Основы языка Python для аналитиков (семинары)

Задание 1. Изменение и сохранение данных

1. Считать данные из файла `kc_house_data.csv` с помощью pandas.

2. Добавить новый признак `price_per_sqft_living`, который будет содержать среднюю стоимость за квадратный метр жилой площади (price / sqft_living).

3. Создать новый признак `age_of_house`, который будет содержать возраст дома (текущий год минус год постройки `yr_built`).

4. Удалить признаки `zipcode`, `lat`, `long` и сохранить измененные данные в новый CSV файл `modified_data.csv`.

Подсказка № 1

Для добавления нового признака используйте выражение `data['price'] / data['sqft_living']`.

Подсказка № 2

Используйте текущий год и вычитайте год постройки `data['yr_built']` из текущего года.

Подсказка № 3

Используйте метод `drop()` для удаления указанных колонок.

Подсказка № 4

Используйте метод `to_csv()` для сохранения обновленного DataFrame в новый файл.

Эталонное решение:

```
import pandas as pd



# Загрузка данных
```

```
data = pd.read_csv('kc_house_data.csv')



# Добавление нового признака: средняя стоимость за квадратный метр
жилой площади

data['price_per_sqft_living'] = data['price'] / data['sqft_living']



# Добавление нового признака: возраст дома

current_year = pd.to_datetime('today').year

data['age_of_house'] = current_year - data['yr_built']



# Создание новых признаков: год и месяц продажи

data['year_ch'] = pd.to_datetime(data['date'],
format='%Y%m%dT000000').dt.year

data['month_ch'] = pd.to_datetime(data['date'],
format='%Y%m%dT000000').dt.month



# Удаление ненужных колонок

data_cleaned = data.drop(columns=['date', 'zipcode', 'lat', 'long'])



# Сохранение измененных данных

data_cleaned.to_csv('modified_data.csv', index=False)
```

Задача 2. Интересные объекты недвижимости

1. Считать данные из файла `kc_house_data.csv`.

2. Найти дома с видом на набережную, стоимостью выше 500000 и более чем 2000 квадратных метров жилой площади.

3. Сохранить информацию о таких домах в новый CSV файл.

Подсказка № 1

Используйте логические условия для фильтрации домов по цене, площади и виду на набережную.

Подсказка № 2

Используйте метод `to_csv()` для записи отфильтрованных данных в новый файл.

Эталонное решение:

```
import pandas as pd



# Загрузка данных

data = pd.read_csv('kc_house_data.csv')



# Фильтрация данных

filtered_homes = data[(data['waterfront'] == 1) &

                      (data['price'] > 500000) &

                      (data['sqft_living'] > 2000)]



# Сохранение данных

filtered_homes.to_csv('high_value_waterfront_homes.csv',
index=False)
```

Задача 3. Анализ жилых и общих площадей

1. Считать данные из файла `kc_house_data.csv`.

2. Построить DataFrame с колонками `sqft_living`, `condition` и `price`.

3. Определить среднюю стоимость за квадратный метр жилой площади для разных состояний домов и сохраните в текстовый файл.

Подсказка № 1

Загрузите данные из файла `kc_house_data.csv` с помощью функции `pd.read_csv()`. Убедитесь, что путь к файлу указан правильно, и проверьте, что данные корректно загружаются.

Подсказка № 2

Создайте новый DataFrame, включающий только столбцы `sqft_living`, `condition`, и `price`. Это можно сделать, выбрав нужные столбцы из загруженного DataFrame, например, используя `data[['sqft_living', 'condition', 'price']]`.

Подсказка № 3

В новом DataFrame добавьте столбец, который будет рассчитывать стоимость за квадратный метр жилой площади, используя формулу `price / sqft_living`. Затем сгруппируйте данные по `condition` и найдите среднее значение этого столбца для каждого состояния дома с помощью методов `groupby()` и `mean()`.

Подсказка № 4

Сохраните результаты анализа в текстовый файл. Откройте файл для записи с помощью `open('living_vs_condition_analysis.txt', 'w')` и используйте метод `write()` для записи результатов. Не забудьте закрыть файл после записи данных.

