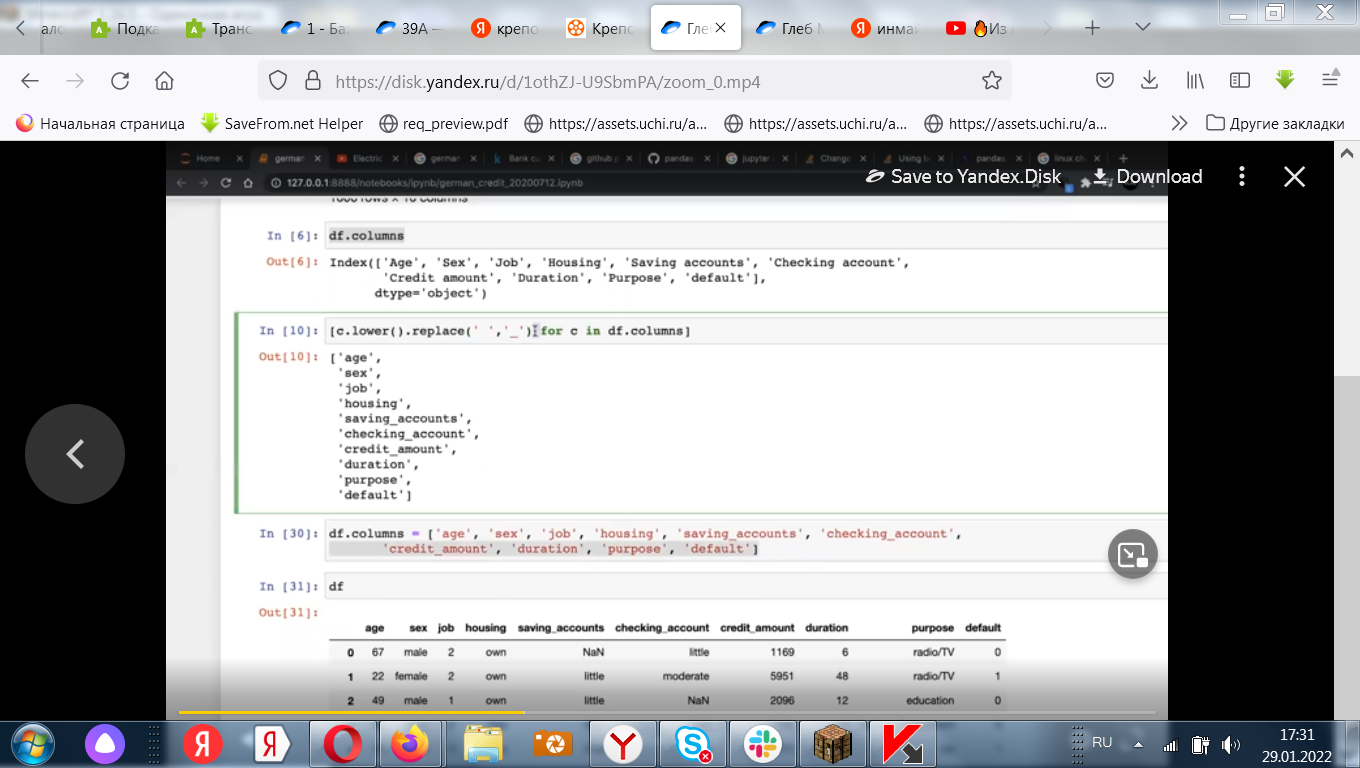
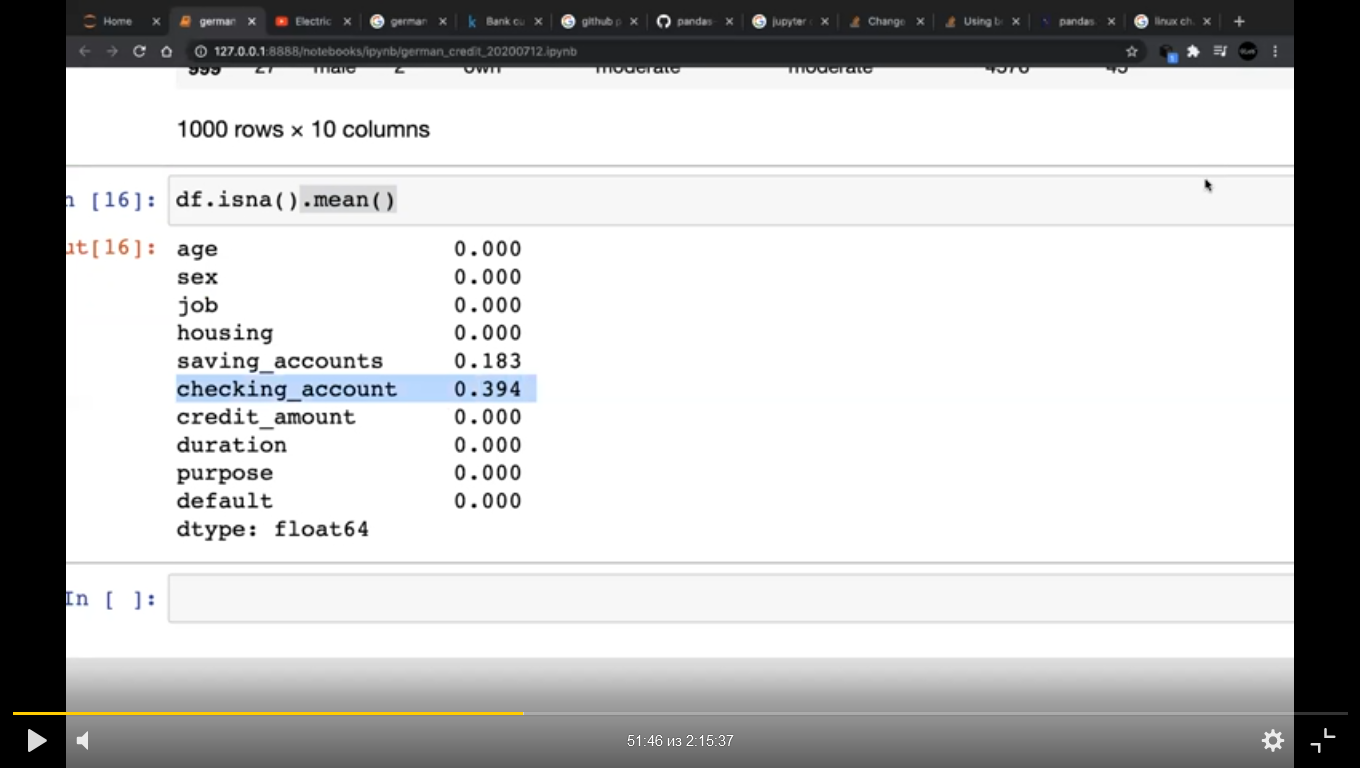
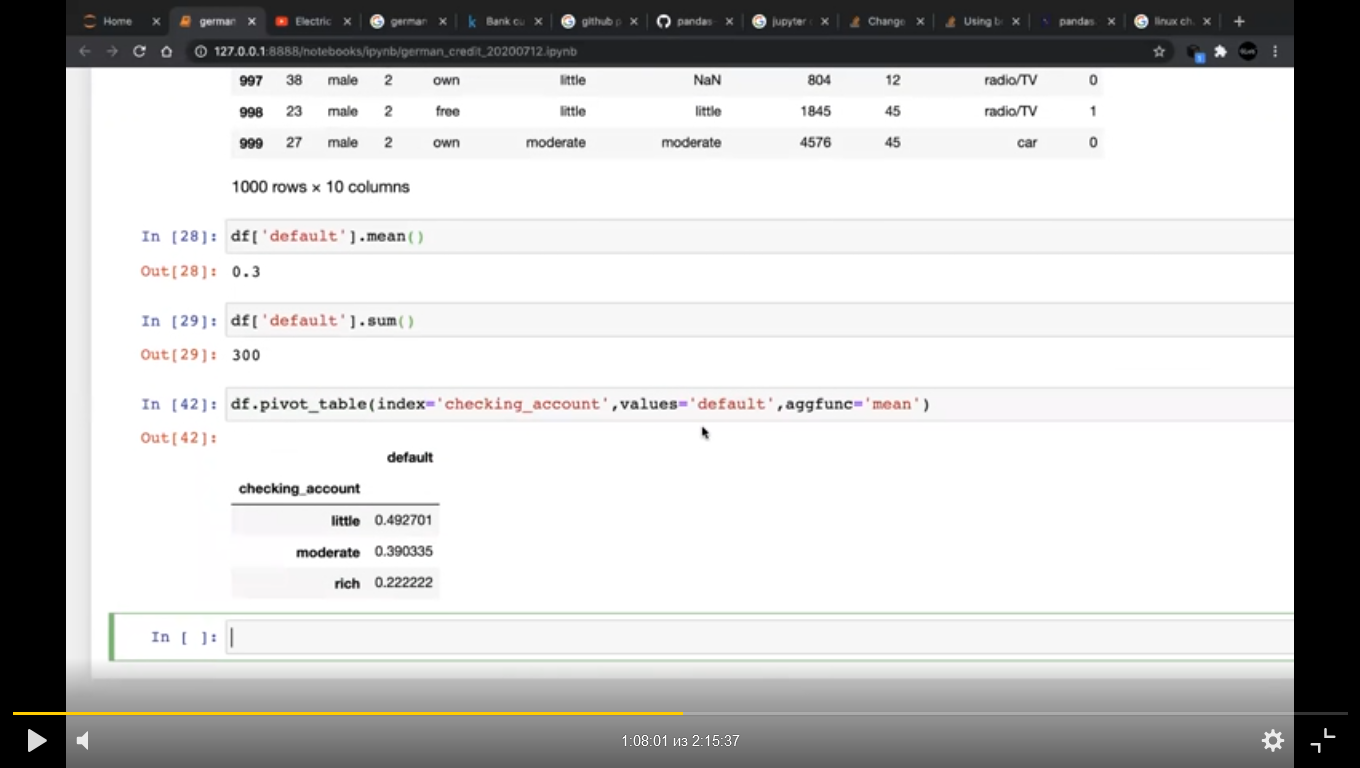
Переименовать названия столбцов

