## 1) Лекция 10. Визуализация данных с Matplotlib и Seaborn

## 2) Краткая цель лекции:

a) Познакомиться с видами графиков

b) Научиться строить и интерпретировать графики

c) Рассмотреть визуальный анализ данных

d) Понять, как можно анализировать геоданные

## 3) Дополнительные материалы к лекции

(ссылки, видео, файлы и т.п.)

## 4) Краткая выжимка, о чём говорилось в предыдущей лекции (если она была), 2-3 предложения

На предыдущей лекции знакомились с анализом данных с помощью библиотеки Pandas. Обсудили создание новых характеристик данных, группировки, объединения и встроенную визуализацию.

## 5) Термины лекции

**Гистограмма** - способ представления табличных данных ввиде столбчатой диаграммы.

**Количественные (вещественные) признаки** - это признаки, представленные в числовом формате.

**Качественные (категориальные) признаки** - это признаки, значения которых можно разделить на группы или категории.

**Визуальный анализ данных** (exploratory data analysis, EDA) - это эффективный способ интерпретации данных и представления результатов анализа в виде графиков и диаграмм.

**Корреляционная** **матрица** (**матрица корреляций**) – это квадратная таблица, заголовками строк и столбцов которой являются признаки, а на пересечении строк и столбцов выводятся коэффициенты **корреляции** для соответствующей пары признаков.

**Коэффициент корреляции** – это статистическая мера, которая вычисляет силу связи между двумя переменными. Значения коэффициента корреляции находятся в диапазоне от -1 до 1.

## 6) Краткий план лекции

(Может собираться на основе подзаголовков из текста лекции, как содержание)

1. Устройство графиков
2. Виды графиков
   1. Линейный график
   2. Гистограмма
   3. Диаграмма рассеяния
   4. Столбчатые диаграммы
      1. Многорядовые
      2. Сложенные
   5. Ящик с усами
   6. Круговая диаграмма
3. Визуальный анализ данных
   1. Анализ вещественных признаков
   2. Анализ категориальных признаков
   3. Анализ взаимного распределения признаков
      1. Вещественные признаки
      2. Категориальные признаки
      3. Матрица корреляций
4. Инструменты для визуализации геоданных
   1. Matplotlib
   2. Folium
   3. KeplerGL

## 7) Подробный текст лекции

В этом уроке знакомимся с визуальным анализом данных с помощью библиотек Matplotlib и Seaborn.

В связи с особенностями человеческого мозга в обрабатывании информации, проще использовать диаграммы или графики для визуализации больших объемов сложных данных, чем разбираться с таблицами или отчетами.

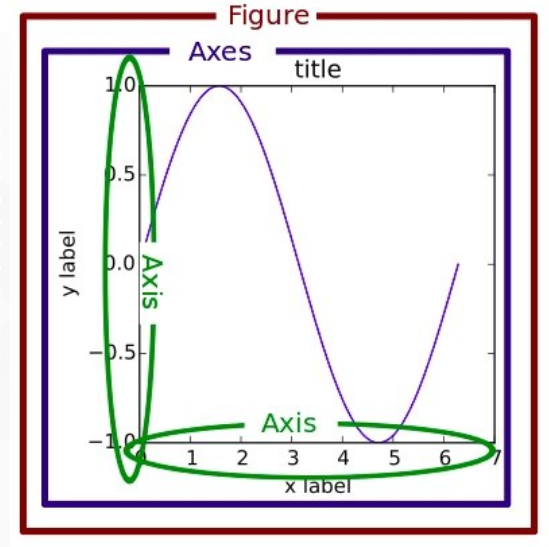
Хорошая визуализация выделяет полезную информацию. Визуализация данных позволяет определить области, которые требуют внимания или улучшения, выяснить, какие характеристики влияют на поведение клиентов, а также понять и объяснить сложные процессы.

Визуализация делает данные более запоминающимися.

### Устройство графиков

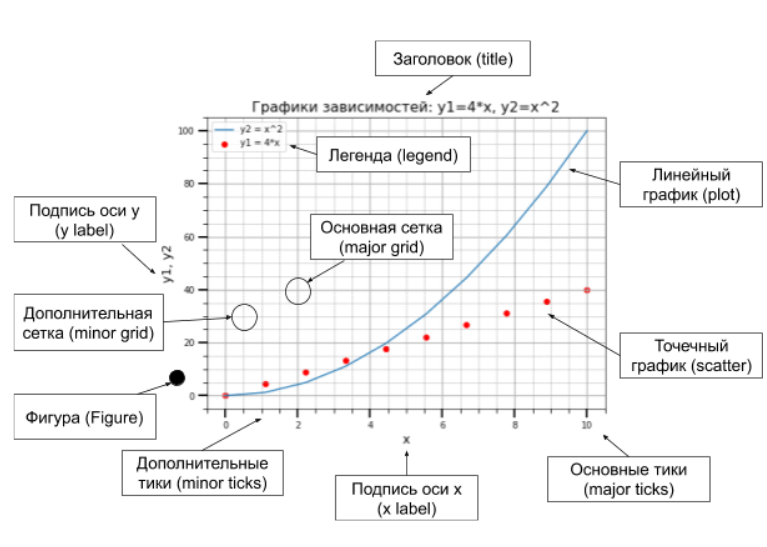
Для начала обсудим устройство графиков в этих двух библиотеках.

Есть объект Figure (фигура), в которой можно класть любое количество Axes (осей). А уже в осях можно рисовать много графиков:

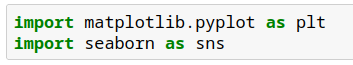


Чуть больше деталей про сами графики:

1. title. Заголовок графика
2. y label/x label. Именования осей х и у
3. legend. Легенда к графикам
4. major/minor ticks. Основной и дополнительный масштаб осей
5. major/minor grid. Основная и дополнительная сетка на графике

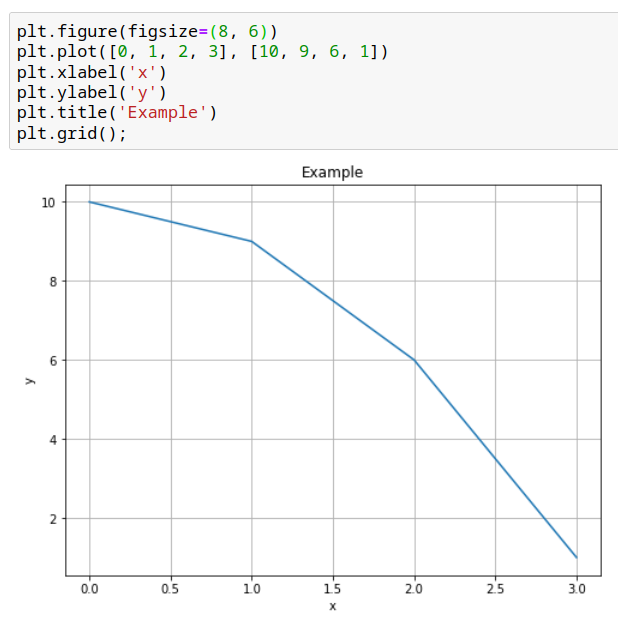


Чтобы начать работу с библиотеками, их нужно импортировать. Matplotlib имеет псевдоним plt, а seaborn - сокращение sns:



Давайте создадим простой график, используя стандартный синтаксис:

* Вначале нужно создать объект фигуры и указать его размер plt.figure(figsize=(8, 6))
* Затем нарисовать график. Мы нарисуем 4 точки. Первые значения расположены по оси x, а вторые по оси y. plt.plot([0, 1, 2, 3], [10, 9, 6, 1])
* Можно добавить наименования осей plt.xlabel('x')
* Можно добавить заголовок plt.title('Example')
* Можно добавить сетку plt.grid()



### Виды графиков

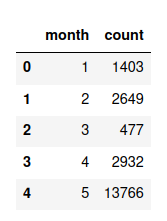
Перед тем, как перейти к визуальному анализу, посмотрим на основные виды графиков и как их читать. Здесь приведен не исчерпывающий список, а только самые основные и самые популярные виды.

#### Линейный график

Данный вид графика уже наблюдали выше при знакомстве с синтаксисом. Из себя представляет точки, соединенные линией. Чтобы его отрисовать нужно передавать список точек по x и список точек по у.

Изучим с помощью линейного графика, как много у нас появляется клиентов в зависимости от месяца.

