Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №1
по дисциплине
«Методы машинного обучения»
на тему
«Разведочный анализ данных. Исследование и визуализация данных»

Выполнил:
студент группы РТ5-61Б
Корякин Д.

_____

Москва — 2019 г.

1. Лабораторная работа 1

1.1. 1) Текстовое описание набора данных

В качестве набора данных мы будем использовать набор данных по курсам на Udemy - https://www.kaggle.com/andrewmvd/udemy-courses/data

Датасет состоит из одного файла: - udemy_cources.csv - обучающая выборка

Файл содержит следующие колонки:

- course_id - id курса
- course_title - назввание курса
- url - ссылка на курс
- is_paid - платный ли курс
- price - цена курса
- num_subscribers - количество подписчиков
- num_reviews - количество просмотров
- num_lectures - количество лекций
- level - уровень
- content_duration - длительность контента
- published_timestamp - дата публикации
- subject - предмет

2. Импорт библиотек

Импортируем библиотеки с помощью команды import. Как правило, все команды import размещают в первой ячейке ноутбука, но мы в этом примере будем подключать все библиотеки последовательно, по мере их использования.

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

3. Загрузка данных

Загрузим файлы датасета в помощью библиотеки Pandas.

Не смотря на то, что файлы имеют расширение txt они представляют собой данные в формате CSV (https://ru.wikipedia.org/wiki/CSV). Часто в файлах такого формата в качестве разделителей используются символы “,”, “;” или табуляция. Поэтому вызывая метод read_csv всегда стоит явно указывать разделитель данных с помощью параметра sep. Чтобы узнать какой разделитель используется в файле его рекомендуется предварительно посмотреть в любом текстовом редакторе.

[2]:
```
# Будем анализировать данные только на обучающей выборке
data = pd.read_csv('data/udemy_courses.csv', sep=",")
```

4. 2) Основные характеристики датасета

[3]:
```
# Первые 5 строк датасета
data.head()
```

[3]:
```
   course_id                                       course_title  \
0    1070968                 Ultimate Investment Banking Course
1    1113822  Complete GST Course & Certification - Grow You…
2    1006314  Financial Modeling for Business Analysts and C…
3    1210588  Beginner to Pro - Financial Analysis in Excel …
4    1011058       How To Maximize Your Profits Trading Options

                                                 url  is_paid ▫
 ↪price  \
0  https://www.udemy.com/ultimate-investment-bank…     True    200
1      https://www.udemy.com/goods-and-services-tax/     True     ▫
 ↪75
2  https://www.udemy.com/financial-modeling-for-b…     True     45
3  https://www.udemy.com/complete-excel-finance-c…     True     95
4  https://www.udemy.com/how-to-maximize-your-pro…     True    200

   num_subscribers  num_reviews  num_lectures              level▫
 ↪ \
0             2147           23            51          All Levels
1             2792          923           274          All Levels
2             2174           74            51  Intermediate Level
3             2451           11            36          All Levels
4             1276           45            26  Intermediate Level

   content_duration    published_timestamp           subject
0               1.5  2017-01-18T20:58:58Z  Business Finance
1              39.0  2017-03-09T16:34:20Z  Business Finance
2               2.5  2016-12-19T19:26:30Z  Business Finance
3               3.0  2017-05-30T20:07:24Z  Business Finance
4               2.0  2016-12-13T14:57:18Z  Business Finance
```

[4]:
```
# Размер датасета - 8143 строк, 7 колонок
data.shape
```

[4]:
```
(3678, 12)
```

[5]:
```
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
```

```
Всего строк: 3678
```

[6]:
```
# Список колонок
data.columns
```

[6]:
```
Index(['course_id', 'course_title', 'url', 'is_paid', 'price',
       'num_subscribers', 'num_reviews', 'num_lectures', 'level',
       'content_duration', 'published_timestamp', 'subject'],
      dtype='object')
```

[7]:
```
# Список колонок с типами данных
data.dtypes
```

[7]:
```
course_id               int64
course_title           object
url                    object
is_paid                  bool
price                   int64
num_subscribers         int64
num_reviews             int64
num_lectures            int64
level                  object
content_duration      float64
published_timestamp    object
subject                object
dtype: object
```

[8]:
```
# Проверим наличие пустых значений
# Цикл по колонкам датасета
for col in data.columns:
    # Количество пустых значений - все значения заполнены
    temp_null_count = data[data[col].isnull()].shape[0]
    print('{} - {}'.format(col, temp_null_count))
```

```
course_id - 0
course_title - 0
url - 0
is_paid - 0
price - 0
num_subscribers - 0
num_reviews - 0
num_lectures - 0
level - 0
content_duration - 0
published_timestamp - 0
subject - 0
```

[9]:
```
# Основные статистические характеристки набора данных
data.describe()
```

[9]:
```
          course_id        price  num_subscribers   num_reviews ▫
  ↪num_lectures  \
count  3.678000e+03  3678.000000      3678.000000   3678.000000  ▫
  ↪3678.000000
```

```
mean    6.759720e+05    66.049483      3197.150625    156.259108
  ↪  40.108755
std     3.432732e+05    61.005755      9504.117010    935.452044
  ↪  50.383346
min     8.324000e+03     0.000000         0.000000      0.000000
  ↪   0.000000
25%     4.076925e+05    20.000000       111.000000      4.000000
  ↪  15.000000
50%     6.879170e+05    45.000000       911.500000     18.000000
  ↪  25.000000
75%     9.613555e+05    95.000000      2546.000000     67.000000
  ↪  45.750000
max     1.282064e+06   200.000000    268923.000000  27445.000000
  ↪ 779.000000

       content_duration
count       3678.000000
mean           4.094517
std            6.053840
min            0.000000
25%            1.000000
50%            2.000000
75%            4.500000
max           78.500000
```

[11]:
```
# Определим уникальные значения для целевого признака
data['price'].unique()
```

[11]:
```
array([200,  75,  45,  95, 150,  65, 195,  30,  20,  50, 175,
  ↪140, 115,
       190, 125,  60, 145, 105, 155, 185, 180, 120,  25, 160,
  ↪40,   0,
       100,  90,  35,  80,  70,  55, 165, 130,  85, 170, 110,
  ↪135])
```

Целевой признак является бинарным и содержит только значения 0 и 1.

