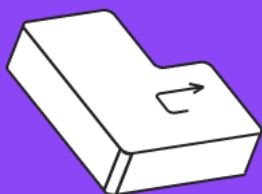


Первичный и визуальный анализ данных

Библиотеки Python для Data
Science



Оглавление

Введение	2
Термины, используемые в лекции	4
Этапы работы над Data Science проектом	4
Первичный и визуальный анализ данных	10
Заключение.	27
Что можно почитать еще?	28
Используемая литература	28

Введение

Наш новый курс посвящен Библиотекам Python для Data Science. В течении этого курса мы с вами пройдем по всем этапам работы с Data Science проектом, от первичной обработки до построения и настройки модели.

Научимся понимать данные, работать с ними, и осуществлять «магию» машинного обучения, то есть, строить математическую модель, которая может предсказать значения или классифицировать объекты.

Данный курс даст вам понимание работы классического машинного обучения, научит собирать данные, обрабатывать и строить математические модели.

В данной лекции мы разберем из каких этапов состоит работа над Data Science проектом, первичный и визуальный анализ данных . А также рассмотрим один из самых важных этапов работы над проектом, постановка задачи, получение данных, EDA. Подготовка и обработка данных определяет качество вашей модели. Данные, в которых нет пропущенных значений, исключены аномалии и выбросы обеспечит наилучшее качество.

Визуализация позволит вам наилучшим образом понять данные, определить поведение признаков, оценить зависимость признаков. Основными инструментами разведочного анализа являются изучение распределения вероятностей переменных, построение и анализ корреляционных матриц, факторный анализ, дискриминантный анализ и многомерное шкалирование.

Часто, когда все понятно, остальное не так сложно и становится делом техники. То же самое верно и в науке о данных, где этап, когда все "понятно", называется стадией Exploratory Data Analysis (EDA) или разведочный анализ данных. Таким образом, это, пожалуй, самый важный этап в проекте по науке о данных, но я знаю из собственного опыта, что этот этап всегда занимает 70-80% от общего времени проекта. Чем лучше вы знаете свой набор данных, тем лучше вы сможете его использовать.

Если вы хотите изучить машинное обучение с помощью Python, ожидается, что вы знаете:

- Основы языка Python
- Линейную алгебру
- Основы статистики и теорию вероятности

- Библиотеки Python для DataScience (Numpy, Pandas, Sklearn)

На этой лекции вы найдете ответы на такие вопросы как:

- С чего мы начинаем работу над проектом?
- Как проводить первичный анализ?
- Для чего нужен визуальный анализ?

Термины, используемые в лекции

Data Science проект — это прикладные исследования, состоит из таких этапов, как формулировка гипотез, проектирование экспериментов и, конечно, оценка результатов и их пригодности для решения конкретных случаев.

Разведочный анализ данных (Exploratory Data Analysis) – предварительное исследование Датасета (Dataset) с целью определения его основных характеристик, взаимосвязей между признаками, а также сужения набора методов, используемых для создания Модели (Model) Машинного обучения (Machine Learning).

Визуализация данных – это наглядное представление массивов различной информации.

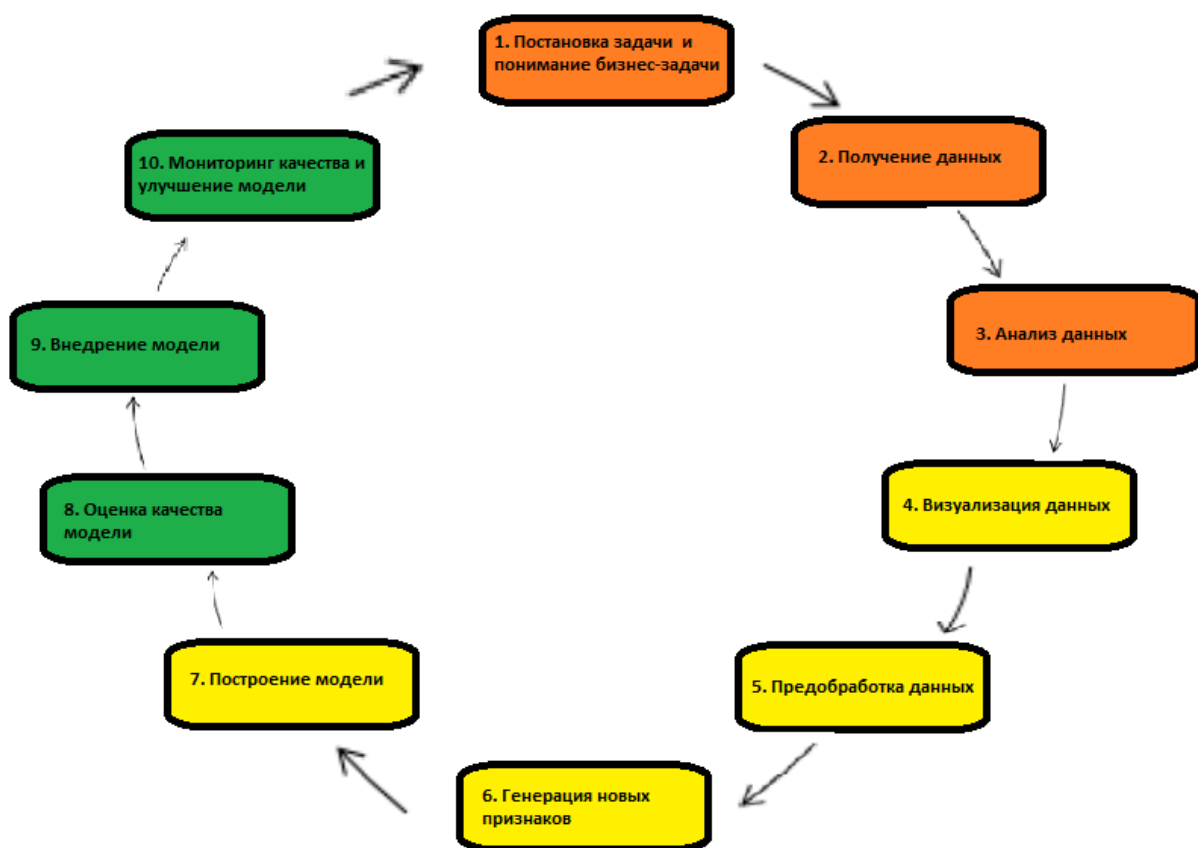
Pandas — это библиотека Python, предоставляющая широкие возможности для анализа данных.

Этапы работы над Data Science проектом

Давайте посмотрим на этапы работы Data Science (это прикладные исследования, состоит из таких этапов, как формулировка гипотез, проектирование экспериментов и, конечно, оценка результатов и их пригодности для решения конкретных случаев) проекта и более подробно разберем начальные пункты плана:

1. Постановка задачи и понимание бизнес-задачи
2. Получение данных

3. Анализ данных
4. Визуализация данных
5. Предобработка данных
6. Генерация новых признаков
7. Построение модели
8. Оценка качества модели
9. Внедрение модели
10. Мониторинг качества и улучшение модели



Процесс работы над проектом начинается с **постановки задачи или понимания бизнес-задачи**.

Мы должны вникнуть в постановку задачи, понять, какой итог требуется получить от проекта, узнать все про участников и заинтересованных лиц. Данный этап как фундамент для всей дальнейшей работы: без него ничего не построишь. На нем нужно определить цель исследования:

- Что представляет собой проблема?
- Почему проблема должна быть решена?
- Кого задевает проблема?
- Какие имеются альтернативы?

Иными словами, нужно четко определить цель заказчика. К примеру, заказчик спрашивает: можем ли мы уменьшить издержки на определенную работу? Нужно уточнить: представляет собой целью увеличить эффективность данной работы или увеличить доход от предпринимательства? Далее в соответствии с некой задачей мы должны решить, каким методом проблема будет решаться.

Подход следует выбирать в зависимости от типа требуемого ответа. Если требуются вещественные значения, то правильным выбором будут регрессионные модели. Если задача заключается в определении принадлежности к определенному классу — правильный выбор — модели классификации.