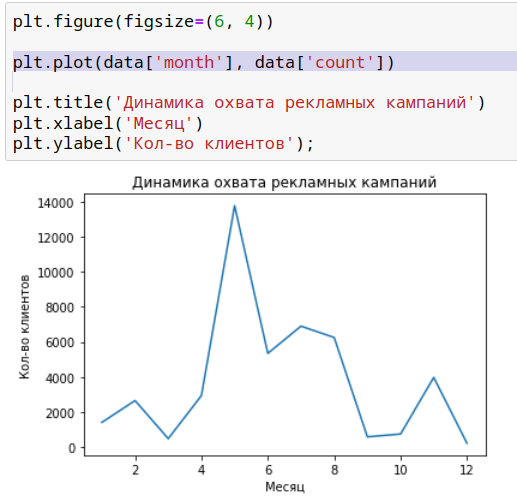
Небольшой набор исходных данных, которые будем рисовать. month - это месяц, count - количество клиентов в этом месяце:



Название функций разное в зависимости от библиотеки:

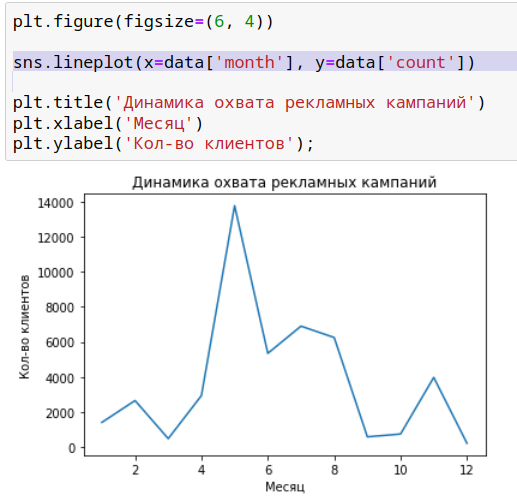
1. Matplotlib

plt.plot()



1. Seaborn

sns.lineplot()



Видим, что в начале года клиентов было мало, но зато к 5 месяцу (май), клиентов появилось больше, видимо провели очень качественную рекламную кампанию.

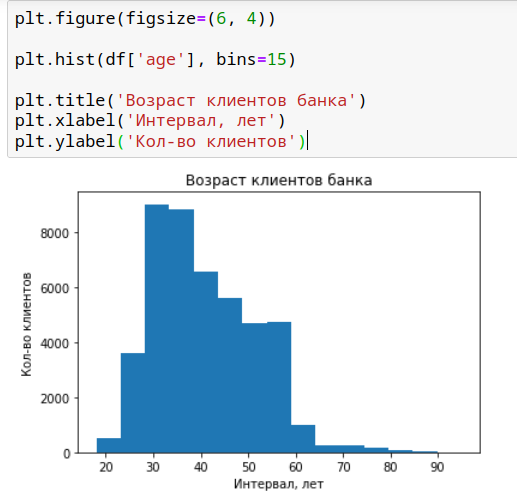
#### Гистограмма

**Гистограмма** - способ представления табличных данных ввиде столбчатой диаграммы.

Чтобы его отрисовать нужно передавать список точек x - это и есть значения признака. К примеру, мы хотим изучить признак возраста клиентов. Для этого pd.Series с возрастом df[‘age’] передаем в функцию. Название функций разное в зависимости от библиотеки:

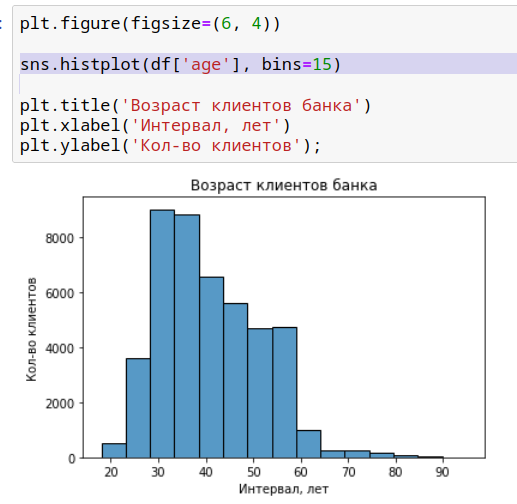
1. Matplotlib

plt.hist(x)



1. Seaborn

sns.histplot()



По оси х будут расположены значения возраста, а по оси у - количество клиентов с таким возрастом. Чем выше столбик, тем больше таких клиентов. Каждый столбик (бин, интервал) - это диапазон значений, в который попадают разные значения возраста. Получаем, что большинство наших клиентов в возрасте от 28 до 43, а совсем молодых (до 28) и возрастных (от 43) очень мало.

#### Диаграмма рассеяния

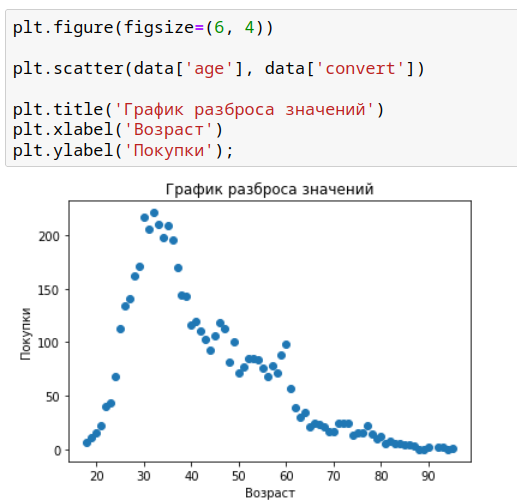
Является полезным типом графика для исследования пар числовых признаков. Наглядно показывает наличие взаимосвязей и закономерностей в данных, и при наличии закономерностей мы можем более тщательно изучать данные признаки.

С помощью такой диаграммы изучим, как много открывается депозитов в банке нашими клиентами в зависимости от возраста.

Чтобы отрисовать график нужно передать список точек по x (возраст клиентов) и список точек по у (факт открытия депозита) в функцию. Название функций разное в зависимости от библиотеки:

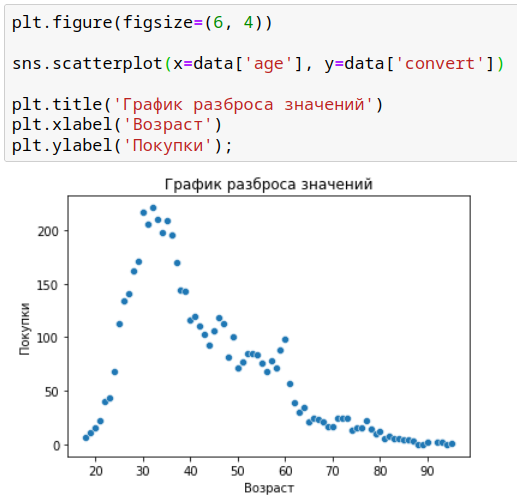
1. Matplotlib

plt.scatter()



1. Seaborn

sns.scatterplot()



По оси x откладываем возраст клиента, а по оси y - количество приобретенных депозитов. Можно сделать вывод, что большинство депозитов в нашем банке берется молодыми людьми в возрасте от 25 до 40 лет.

#### Столбчатые диаграммы

Еще один распространенный тип графиков — столбчатые диаграммы. Они похожа на гистограммы, но на оси x тут располагаются не числовые значения, а категории.

Изучим сколько клиентов в разных профессиях есть в нашем банке.

Чтобы отрисовать график нужно передавать список точек по x (названия профессий) и список точек по у (количество клиентов). При этом, названия профессий не будет помещаться на оси x, названия будут накладываться друг на друга:

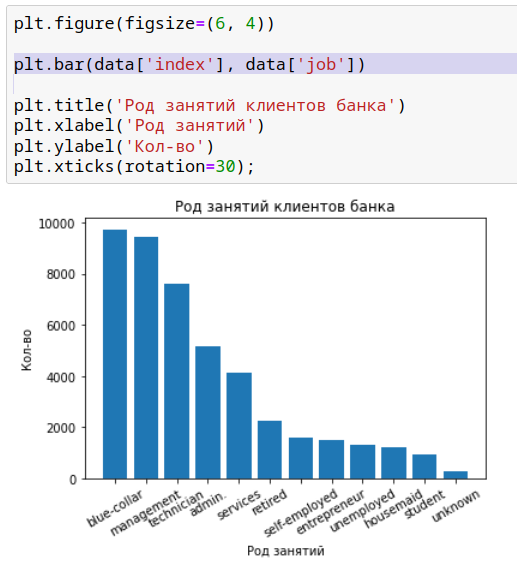


Поэтому мы их немного повернем, для этого используем атрибут rotation, который передаем в plt.xticks, который отвечает за ось х.