5. 3) Визуальное исследование датасета

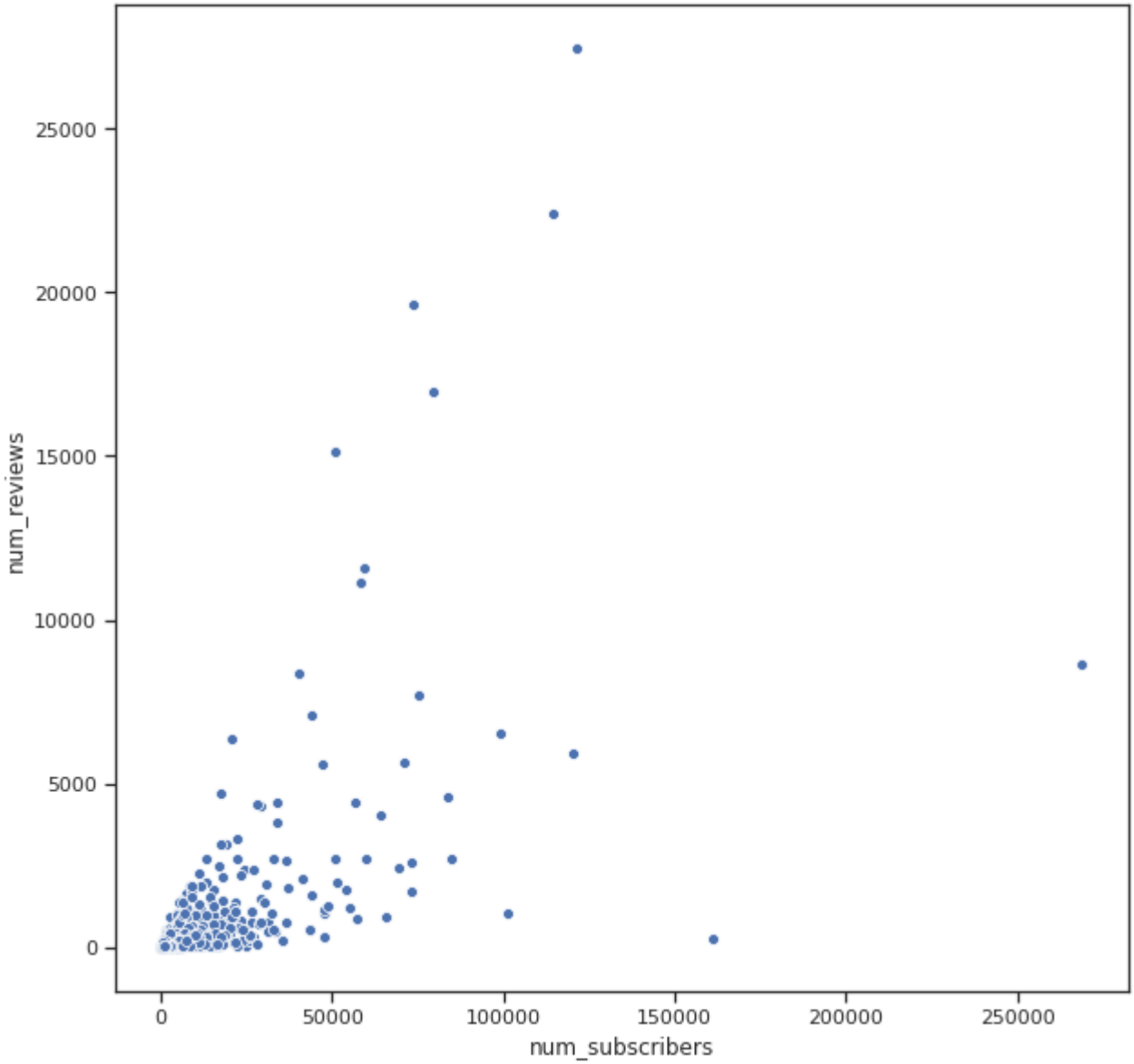
Для визуального исследования могут быть использованы различные виды диаграмм, мы построим только некоторые варианты диаграмм, которые используются достаточно часто.

5.0.1. Диаграмма рассеяния

Позволяет построить распределение двух колонок данных и визуально обнаружить наличие зависимости. Не предполагается, что значения упорядочены (например, по времени).

[12]:
```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='num_subscribers', y='num_reviews',
  ↪data=data)
```