Результатом такого шага станут требования к данным: ок, проблема ясна, сейчас подумаем, а что нам может понадобиться для благоприятного решения?

Получение данных.

После того как четко определены цели исследования и выбран подход, то есть вы понимаете, какие ответы можно получить на интересующие вас вопросы, необходимо определить тип данных, которые могут дать искомые ответы. Необходимо подготовить требуемые данные, включая содержание, формат источника данных, которые будут использоваться на следующем этапе.

Этот этап включает в себя сбор данных из доступных источников. Убедитесь, что Вы выбрали доступные, надежные источники, которые позволяют получить искомые данные в желаемом качестве. После завершения первоначального сбора данных важно проверить, были ли получены желаемые данные. На этом этапе проверить требования к данным и принять решение о необходимости получения дополнительных данных. Можно выявить пробелы в данных и разработать планы по заполнению пробелов или поиску альтернативных данных.

На этом этапе вы можете получить данные путем запроса к базе данных с использованием технических навыков, таких как MySQL, данные также могут быть получены в таких форматах, как Microsoft Excel, PostgreSQL, Oracle или нереляционные базы данных (NoSQL), такие, как MongoDB. Другим способом получения данных является использование таких инструментов, как BeautifulSoup,

для анализа веб-сайта. Еще один известный метод — подключение к веб-интерфейсу API для сбора данных. И, конечно, самый традиционный метод получения данных — непосредственно из файлов, сохраненных в формате CSV (значения, разделенные запятыми) или TSV (значения, разделенные вкладками), например, загруженных с сайта Kaggle или существующих данных компании.

Поиск и извлечение данных часто требует экспертизу, важно хоть немного разбираться в теме, и интуицию, которая выходит за рамки технического исследования, понимания природы и формата данных, что может обеспечить только опыт и знание рассматриваемой области практики.

Следующим этапом, который необходимо рассмотреть подробно, является **анализ данных**. Проще говоря, этот этап требует ответа на вопрос: "Являются ли собранные данные репрезентативными для рассматриваемой проблемы?".

Здесь необходима описательная статистика. Изучите меры центральной тенденции (среднее значение, медиана, мода), найдите выбросы и оцените изменчивость (обычно это межквартильный диапазон, дисперсия и стандартное отклонение). Также строится гистограмма распределения переменных. Гистограммы — хороший инструмент для понимания того, как распределены значения данных и какие приготовления необходимо сделать, чтобы переменные были максимально полезны при построении модели. Другие инструменты визуализации, такие как усы, также могут быть полезны.

Затем проводится попарное сравнение. Вычисляются корреляции между переменными, чтобы определить, какие переменные связаны между собой и в какой степени. Если между переменными обнаружены значительные корреляции, некоторые из них могут быть отброшены как лишние.

Одним из очень важных этапов является **подготовка данных** (включает в себя визуализацию, предобработку и генерацию новых признаков).

Наряду со сбором и анализом данных, подготовка данных является одним из самых ресурсоёмких видов деятельности в проекте. Эта фаза может занимать 70% или даже 90% времени проекта. На этом этапе данные обрабатываются в форме, удобной для работы. Это включает удаление дубликатов данных, обработку недостающих или неверных данных, проверку ошибок форматирования и их исправление при необходимости.

На данном этапе мы визуализируем данные для лучшего понимания ваших данных.

Также формируется набор элементов, которые машинное обучение будет обрабатывать на следующем этапе. Извлекаются и отбираются атрибуты, которые могут быть полезны при решении бизнес-задач. На этом этапе особое внимание следует уделить ошибкам, поскольку они могут оказаться фатальными для всего проекта. Слишком много признаков приведет к переобучению модели, а слишком мало — к недообучению. В данном случае, нужно придерживаться золотой середины, так как проклятие размерности — одна из самых больших проблем в машинном обучении: чем выше размерность, тем более разрежены данные. Иными словами, по мере роста количества признаков объем данных, которые нам нужно обобщить, растет экспоненциально.

Следующий шаг — **построение модели**

Выбор модели, как мы видели, осуществляется в начале работы и зависит от бизнес-задачи. Поэтому, как только тип модели определен и имеется обучающая выборка, аналитик разрабатывает модель и проверяет, как она работает с набором функций, созданным на предыдущем этапе.

Другим важным шагом является **разработка модели**. На данном этапе мы, в случае необходимости, проводим **оценку качества модели**.

Разработка модели тесно связана с построением модели. Расчеты перемежаются с подгонкой модели. На этом этапе необходимо ответить на вопрос, удовлетворяет ли построенная модель бизнес-задаче.

В расчете модели есть два этапа. Проводятся диагностические измерения, чтобы понять, работает ли модель так, как предсказано. Если используется прогностическая модель, можно использовать дерево решений, чтобы понять, соответствует ли выход модели первоначальному плану. На втором этапе проводится проверка статистической значимости гипотез. Это необходимо для того, чтобы убедиться, что данные модели используются и интерпретируются правильно и что результат находится в пределах статистической погрешности.

По-настоящему захватывающим и кульминационным этапом является **внедрение!**

Если модель дает удовлетворительный ответ на наш вопрос, то она обязательно будет полезной. Когда модель разработана и аналитики уверены в своей работе, им необходимо представить разработанный инструмент своему


клиенту. При этом важно привлечь не только владельца продукта, но и маркетинг, разработчиков, системных администраторов и других заинтересованных лиц, на которых можно повлиять для более эффективного использования результатов проекта. Следующий шаг - переход к внедрению.


Внедрение можно проводить поэтапно, например, с ограниченной группой пользователей или в тестовой среде. Необходимо также создать систему обратной связи, чтобы отслеживать, насколько хорошо работает разработанная модель. Через некоторое время эта обратная связь может быть полезна для улучшения модели. Это также может привести к появлению новых источников данных и новых заинтересованных сторон и улучшить саму бизнес-задачу.


Таким образом, нет предела совершенству, и даже внедренные модели не являются совершенными.

Этап **мониторинга качества и улучшения модели** очень важен для сохранения жизнедеятельности вашей модели. Это заключительный этап, на котором модель машинного обучения переобучается в производственной среде при получении новых данных или при необходимых действиях для поддержания производительности модели машинного обучения.

Четко определенный рабочий процесс облегчает работу специалистов по исследованию данных. Приведенный выше жизненный цикл не является окончательным и может время от времени меняться для повышения эффективности конкретного проекта.

 Соберите все имеющиеся в вашем распоряжении данные. Вы не знаете точно, какие данные вам понадобятся, и у вас может быть только один шанс собрать их.

 Несколько источников данных дают более полную картину и позволяют лучше оценить поведение пользователей в контексте, но обратной стороной является разнообразие форматов и стоимость их интеграции в аналитическую систему. Не всегда все данные возможно собрать воедино, а если и возможно, то не всегда это необходимо.

 Все, как менеджеры, так и аналитики, предпочитают бесплатные данные, но иногда качественная информация доступна только в том случае, если за нее

приходится платить. В этих случаях следует рассмотреть обоснование покупки и сравнить стоимость данных с их ценностью.

Если у вас много данных, можете ли вы обеспечить их хранение и обработку? Ценные данные не всегда доступны в больших объемах, и наоборот.

