.955962	0.250
8	1.000	0.257242	0.467290	0.833333	0.954478	0.625

Рисунок 56, нормированные данные

Линейная свертка критериев

Линейная свертка:

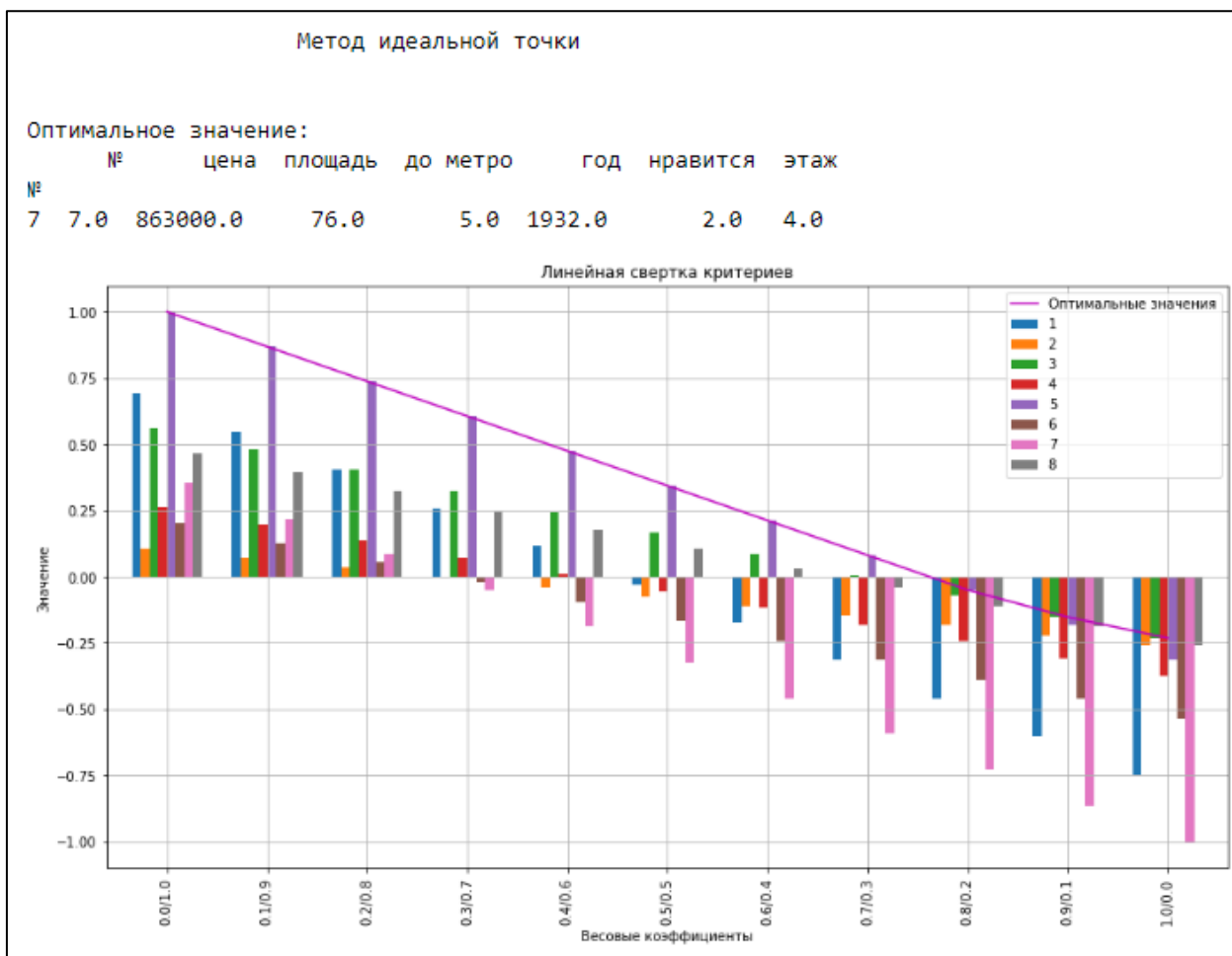
	0.0/1.0	0.1/0.9	0.2/0.8	0.3/0.7	0.4/0.6	0.5/0.5	0.6/0.4
1	0.691589	0.547691	0.403792	0.259894	0.115996	-0.027902	-0.171800
2	0.107477	0.071005	0.034533	-0.001939	-0.038411	-0.074883	-0.111355
3	0.560748	0.481498	0.402248	0.322998	0.243749	0.164499	0.085249
4	0.261682	0.198318	0.134954	0.071590	0.008226	-0.055138	-0.118502
5	1.000000	0.868714	0.737428	0.606141	0.474855	0.343569	0.212283
6	0.200935	0.127191	0.053448	-0.020296	-0.094039	-0.167783	-0.241527
7	0.355140	0.219626	0.084112	-0.051402	-0.186916	-0.322430	-0.457944
8	0.467290	0.394837	0.322383	0.249930	0.177477	0.105024	0.032571