[12]: <matplotlib.axes._subplots.AxesSubplot at 0x7f5b1428a510>

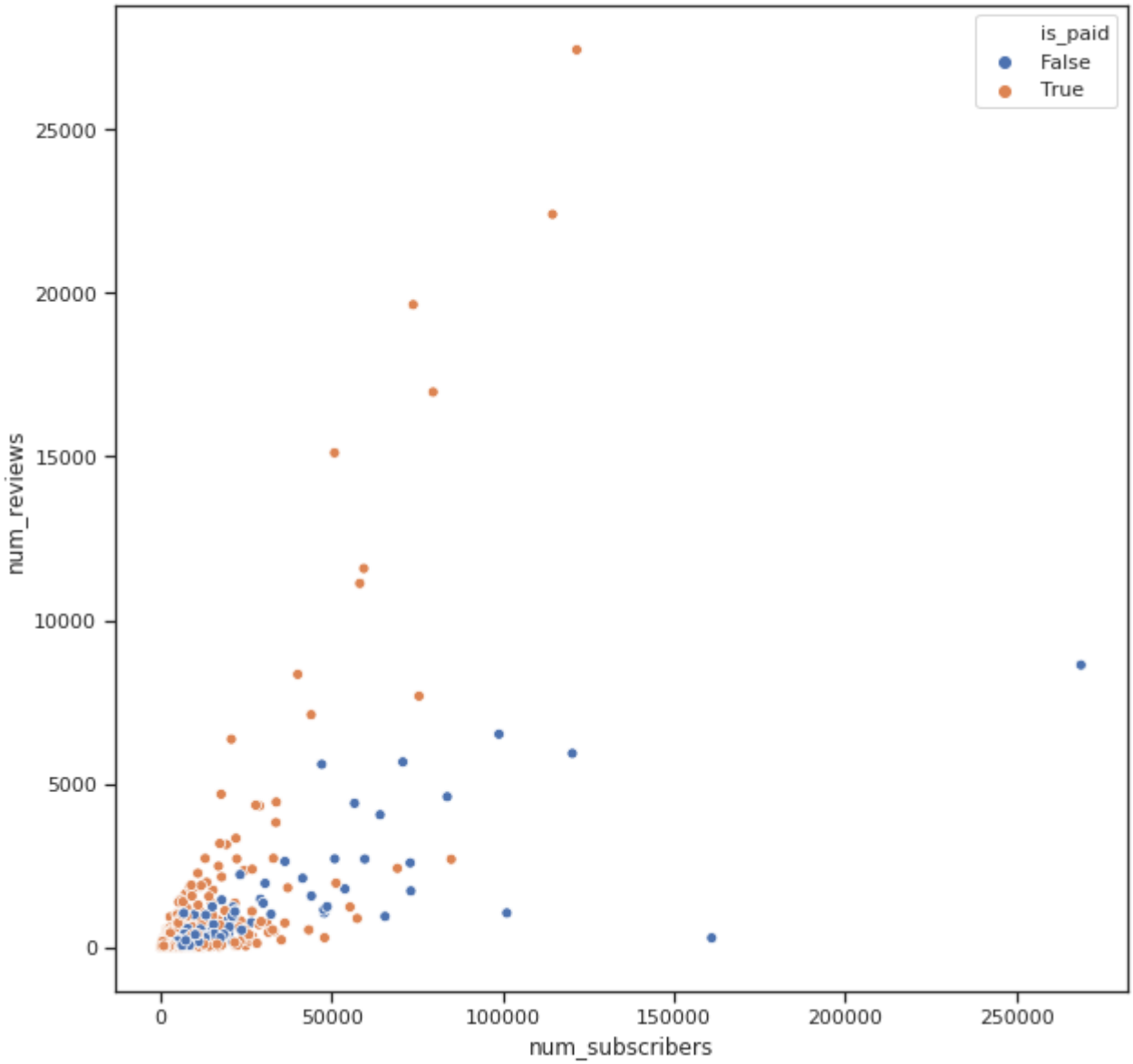


Можно видеть что между полями Humidity и HumidityRatio пристутствует почти линейная зависимость.

Посмотрим насколько на эту зависимость влияет целевой признак.

[13]:
```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='num_subscribers', y='num_reviews',
 ↪data=data, hue='is_paid')
```

[13]: <matplotlib.axes._subplots.AxesSubplot at 0x7f5ae8419150>

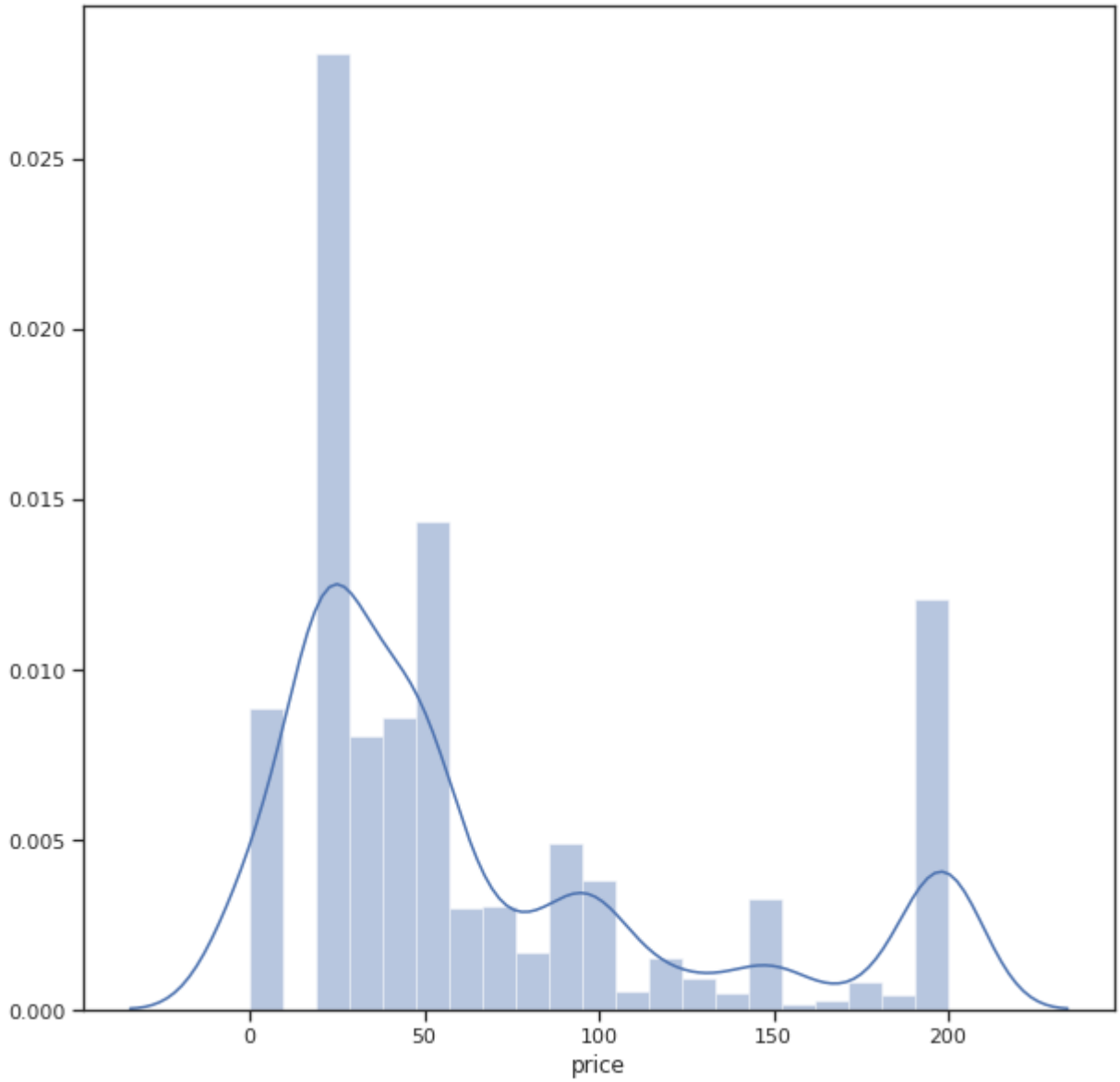


5.1. Гистограмма

Позволяет оценить плотность вероятности распределения данных.