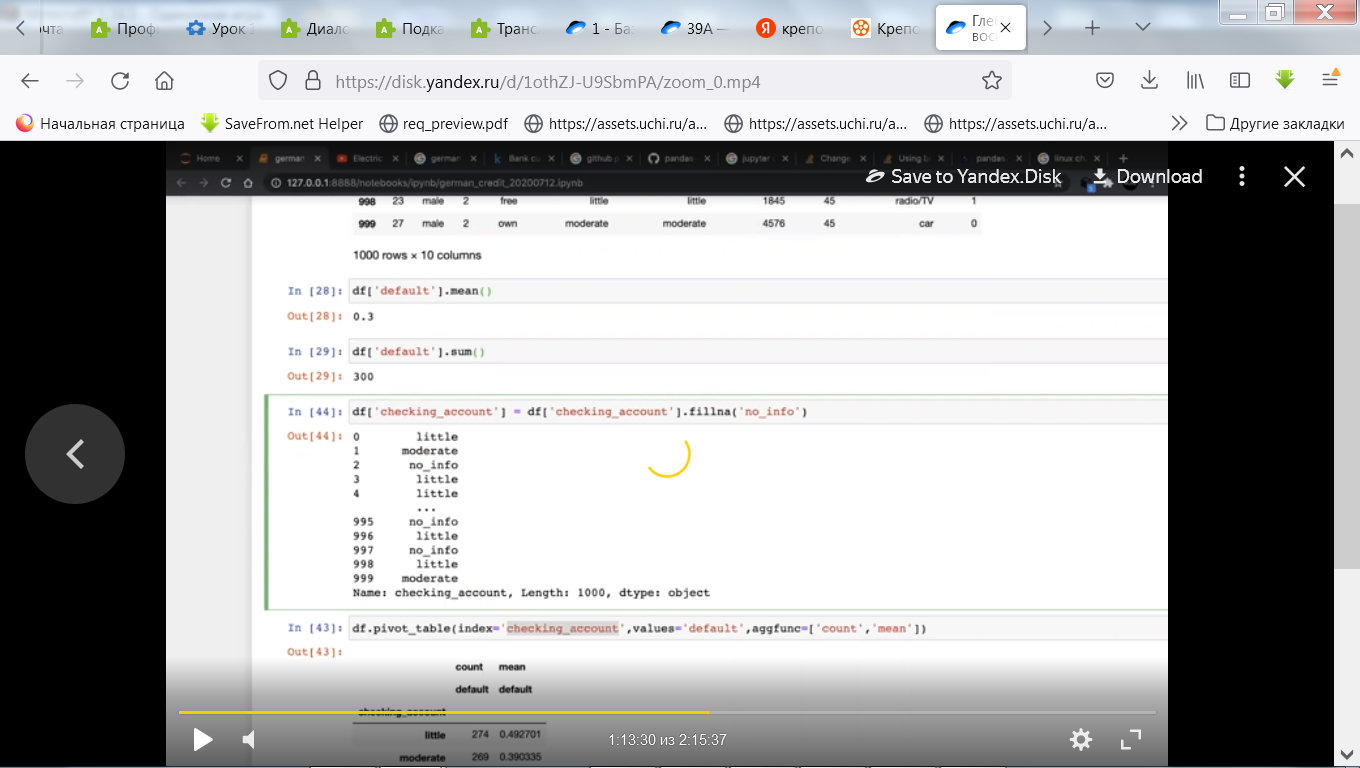


Сосчитать процент пропущенных значений

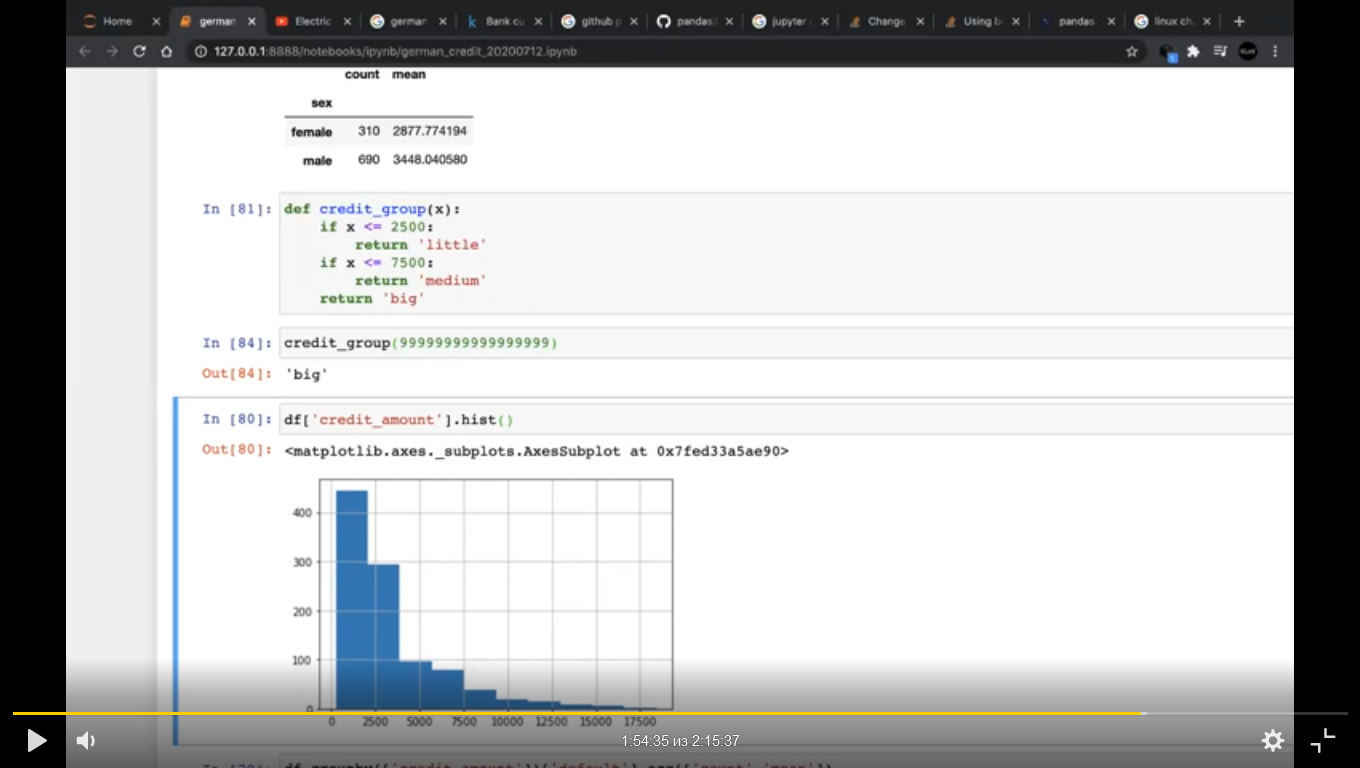


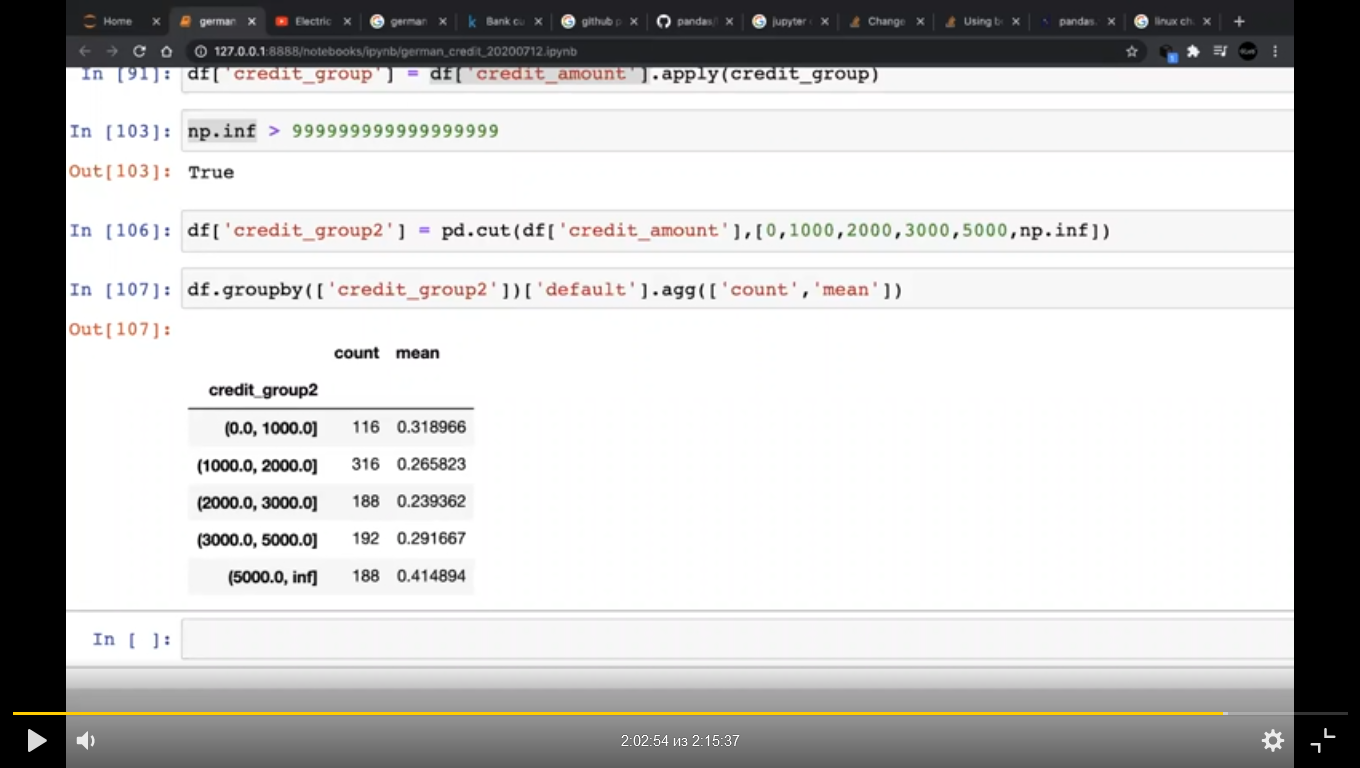
Как создавать сводные таблицы

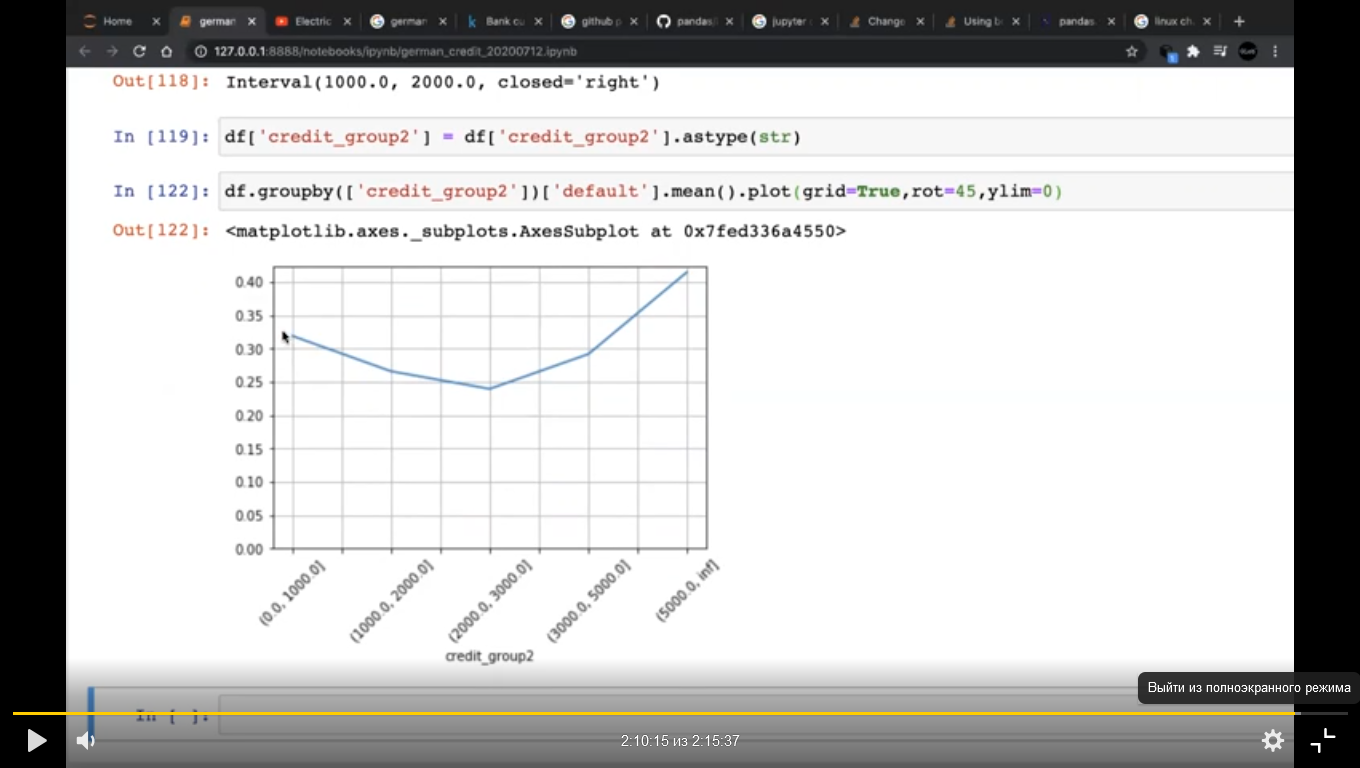