Название функций для построения столбчатого графике разное в зависимости от библиотеки:

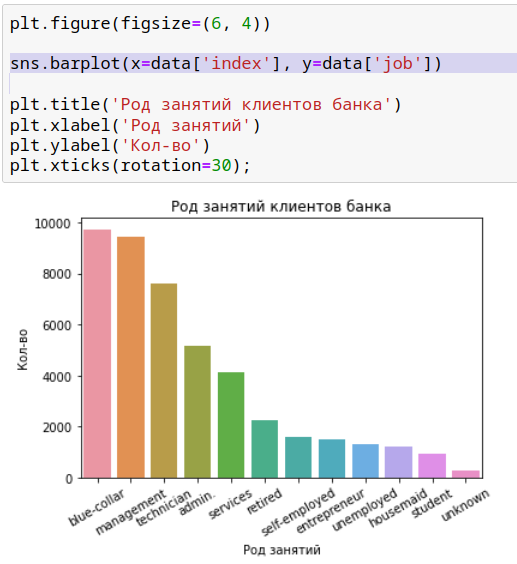
1. Matplotlib

plt.bar()



1. Seaborn

sns.barplot()



Видим из графиков, что у нас очень много представителей так называемых синих воротничков, менеджеров, технических специалистов. И совсем мало студентов, уборщиков, безработных.

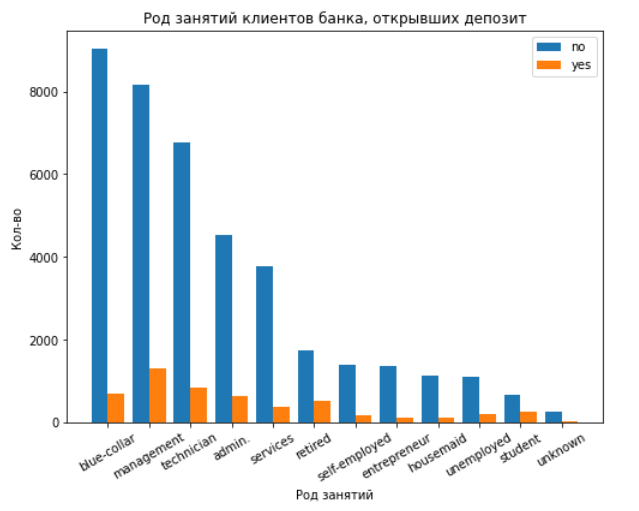
##### Многорядовые

С помощью такого типа графика можем строить по несколько категорий в рамках одной категории. К примеру, хотим проанализировать, клиенты с каким родом деятельности чаще берут депозиты в банке.

Для этого нужно создать два ряда столбиков:

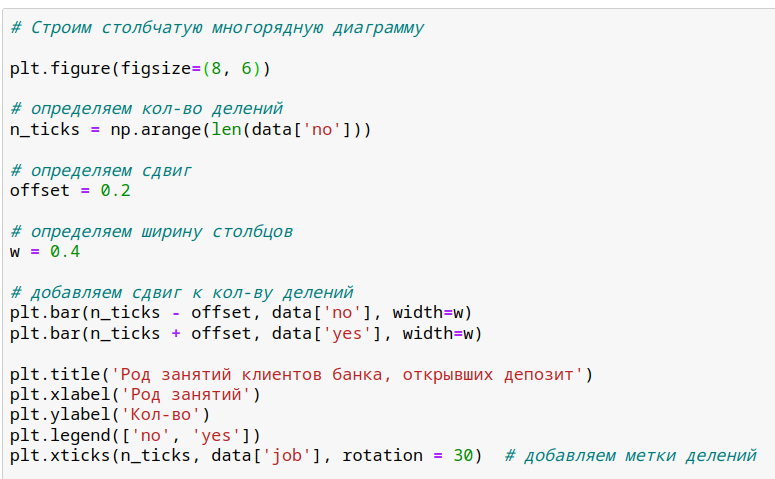
1. Для клиентов по профессиям, кто не взяли депозит
2. Для клиентов по профессиям, кто взяли депозит

Получим подобную визуализацию, где синие столбцы - это первая категория людей без депозитов, а оранжевые - это вторая категория людей с депозитами. Получается, что больше всего у нас берут депозиты менеджеры:



Как это можно устроить с точки зрения синтаксиса? А нужно добавить дополнительные деления на оси х при отрисовки, для этого:

1. Генерируем деления в количестве профессий: n\_ticks = np.arange(len(data['no']))
2. Определяем сдвиг, насколько синие и оранжевые столбики будут сдвигаться: offset = 0.2
3. Определяем ширину столбцов. Главное, чтобы они не пересекались друг с другом, если вы того явно не хотите: w = 0.4
4. Отрисовываем первый столбчатый график для синих столбцов. При этом из их делений вычитаем сдвиг, чтобы столбцы оказались чуть левее: plt.bar(n\_ticks - offset, data['no'], width=w)
5. Отрисовываем второй столбчатый график для оранжевых столбцов. К их делением добавляем сдвиг, чтобы столбцы оказались чуть правее: plt.bar(n\_ticks + offset, data['yes'], width=w)



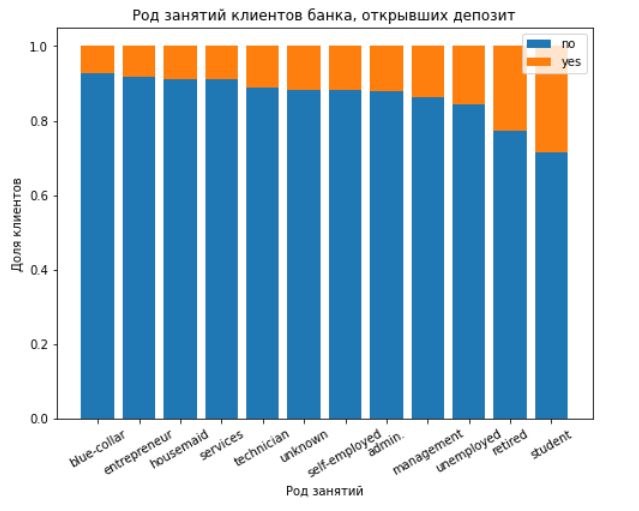
Отобразив многорядовую столбчатую диаграмму можем сделать вывод, что больше всего депозитов у нас приобретают менеджеры, затем технические специалисты и только потом самые многочисленные синие воротнички.

##### Сложенные

Еще один способ представления столбчатого графика — сложенная форма, где каждый столбец установлен поверх другого. Это особенно полезно в том случае, когда нужно показать общее значение суммы всех столбцов.

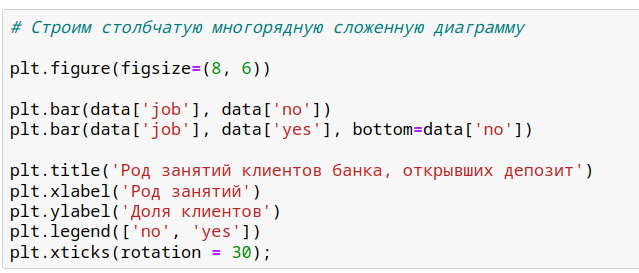
Для превращения обычного столбчатого графика в сложенный нужно добавить аргумент bottom в функцию bar(), которая будет стоять наверху.

Получаем, что процент открытия депозитов самый большой среди студентов и людей на пенсии, а среди синих воротничков процент открытия депозита самый маленький:



Синие столбики строим как обычно: plt.bar(data['job'], data['no'])

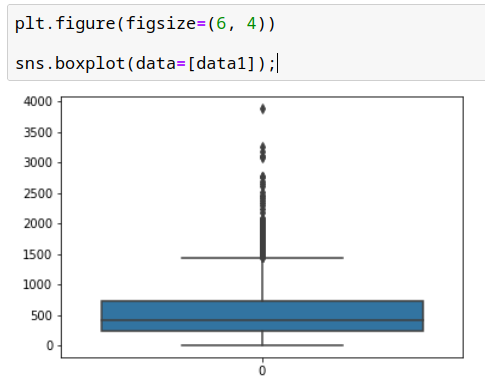
А вот чтобы построить оранжевые столбцы, нужно к ним добавить аргумент bottom, который будет равняться значениям синих столбиков, ведь поверх них и нужно будет строить оранжевые:



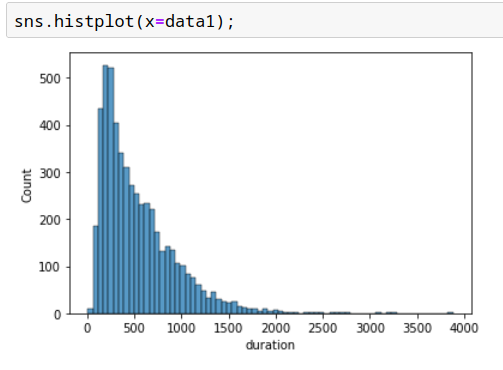
#### Ящик с усами

Диаграмму ящик с усами (боксплот) используют для быстрой оценки данных. Она также как и гистограмма показывает распределение значений в выборке. Основное преимущество диаграммы боксплот в том, что она компактно и наглядно показывает основные статистические показатели:

* Медиану (50% квартиль) - это линия по середине прямоугольника. **Медиана** — это значение делящее распределение пополам. Другими словами это значение ниже которого находятся 50% значений, и выше также 50% всех значений в распределении.
* Верхний и нижний квартили (25% и 75%) - верхняя и нижняя граница прямоугольника. **Квантиль** — это значение, ниже которого лежит определенное число наблюдений, соответствующих выбранной частоте.
* Наблюдаемые минимумы и максимумы
* А также выбросы в данных, в виде отдельных точек



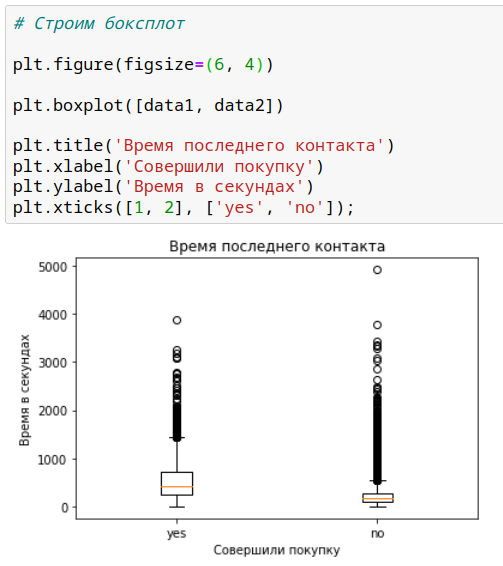
К ящику с усами можно относится как к виду сверху на гистограмму, здесь тоже минимум виден и где больше всего данных находится, и выбросы виднеются:



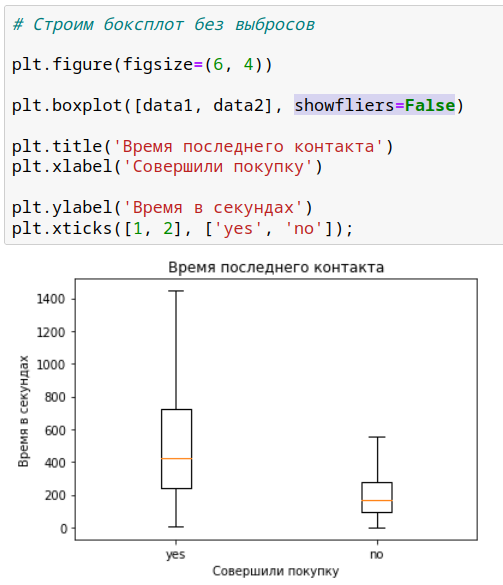
Посмотрим, а есть ли какая-то связь между временем общения оператора с клиентом и оформлением депозита:

1. Matplotlib

plt.boxplot()

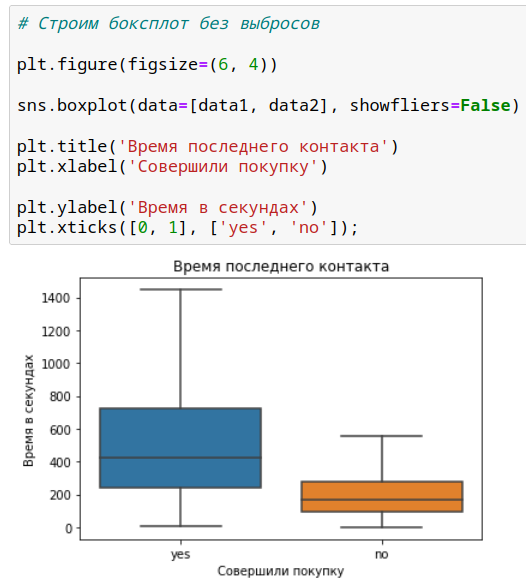


Наблюдаем выбросы - это черные круги. Такие объекты, которые не похожи на основную массу объектов. Можно их не отрисовывать на диаграмме, если добавим атрибут showfliers=False:



1. Seaborn

sns.boxplot()



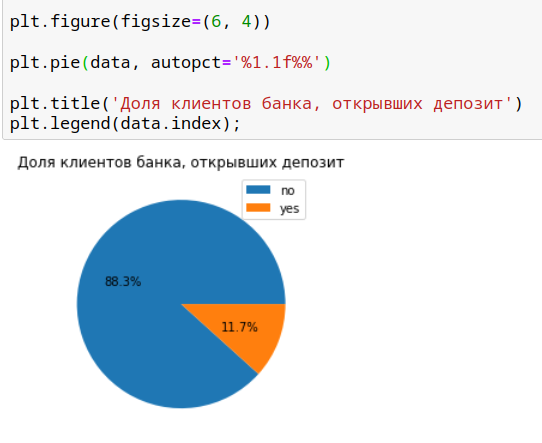
Видим, что если с клиентом дольше общаться, то вероятность совершения депозита увеличивается. Ящик у группы yes находится выше, чем у группы no.

#### Круговая диаграмма

Еще один способ представления данных — круговая диаграмма. Для нее нужно передать основной аргумент, представляющий собой список значений. Это могут быть как числовые значения, так и проценты.

Но и это не все, что может быть на диаграмме. У нее нет осей, поэтому сложно передать точное разделение. Чтобы решить эту проблему, можно использовать autopct, который добавляет в центр каждой части текст с соответствующим значением. В нашем примере мы передаем в autopct строку ‘%1.1f%%’, два процента в начале и в конце пишутся всегда, единица до точки будет отображать целое значение, единица после точки показывает, что в нашем значении будет точность в один знак после запятой, f показывает, что значение будет типом данных float и в конце строки добавляется знак процента.

Хотим посмотреть процентное соотношение по клиентам, открывшим депозит и не открывшим:



С основными типами графиков разобрались, теперь можно плавно переходить к визуальному анализу данных.

### Визуальный анализ данных

Но чтобы лучше подбирать способ визуализации для анализа нужно уметь разделять типы признаков на две группы:

1. **Количественные (вещественные) признаки** - это признаки, представленные в числовом формате.

Приведем примеры таких признаков:

* Возраст человека
* Количество товаров, которое осталось на складе
* Температура воздуха

1. **Качественные (категориальные) признаки** - это признаки, значения которых можно разделить на группы или категории.

Приведем примеры:

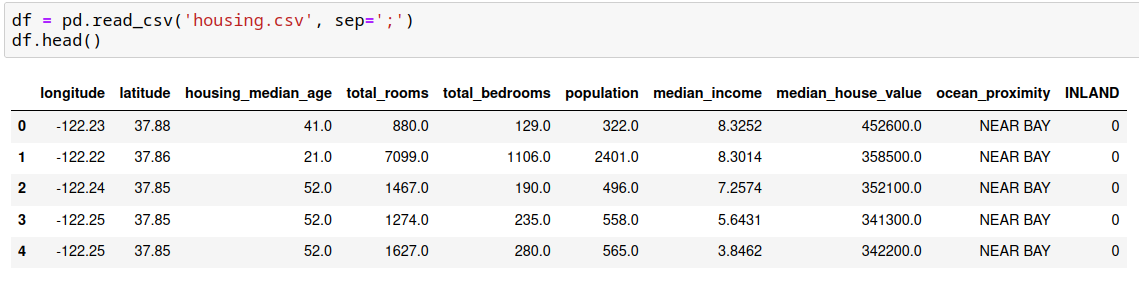
* Цвет машины
* Уровень образования
* Семейное положение

**Визуальный анализ данных** (exploratory data analysis, EDA) - это эффективный способ интерпретации данных и представления результатов анализа в виде графиков и диаграмм.

Есть три шага в EDA:

1. Анализ вещественных характеристик
2. Анализ категориальных признаков
3. Анализ взаимного распределения признаков

Пройдемся по каждому этапу отдельно и будем визуально изучать набор данных по стоимости недвижимости.