Первичный и визуальный анализ данных

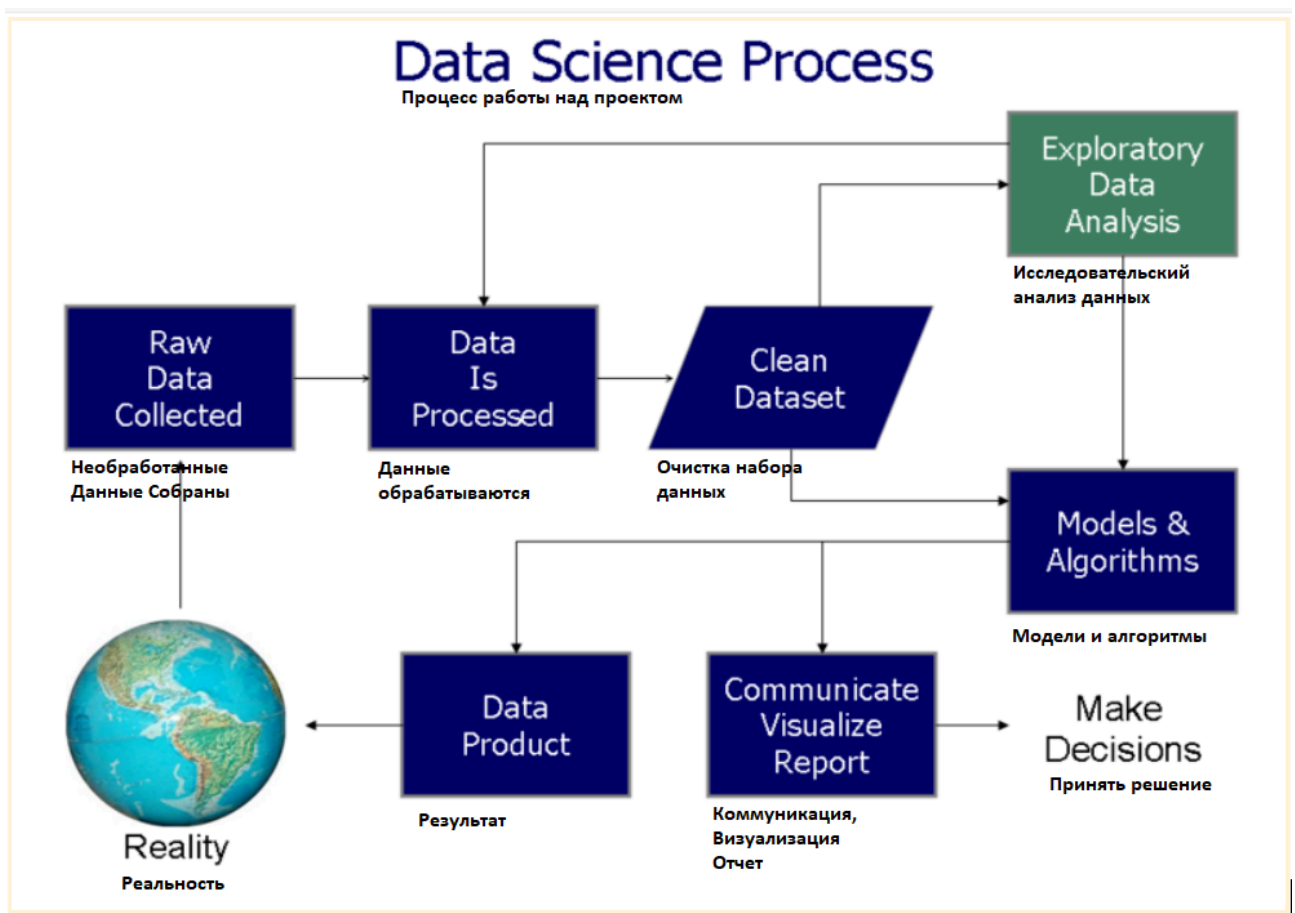
Итак, вы собрали данные, определили их.

Возьмем для примера задачу по предсказанию стоимости жилья. Цель — продемонстрировать умение работы с сырыми данными — их подготовка, чистка для последующего анализа, исследование и визуализация данных.

Для того чтобы начать работу с данными мы будем использовать уже известные Вам библиотеки Pandas и numpy.

Часто самое сложное — это не рисовать на картинке, а рисовать на чистом листе бумаги. Точно так же и в науке о данных бывает трудно понять, с чего начать, когда у вас есть набор данных. Именно здесь в игру вступает разведочный анализ данных (EDA).

Прежде чем перейти к рассмотрению EDA, важно понять, как EDA вписывается в общий процесс обработки данных.



[Farcaster из Википедия](#)

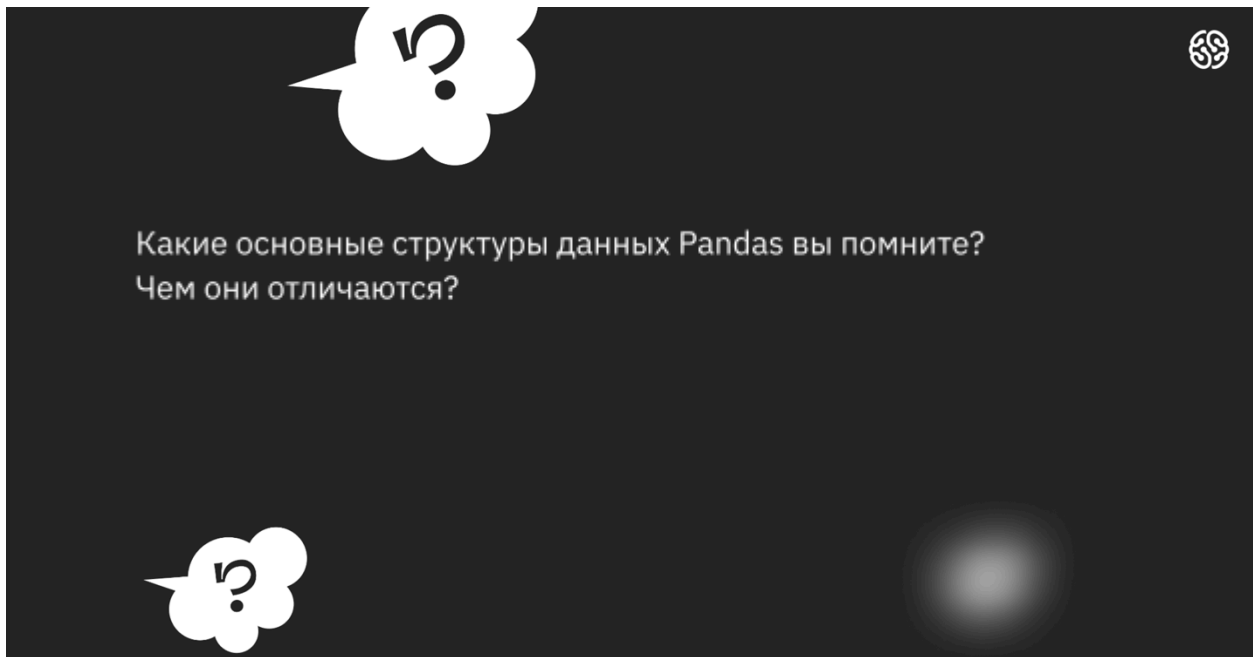
Исследовательский анализ данных — это процесс анализа или понимания данных и извлечения идей или основных характеристик данных. EDA обычно подразделяют на два метода, т.е. визуальный анализ и первичный анализ.

Технически основной целью EDA является:

- Изучение распределение данных
- Обработка отсутствующих значений набора данных (наиболее распространенная проблема с каждым набором данных)
- Обработка выбросов
- Удаление повторяющихся данных
- Кодирование категориальных переменных
- Нормализация и масштабирование

Pandas — это библиотека Python, предоставляющая широкие возможности для анализа данных. Данные, с которыми работают дата-саентисты, часто хранятся в форме табличек — например, в форматах .csv, .tsv или .xlsx. С помощью библиотеки

Pandas такие табличные данные очень удобно загружать, обрабатывать и анализировать.



Задание: Какие основные структуры данных Pandas, чем они отличаются?



Основными структурами данных Pandas являются классы **Series** и **DataFrame**. **Series** представляет собой одномерную индексированную таблицу данных фиксированного типа; **DataFrame** - двумерную структуру данных, где каждый столбец таблицы содержит данные одного типа; его можно представить как

словарь объектов типа Series; структура DataFrame отлично подходит для представления реальных данных. Структура DataFrame отлично подходит для представления реальных данных. Ее строки соответствуют описанию атрибутов каждого объекта, а столбцы — его атрибутам.