[14]:
```
fig, ax = plt.subplots(figsize=(10,10))
sns.distplot(data['price'])
```

[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5ae85f2ed0>
```

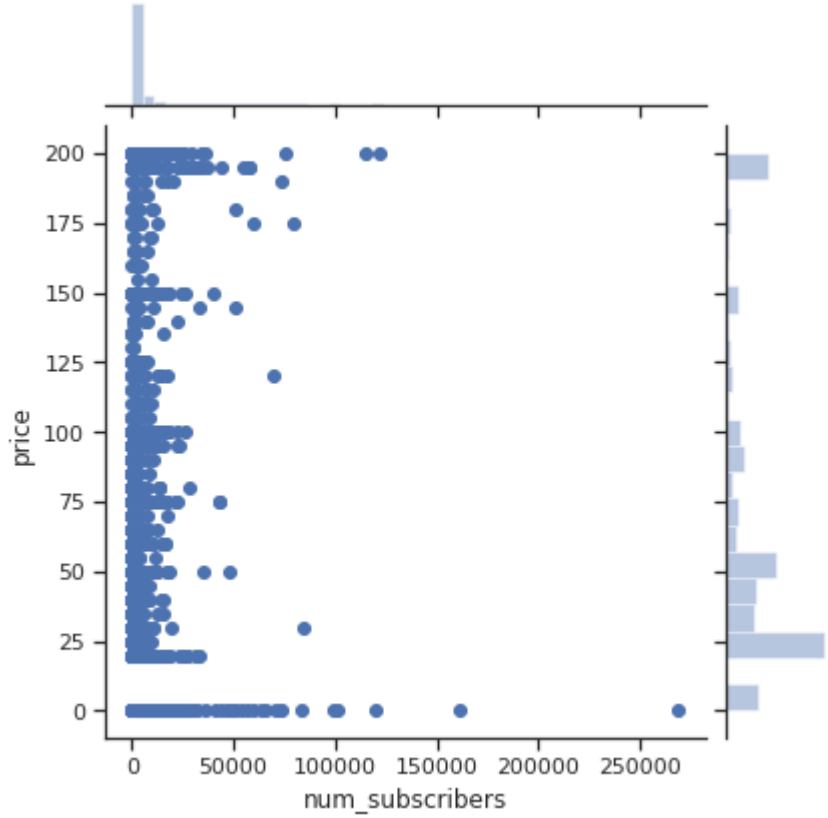


5.1.1. Jointplot

Комбинация гистограмм и диаграмм рассеивания.

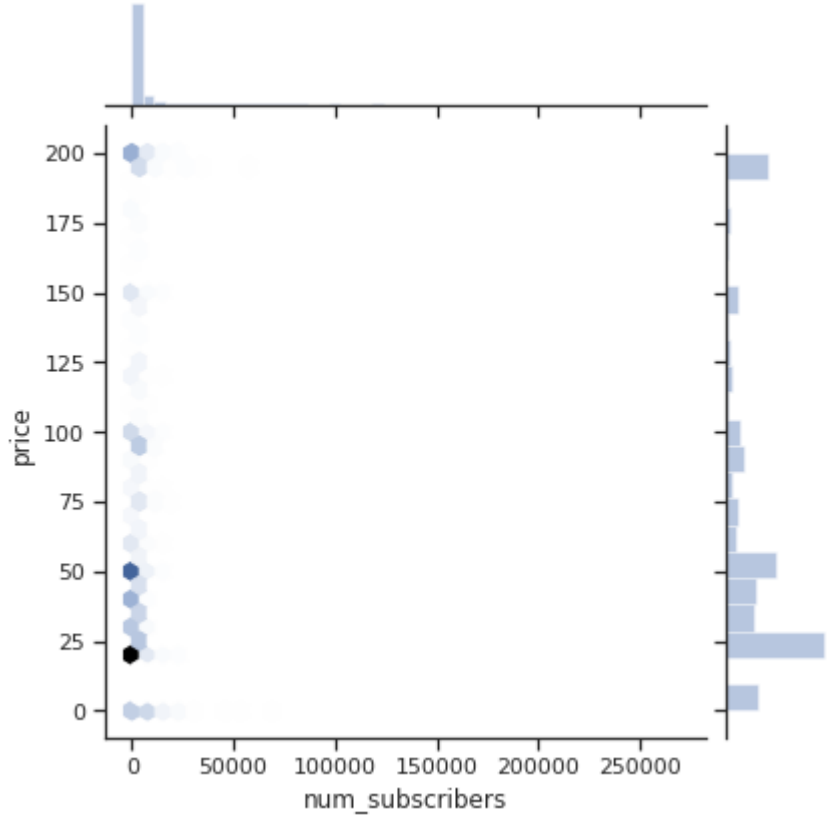
[15]:
```
sns.jointplot(x='num_subscribers', y='price', data=data)
```

[15]:
```
<seaborn.axisgrid.JointGrid at 0x7f5ae8512c50>
```



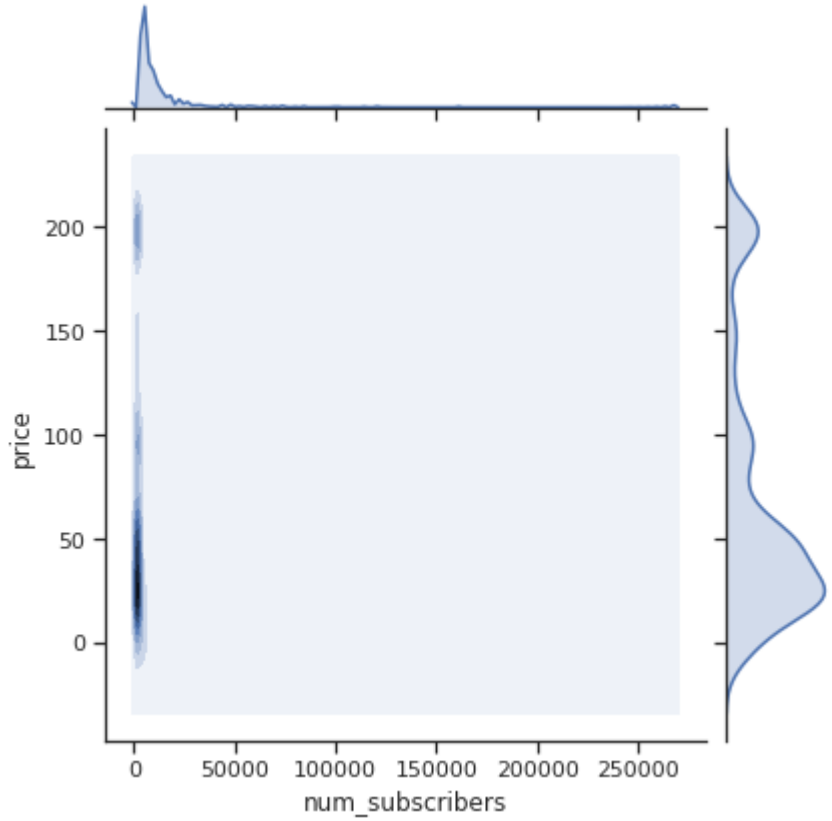
[16]:
```