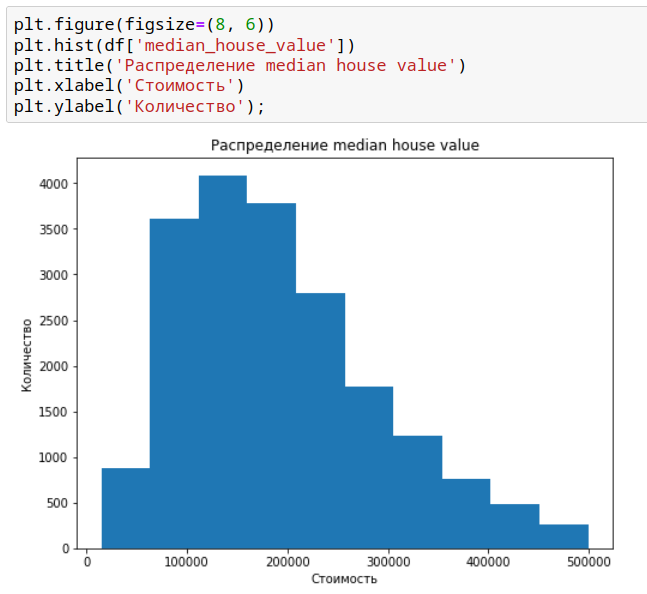


#### Анализ вещественных характеристик

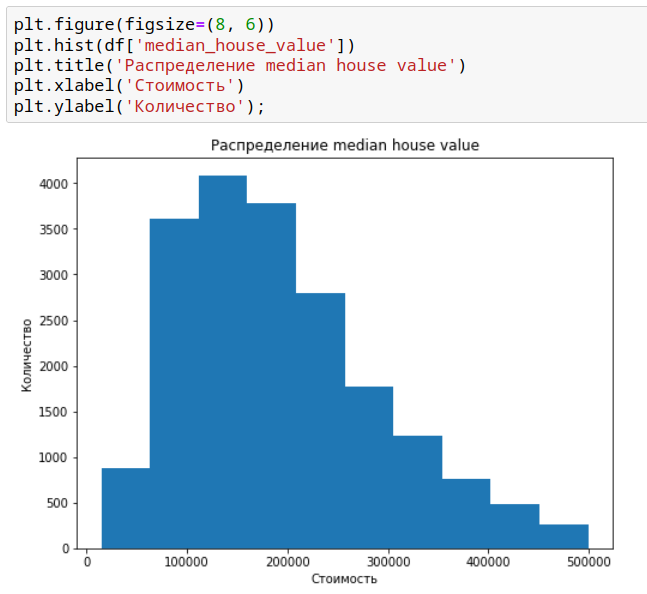
Для таких признаков стоит изучить распределение, а это можно сделать построив гистограмму.

Изучим признак стоимости недвижимости. Для этого напишим следующий синтаксис:

* Создаем фигуру
* Рисуем график гистограммы plt.hist(df['median\_house\_value'])

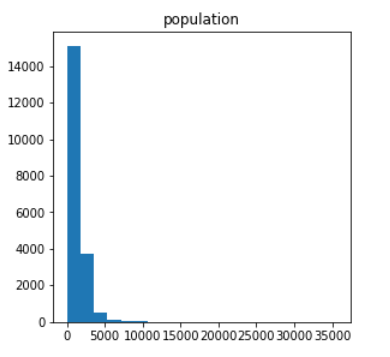


Видим, что по оси х расположены значения стоимости дома, а по оси у - количество домов с такой стоимостью. Чем выше столбик, тем больше таких значений. Каждый столбик (бин, интервал) - это диапазон значений, в который попадают разные значения стоимости. Получаем, что большинство наших домов стоят где-то от 70к до 250к, а дешевых (до 70к) и очень дорогих (от 250к) очень мало:



Признак, который посмотрели выше имеет очень приятное распределение, нет явных выбросов - очень больших, либо очень маленьких значений, которых мало.

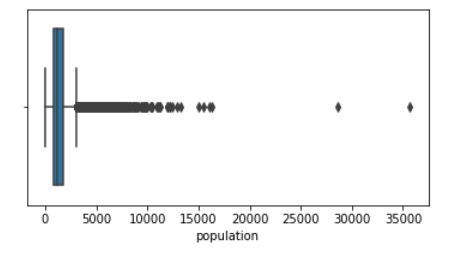
Давайте возьмем признак населения и поизучаем его:



А здесь в большинстве своем население проживающее в доме изменяется от 0 до 5000. Дома, которые могут в себя вместить больше 5 тысяч людей, встречаются очень редко - а это очень похоже на выбросы.

Как еще можно найти выбросы с помощью визуализаций?

С помощью ящика с усами:

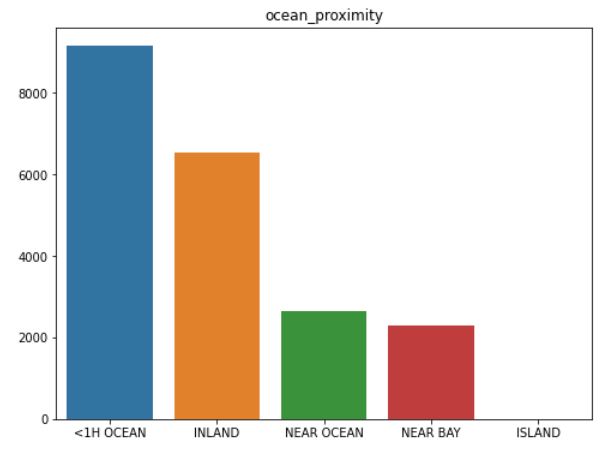


Все значения, которые находятся правее максимального уса похожи на выбросы. Значит лучше их убрать из набора данных.

#### Анализ категориальных признаков

Для таких признаков тоже стоит изучить распределение, но уже используя столбчатые диаграммы.

Посмотрим, на характеристику удаленности от океана. По оси х находятся названия категорий признака, а по оси у - их количество. Видим, что домов, которые находятся в менее часа езды от океана, очень много, а самая малочисленная категория - островные дома.



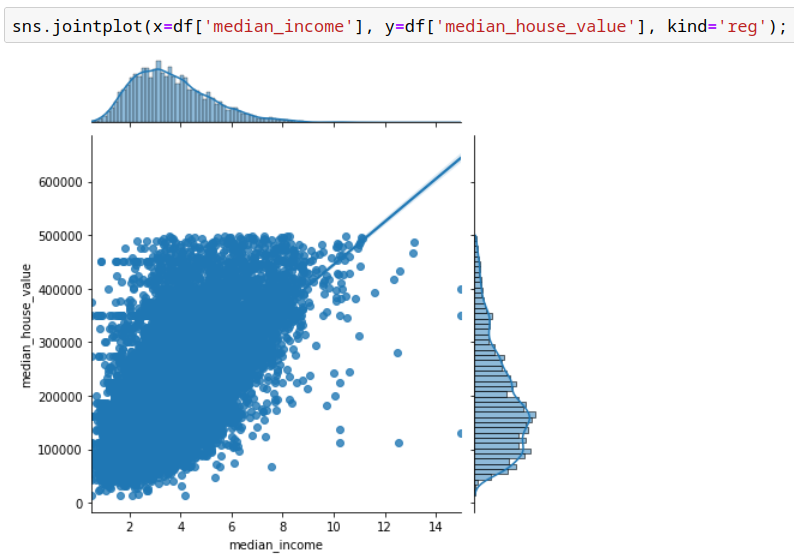
#### Анализ взаимного распределения признаков

На этом этапе полезно будет находить закономерности между признаками, для того, чтобы знать, как один признак влияет на другой.

##### Вещественные признаки

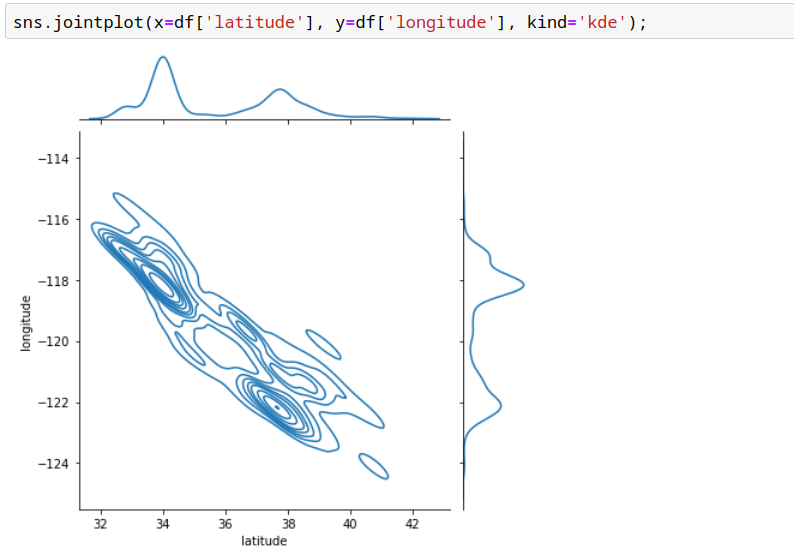
Для анализа подойдут графики взаимного распределения. К примеру, хотим проанализировать зависимость среднего заработка семьи и стоимость недвижимости. У нас есть гипотеза, что семья покупает дом в зависимости от своих доходов.

Чтобы построить график взаимного распределения нужно в sns.jointplot передать первый признак и второй:



По оси х располагается первый признак - средний заработок семьи, а по оси у - второй признак - стоимость недвижимости. По середине точечный график их взаимного распределения. В идеале, чтобы здесь вырисовывалась какая-то закономерность. И действительно, у нас есть неплохая связь: чем больше median\_income, тем больше median\_house\_value, так что наша гипотеза подтверждается.