Посмотрим пример анализа данных на примере датасета предсказание стоимости жилья

```
[1] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Библиотеку Numpy мы будем использовать для математических вычислений. Matplotlib и Seaborn нужны для визуализации данных.

Загружаем датасет:

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

Как мы уже знаем, посмотреть на первые или последние несколько (по умолчанию, пять) значений можно с помощью методов .head() и .tail() соответственно.

Мы увидим таблицу наших данных:

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_basement	yr_built	yr_renovated	street	city	statezip	country
0	2014-05-02 00:00:00	313000.0	3.0	1.50	1340	7912	1.5	0	0	3	1340	0	1955	2005	18810 Denamore Ave N	Shoreline	WA 98133	USA
1	2014-05-02 00:00:00	2384000.0	5.0	2.50	3650	9050	2.0	0	4	5	3370	280	1921	0	709 W Blaine St	Seattle	WA 98119	USA
2	2014-05-02 00:00:00	342000.0	3.0	2.00	1930	11947	1.0	0	0	4	1930	0	1966	0	26206-26214 143rd Ave SE	Kent	WA 98042	USA
3	2014-05-02 00:00:00	420000.0	3.0	2.25	2000	8030	1.0	0	0	4	1000	1000	1963	0	857 170th Pl NE	Bellevue	WA 98008	USA
4	2014-05-02 00:00:00	550000.0	4.0	2.50	1940	10500	1.0	0	0	4	1140	800	1976	1992	9105 170th Ave NE	Redmond	WA 98052	USA

Выведем дополнительную информацию о датасете

```
0
сек. data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 18 columns):
#   Column                Non-Null Count  Dtype
---  -
0   date                   4600 non-null   object
1   price                  4600 non-null   float64
2   bedrooms               4600 non-null   float64
3   bathrooms              4600 non-null   float64
4   sqft_living            4600 non-null   int64
5   sqft_lot               4600 non-null   int64
6   floors                 4600 non-null   float64
7   waterfront             4600 non-null   int64
8   view                   4600 non-null   int64
9   condition              4600 non-null   int64
10  sqft_above             4600 non-null   int64
11  sqft_basement          4600 non-null   int64
12  yr_built                4600 non-null   int64
13  yr_renovated           4600 non-null   int64
14  street                 4600 non-null   object
15  city                   4600 non-null   object
16  statezip               4600 non-null   object
17  country                4600 non-null   object
dtypes: float64(4), int64(9), object(5)
memory usage: 647.0+ KB
```

Итак, в датасете представлены 4600 объектов и 18 признаков, которые категориально можно разделить на:

- внутренние характеристики дома (bathrooms, sqft_living (площадь жилья), sqft_lot (общая жилая площадь) и другие...)
- переменные локации дома (street, city, statezip, country)
- дата и цена

Типы наших переменных:

Дискретные переменные — bedrooms, sqft_living, sqft_lot, bedrooms, sqft_above, sqft_basement, yr_built, date (могут принимать набор четко разделяемых значений)

Непрерывные переменные — price, floors (могут принимать любое значение в определенном интервале)

Номинативные переменные — street, city, statezip, country, waterfront (каждое значение переменной указывает на принадлежность объекта к группе (категории))

Ранговые переменные — view, condition (переменные, в которых можно установить порядок между значениями)

Обратим внимание, что признак datetime имеет тип данных object, что не соответствует действительности, данный признак будет более информативен как дата:

```
1 data['date'] = pd.to_datetime(data['date'])
```

```
data['date'] = pd.to_datetime(data['date'])
```

Мы видим, что признак поменял тип данных

```
0    date    4600 non-null    datetime64[ns]
```

И в заключение нашей подготовки данных давайте переведем измерения sqft_living и sqft_basement из кв.фут в кв.м для более четкого понимания объема площади недвижимости.

```
1 data['sqm_living'] = round(data.sqft_living/10.764,2)
2 data['sqm_basement'] = round(data.sqft_basement/10.764,2)
```

```
data['sqm_living'] = round(data.sqft_living/10.764,2)
```

```
data['sqm_basement'] = round(data.sqft_basement/10.764,2)
```

Проверим, есть ли в наших данных пропуски. В результате использование info(), можно сделать вывод о том, что пропусков нет, но можно посмотреть и с помощью других функций:

```
1 data.isna().sum()
```

```
data.isna().sum()
```

Мы убеждаемся, что пропусков нет, пропуски негативно влияют на работу модели.

```
date          0
price         0
bedrooms      0
bathrooms     0
sqft_living   0
sqft_lot      0
floors        0
waterfront    0
view          0
condition     0
sqft_above    0
sqft_basement 0
yr_built      0
yr_renovated  0
street        0
city          0
statezip      0
country       0
dtype: int64
```

При наличии пропусков нужно решить, что можно с ними сделать:

1. Можно удалить, в том случае, если пропусков немного относительно всей выборки.
2. Если пропусков много (более 50% от общего количества объектов), то, вероятно, от признака стоит избавиться.
3. Пропуски можно заменить медианой или модой (в случае категориальных признаков). Медиана более стабильна нежели среднее арифметическое. Выбросы оказывают сильное влияние на среднее арифметическое.
4. В ряде случаев пропуски можно заменить, используя взаимное влияние, то есть, с учетом других признаков восстановить пропуски в нужном признаке.
5. Можно использовать модель для восстановления пропущенных значений.

Небольшая обработка данных завершена, поэтому, давайте переходить к более детальному исследованию данных

Обратим внимание на признак statezip:

```
1 data.statezip.value_counts()
```

```
1 WA 98103      148
2 WA 98052      135
3 WA 98117      132
4 WA 98115      130
5 WA 98006      110
6      ...
7 WA 98047        6
8 WA 98288        3
9 WA 98050        2
10 WA 98354        2
11 WA 98068        1
12 Name: statezip, Length: 77, dtype: int64
```

Мы видим, что в нашем датасете 77 районов, попробуем сгруппировать данные:

```
1 data.groupby('city')\
2     .agg({'country':'count'})\
3     .sort_values(by='country',ascending=False)\
4     .rename(columns={'country':'amount_of_sale_houses'})\
5     .count()
```

Получим результат:

```
1 amount_of_sale_houses      44
2 dtype: int64
```

Обратим внимание на саму таблицу:

```
1 data.groupby('city')\
2     .agg({'country':'count'})\
3     .sort_values(by='country',ascending=False)\
4     .rename(columns={'country':'amount_of_sale_houses'})\
5     .head(10)
```

amount_of_sale_houses	
city	
Seattle	1573
Renton	293
Bellevue	286
Redmond	235
Issaquah	187
Kirkland	187
Kent	185
Auburn	176
Sammamish	175
Federal Way	148

В нашем наборе данных вся недвижимость США расположена в одном штате - Вашингтон. Количество городов, в которых дома выставлены на продажу, составляет 44, а в десятку городов с наибольшим количеством продаж недвижимости входят Сиэтл, Рентон, Белвью, Редмонд, Иссакуа, Киркланд, Кент, Оберн, Сан-Мамиш и Федерал Уэй.

Выведем описательную статистику переменных

```
1 data.describe()
```

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_basement	yr_built	yr_renovated	sqm_living	sqm_basement
count	4.600000e+03	4600.000000	4600.000000	4600.000000	4.600000e+03	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000
mean	5.519630e+05	3.400870	2.160815	2139.346957	1.485252e+04	1.512065	0.007174	0.240652	3.451739	1827.265435	312.081522	1970.786304	808.608261	198.750185	28.993000
std	5.638347e+05	0.908848	0.783781	963.206916	3.588444e+04	0.538288	0.084404	0.778405	0.677230	862.168977	464.137228	29.731848	979.414536	89.484119	43.119283
min	0.000000e+00	0.000000	0.000000	370.000000	6.380000e+02	1.000000	0.000000	0.000000	1.000000	370.000000	0.000000	1900.000000	0.000000	34.370000	0.000000
25%	3.228750e+05	3.000000	1.750000	1460.000000	5.000750e+03	1.000000	0.000000	0.000000	3.000000	1190.000000	0.000000	1951.000000	0.000000	135.640000	0.000000
50%	4.609435e+05	3.000000	2.250000	1980.000000	7.683000e+03	1.500000	0.000000	0.000000	3.000000	1590.000000	0.000000	1976.000000	0.000000	183.950000	0.000000
75%	6.549625e+05	4.000000	2.500000	2620.000000	1.100125e+04	2.000000	0.000000	0.000000	4.000000	2300.000000	610.000000	1997.000000	1999.000000	243.400000	56.670000
max	2.659000e+07	9.000000	8.000000	13540.000000	1.074218e+06	3.500000	1.000000	4.000000	5.000000	9410.000000	4820.000000	2014.000000	2014.000000	1257.900000	447.790000

Согласно данным, средний профиль выставленных на продажу объектов оценивается в \$551 1963, с тремя спальнями, двумя ванными комнатами, 198,7 квадратных футов жилой площади и 28,7 квадратных футов подвала, с оценкой

качества 3,45. Дома, выставленные на продажу, являются объектами вторичного рынка, со средним годом постройки 1970.

Посмотрим на нашу целевую переменную. Так как, наиболее интуитивно понятная задача, которую можно поставить по этим данным - предсказание стоимости недвижимости.