sns.jointplot(x='num_subscribers', y='price', data=data,
  ↪kind="hex")
```

[16]: <seaborn.axisgrid.JointGrid at 0x7f5ae81113d0>



[17]:
```
sns.jointplot(x='num_subscribers', y='price', data=data,
↪kind="kde")
```

[17]: <seaborn.axisgrid.JointGrid at 0x7f5ae7f78190>

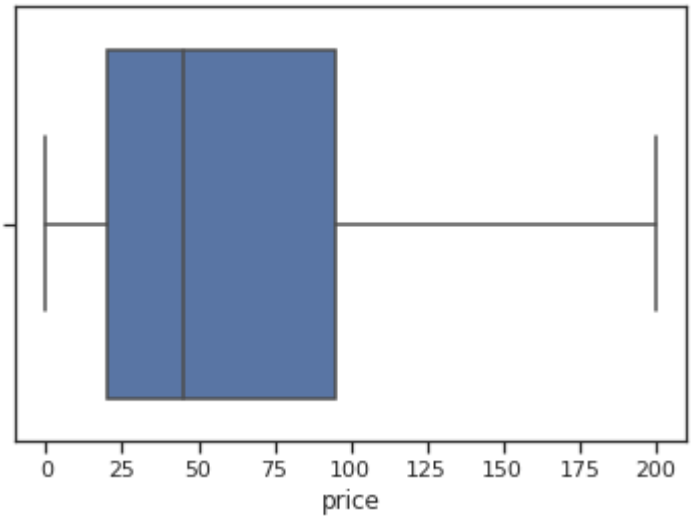


5.1.2. Ящик с усами

Отображает одномерное распределение вероятности.

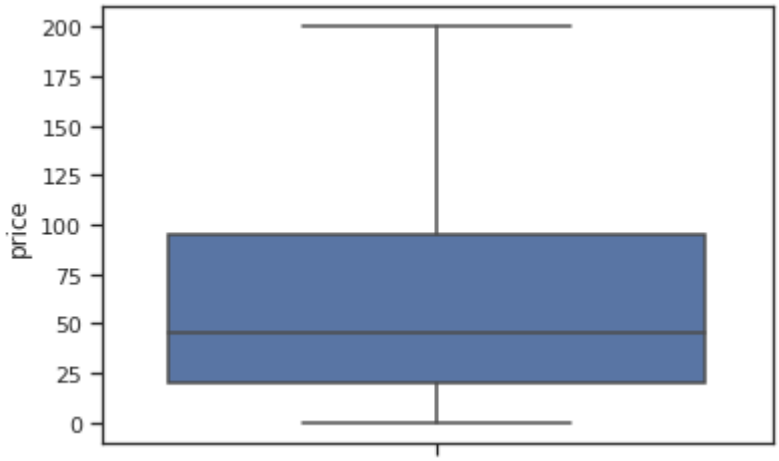
[18]:
```
sns.boxplot(x=data['price'])
```

[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5ae7f78c90>
```