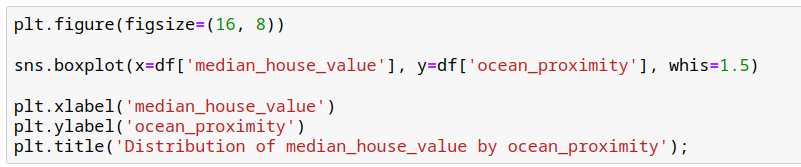
Или же еще один пример. Зависимости признака долготы и широты. Их тоже можно изучать с помощью графика взаимного распределения sns.jointplot. Только можем поменять атрибут kind=’kde’:



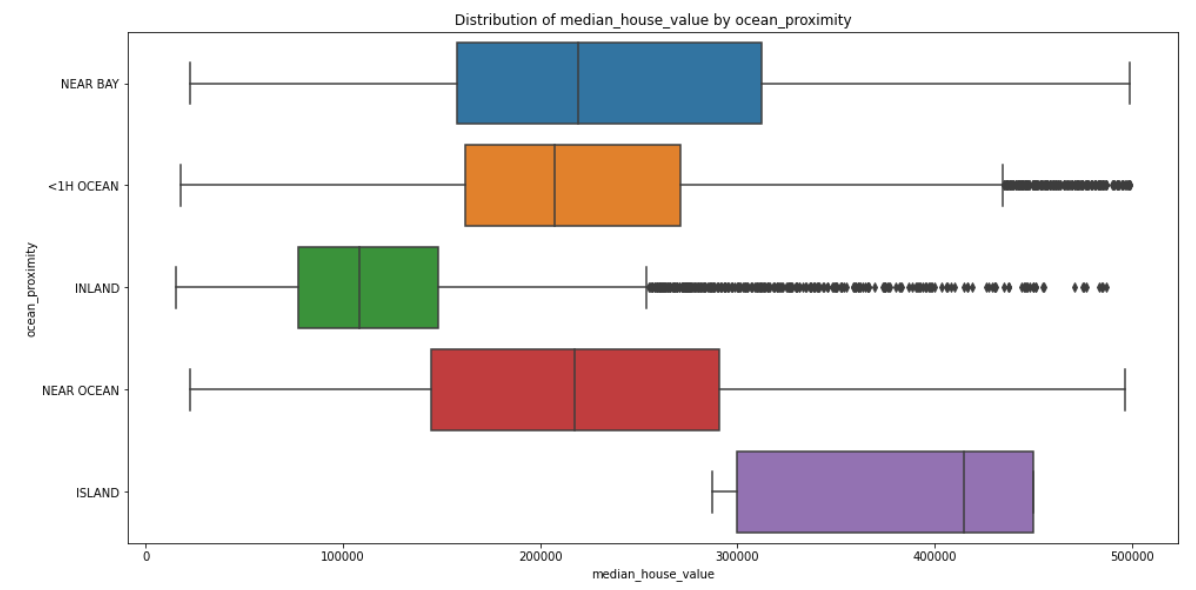
На этом графике чем более скученные эллипсы, тем в этих значениях находятся больше точек. Значит в наших данных точно есть какие-то два города.

##### Категориальные признаки

Для анализа подойдут графики боксплот. К примеру, хотим проанализировать стоимость недвижимости в зависимости от удаленности от океана. Для этого в ящик с усами передаем median\_house\_value и ocean\_proximity:



Получаем, что самые дорогие дома находятся на островах (ISLAND), раз у них ящик находится правее всех, а самые дешевые дома, напротив, имеют ящик левее всех и это дома материковые (INLAND). А вот остальные три категории друг от друга практически не отличаются, из стоимости имеют одинаковые распределения:



##### Анализ матрицы корреляций

Данный этап поможет нам найти линейные связи между признаками, для того, чтобы знать, как один признак влияет на другой.

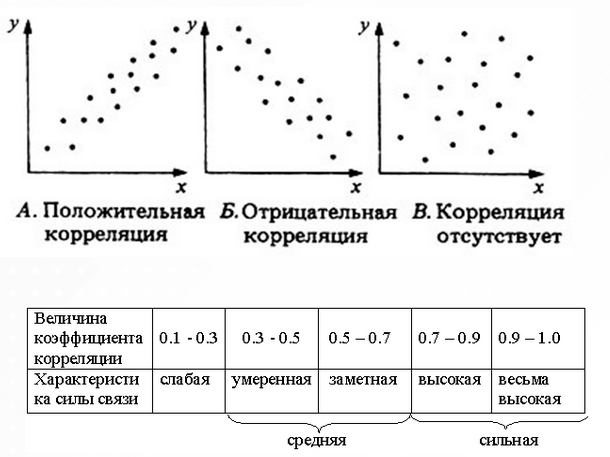
**Корреляционная** **матрица** (**матрица корреляций**) – это квадратная таблица, заголовками строк и столбцов которой являются признаки, а на пересечении строк и столбцов выводятся коэффициенты **корреляции** для соответствующей пары признаков.

**Коэффициент корреляции** – это статистическая мера, которая вычисляет силу связи между двумя переменными. Значения коэффициента корреляции находятся в диапазоне от -1 до 1.

Когда коэффициент равен 1, то между признаками очень сильная прямая связь, как на картинке **А**. А значит, когда увеличивается первый признак, то увеличивается и второй признак.

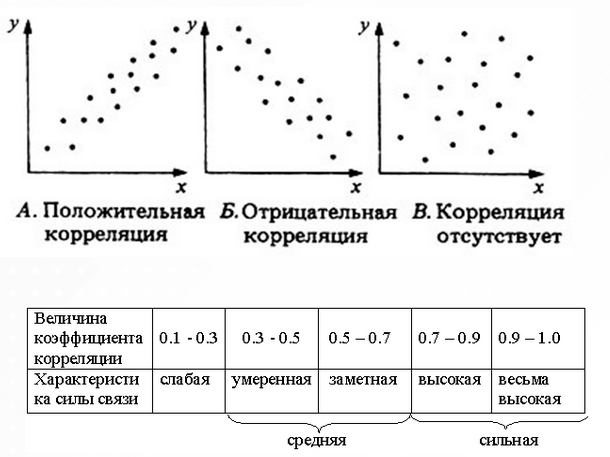
Когда коэффициент равен -1, то между признаками очень сильная обратная линейная связь, как на картинке **Б**. А значит, если увеличивается первый признак, то уменьшается второй.

А если коэффициент корреляции находится в районе 0, то между признаками нет линейной связи, эта ситуация показана на картинке **В**.

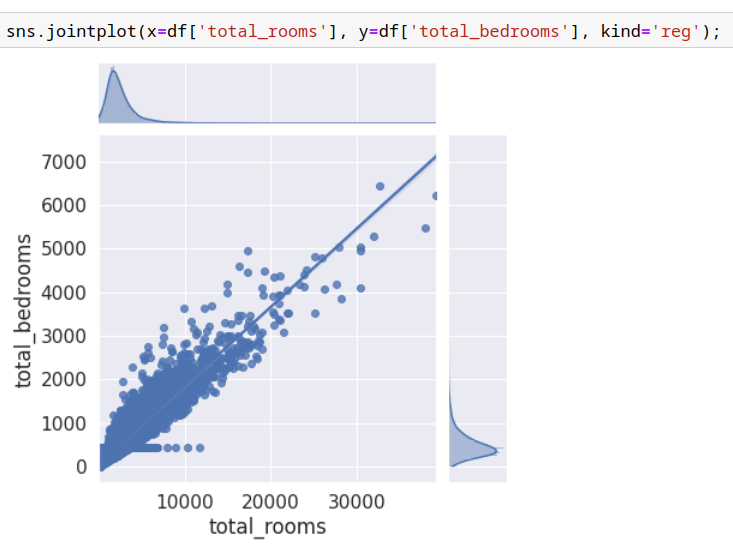


Получаем, что если коэффициент корреляции по модулю

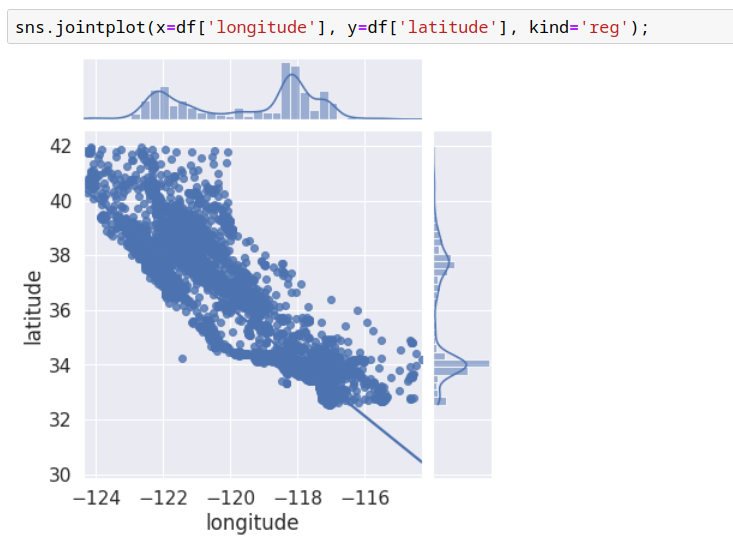
* меньше 0.3, то линейная связь слабая
* от 0.3 до 0.7 - это средняя линейная связь
* больше 0.7, то связи сильная



