## Лабораторная работа 1

1. Установить *Python* с официального сайта сайта https://www.python.org/. При установке рекомендуется включить параметр «Add Python to PATH».
2. Для сдачи лабы достаточно показать работу с основными структурами данных (рассказать особенности)

## Лабораторная работа 2

1. Установка пакетов : **File-> Settings(Ctrl+Alt+S)->Project:Название проекта->Python interpreter->Через плюсик добавляем**
2. Массивы Numpy аналогичны массивам в других ЯП
3. **Series** - это структура данных, которая сочетает свойства одномерного массивы NumPy и словаря Python, т.е. доступ к каждому элементу может быть получен, либо с помощью индекса, либо с помощью идентификатора (ключа).

**Dataframe —** это табличная структура данных, напоминающая таблицы из Microsoft Excel.

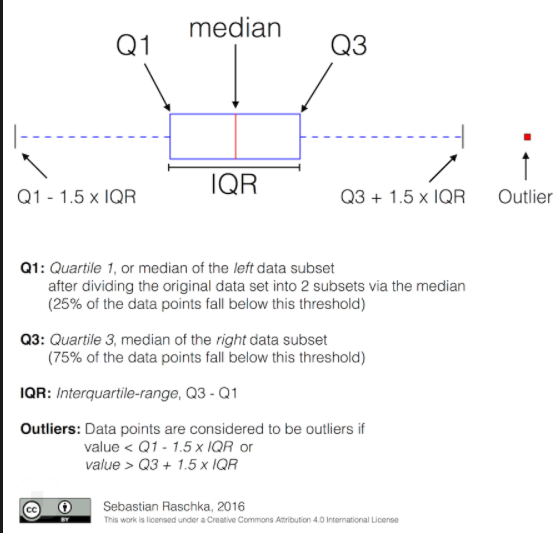
1. Простой график Matplotlib

## Лабораторная работа 3

1. Медиана – это среднее число в отсортированном, возрастающем или убывающем, списке чисел.
2. Задания кроме, 5 – построение графиков, использование dataframe, а также методов из библиотек (csv файл содержит долю браузеров за промежутки времени, mean- вычисление среднего значения). Далее требовалось работать с каким-либо одним столбцом, я выбрал с firefox



1. Про boxplot: <https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.boxplot.html>



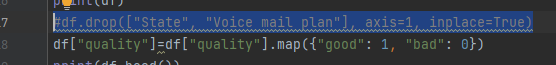
1. Метод describe выводит различную информацию (min,max,avg). При выводе 25% - значение означает, что 25% всех значений в dataframe меньше, чем указанное

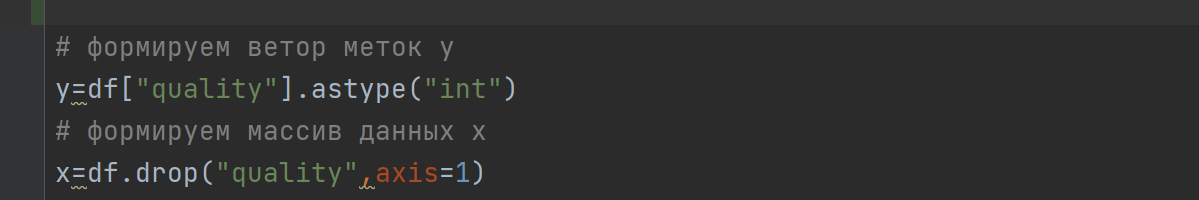
## Лабораторная работа 4

1. В лабе будем искать пропущенные данные, выводить информацию о них. В коде комментариями выделены задания. График с красными прямоугольниками отображает NaN значения в датасете(NaN выделены красным цветом) .
2. По графикам (Синие гистограммы) нужно сделать выводы о эффективности применения замены пропусков. Всего четыре графика: на первом отображены исходные данные с пропусками. 2 график показывает данные, где NaN заменены средним значением по столбу. 3 график показывает данные, где NaN заменены модой(наиболее часто встречающемся значением среди столбца Girth, среди всего датасета вроде бы нельзя). 4 график показывает данные, где NaN заменены предыдущим значением по столбцу.

## Лабораторная работа 5

1. ОБЯЗАТЕЛЬНО использовать уникальный dataset, скачать можно отсюда <https://www.kaggle.com/datasets?search=binary+classification&fileType=csv&sizeEnd=25%2CMB>
2. По лабе есть отчёт, он в папке report
3. Если есть строковые столбцы или bool (или подобные) их надо заменить или удалить