	0.7/0.3	0.8/0.2	0.9/0.1	1.0/0.0
1	-0.315698	-0.459596	-0.603495	-0.747393
2	-0.147827	-0.184298	-0.220770	-0.257242
3	0.006000	-0.073250	-0.152500	-0.231750
4	-0.181866	-0.245230	-0.308594	-0.371958
5	0.080997	-0.050290	-0.181576	-0.312862
6	-0.315270	-0.389014	-0.462757	-0.536501
7	-0.593458	-0.728972	-0.864486	-1.000000
8	-0.039883	-0.112336	-0.184789	-0.257242

Оптимальные значения:

При важности 0.0/1.0 оптимальный выбор под номером - 5
 При важности 0.1/0.9 оптимальный выбор под номером - 5
 При важности 0.2/0.8 оптимальный выбор под номером - 5
 При важности 0.3/0.7 оптимальный выбор под номером - 5
 При важности 0.4/0.6 оптимальный выбор под номером - 5
 При важности 0.5/0.5 оптимальный выбор под номером - 5
 При важности 0.6/0.4 оптимальный выбор под номером - 5
 При важности 0.7/0.3 оптимальный выбор под номером - 5
 При важности 0.8/0.2 оптимальный выбор под номером - 5
 При важности 0.9/0.1 оптимальный выбор под номером - 3
 При важности 1.0/0.0 оптимальный выбор под номером - 3

Рисунок 57. линейная свёрстка



идеальная точка

6.5.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

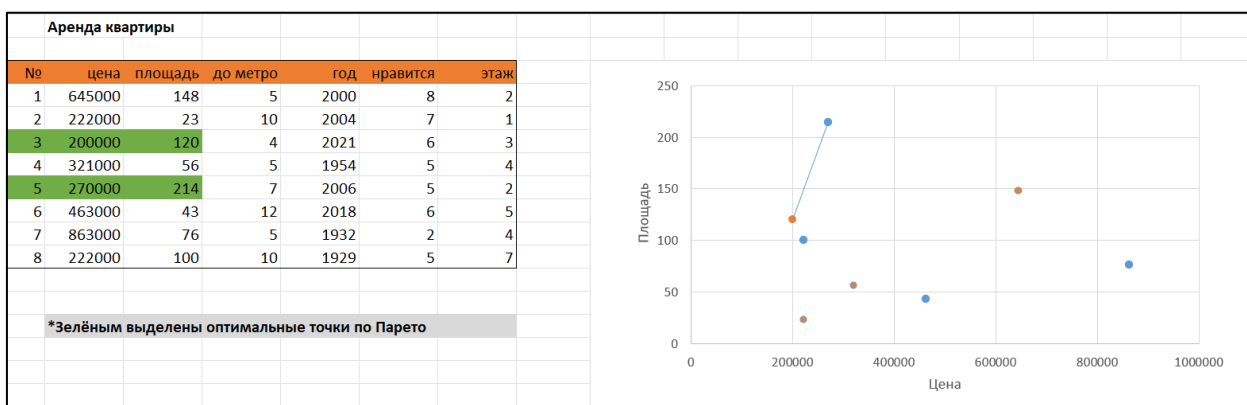


Рисунок 58. Паретооптимально

Скорость использования алгоритма	Высокая	Средняя
Простота использования	Высокая	Низкая
Надёжность	Высокая	Низкая
Точность	Высокая	Средняя

Мы считаем, что представленный рукописный код на языке Python лучше, потому что удобнее, быстрее и проще, имеет функцию импорта исходных данных, а также более наглядную визуализацию относительно Excel. Улучшением кода может послужить добавление времени выполнения запроса, более детальной выводимой информации, ручного ввода данных, в том числе генерация случайных данных.