[19]:
```
# По вертикали
sns.boxplot(y=data['price'])
```

[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5ae7e0c290>
```

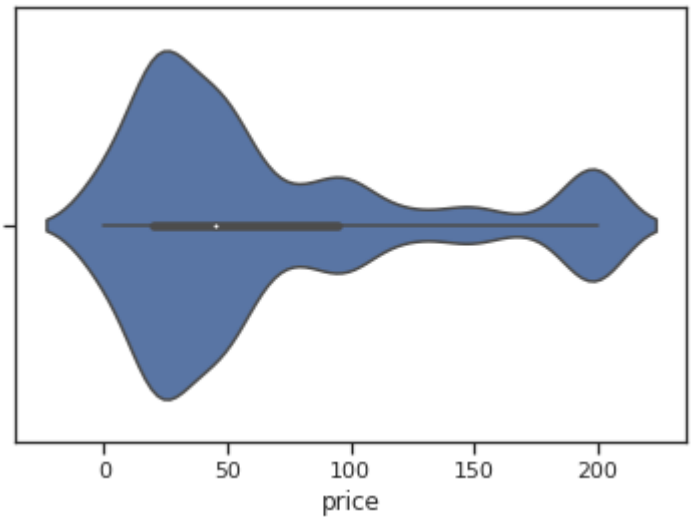


5.1.3. Violin plot

Похоже на предыдущую диаграмму, но по краям отображаются распределения плотности - https://en.wikipedia.org/wiki/Kernel_density_estimation

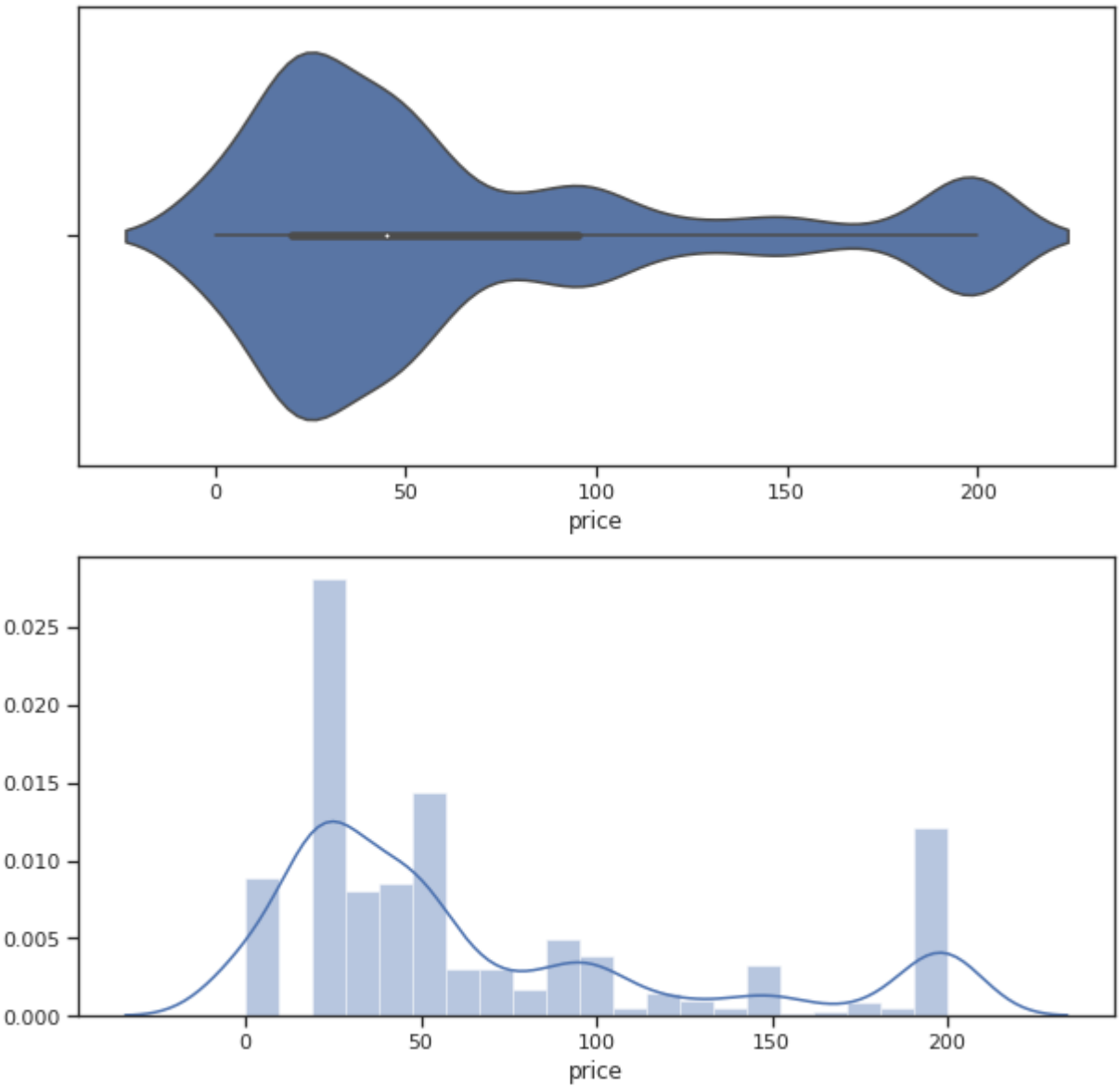
[22]:
```
sns.violinplot(x=data['price'])
```

[22]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5acf0d9890>
```



[23]:
```
fig, ax = plt.subplots(2, 1, figsize=(10,10))
sns.violinplot(ax=ax[0], x=data['price'])
sns.distplot(data['price'], ax=ax[1])
```

```
[23]: <matplotlib.axes._subplots.AxesSubplot at 0x7f5acdfc23d0>
```



6. 4) Информация о корреляции признаков

Проверка корреляции признаков позволяет решить две задачи: 1. Понять какие признаки (колонки датасета) наиболее сильно коррелируют с целевым признаком (в нашем примере это колонка “Price”). Именно эти признаки будут наиболее информативными для моделей машинного обучения. Признаки, которые слабо коррелируют с целевым признаком, можно попробовать исключить из построения модели, иногда это повышает качество модели. Нужно отметить, что некоторые алгоритмы машинного обучения автоматически определяют ценность того или иного признака для построения модели. 1. Понять какие нецелевые признаки линейно зависимы между собой. Линейно зависимые признаки, как правило, очень плохо влияют на качество моделей. Поэтому если несколько признаков линейно зависимы, то для построения модели из них выбирают какой-то один признак.

[30]:
```
data.corr()
```

[30]:
```
                  course_id   is_paid     price  num_subscribers ▫
  ↪num_reviews  \
course_id          1.000000 -0.013679  0.142319        -0.167856 ▫
  ↪   -0.058550
is_paid           -0.013679  1.000000  0.328513        -0.266159 ▫
  ↪   -0.087471
price              0.142319  0.328513  1.000000         0.050769 ▫
  ↪    0.113696
num_subscribers   -0.167856 -0.266159  0.050769         1.000000 ▫
  ↪    0.649946
num_reviews       -0.058550 -0.087471  0.113696         0.649946 ▫
  ↪    1.000000
num_lectures      -0.024646  0.112574  0.330160         0.157746 ▫
  ↪    0.243029
content_duration  -0.057223  0.094417  0.293450         0.161839 ▫
  ↪    0.228889

                  num_lectures  content_duration
course_id            -0.024646         -0.057223
is_paid               0.112574          0.094417
price                 0.330160          0.293450
num_subscribers       0.157746          0.161839
num_reviews           0.243029          0.228889
num_lectures          1.000000          0.801647
content_duration      0.801647          1.000000
```

Корреляционная матрица содержит коэффициенты корреляции между всеми парами признаков.

Корреляционная матрица симметрична относительно главной диагонали. На главной диагонали расположены единицы (корреляция признака самого с собой).

На основе корреляционной матрицы можно сделать следующие выводы: - Целевой признак наиболее сильно коррелирует с количеством лекций (0.3) и длительностью контента (0.3). Эти признаки обязательно следует оставить в модели. - Количество подписчиков сильно коррелирует с количеством просмотров (0.64) - Целевой признак слабо коррелирует с количеством подписчкиов (0.05) и количеством просмотров (0.11). Скорее всего эти признаки стоит исключить из модели, возможно они только ухудшат качество модели. - Количество лекций и длительность курса очень сильно коррелируют между собой (0.8).Поэтому из этих признаков в модели можно оставлять только один.

Описание метода corr - https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.

По умолчанию при построении матрицы используется коэффициент корреляции Пирсона. Возможно также построить корреляционную матрицу на основе коэффициентов корреляции Кендалла и Спирмена. На практике три метода редко дают значимые различия.

