**Для выполнения заданий рекомендованы (но не обязательны) Python 3.6 + Anaconda.**

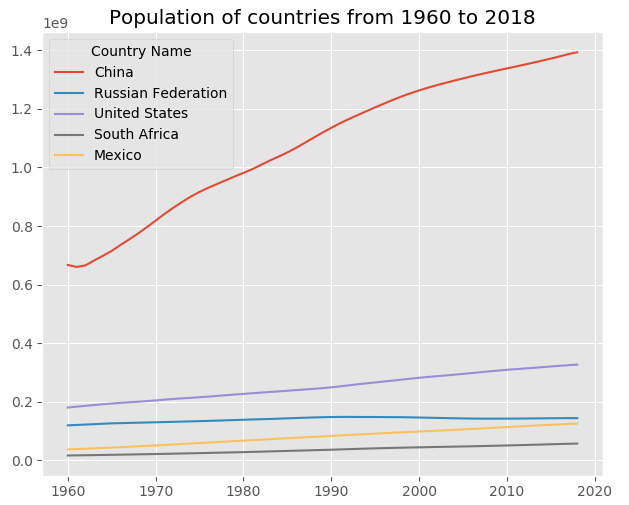
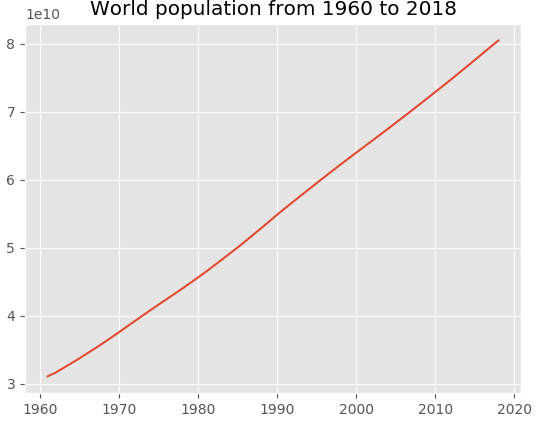
**Материалы для подготовки:**

1) [Основы машинного обучения](https://coderlessons.com/tutorials/python-technologies/uznaite-mashinnoe-obuchenie-s-python/mashinnoe-obuchenie-s-python-kratkoe-rukovodstvo) (основы Python и настройка окружения, задачи классификации, регрессии и кластеризации)

Задание № 1. Обработка csv-файлов

1. Скачать файл \*.csv по ссылке согласно варианту.
2. Установить библиотеки pandas и matplotlib.
3. С помощью python обработать данные из файла и нарисовать два графика: тенденция по годам по всем странам и тенденция по топ-5 странам по годам.

Пример результата работы программы:



Вариант 1. [Доступ к электричеству](https://data.worldbank.org/indicator/EG.ELC.ACCS.ZS?view=chart)

Вариант 2. [Выбросы углекислого газа](https://data.worldbank.org/indicator/EN.ATM.CO2E.PC?end=2018&start=1960&view=chart)

Вариант 3. [Площадь лесов](https://data.worldbank.org/indicator/AG.LND.FRST.K2?view=chart)

Вариант 4. [ВВП](https://data.worldbank.org/indicator/NY.GDP.MKTP.CD?view=chart)

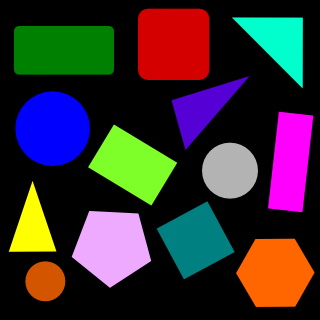
Вариант 5. [Уровень инфляции](https://data.worldbank.org/indicator/FP.CPI.TOTL.ZG?view=chart)

Вариант 6. [Уровень безработицы](https://data.worldbank.org/indicator/SL.UEM.TOTL.ZS)

Задание №2. Распознавание контура фигур

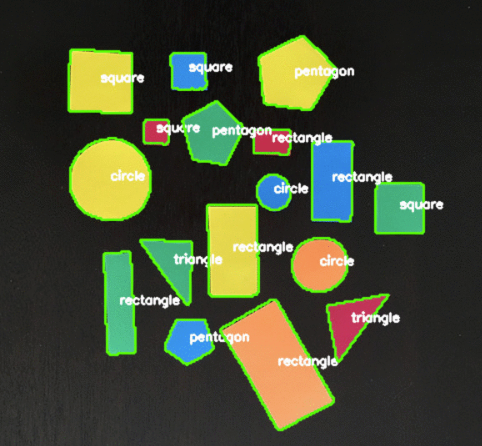
1. Установить библиотеку opencv.
2. Написать скрипт генерации N картинок со случайными геометрическими фигурами случайных цветов на черном фоне (N задается с клавиатуры). Скрипт должен быть оформлен в отдельный .py файл. На каждом изображении должны быть круг, квадрат, прямоугольник, треугольник, пятиугольник. (Продвинутый вариант – добавить закругленные углы).