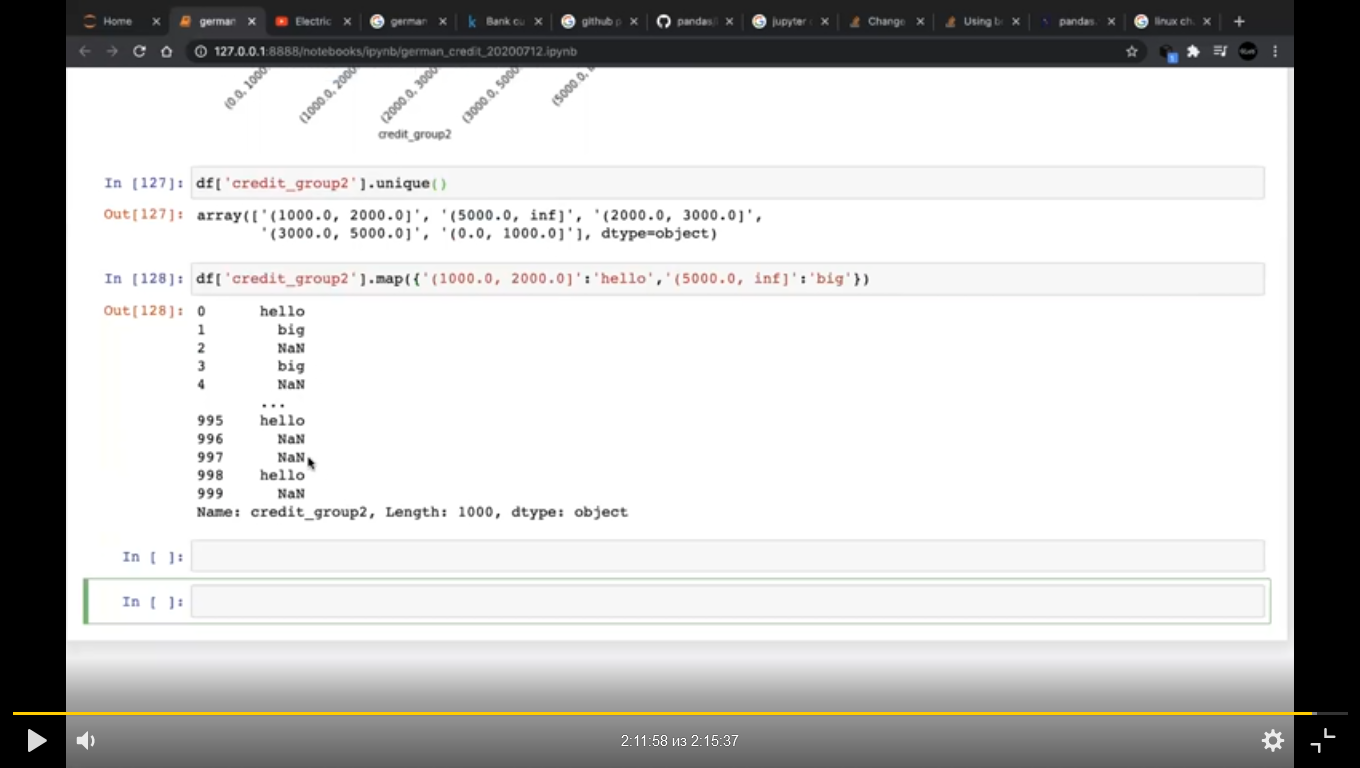


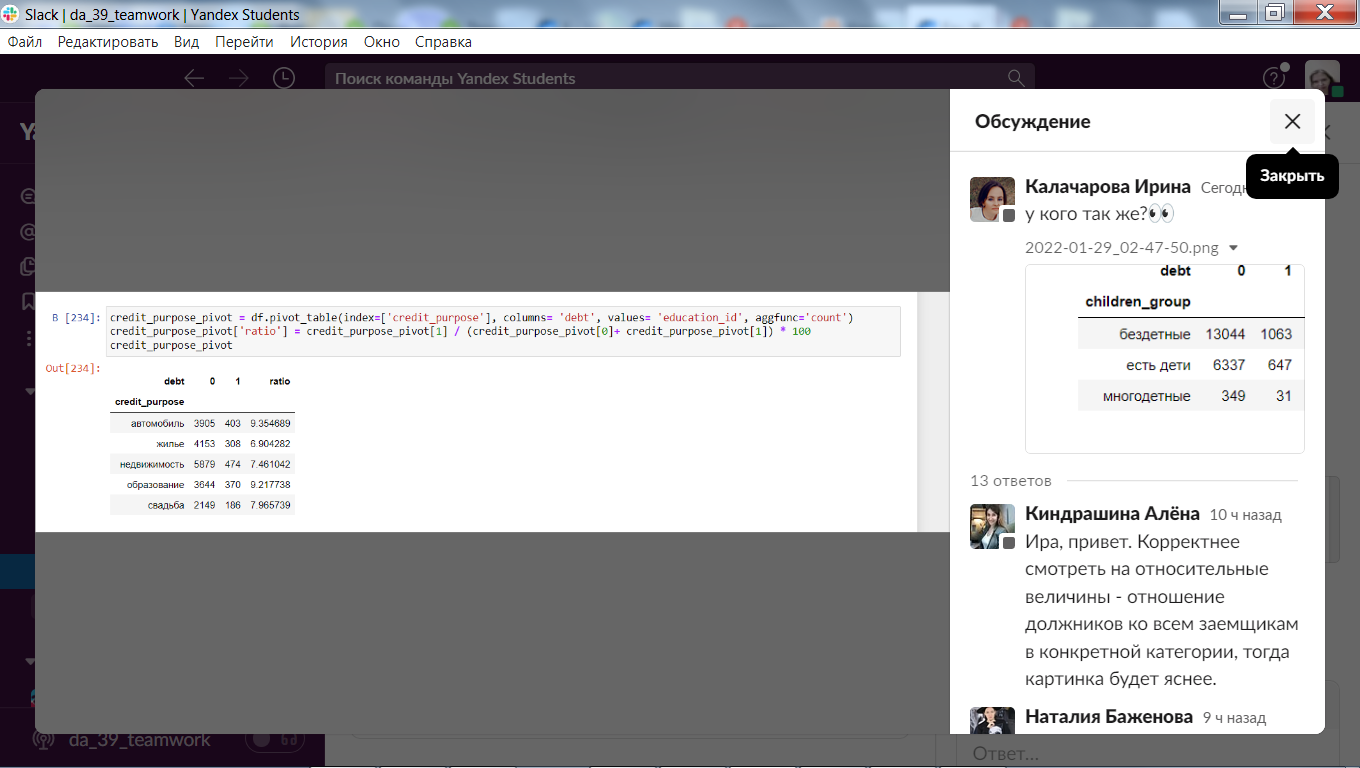
Как поделить на группы по доходам

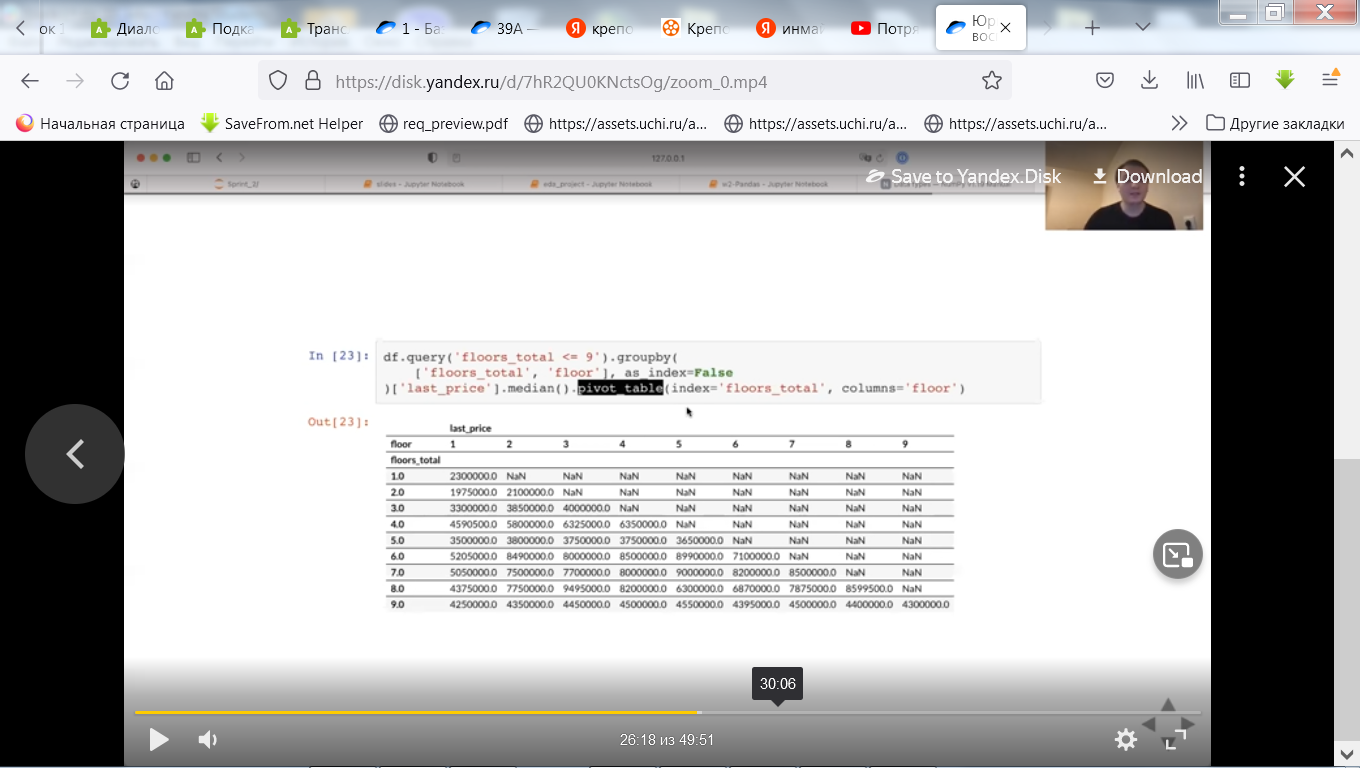




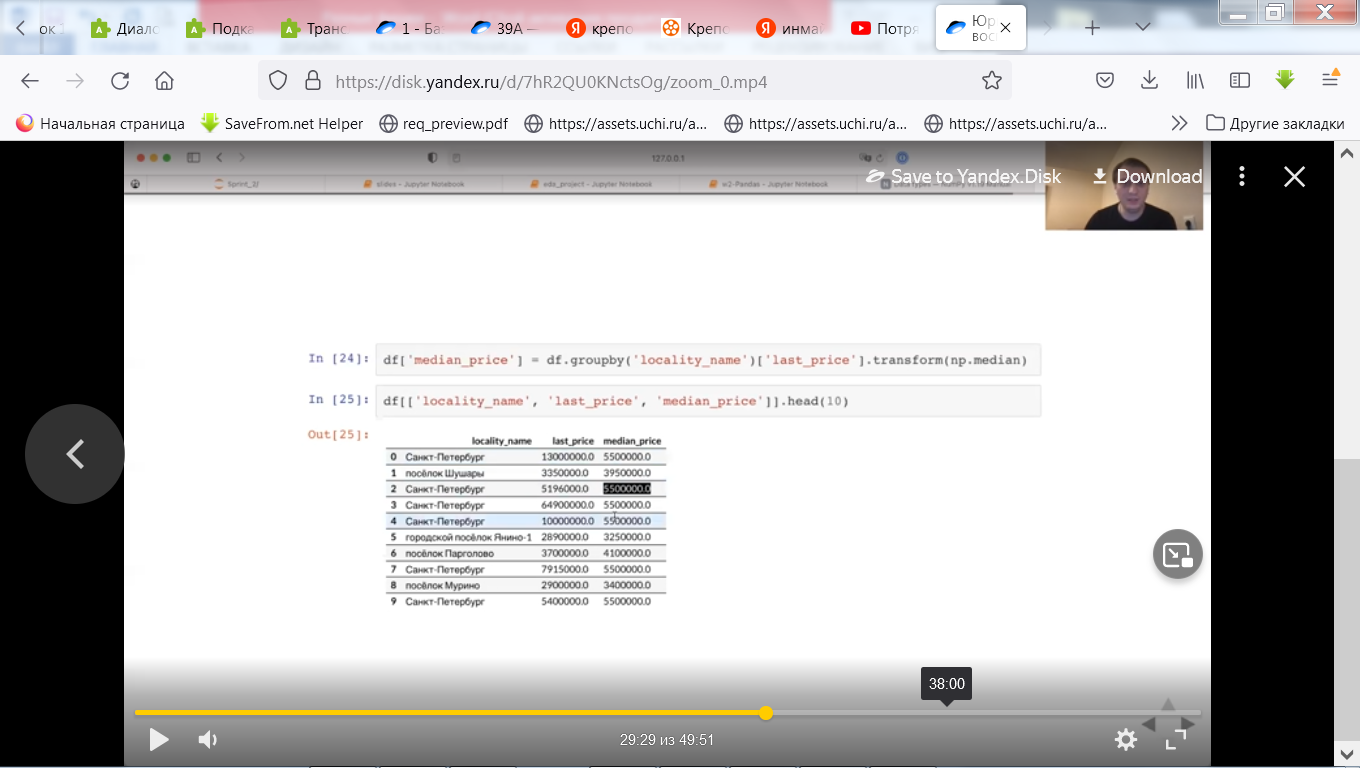