```
1 data.price.value_counts()
```

```
0.0          49
300000.0      42
400000.0      31
440000.0      29
450000.0      29
..          ..
684680.0       1
609900.0       1
1635000.0      1
1339000.0      1
220600.0       1
Name: price, Length: 1741, dtype: int64
```

В наших данных есть недвижимость , стоимость которой составляет \$0. Мы понимаем, что цена дома не может составлять \$0, такая картина говорит нам о недостаточности данных по этим объектам. Возможно, при сборе этого датасета не удалось найти цены по всем объектам, которые выставались на продажу.

У нас есть дома без спален и ванных комнат. Давайте попробуем найти эти объекты:

```
1 data.groupby(['bedrooms','bathrooms'])\
2   .agg({'price':['count','min','max']})
```

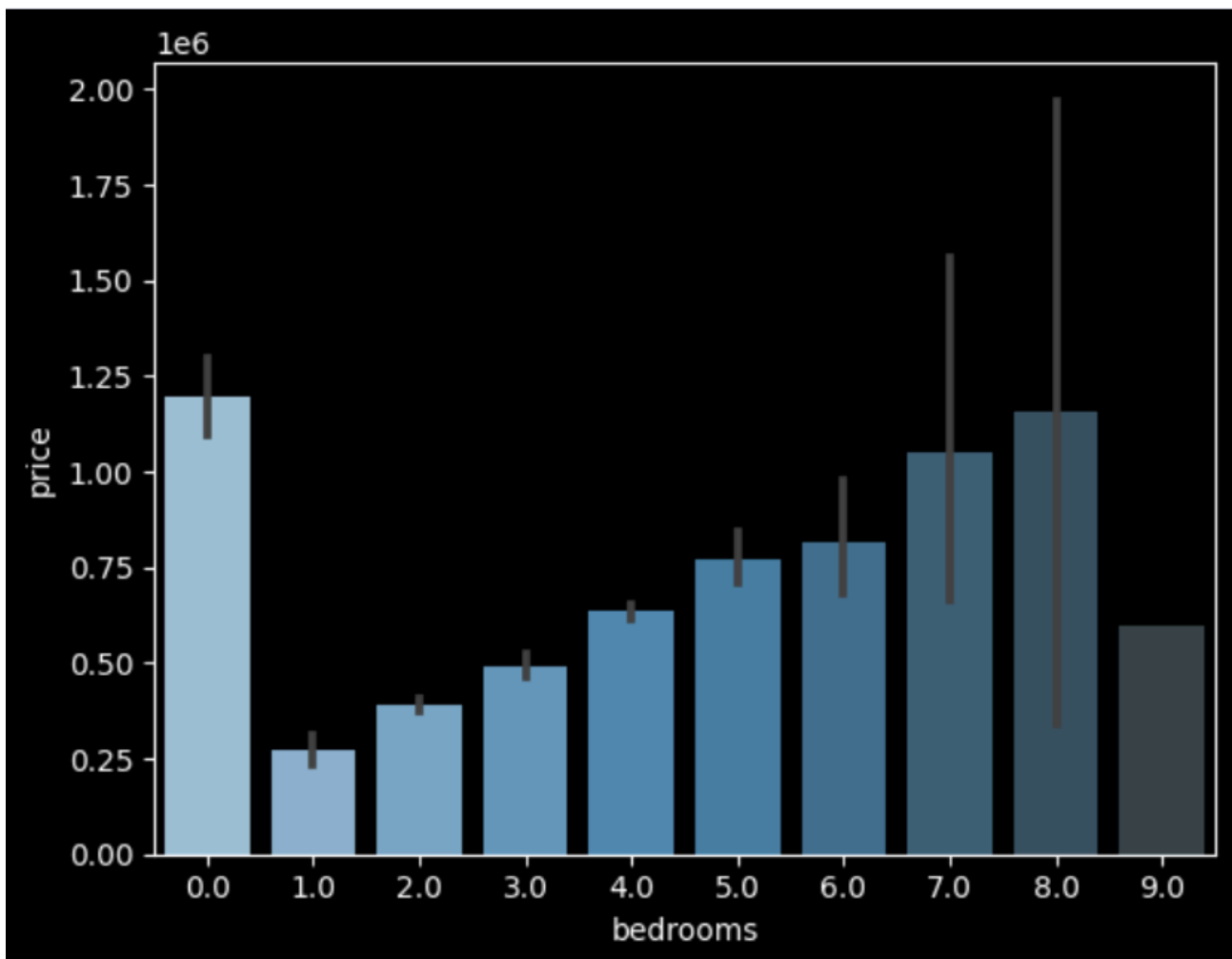
		price		
		count	min	max
bedrooms	bathrooms			
0.0	0.00	2	1095000.0	1295648.0
1.0	0.75	8	80000.0	527550.0
	1.00	23	0.0	540000.0
	1.25	1	516500.0	516500.0
	1.50	3	235000.0	410000.0
...	
7.0	5.75	1	540000.0	540000.0
	8.00	1	2280000.0	2280000.0
8.0	2.75	1	340000.0	340000.0
	3.50	1	1970000.0	1970000.0
9.0	4.50	1	599999.0	599999.0
[102 rows x 3 columns]				

Мы видим, что таких объектов - 2.

Есть очень хороший принцип, если не понимаешь - рисуй. Далее, чтобы удобно было работать и исследовать данные будем их визуализировать.

Посмотрим, на выбросы данные по спальням, используя barplot библиотека seaborn