Пример исходного изображения:



1. С помощью линейных фильтров на исходных сгенерированных изображений найти фигуры, выделить их контуры и подписать.

Пример результата работы программы:



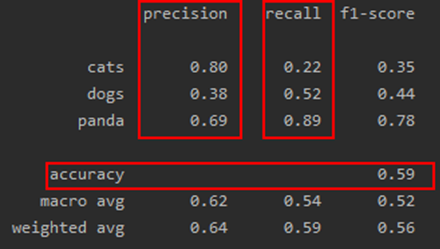
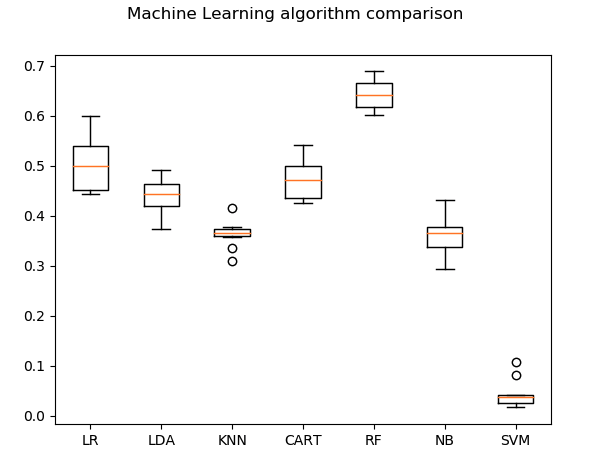
Задание №3. Классификация покемонов.

[Статья](https://gogul.dev/software/image-classification-python) для вдохновения.

1. Установить библиотеки sklearn, h5py, mahotas
2. Скачать [датасет](https://www.kaggle.com/thedagger/pokemon-generation-one) с покемонами по варианту (для каждого покемона отдельная директория).
3. Выбрать фичи («признаки»), для сравнения изображений (цвет, размер, форма, наличие хвоста, бровей и т.д).

([статья](https://intalent.pro/article/pro-kotikov-sobak-mashinnoe-obuchenie-i-deep-learning.html) о фичах).

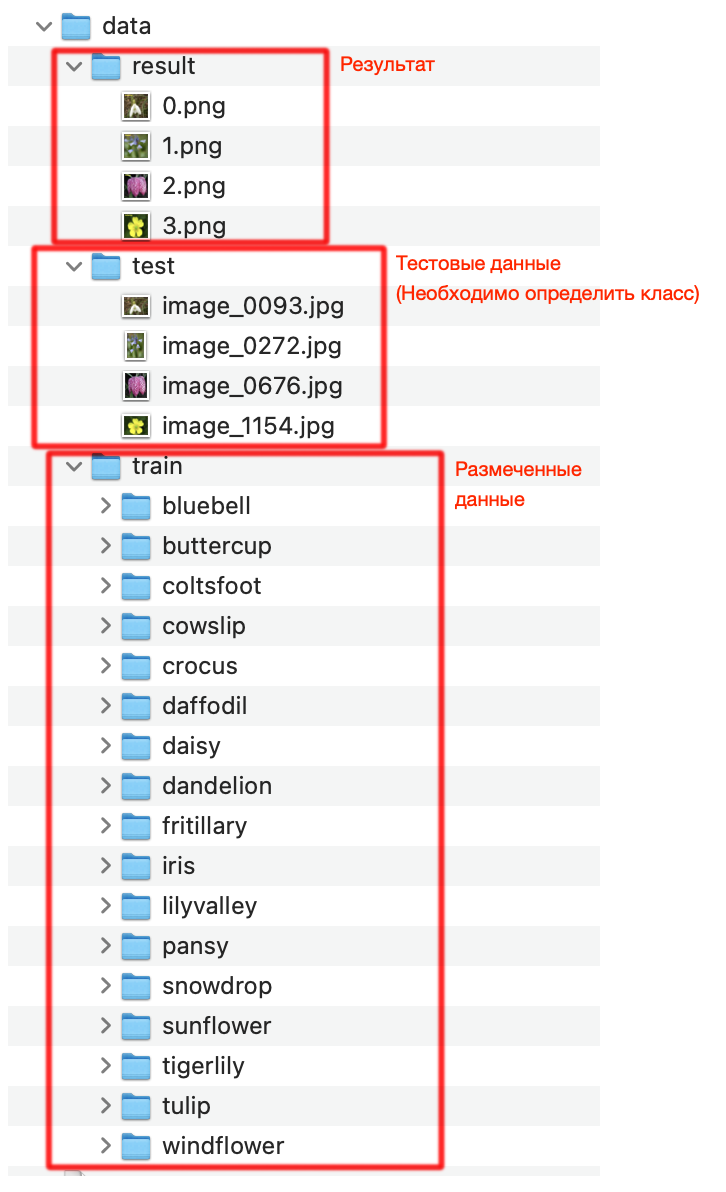
1. Подобрать фильтры для каждой фитчи.
2. Извлечь фичи из каждой картинки обучающей выборки и записать в отдельный датасет.
3. Разделить датасет на обучающую и тестовую выборку.
4. Построить модели GaussianNB, Logistic Regression, Decision Tree,  SVM, Random Forest, обучить, оценить полноту, точность и аккуратность и визуализировать с помощью «ящика с усами».