[31]:
```
data.corr(method='pearson')
```

[31]:
```
                  course_id   is_paid     price  num_subscribers ▫
  ↪num_reviews  \
```

```
course_id          1.000000 -0.013679  0.142319         -0.167856
 ↪  -0.058550
is_paid           -0.013679  1.000000  0.328513         -0.266159
 ↪  -0.087471
price              0.142319  0.328513  1.000000          0.050769
 ↪   0.113696
num_subscribers   -0.167856 -0.266159  0.050769          1.000000
 ↪   0.649946
num_reviews       -0.058550 -0.087471  0.113696          0.649946
 ↪   1.000000
num_lectures      -0.024646  0.112574  0.330160          0.157746
 ↪   0.243029
content_duration  -0.057223  0.094417  0.293450          0.161839
 ↪   0.228889

                  num_lectures  content_duration
course_id            -0.024646         -0.057223
is_paid               0.112574          0.094417
price                 0.330160          0.293450
num_subscribers       0.157746          0.161839
num_reviews           0.243029          0.228889
num_lectures          1.000000          0.801647
content_duration      0.801647          1.000000
```

[32]:
```
data.corr(method='kendall')
```

[32]:
```
                  course_id   is_paid     price  num_subscribers
 ↪num_reviews  \
course_id          1.000000 -0.013527  0.090562         -0.122772
 ↪  -0.116404
is_paid           -0.013527  1.000000  0.413177         -0.255425
 ↪  -0.218237
price              0.090562  0.413177  1.000000          0.052506
 ↪   0.105034
num_subscribers   -0.122772 -0.255425  0.052506          1.000000
 ↪   0.602175
num_reviews       -0.116404 -0.218237  0.105034          0.602175
 ↪   1.000000
num_lectures      -0.020397  0.138209  0.274946          0.143285
 ↪   0.239496
content_duration  -0.058723  0.125193  0.258408          0.119035
 ↪   0.230950

                  num_lectures  content_duration
course_id            -0.020397         -0.058723
is_paid               0.138209          0.125193
price                 0.274946          0.258408
num_subscribers       0.143285          0.119035
num_reviews           0.239496          0.230950
```

```
num_lectures          1.000000          0.639178
content_duration      0.639178          1.000000
```

[33]:
```
data.corr(method='spearman')
```

[33]:
```
                  course_id   is_paid     price  num_subscribers ▫
  ↪num_reviews  \
course_id          1.000000 -0.016565  0.129117        -0.180870 ▫
  ↪   -0.170728
is_paid           -0.016565  1.000000  0.484908        -0.312580 ▫
  ↪   -0.264634
price              0.129117  0.484908  1.000000         0.067184 ▫
  ↪    0.143543
num_subscribers   -0.180870 -0.312580  0.067184         1.000000 ▫
  ↪    0.785341
num_reviews       -0.170728 -0.264634  0.143543         0.785341 ▫
  ↪    1.000000
num_lectures      -0.030456  0.167880  0.386699         0.210167 ▫
  ↪    0.341328
content_duration  -0.085031  0.147965  0.352967         0.169809 ▫
  ↪    0.323214

                  num_lectures  content_duration
course_id            -0.030456         -0.085031
is_paid               0.167880          0.147965
price                 0.386699          0.352967
num_subscribers       0.210167          0.169809
num_reviews           0.341328          0.323214
num_lectures          1.000000          0.805285
content_duration      0.805285          1.000000
```

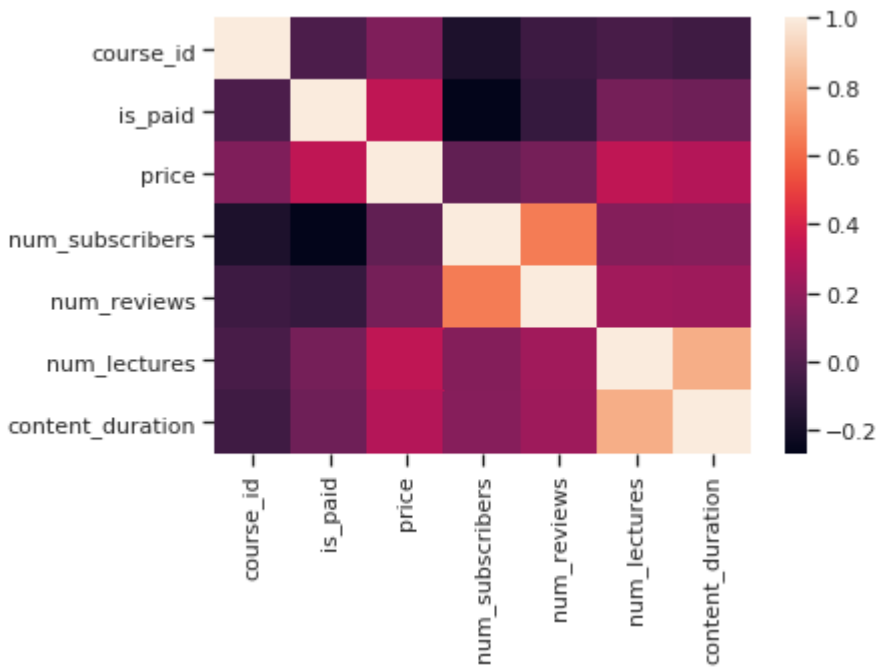
В случае большого количества признаков анализ числовой корреляционной матрицы становится неудобен.

Для визуализации корреляционной матрицы будем использовать “тепловую карту” heatmap которая показывает степень корреляции различными цветами.

Используем метод heatmap библиотеки seaborn - https://seaborn.pydata.org/generated/seaborn.heatma