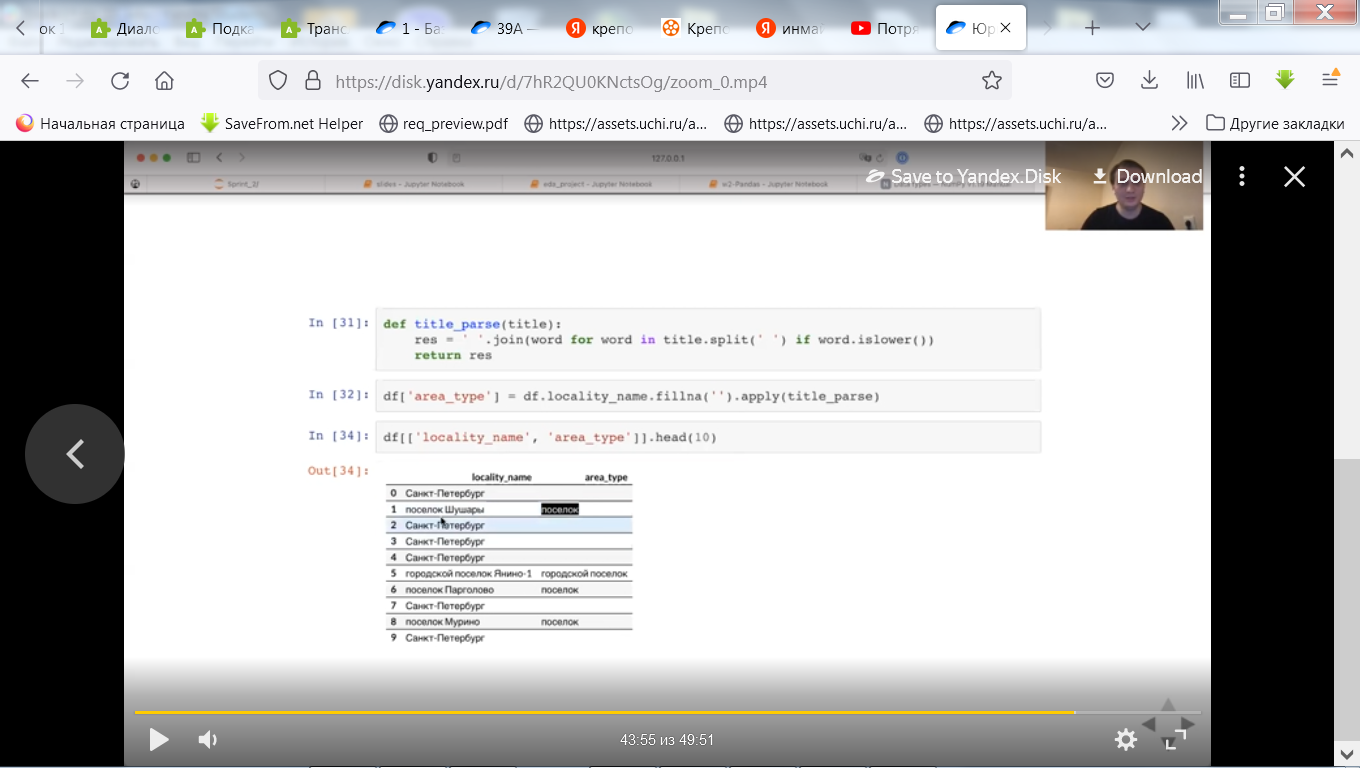


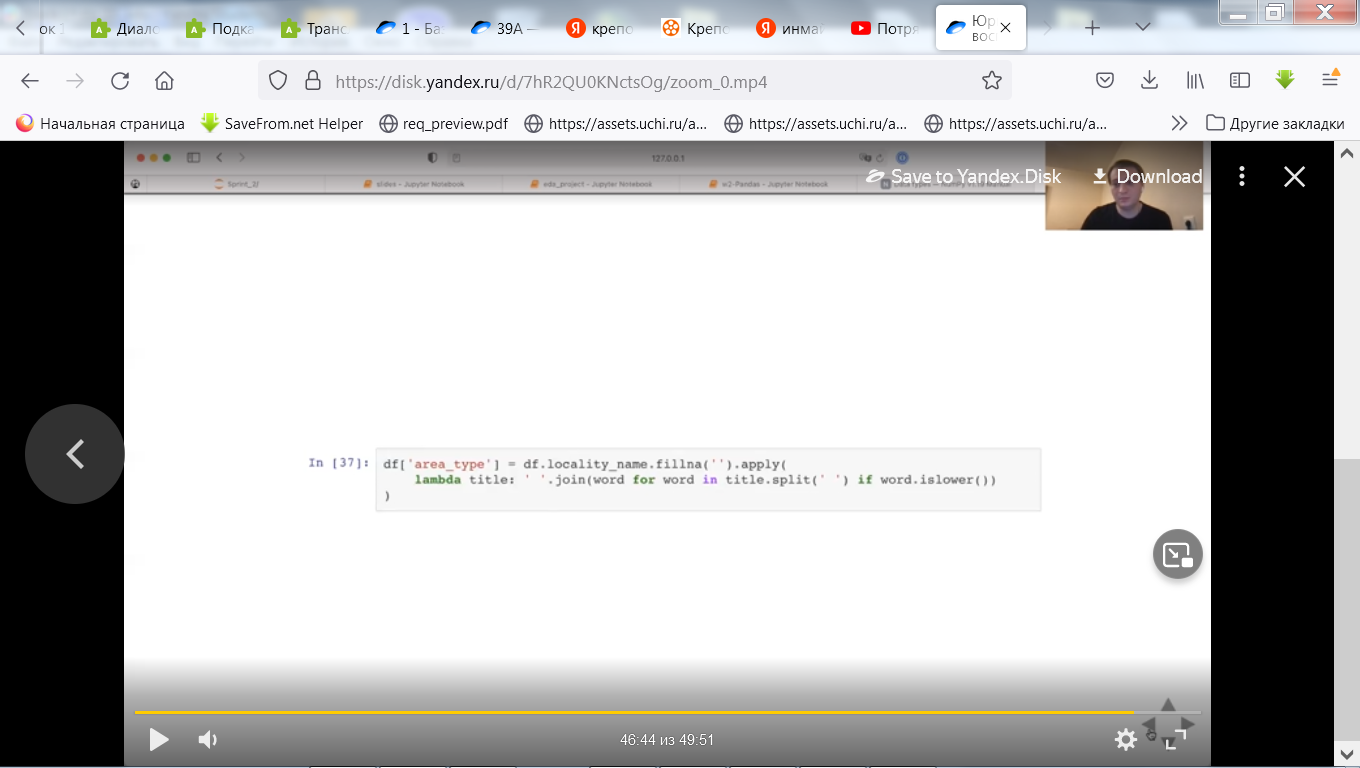


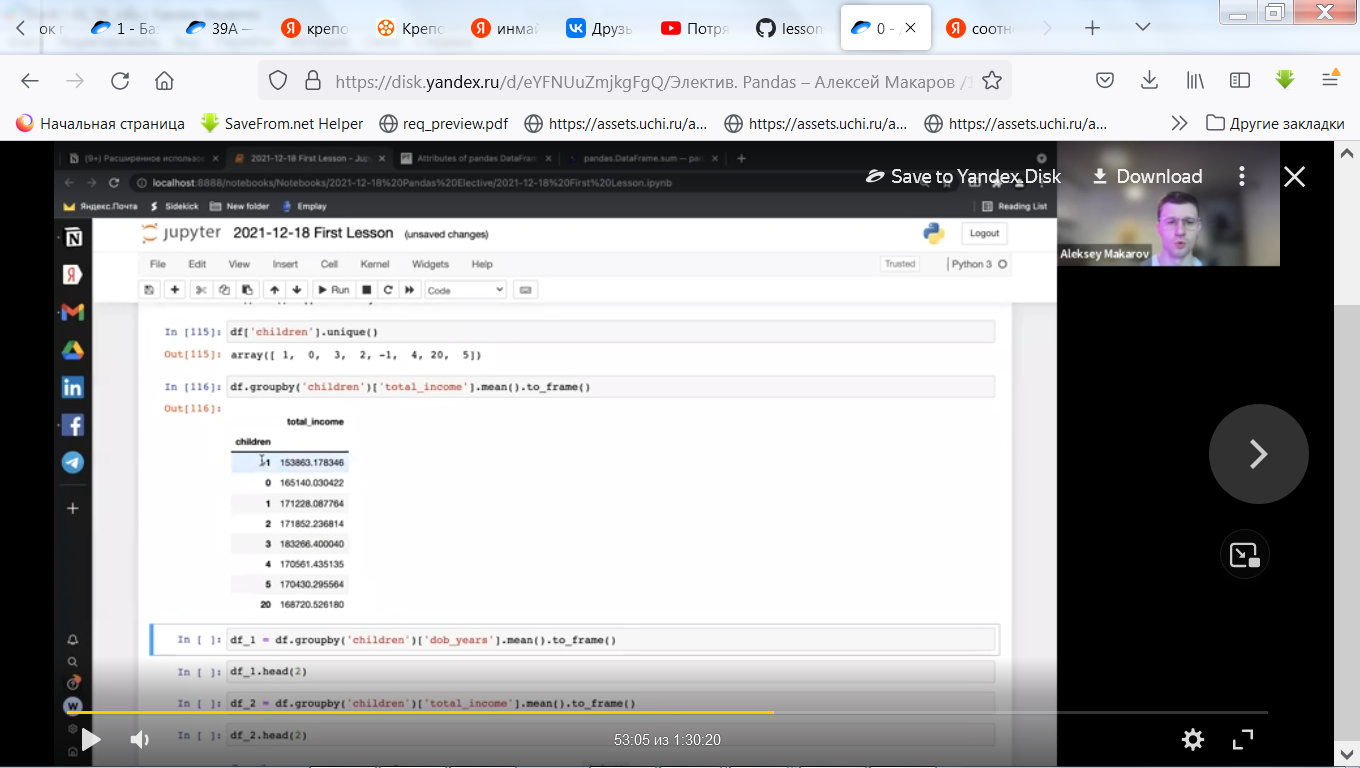


Как записать в новый столбец для каждой строки название группы









ПРО СЛОВАРЬ

mean\_income = pd.DataFrame({'mean\_income':mean\_income\_group}).reset\_index()

привет! Давай разбираться.  
mean\_income\_group - это pd.Series. Такой пандасовский словарик по сути где каждый элемент проиндексирован.  
{'mean\_income':mean\_income\_group} - ты упаковываешь его в словарь.  
pd.DataFrame({'mean\_income':mean\_income\_group}) - словарь преобразуется в pd.DataFrame. mean\_income - название единственного столбца, а индексы mean\_income\_group становятся индексами строк датафрейма.  
pd.DataFrame({'mean\_income':mean\_income\_group}).reset\_index() Индексы

датафрейма становятся вторым (или сколько там у тебя уровней индекса) столбцом. Создаётся новый целочисленный индекс для строк.