```
1 sns.barplot(x=data.bedrooms, y=data.price, palette="Blues_d")
```



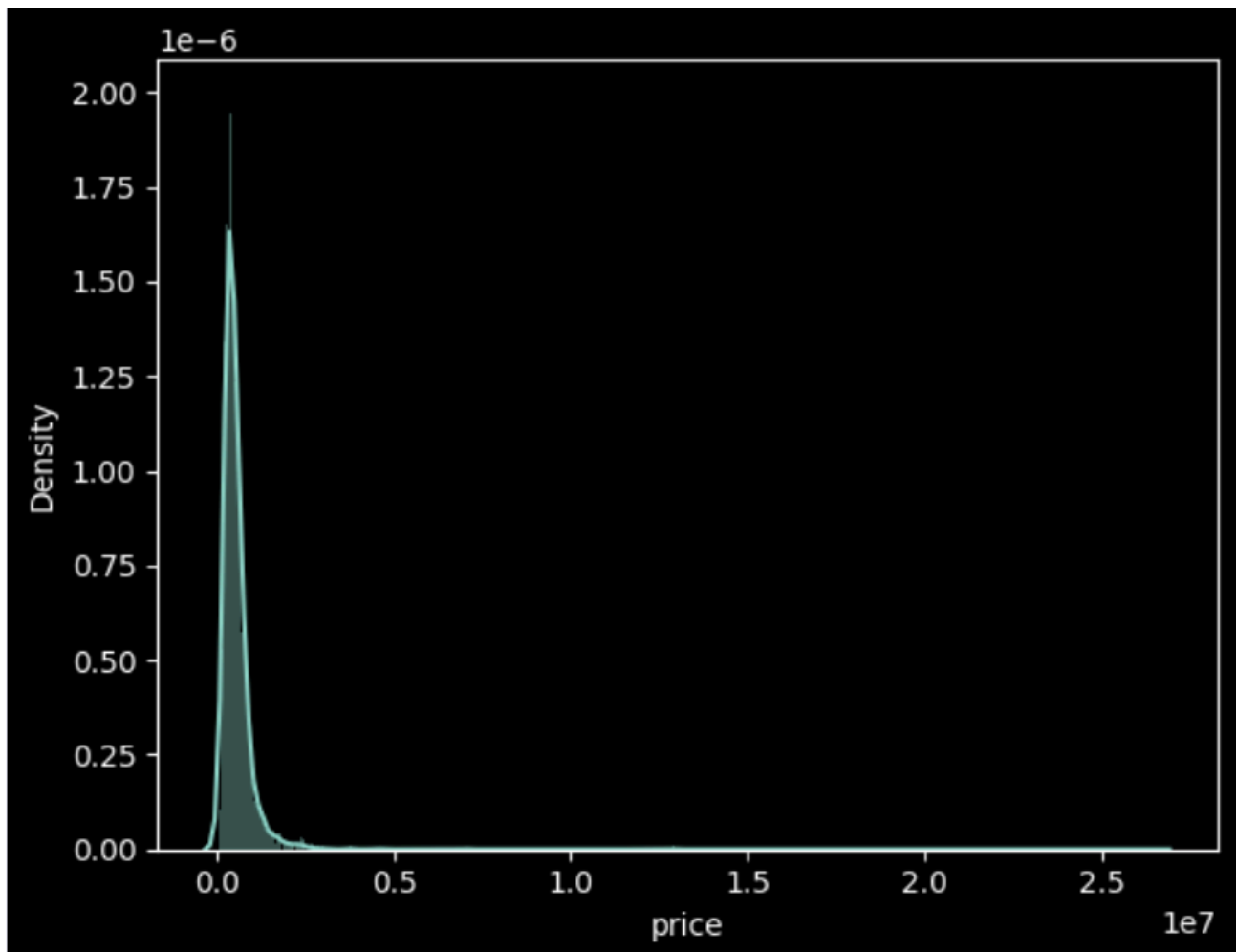
Этот barplot иллюстрирует еще один выброс — объект, который имеет 9 спален, но имеет цену меньше чем у дома с 4 спальнями. Такое поведение переменной можно объясниться тем, что дом может находиться далеко от центра города или за пределами города, где спрос на такую недвижимость очень низкий, поэтому стоимость такой недвижимости значительно ниже.

Еще один вывод, который можно сделать из графика, что недвижимость с 7-8 спальнями имеет наиболее большой разброс по ценам. Покупатель на рынке может найти недвижимость с 8 спальнями по цене недвижимости с 1-й и 2-й спальнями.

Такие недочеты следует находить именно при EDA анализе, так как, во-первых, это дает нам более глубокое понимание структуры данных, а во-вторых, при построении моделей, например регрессионных моделей, на основе этих данных от таких выбросов следует избавляться, чтобы модель была более точной.

Посмотрим на распределение целевой переменной. Целевой переменной мы называем признак, значение, которого мы хотим предсказать.

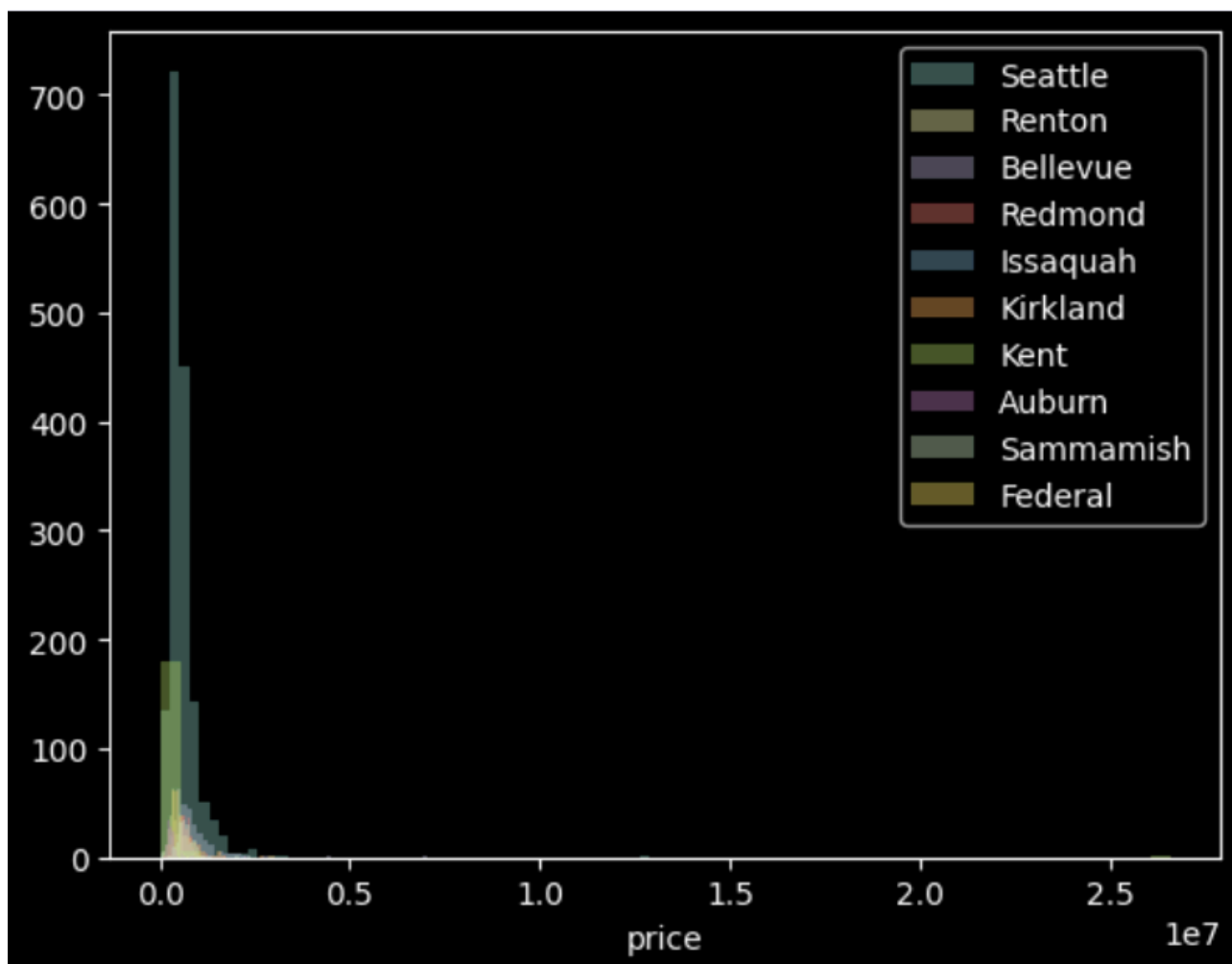
```
1 sns.distplot(data['price'],bins=1000)
```



Какие выводы мы можем сделать:

Displot показывает нам, что цена объектов недвижимости распределена не нормально и имеет вытянутый хвост вправо, что говорит, что датасет включает недвижимость с высокой стоимостью, попробуем здесь разобраться.