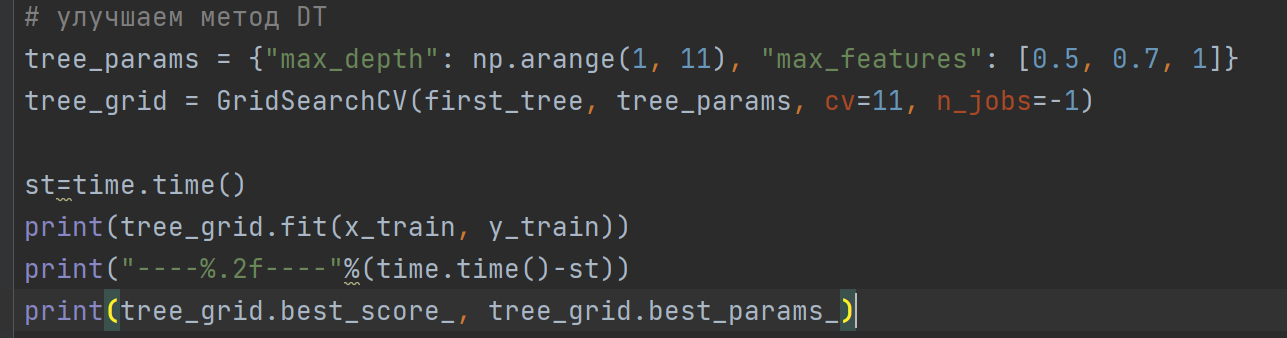


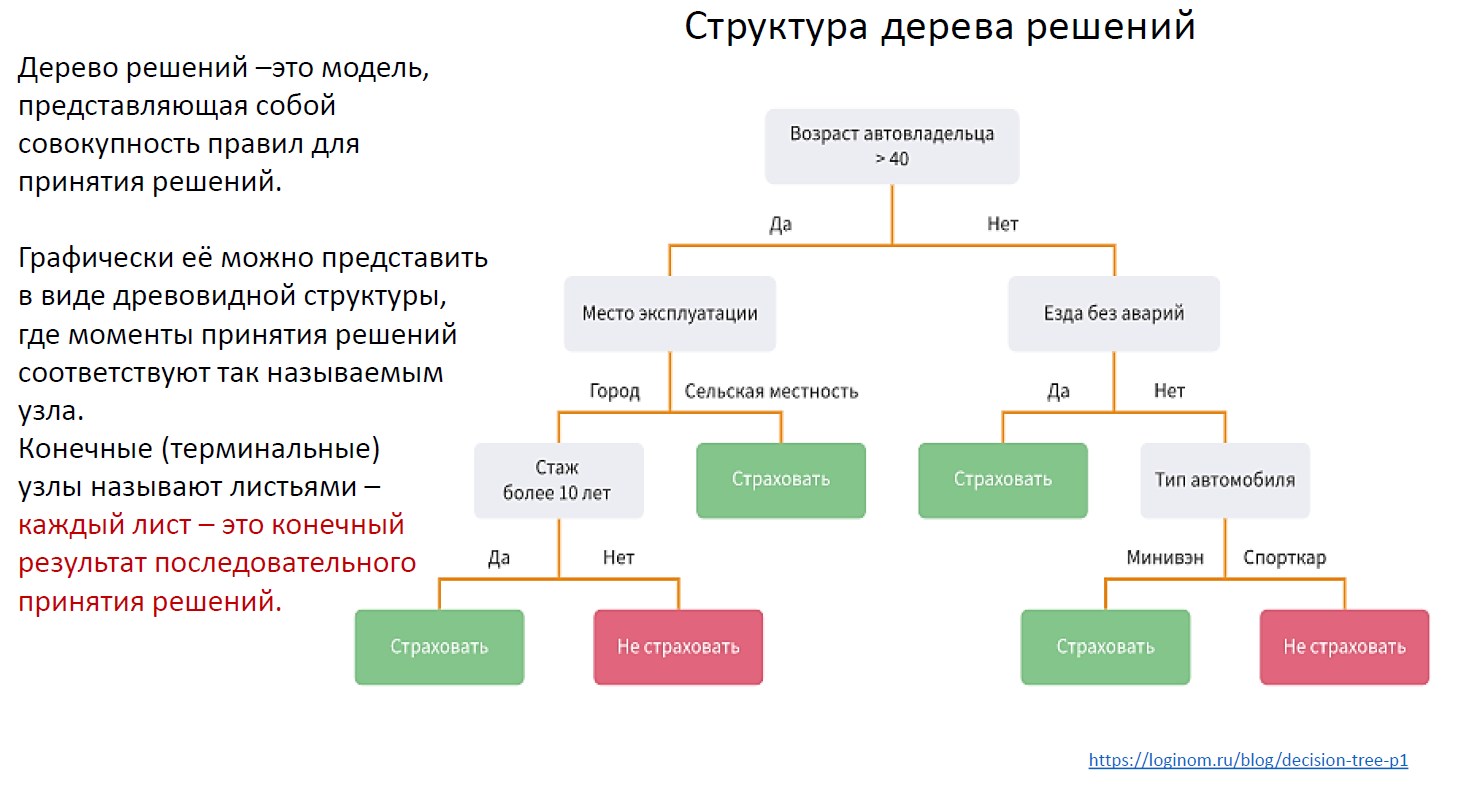
1. 

Здесь мы выбираем столбцы, по которым будем определять хорошее вино или плохое, в данном случае в х пойдут все столбцы кроме столбца качество

1. делим данные на две части: на которую будем обучать модель(train) и с которой будем сравнивать(valid). Параметр test\_size определяет соотношение сколько данных (в данном случае 70% ) будут использоваться для обучения модели и со скольки будем сравнивать (30%). Параметр рандом отвечает за случайность выборки (не важен )



1. Здесь улучшаем метод дерева решений, перебирая параметры. Потом выводим время вычислений, максимальное значение и параметры, при которых оно было достигнуто. В DT параметры -глубина дерева, в KNN количество соседей
2. Для сдачи нужно прочитать про алгоритмы DT(decision tree)



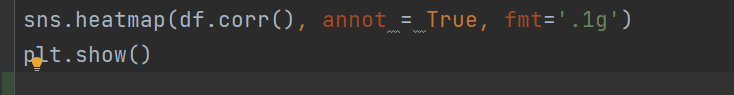
и KNN(k-nearest neighbors algorithm)

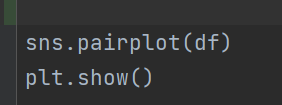
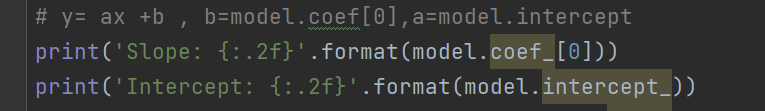


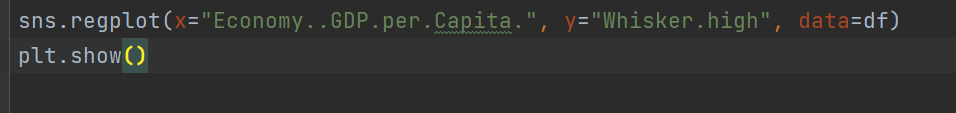
1. Конвертируем .dot в картинку https://onlineconvertfree.com/complete/dot-png/

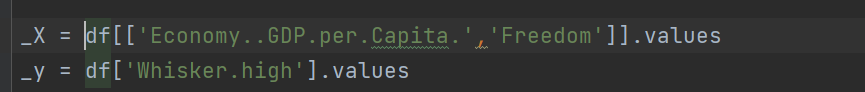
## Лабораторная работа 6

1. Прочитать ответы на вопросы в методе /task/lab6, можно и лекцию
2. Тепловая карта- по ней можно увидеть насколько связаны столбцы (их коэффициент корреляции), чем ближе к 1, тем сильнее связь. Брать стоит наиболее связанные



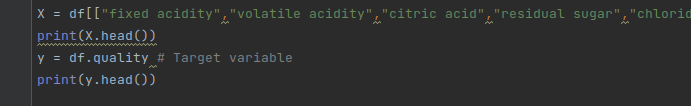
1. Строим диаграмму рассеивания значений столбцов 
2. По тепловой карте и диаграмме рассеивания выбираем столбцы имеющие большой коэффициент корреляции
3. Так как регрессия линейная, то она описывается уравнением 
4. Далее выводим полученную



1. Делаем тоже самое, что и ранее, но берём несколько параметров 

## Лабораторная работа 7

1) Логистическая регрессия — статистическая модель, используемая для прогнозирования вероятности возникновения некоторого события путём его сравнения с логистической кривой. Эта регреcсия выдаёт ответ в виде вероятности бинарного события (1 или 0). Для этого мы берём некоторые параметры, в переменную X их добавляем. На основе данных параметров будем определять хорошее вино или плохое(1 или 0), параметр quality.