1. Выбрать несколько картинок покемонов (не входящих в обучающую и тестовую выборки), определить их класс с помощью лучшего классификатора и подписать на картинках.

Пример результата работы программы:



Рекомендованная организация данных:

Вариант 1: Bulbasaur, Ivysaur, Venusaur, Charmander, Charmeleon, Charizard

Вариант 2: Squirtle, Wartortle, Blastoise, Pidgey, Pidgeotto, Pidgeot

Вариант 3: Pikachu, Raichu, Drowzee,Clefable, Clefairy, Ditto

Вариант 4: Vulpix, Ninetales, Oddish, Gloom, Bellsprout, Vileplume

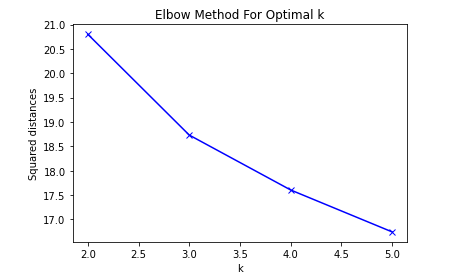
Вариант 5: Machop, Machoke, Machamp, Arcanine, Growlithe, Golem

Вариант 6: Slowpoke, Slowbro, Muk, Grimer, Lickitung, Chansey

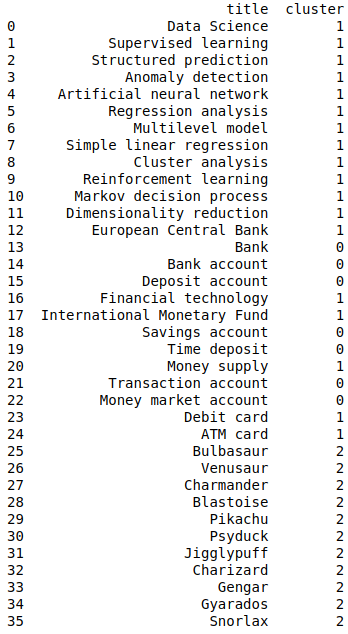
Задание №4. Кластеризация статей Wikipedia

[Статья](https://towardsdatascience.com/clustering-documents-with-python-97314ad6a78d) для вдохновения.