У меня вот, так  #Создадим ключи к словарю в категории family\_status'  
family\_log={0: 'женат / замужем', 1:'гражданский брак', 2: 'вдовец / вдова', 3: 'в разводе', 4: 'не женат / не замужем'}  
df['family\_status\_id'].map(family\_log)  
family\_result =family\_log.get(0)                                                    А столбец  df['family\_final ']= df['family\_status\_id'].map(family\_log)?  верно?  Т.е ключ тянет из  family\_log?  В   df['family\_final ']?

**Виды графиков для различной информации**

Как узнать, какой вид графика подходит?

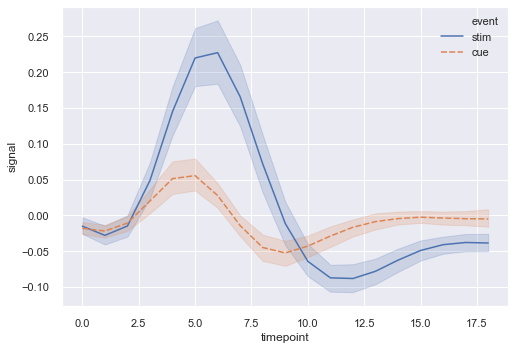
Можно попробовать все виды и определить лучший. Однако быстрее — понять, что именно вы хотите отобразить на графике и выбрать подходящий вариант.

Что чаще всего отображают на графиках?

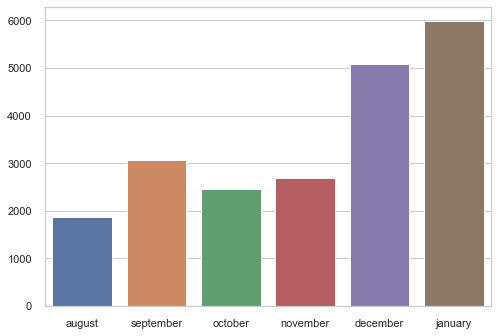
**Сравнение во времени**

Сколько временных периодов нужно отразить?

* Если много, рисуйте линейный график:



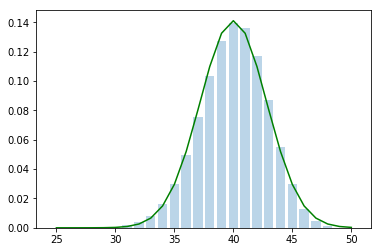
* Если несколько (например, для нескольких месяцев), нагляднее столбчатая гистограмма:



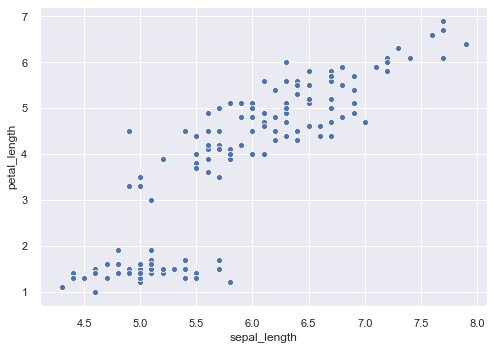
**Распределение** **набора данных**

Сколько переменных нужно показать?

* Одну. Подойдёт столбчатая гистограмма. Также можно добавить наложение линейного графика, который отображает, например, нормальное распределение.

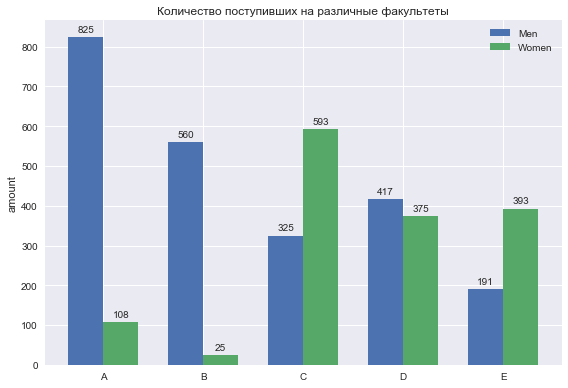


* Две. Диаграмма рассеяния. Она также подойдёт, чтобы показать взаимосвязь данных:



**Распределение данных по категориям**

Если важно наблюдать изменения во времени — стройте столбчатую гистограмму. Если нет, то круговую диаграмму, но будьте внимательны с отображением долей.



**Воронки и другие специальные графики**

Эти графики не классические, но их часто нужно строить. К специальным графикам относится, например, тепловая карта:



После выбора диаграммы, кажется, можно выдохнуть. Только график — система из нескольких элементов. Убедитесь, что:

* **цвет**
  + подобран согласно цветовой гамме;
  + не помешает восприятию в случае чёрно-белой печати;
  + не вызовет проблем у людей с нарушениями цветового зрения;
  + контрастирует с фоном.
* **текст**
  + читаемый;
  + расположен горизонтально;
  + данные на графике подписаны;
  + есть легенда.
* **линии**
  + линии сетки (если есть) скрыты;
  + у графика нет рамки;
  + на осях нет ненужных отметок;
  + на графике одна горизонтальная и одна вертикальная ось.
* **общий вид**
  + пропорции соблюдены;
  + расстояния между осями равные;
  + график двухмерный;
  + минимум украшательств.

Сохраните этот чек-лист для проверки результата визуализации. График должен поддерживать выводы и результаты анализа.

Вот памятка по методам, которые помогут построить информативный график в *matplotlib*.

1. Заголовок графика добавляют методом **set\_title()**:

Скопировать кодPYTHON

import matplotlib.pyplot as plt

fig, test = plt.subplots()

test.set\_title('Нормальное распределение')

1. Подписи к осям — методами **set\_xlabel()** и **set\_ylabel()**:

Скопировать кодPYTHON

import matplotlib.pyplot as plt

fig, test = plt.subplots()

test.set\_xlabel('Подпись для оси X')

test.set\_ylabel('Подпись для оси Y')

1. Легенду — методом **legend()**:

Скопировать кодPYTHON

import matplotlib.pyplot as plt

fig, test = plt.subplots()

test.plot(x, y1, label='Подпись 1')

test.plot(x, y2, label='Подпись 2')

test.plot(x, y3, label='Подпись 3')

test.legend()

1. Изменяют размеры графика методами **set\_figheight()** и **set\_figwidth()**:

Скопировать кодPYTHON

import matplotlib.pyplot as plt

fig, test = plt.subplots()

fig.set\_figheight(10) *# высота*

fig.set\_figwidth(8) *# ширина*

Перейти к заданию

Чат поддержки