2) Матрица сопряжённости



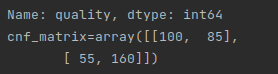
Строка 1 столбец 1- верно классифицированные положительные значения

Строка 1 столбец 2- неверно классифицированные положительные значения(если про вино, то это хорошее вино, классифицированное как плохое)

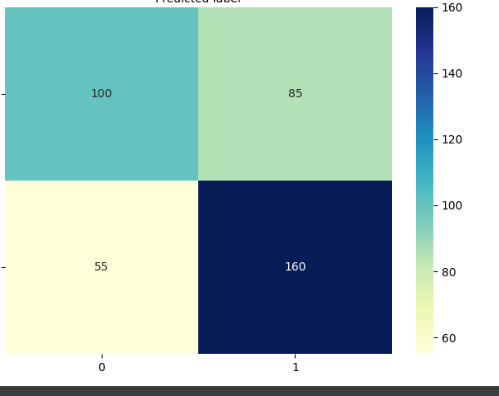
Строка 2 столбец 1- противоположно предыдущему абзацу(т.е. плохое вино классифицированное как хорошее)

Строка 2 столбец 2 - верно классифицированные отрицательные значения

В коде

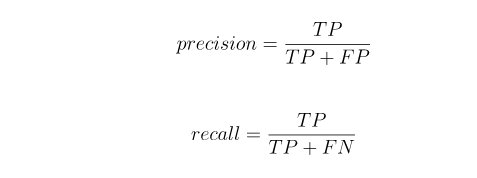


Графически



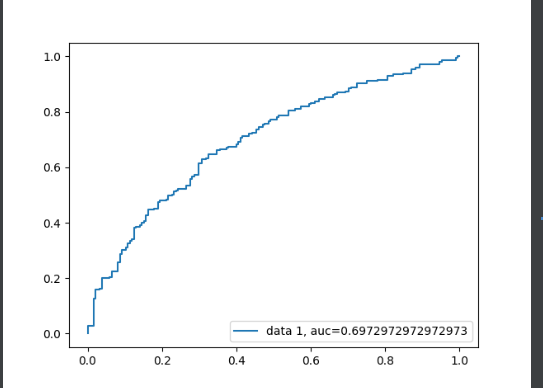
*accuracy- доля правильных ответов алгоритма:*

*Precision можно интерпретировать как долю объектов, названных классификатором положительными и при этом действительно являющимися положительными, а recall показывает, какую долю объектов положительного класса из всех объектов положительного класса нашел алгоритм.Т.е.*



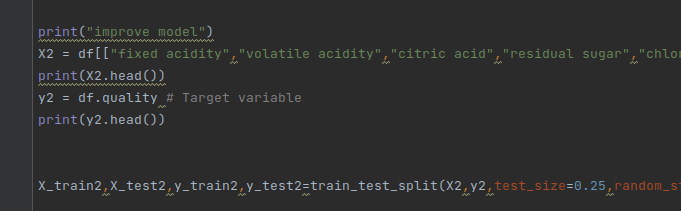
ROC-кривая показывает зависимость количества вернл классифицированных положительных примеров от количества неверно

классифицированных отрицательных примеров.



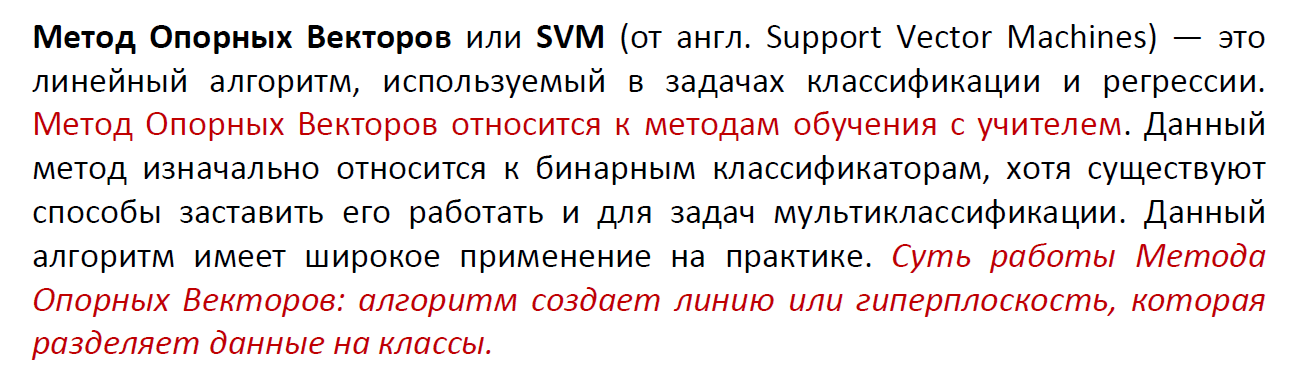
*Численный показатель площади под кривой называется AUC (Area Under Curve).С большими допущениями можно считать, что чем больше показатель AUC, тем лучшей прогностической силой обладает модель.*