В наших данных очень большой прямой линейно связью обладают признаки общее количество комнат и спален. Когда комнат становится больше, то спален тоже становится больше:



А пример обратной связи можно посмотреть на признаках широты и долготы. Когда долгота увеличивается, то уменьшается широта. Это связано с тем, что дома расположены вдоль побережья:

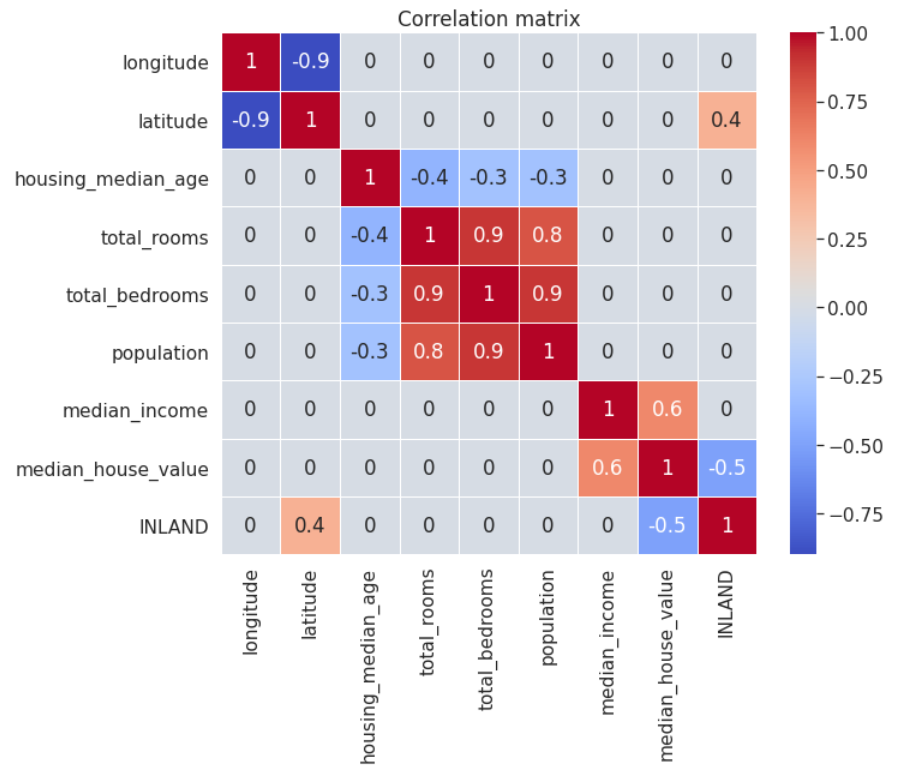


А чтобы посмотреть всю матрицу корреляций можно вызывать у датафрейма метод corr(), который подсчитает все попарные корреляции:



По диагонали корреляция признака самим с собой, поэтому она равняется 1, поэтому лучше изучать корреляции снизу либо сверху диагонали. А чтобы еще наглядней и быстрее проводить аналитику линейных связей, можно отрисовать тепловую карту из seaborn с помощью функции heatmap():





И видим, что наиболее коррелируемые признаки - это

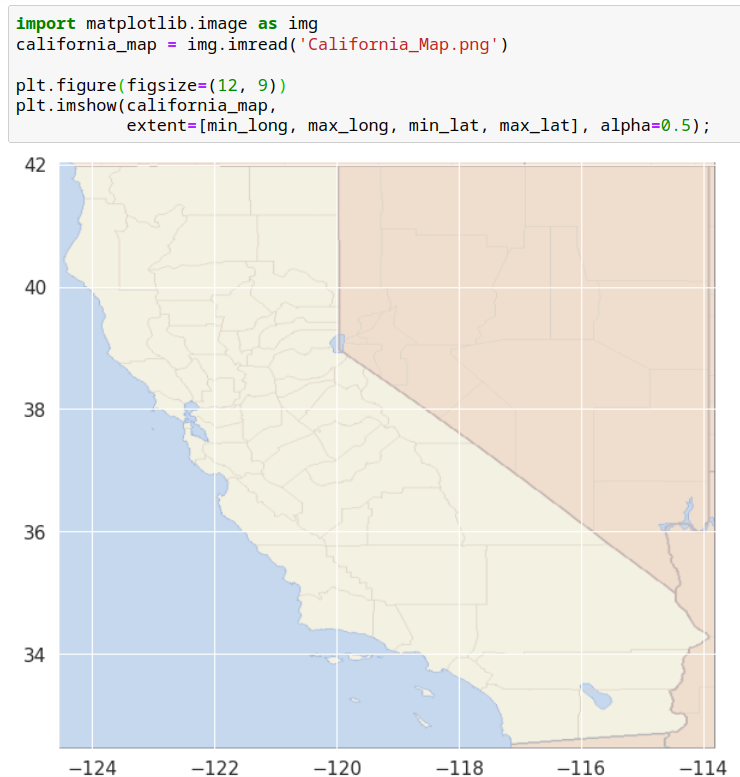
1. Широта и долгота. У них сильная обратная линейная связь
2. Общее количество комнат, спален и население. Между ними сильная прямая связь
3. Средний заработок и стоимость дома. Прямая средняя линейная связь.

Таким образом, мы быстро улавливаем, где есть линейные связи между признаками.

### Инструменты для визуализации геоданных

#### Matplotlib

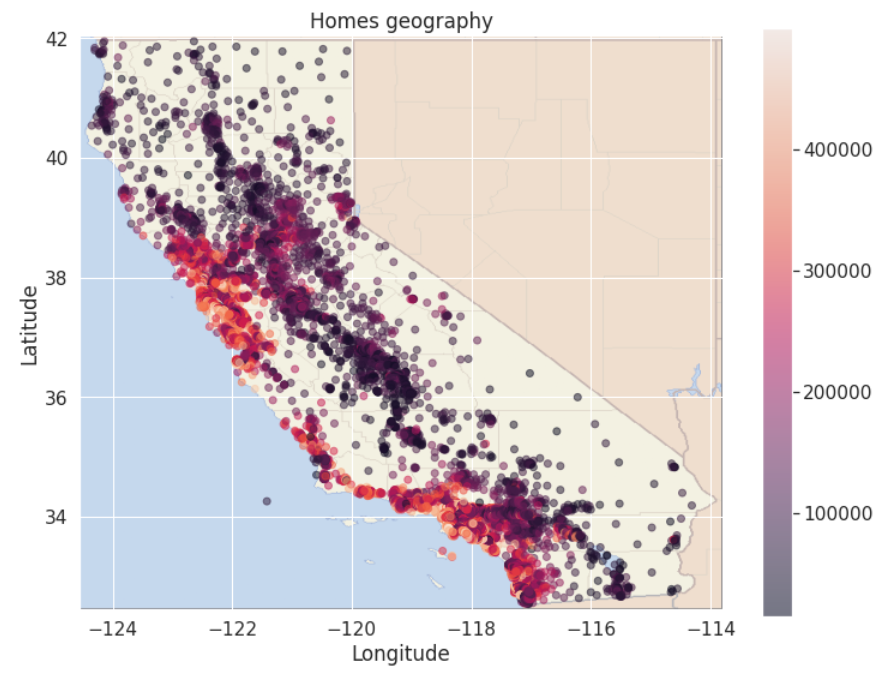
Можно заранее взять карту необходимой местности из интернета и визуализировать ее через функцию imshow(), при этом можем этому изображению задать границы по осям через атрибут extent - это и будут наши значения широты и долготы:



А затем на эту карту можем нанести наши точки через точечный график от долготы и широты, добавим прозрачность с помощью атрибута **alpha** и раскрашивание через атрибут **c** в зависимости от стоимости недвижимости:



И видим, как все наши дома отрисовались на карте, при этом цвет точки показывает стоимости дома. Действительно дома находящиеся на материке более дешевые по сравнению с островными домами. Также более дорогая недвижимости находится у океана:



Получили довольно полезную статичную карту с домами из набора данных.