```
1 sns.distplot(data.query('city = "Seattle"').price, kde=False,
  label="Seattle")
2 sns.distplot(data.query('city = "Renton"').price, kde=False,
  label="Renton")
3 sns.distplot(data.query('city = "Bellevue"').price, kde=False,
  label="Bellevue")
4 sns.distplot(data.query('city = "Redmond"').price, kde=False,
  label="Redmond")
5 sns.distplot(data.query('city = "Issaquah"').price, kde=False,
  label="Issaquah")
6 sns.distplot(data.query('city = "Kirkland"').price, kde=False,
  label="Kirkland")
7 sns.distplot(data.query('city = "Kent"').price, kde=False,
  label="Kent")
8 sns.distplot(data.query('city = "Auburn"').price, kde=False,
  label="Auburn")
9 sns.distplot(data.query('city = "Sammamish"').price, kde=False,
  label="Sammamish")
10 sns.distplot(data.query('city = "Federal"').price, kde=False,
  label="Federal")
11
12
13 plt.legend()
14 plt.show()
```

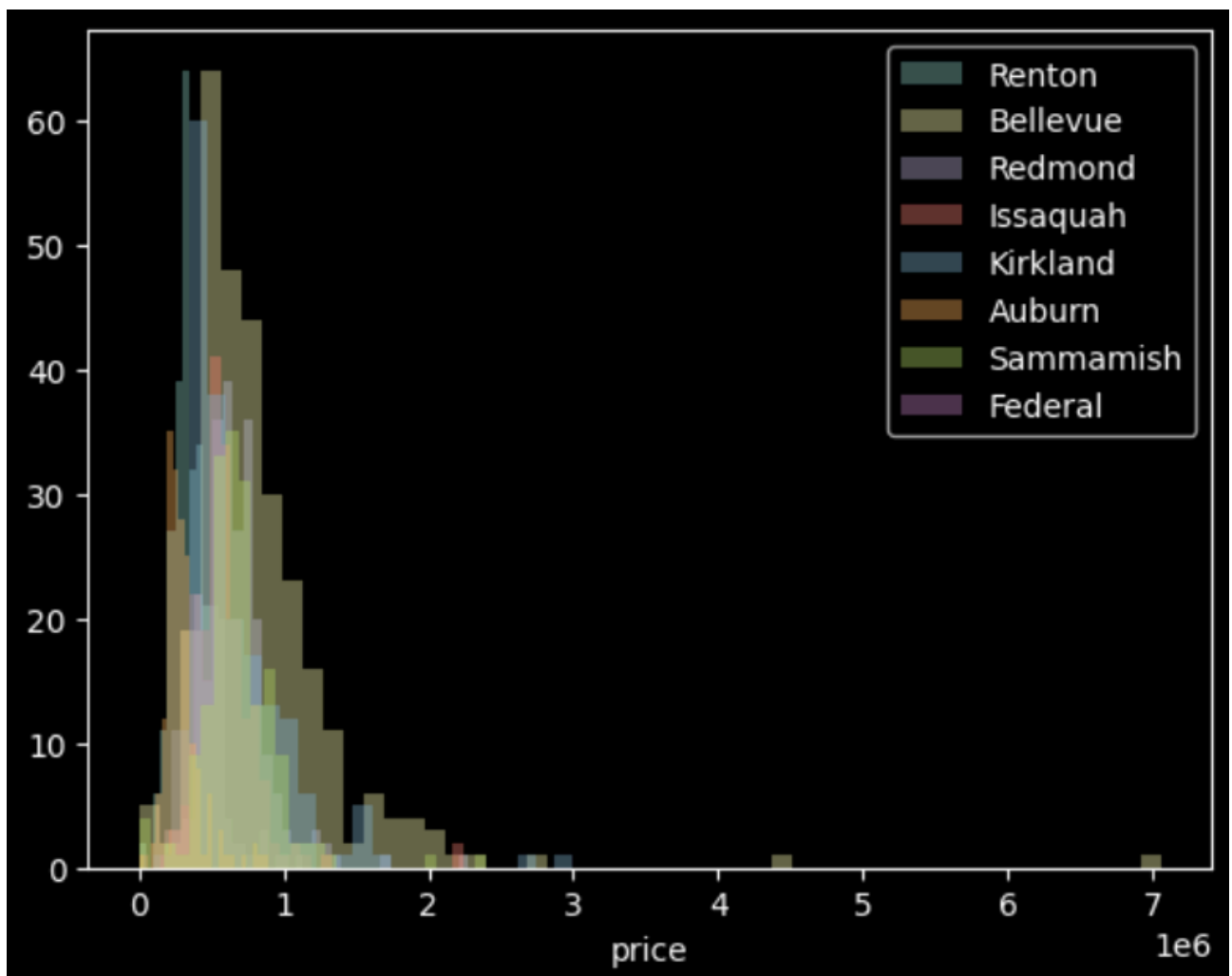


Из графика видно, что наиболее доступная по цене недвижимость располагается в городе Seattle, в городе Kent. Но давайте попробуем убрать наиболее выбивающиеся значения, чтобы посмотреть распределение по другим городам.


```

1 sns.distplot(data.query('city = "Renton"').price, kde=False,
  label="Renton")
2 sns.distplot(data.query('city = "Bellevue"').price, kde=False,
  label="Bellevue")
3 sns.distplot(data.query('city = "Redmond"').price, kde=False,
  label="Redmond")
4 sns.distplot(data.query('city = "Issaquah"').price, kde=False,
  label="Issaquah")
5 sns.distplot(data.query('city = "Kirkland"').price, kde=False,
  label="Kirkland")
6 sns.distplot(data.query('city = "Auburn"').price, kde=False,
  label="Auburn")
7 sns.distplot(data.query('city = "Sammamish"').price, kde=False,
  label="Sammamish")
8 sns.distplot(data.query('city = "Federal"').price, kde=False,
  label="Federal")
9
10
11 plt.legend()
12 plt.show()

```



Из этого графика видно, что наиболее дорогая недвижимость находится в городе Bellevue, а также распределение цены в городе Sammamish имеет хвост, который уходит вправо.

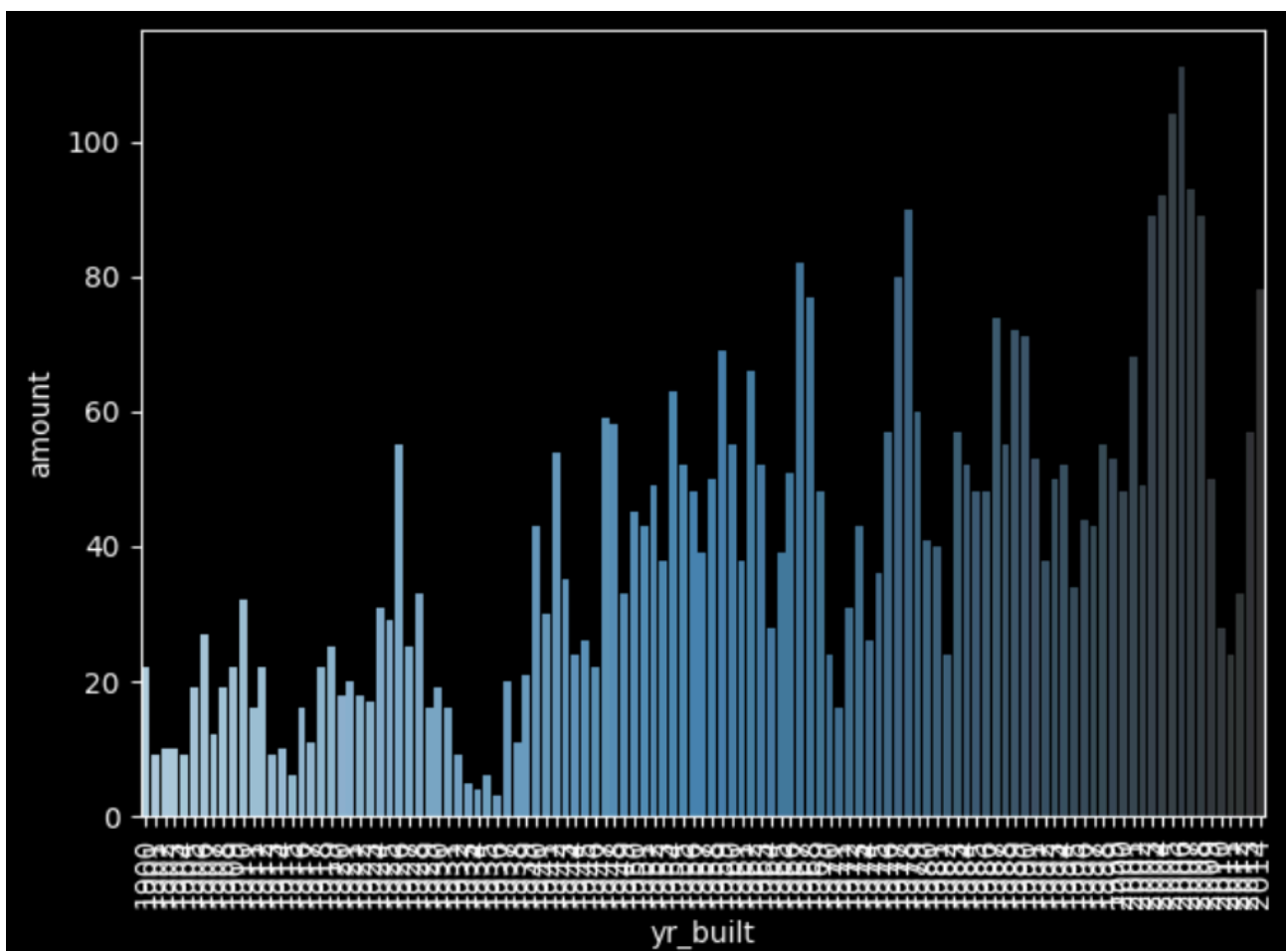
К самым низким по цене недвижимости и количеству недвижимости относится город Auburn.

Рассмотрим признак, который определяет год постройки. Сгруппируем данные по цене:

```
1 year_data = data.groupby('yr_built', as_index=False)\
2   .agg({'price': 'count'})\
3   .rename(columns={'price': 'amount'})
```

Отобразим на графике, чтобы отследить тенденцию

```
1 sns.barplot(x=year_data.yr_built, y=year_data.amount,
2             palette="Blues_d")
3 plt.xticks(rotation=90)
4 plt.tight_layout()
5 plt.figure(figsize=(60,20))
```



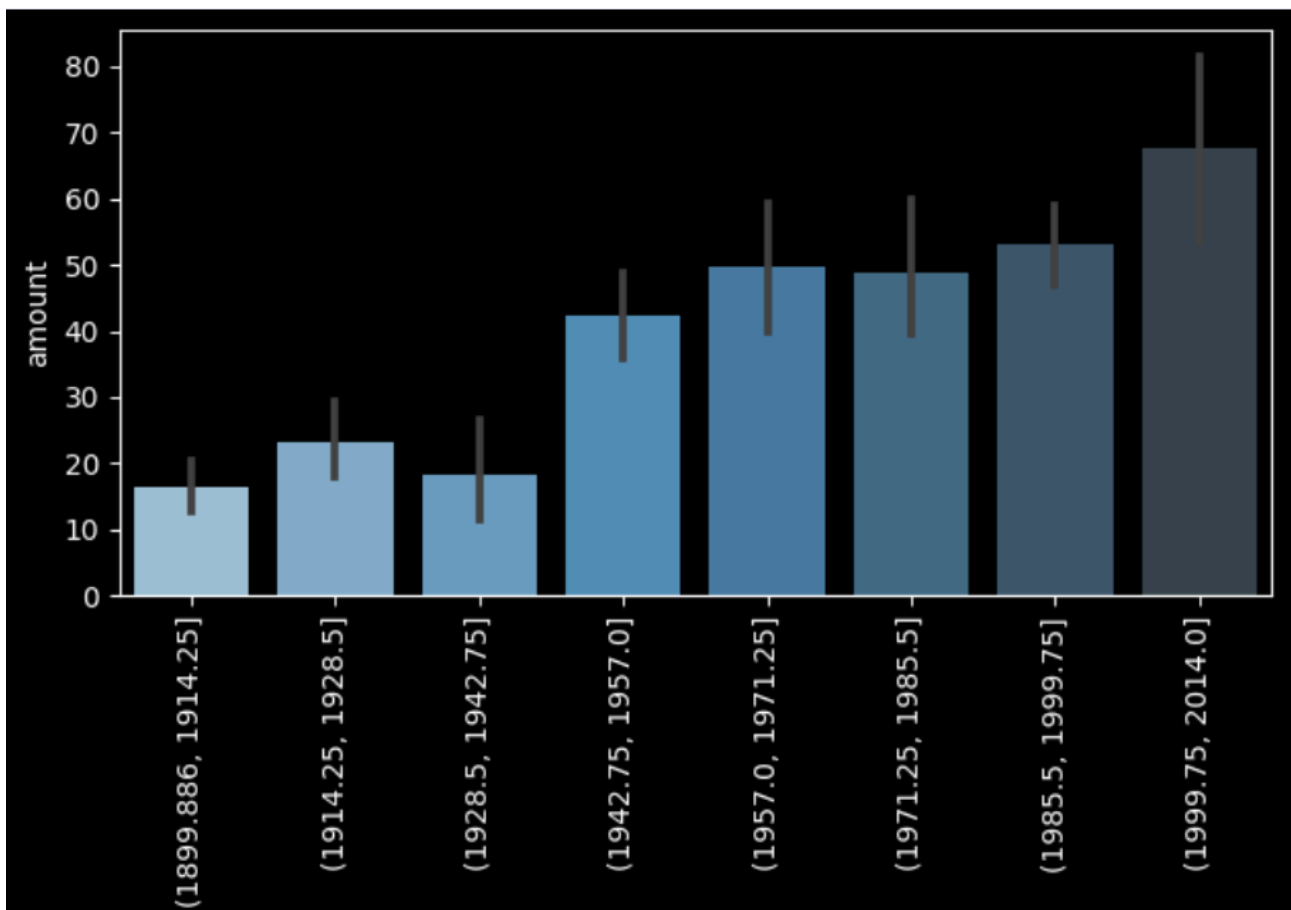
На графике по x-оси слишком много данных, что приводит к затруднениям при анализе графика. Одним из решением проблемы является создание временных батчей, которые объединяют объявления о продаже объектов недвижимости.

Категоризируем наш признак на 8 батчей:

```
1 new=pd.cut(year_data['yr_built'],bins=8)
2 year_data['batch'] = new
3 year_data.head(10)
```

Визуализируем данные:

```
1 sns.barplot(x=year_data.batch, y=year_data.amount, palette="Blues_d")
2 plt.xticks(rotation=90)
3 plt.tight_layout()
4 plt.figure(figsize=(15,8))
```



Анализ распределения количества недвижимости по годам показывает, что чаще всего на продажу выставляются новые объекты недвижимости от 2000 г. и выше годов постройки.

Сегодня мы с вами рассмотрели одномерный EDA.

Заключение.

Одномерный анализ — это анализ переменных по отдельности. Если переменная анализируется отдельно от других переменных, категориальных или непрерывных, это называется одномерным анализом.

Визуализация — это мощный инструмент для донесения мыслей и идей до конечных пользователей, а также помощь в восприятии и анализе данных. Однако, как и любой другой инструмент, визуализация должна применяться в нужное время и в нужном месте. В противном случае информация может быть воспринята медленно или даже неверно.

EDA — один из самых важных этапов проекта по науке о данных. Этот этап не только определяет направление проекта, но и помогает максимально эффективно использовать набор данных.

На лекции мы рассмотрели все ключевые понятия, связанные с процессом EDA. Кроме того, мы шаг за шагом отработали часть основной методологии EDA, используя практические наборы данных.

Однако это только начало — вы увидите, что мир EDA гораздо более разнообразен и подробен. Лучший способ научиться — это попробовать свои собственные EDA с использованием наборов данных для проектов машинного обучения.

Что можно почитать еще?

1. [Первичный анализ данных с Pandas](#)
2. [Разведочный анализ данных \(EDA\)](#)
3. [Exploratory Data Analysis\(EDA\)](#)

Используемая литература

1. EDA под другим углом. Ananiev_Genrih. 20 дек 2019
2. Data Science. Наука о данных с нуля. Джоэл Грас. 2021
3. Machine Learning with Python Cookbook. Chris Albon. 2019