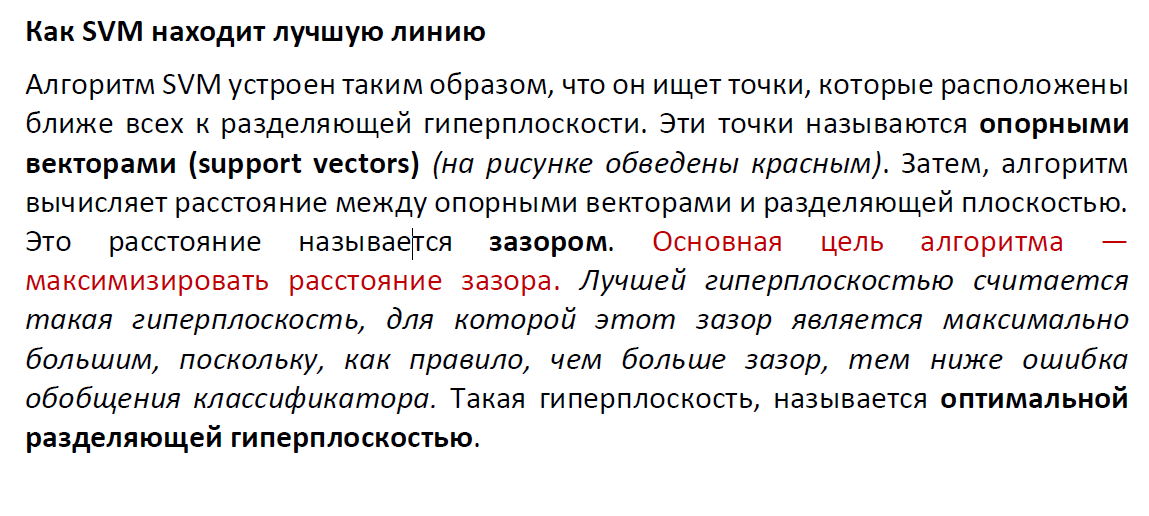
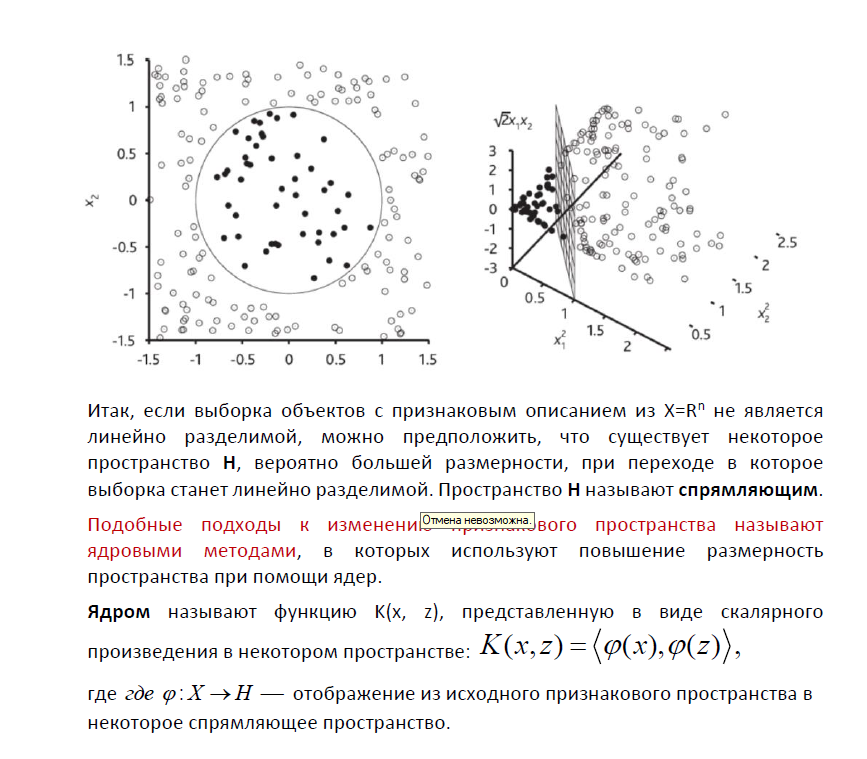
*3) Далее* делаем тоже самое, но в модели будет большие характеристик и она будет точнее.



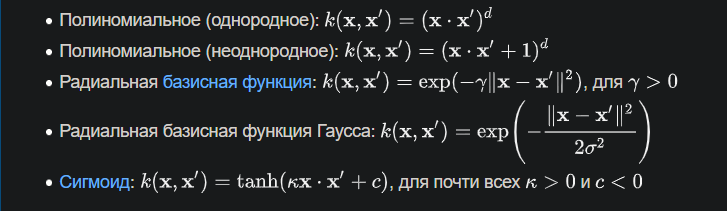
## Лабораторная работа 8

Нужно прочитать ответы на вопросы

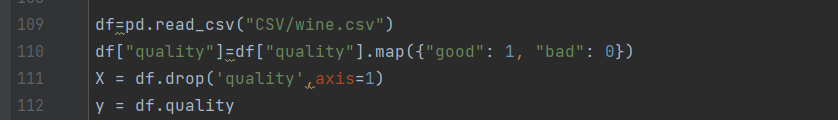
1. 
2. **Метод Опорных Векторов** или **SVM** (от англ. Support Vector Machines) — это линейный алгоритм, используемый в задачах классификации и регрессии.
3. 

Т.е. ядро используется в тех ситуациях, когда невозможно разделить выборку линией. Ядра друг от друга отличаются формулой функции



## Лабораторная работа 9

1. Лаба аналогичная предыдущим, но меняется алгоритм – теперь случайный лес
2. Лаба может очень долго выполнятся – возможно несколько минут
3. Прочитать ответы на вопросы в методе /task/lab9, можно и лекцию
4. Единственное, что нужно изменить (сама лаба начинается со 109 строчки, выше метод для обучения моделей, можно пропустить):



1. Выводится 2 картинки, в каждой по 6 графиков на 2 алгоритма в каждой, они подписаны сверху слева и справа
2. Теперь по графикам:

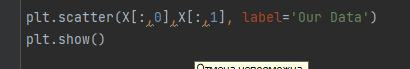
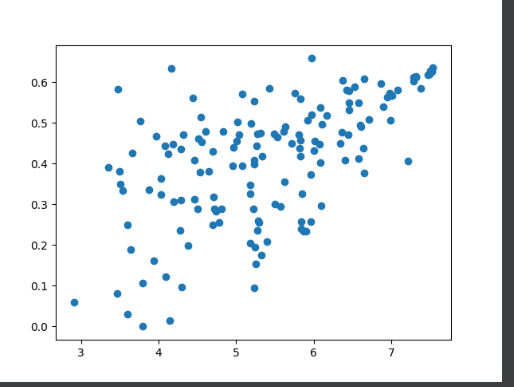
* На первой строке – здесь два набора данных – проверочные данные(красные) и обучаемые данные(зелёные). По оси Х количество тестовых данных (размер датасета), по У – качество. В моём случае в большинстве случаем наблюдается повышения качества модели с увеличением тестовых данных и сохранение значения качества с увеличением количеств проверочных данных.
* На второй строке графиков отображается изменения времени обучения от количества исходных данных. Очевидно, что будет увеличение
* На последней строке графиков отображается зависимость точности модели от затраченного времени на обучение

## Лабораторная работа 10

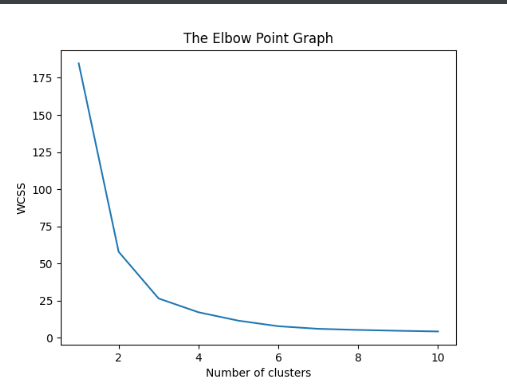
1. Надо выбрать два параметра для проведения “Исследования”, желательно , чтобы они имели какую-то связь