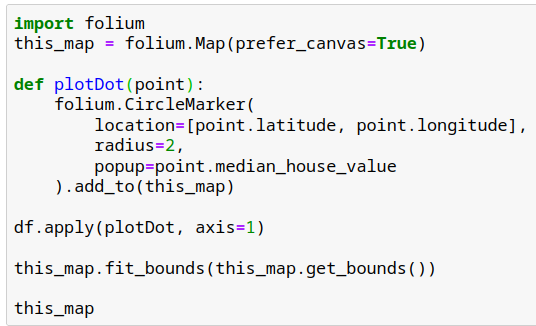
#### Folium

Библиотека folium позволяет не скачивать заранее изображение с картой, а воспользоваться встроенной картой в сам инструмент. И вдобавок это будет интерактивная карта.

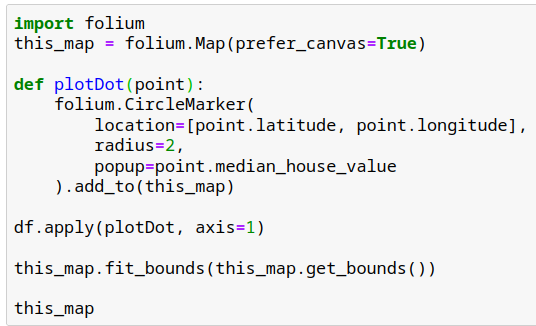
Устанавливаем folium через pip install folium:



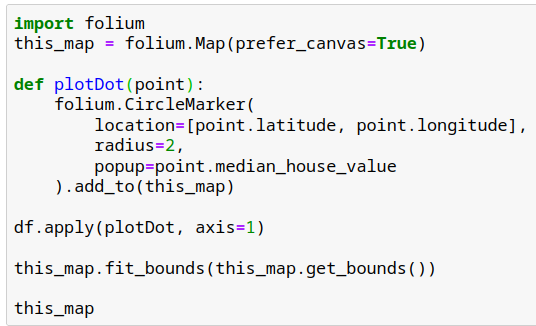
1. Импортируем folium



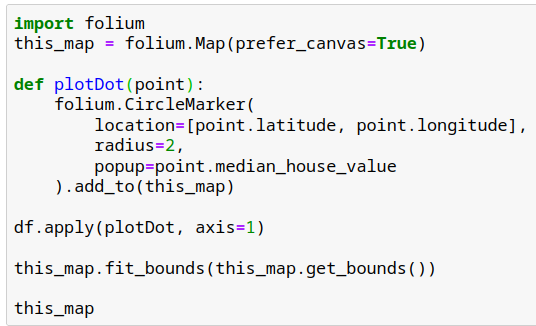
1. Создаем объект карты



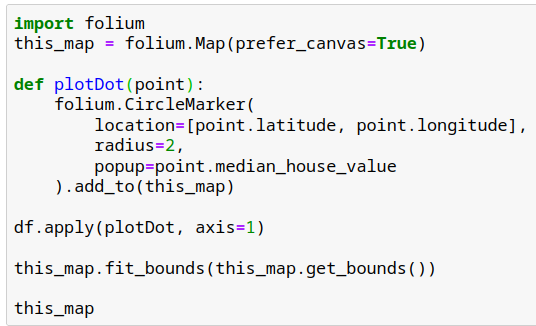
1. Реализуем функцию по отрисовке точек. На вход поступает точка со всеми характеристиками объекта, в частности с координатами широты и долготы. Создается круглый маркер (CircleMarker) на карте из соответствующей позиции с заданным радиусом (radius=2) и добавим еще надпись стоимости недвижимости при клике на точку (popup=point.median\_house\_value).



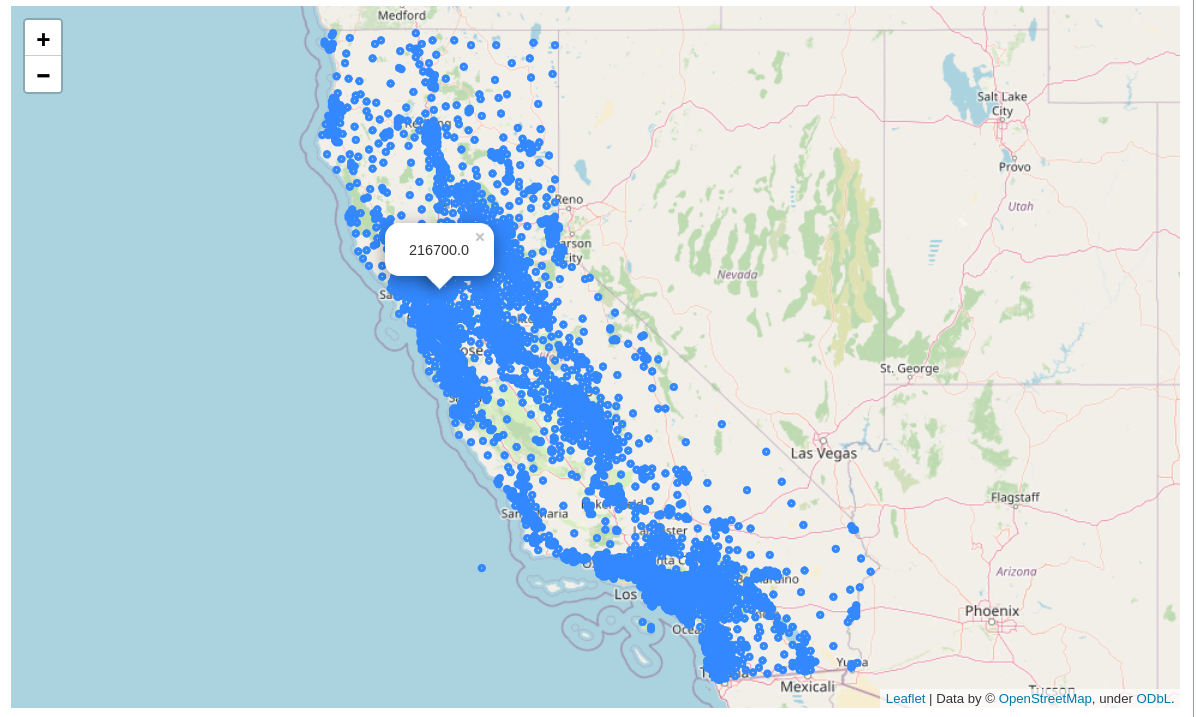
1. Применяем данную функцию ко всем точкам из датафрейма



1. Визуализируем карту



И получаем интерактивную карту, по которой можно перемещаться и изучать наши точки:



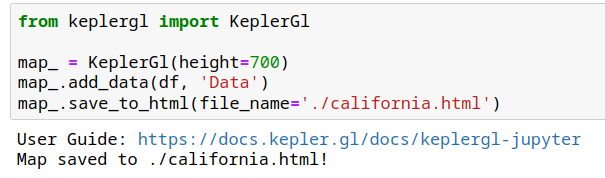
#### KeplerGL

Еще один инструмент для интерактивной карты.

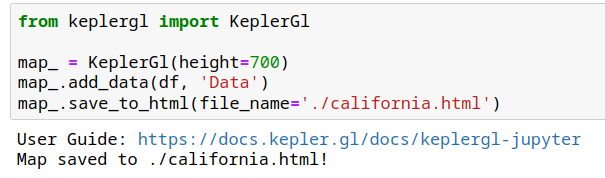
Первым делом нужно себе установить библиотеку:



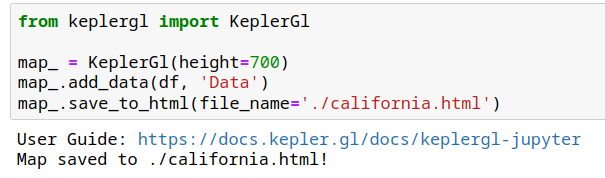
1. Затем импортировать ее



1. Создать объект KeplerGL и в него добавить наш датафрейм



1. Сгенерировать на выход html страницу с картой



Дальше можем открыть получившийся файл html и полюбоваться картой, где каждая точка описывается основными характеристиками дома из нашей таблицы:



## 9) Рекомендуемая дополнительная литература или материалы

1. Типы данных <https://youtu.be/c4Cg3TUIH0E>
2. Документация seaborn <https://seaborn.pydata.org/index.html>
3. Документация matplotlib <https://matplotlib.org/>
4. Краткое руководство по matplotlib <https://pyprog.pro/mpl/mpl_short_guide.html>

## 10) Используемая литература или материалы

1. Документация seaborn <https://seaborn.pydata.org/index.html>
2. Документация matplotlib <https://matplotlib.org/>