Эталонное решение:

```
import pandas as pd



# Загрузка данных

data = pd.read_csv('kc_house_data.csv')



# Расчет среднего соотношения жилой площади к общей площади

data['ratio'] = data['sqft_living'] / data['sqft_lot']

mean_ratio = data['ratio'].mean()



# Сохранение статистики

with open('living_vs_lot_analysis.txt', 'w') as f:
```

```
    f.write(f'Среднее соотношение жилой площади к общей площади:
{mean_ratio}\n')
```

Задача 4. Информация о клиентах и их покупках

1. Создайте датафрейм с покупками и сохраните его в переменную `purchases`.

purchases = pd.DataFrame({

    'purchase_id': [2001, 2002, 2003, 2004, 2005],

    'client_id': [1001, 1002, 1003, 1001, 1005],

    'house_id': [1234567890, 9876543210, 4567891230, 5566778899, 1122334455]

})

2. Создайте датафрейм с информацией о клиентах и сохраните его в переменную `clients_info`.

clients_info = pd.DataFrame({

    'client_id': [1001, 1002, 1003, 1004, 1005],

    'name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],

    'age': [30, 40, 35, 45, 50]

})

3. Присоедините информацию о клиентах к таблице `purchases` через метод `join` и сохраните в переменную `purchases_with_info`.

4. Присоедините информацию о клиентах к таблице `purchases` через метод `merge` и сохраните в переменную `purchases_with_info_merged`.

5. Сохраните результаты в CSV файлы `purchases_with_info.csv` и `purchases_with_info_merged.csv`.

Подсказка № 1

Для создания датафрейма с покупками, вам нужно создать таблицу, в которой будут указаны идентификаторы покупок (`purchase_id`), идентификаторы клиентов (`client_id`), и идентификаторы домов (`house_id`). Убедитесь, что все данные организованы в виде словаря, где ключи представляют собой названия столбцов, а значения — списки данных.

Подсказка № 2

Создайте датафрейм с информацией о клиентах. Этот датафрейм должен содержать столбцы, такие как `client_id` (идентификатор клиента), `name` (имя клиента), и `age` (возраст клиента). Сформируйте его аналогичным образом, как в предыдущем шаге, с соответствующими столбцами и данными.

Подсказка № 3

Чтобы присоединить информацию о клиентах к таблице покупок, используйте метод `join`. Для этого нужно установить `client_id` в качестве индекса в датафрейме с информацией о клиентах и затем присоединить этот датафрейм к таблице покупок по идентификатору клиента. Убедитесь, что индексы совпадают и информация объединена правильно.

Подсказка № 4

Вместо метода `join` можно использовать метод `merge`, который также позволяет объединять два датафрейма по общему столбцу. В данном случае объедините таблицы по столбцу `client_id`, который должен присутствовать в обоих датафреймах. Убедитесь, что вы используете правильный столбец для объединения.

Подсказка № 5

Для сохранения результатов в CSV файлы используйте метод `to_csv()`. Убедитесь, что вы указываете правильные имена файлов и устанавливаете параметр `index=False`, чтобы избежать сохранения индексов датафрейма в файле. Это обеспечит чистоту и читаемость вашего файла.

Эталонное решение:

```
import pandas as pd



# Создание датафрейма с покупками

purchases = pd.DataFrame({

    'purchase_id': [2001, 2002, 2003, 2004, 2005],
```

```
    'client_id': [1001, 1002, 1003, 1001, 1005],

    'house_id': [1234567890, 9876543210, 4567891230, 5566778899,
1122334455]

})



# Создание датафрейма с информацией о клиентах

clients_info = pd.DataFrame({

    'client_id': [1001, 1002, 1003, 1004, 1005],

    'name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],

    'age': [30, 40, 35, 45, 50]

})



# Присоединение информации о клиентах через метод join

purchases_with_info =
purchases.join(clients_info.set_index('client_id'),
on='client_id')



# Присоединение информации о клиентах через метод merge

purchases_with_info_merged = pd.merge(purchases, clients_info,
on='client_id')



# Сохранение результатов в CSV файлы

purchases_with_info.to_csv('purchases_with_info.csv', index=False)

purchases_with_info_merged.to_csv('purchases_with_info_merged.csv'
, index=False)
```