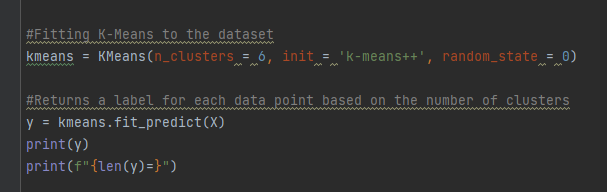
1. Здесь выводим график с исходными данными

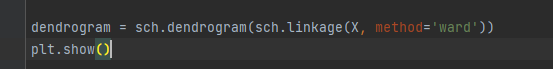
1. Для обоснования количества кластеров вычисляем wcss(WCSS - это сумма квадратов расстояний каждой точки данных во всех кластерах до соответствующих центроидов (начальные центры кластеров для кластеризации, будут видны на графике 5 пункта). Строим зависимость уменьшения wcss от количества кластеров, видно что после значения в 3 кластера убывание снижается незначительно, поэтом можно взять и его (я взял 6)



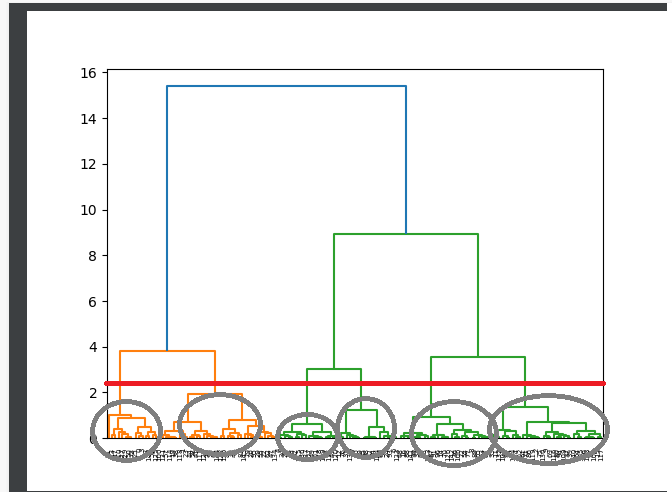
1. Здесь выполняем кластеризацию и выводим значения у, которые соответствуют номеру кластера (0-5) для каждого элемента из Х



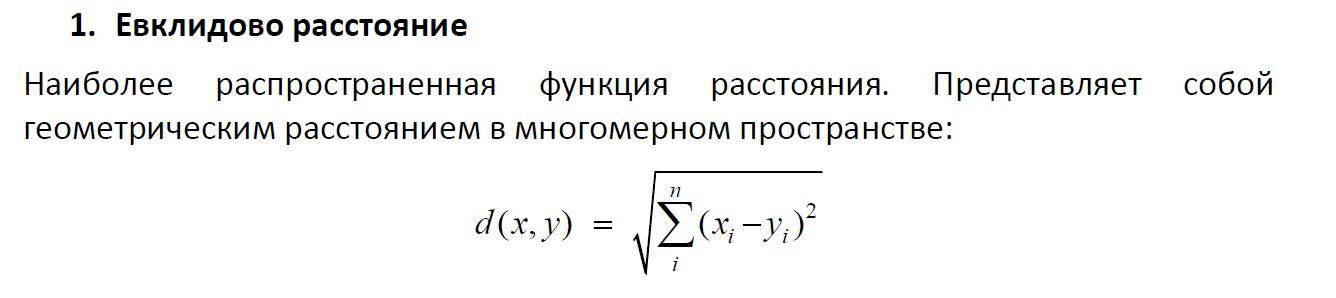
1. С 43 строки по 65 выполняем визуализацию данных, s- размер точки графике, marker – знак отображения (по умолчанию точка). Центроиды выделены ярко голубым цветом. Серый крестик Х -отображение значения одной страны(6 задание), 100 – номер в датасете
2. Строим дейдрограмму

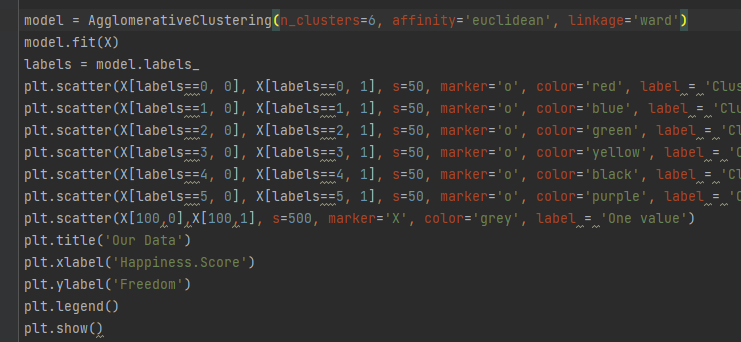


1. По ней определяем количество кластеров для разбиения (моё разбиение выделено красным): для выбора наиболее подходящего следует выделять в разные группы те поддеревья, расстояния между которыми достаточно велики



1. Выполняем кластеризацию и выводим график

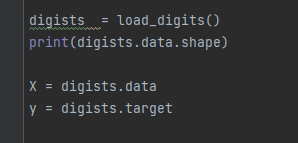


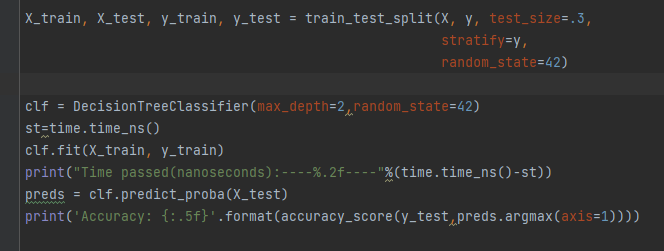


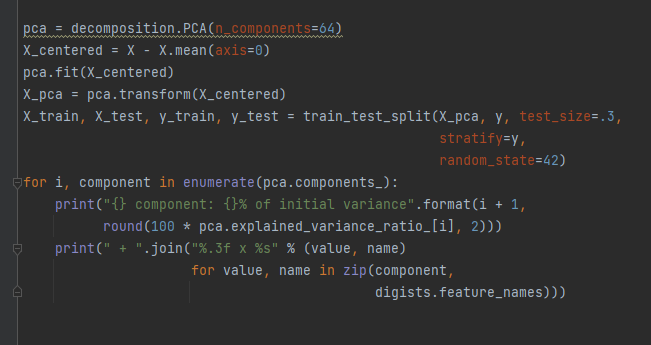
1. Прочитать ответы на вопросы в методе /Task/lab10, можно и лекцию

## Лабораторная работа 11

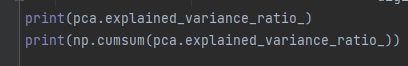
1. Ответы на лабу в доке с заданием. Если непонятен принцип работы алгоритма, то или ко мне или смотрите видео <https://www.youtube.com/watch?v=xZR6Zc8tKiw&t=121s>
2. Здесь скачиваем данные, нужно чтобы было более 30 параметров. Также выводим размерность(цифра 64 понадобится для определения уравнений главных компонент и их влияния на точность).