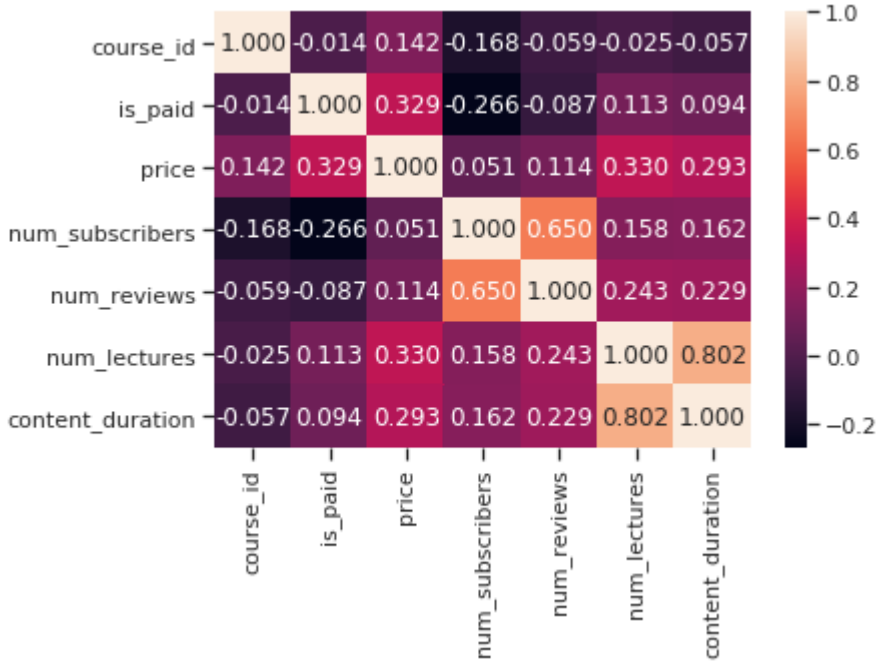
[34]:
```
sns.heatmap(data.corr())
```

[34]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5ac39fc1d0>
```



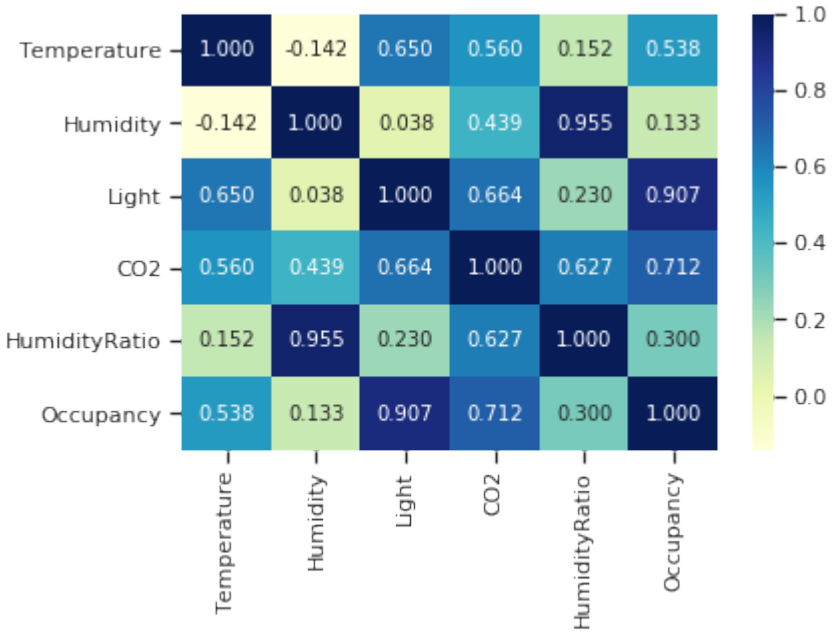
[35]:
```
# Вывод значений в ячейках
sns.heatmap(data.corr(), annot=True, fmt='.3f')
```

[35]: <matplotlib.axes._subplots.AxesSubplot at 0x7f5ac1939d10>



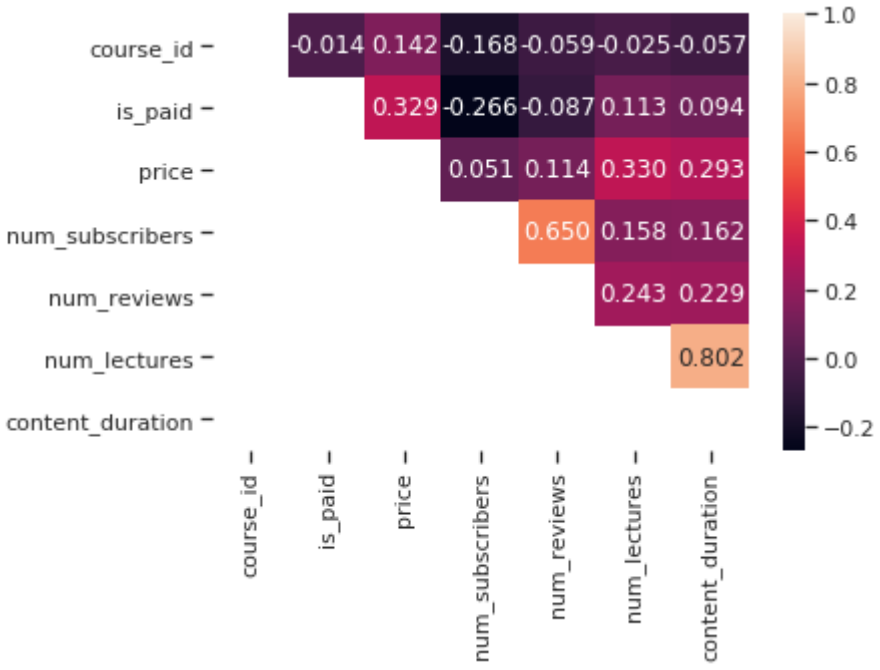
[19]:
```
# Изменение цветовой гаммы
sns.heatmap(data.corr(), cmap='YlGnBu', annot=True, fmt='.3f')
```

[19]: <matplotlib.axes._subplots.AxesSubplot at 0x7f4a28e8c940>



[36]:
```
# Треугольный вариант матрицы
mask = np.zeros_like(data.corr(), dtype=np.bool)
# чтобы оставить нижнюю часть матрицы
# mask[np.triu_indices_from(mask)] = True
# чтобы оставить верхнюю часть матрицы
mask[np.tril_indices_from(mask)] = True
sns.heatmap(data.corr(), mask=mask, annot=True, fmt='.3f')
```

[36]: <matplotlib.axes._subplots.AxesSubplot at 0x7f5ac19216d0>



[37]:
```
fig, ax = plt.subplots(1, 3, sharex='col', sharey='row',
↪figsize=(15,5))
sns.heatmap(data.corr(method='pearson'), ax=ax[0], annot=True,
↪fmt='.2f')
sns.heatmap(data.corr(method='kendall'), ax=ax[1], annot=True,
↪fmt='.2f')
sns.heatmap(data.corr(method='spearman'), ax=ax[2], annot=True,
↪fmt='.2f')
fig.suptitle('Корреляционные матрицы, построенные различными
↪методами')
ax[0].title.set_text('Pearson')
ax[1].title.set_text('Kendall')
ax[2].title.set_text('Spearman')
```