1. Обучаем модель по методу дерева решений, выводим точность и время обучения в нс 
2. Здесь будем находить 2 главные компоненты и выведем их уравнение



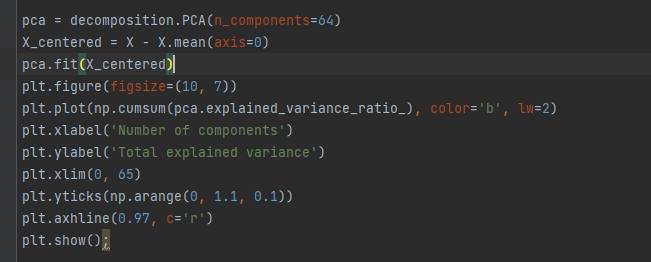
1. Выведем уравнения главных компонент, их процент отклонения, объясняемый каждой из компонент (pca.explained\_variance\_ratio\_), сумму процентов отклонения

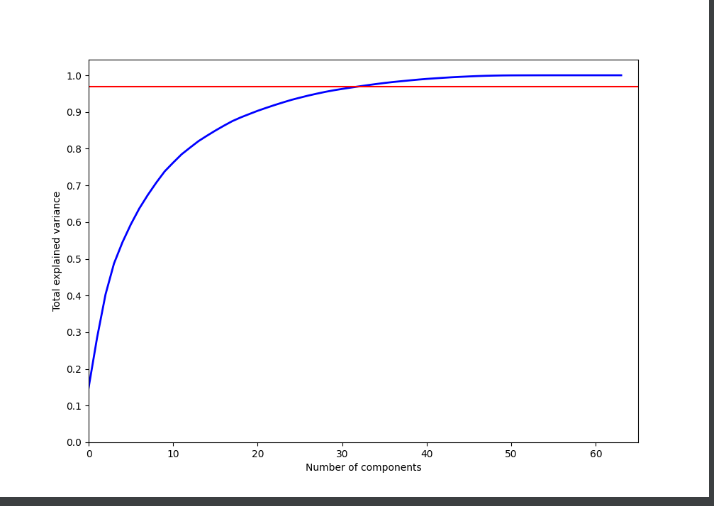


1. Выполняем обучение по методу дерева решений, но уже после выбора двух главных компонент. Точность возрастёт



1. Строим график для определения наиболее подходящего количества компонент, по нему можно выбрать наиболее подходящее количество компонент, я выбрал 30 (plt.axhline – рисует линию, по значению y)

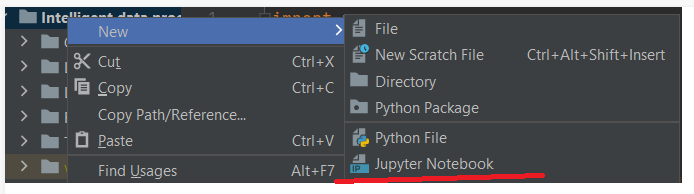


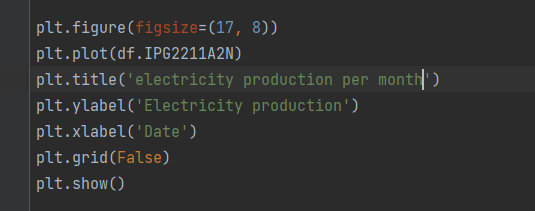


1. Выполняем обучение по методу дерева решений, но уже после выбора двух главных компонент. Точность возрастёт



## Лабораторная работа 12

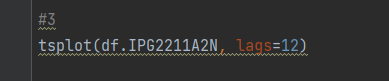
1. В качестве альтернативы можно выбрать датасет с производством пива, скачан в /CSV. Также нужно установить jupyter notebook, можно просто создать файл jupyter notebook, pycharm сам предложит установить всё необходимое .До очистки датасета выберите диапазон для обучения около 200 (1-200) и добавьте 6 следующих строк(201-206) в Actual.csv, они понадобятся для прогноза в дальнейшем (не забудьте про названия столбцов). Удалите лишние строки из датасета.
2. Основные определения из лабы:
3. Временным рядом называется последовательность значений признака y, измеряемого через постоянные временные интервалы:
4. Тренд — плавное долгосрочное изменение уровня ряда. Эту характеристику можно получить, наблюдая ряд в течение достаточно долгого времени
5. Cкользя́щее сре́днее — общее название для семейства функций, значения которых в каждой точке определения равны некоторому среднему значению исходной функции за предыдущий период .Скользящие средние обычно используются с данными временных рядов для сглаживания краткосрочных колебаний и выделения основных тенденций или циклов
6. Временной ряд называется стационарным, если его статистические свойства не изменяются со временем.
7. Гипотезу о стационарности можно проверить с помощью критерия Дики-Фуллера.Не вдаваясь в технические детали теста Дики-Фуллера, он проверяет нулевую гипотезу о наличии единичного корня. Если да, то p> 0 и процесс не стационарный. В противном случае p = 0, нулевая гипотеза отклоняется, и процесс считается стационарным.
8. Автокорреляция — статистическая взаимосвязь между последовательностями величин одного ряда, взятыми со сдвигом, например, для случайного процесса — со сдвигом по времени.
9. График выведет данные из датасета (производство электроэнергии за месяц)



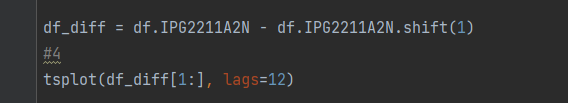
1. Во втором графике показаны данные из датасета после проведения сглаживания ряда моделью скользящего среднего, тут также показана зелёным линия тренда и максимальные(минимальные) значения за период времени в 12 месяцев (параметр)



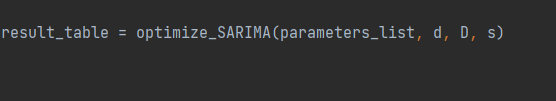
1. На третьем графике показана автокорреляция, значение критерия Дики-Фуллера, частичная автокорреляция



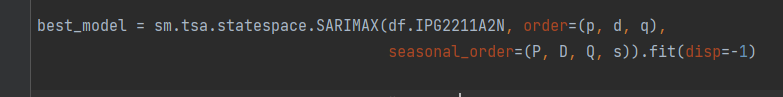
1. Согласно тесту Дики-Фуллера временной ряд, что неудивительно, нестационарен. Также, глядя на график автокорреляции, мы видим, что он очень высокий, и кажется, что нет явной сезонности.Поэтому, чтобы избавиться от высокой автокорреляции и сделать процесс стационарным, возьмем первую разность (если первые разности ряда стационарны, то он называется интегрированным рядом первого порядка). Мы просто вычитаем временной ряд из самого себя с задержкой в один день, и получаем:
2. Вычитаем и строим тот же график, но с новыми значениями



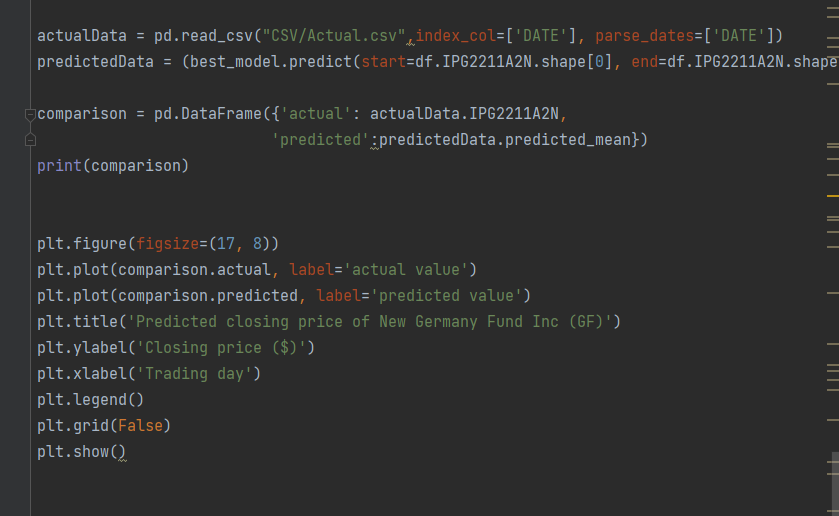
1. Так как модель теперь стационарна, то можно применить алгоритм SARIMA(образована от ARIMA, она является обобщением Авторегрессионной модели скользящей средней (ARMA) Если взять авторегрессионную модель порядка p (AR(p)) и модель скользящего среднего порядка q (MA(q)) и сложить то, что находится у них в правых частях, то в результате получим это модель ARMA(p, q), она выглядит следующим образом).
2. Подбираем лучшие характеристики для модели (в методе)



1. Обучаем и выводим инфу (3 print), MAPE - как средняя абсолютная ошибка в процентах

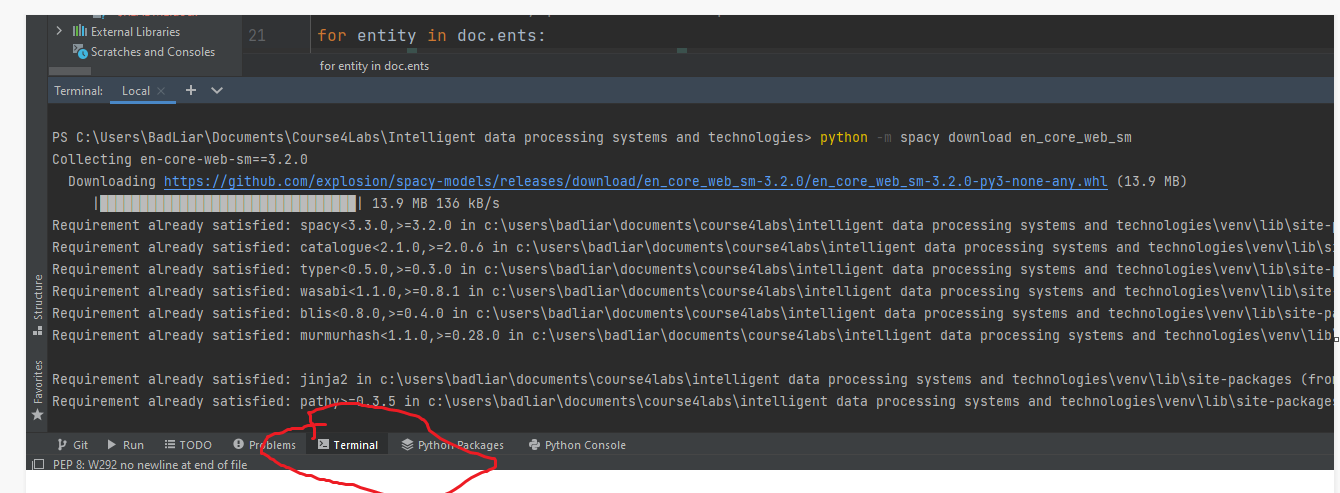


1. Здесь считываем данные из имеющихся значений для прогноза, сравниваем их на графике

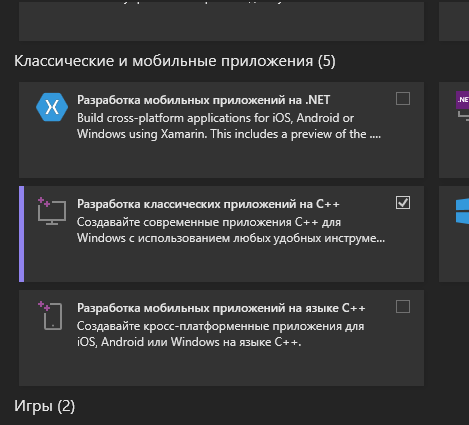


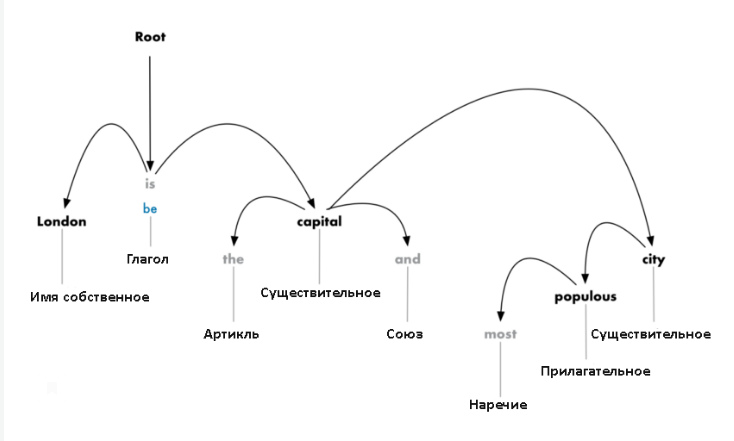
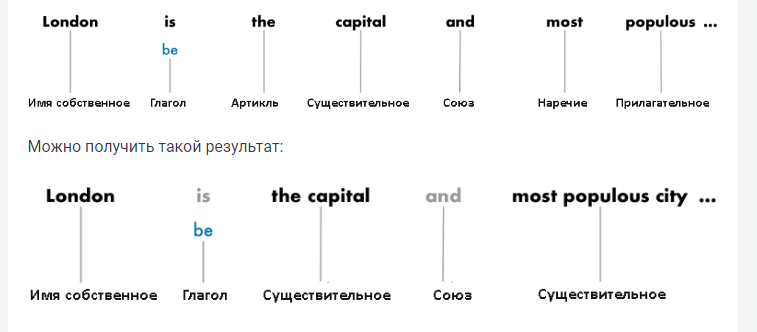
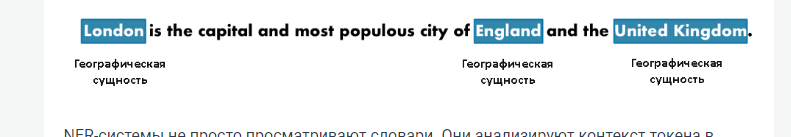
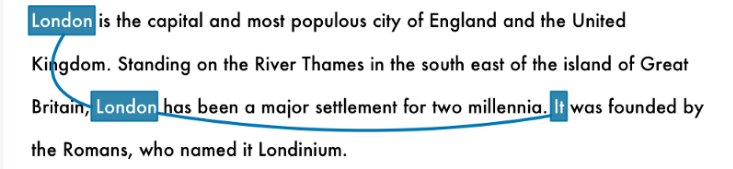
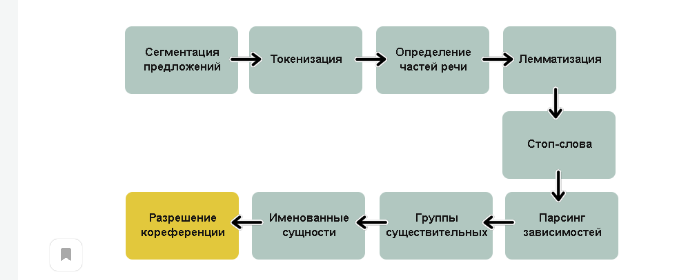
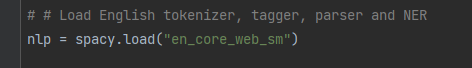
## Лабораторная работа 13

1. В терминале Pycharm(на скрине путь) вводим команду python -m spacy download en\_core\_web\_sm и python -m spacy download ru\_core\_news\_sm

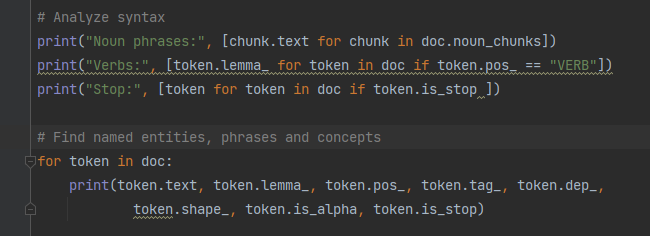
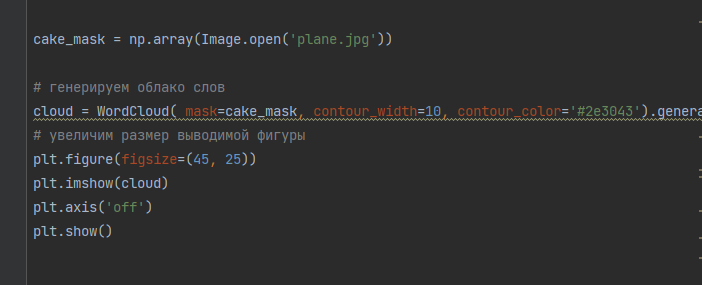
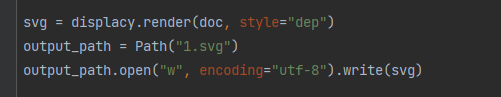
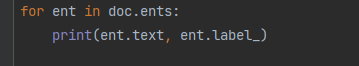


1. Для библиотеки wordcloud нужны плюсы 14+ версии, качаем отсюда(не исключено, что есть другие варианты установки)



1. Алгоритм NLP(Natural language processing ( есть ещё аббревиатура - Распознавание именованных сущностей ( NER)) работает следующим образом :
2. Шаг 1. Выделение предложений
3. Токенизация, или выделение слов
4. Определение частей речи
5. Лемматизация процесс приведения словоформы к лемме — её нормальной (словарной) форме. Например - предложение «I had two ponies» превращается в «I [have] two [pony]».
6. Определение стоп-слов (Например – a, the )
7. Парсинг зависимостей. Теперь необходимо установить взаимосвязь между словами в предложении. Это называется парсингом зависимостей. Конечная цель этого шага – построение дерева, в котором каждый токен имеет единственного родителя. Корнем может быть главный глагол. (определяем подлежащее и сказуемое и т.д.)
8. Поиск групп существительных. Сейчас мы рассматриваем каждое слово в нашем предложении как отдельную сущность. Но иногда имеет смысл сгруппировать токены, которые относятся к одной и той же идее или вещи. Мы можем использовать полученное дерево парсинга, чтобы автоматически объединить такие слова. 
9. Распознавание именованных сущностей (Named Entity Recognition, NER) 
10. Разрешение кореференции. В английском очень много местоимений – слов вроде he, she, it. Это сокращения, которыми мы заменяем на письме настоящие имена и названия. Человек может проследить взаимосвязь этих слов от предложения к предложению, основываясь на контексте. Но NLP-модель не знает о том, что означают местоимения, ведь она рассматривает всего одно предложение за раз. Разрешением кореференции называется отслеживание местоимений в предложениях с целью выбрать все слова, относящиеся к одной сущности. 
11. Итого : 
12. Теперь по коду :
13. Загрузка NLP модели 
14. Парсинг текста с помощью spaCy



1. Выводим различную инфу о тексте (вся инфа в doc) 
2. Строим облако слов, для английского текста удалять стоп-слова не надо, библиотека их удалит сама 
3. Строим дерево зависимостей, будет лежат в директории проекта 
4. Выводим слова, являющиеся именованными сущностями 
5. Всё тоже самое и для русского языка, меняется NER, также проводится удаление стоп-слов. Больше изменений нет