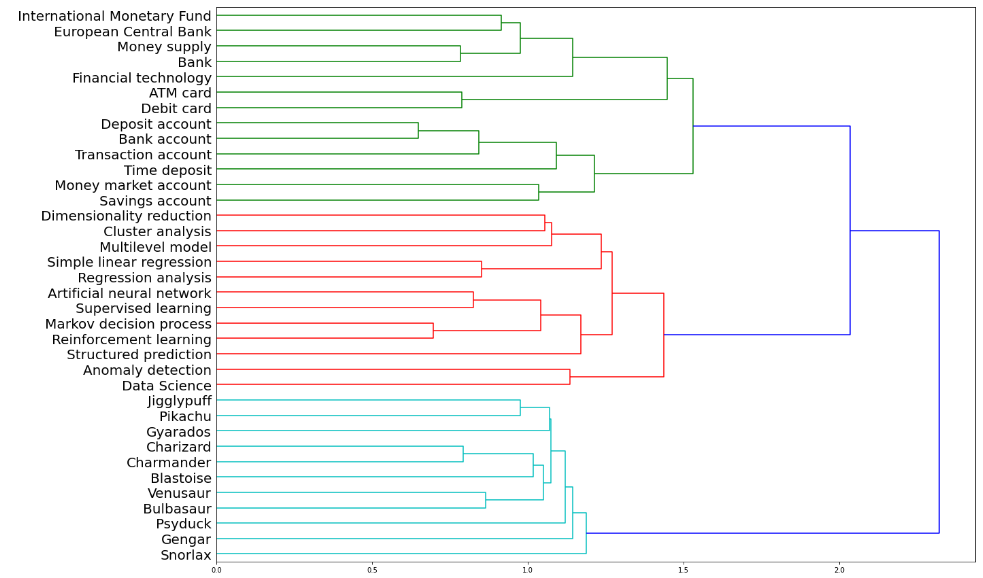
1. Установить библиотеки wikipedia, scipy.
2. Найти список статей англоязычной Википедии по темам согласно варианту (минимум 15 статей по каждой теме) и записать в отдельный массив.
3. Получить тест статей (документ) и записать в отдельный массив.
4. Извлечь фичи из текстов на английском языке с помощью [TF-IDF](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html?highlight=tfidfvectorizer#sklearn.feature_extraction.text.TfidfVectorizer) (признаки).
5. Определить число кластеров с помощь [elbow метода](https://en.wikipedia.org/wiki/Elbow_method_(clustering)) и нарисовать график.



1. Разбить массив признаков на кластеры с помощью алгоритмов [KMeans](https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html), [MiniBatchKMeans](https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html?highlight=minibatchkmeans#sklearn.cluster.MiniBatchKMeans), [DBSCAN](https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html?highlight=dbscan#sklearn.cluster.DBSCAN) и вывести результат работы алгоритмов в консоль.



1. Разбить массив признаков на кластеры с помощью [иерархической кластеризации](https://ru.wikipedia.org/wiki/%D0%98%D0%B5%D1%80%D0%B0%D1%80%D1%85%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BA%D0%BB%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) (из библиотеки [scipy](https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html)) и представить результат в виде дендрограммы.



Список статей по темам можно найти [здесь](https://en.wikipedia.org/wiki/Wikipedia:Vital_articles).

Вариант 1: Ancient history, English literature, Diseases

### Вариант 2: Biochemistry, Cryptography, Modern history

### Вариант 3: Mathematicians, History of Asia, Media

Вариант 4: Mountains, Cooking, Government

Вариант 5: Religious figures, Arctic Ocean, Drugs

Вариант 6: Buddhism, Botany, Transport

Задание №5. Снижение размерности и визуализация

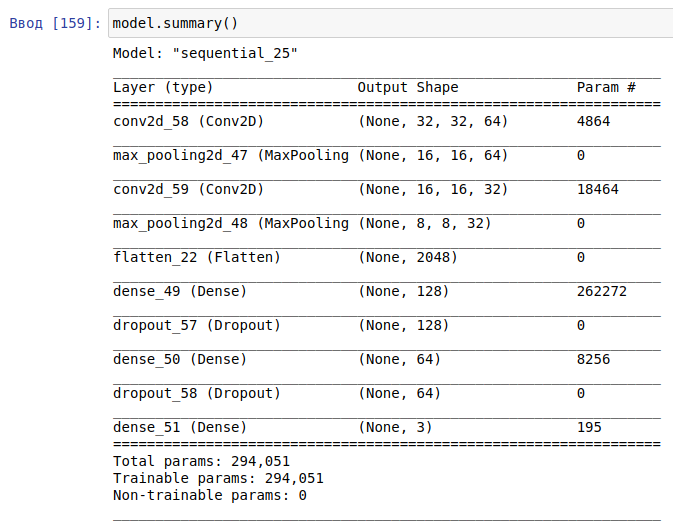
1. На основе признаков из заданий №3 и №4 построить [PCA](https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html) и [TSNE](https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html) распределения в двумерном пространстве.
2. Визуализировать результат с помощью scatter графика.

|  |  |
| --- | --- |
| Плохо: | Хорошо:https://miro.medium.com/max/595/1*NX1uR7kPf5DKJHoAw3sUEQ.png |

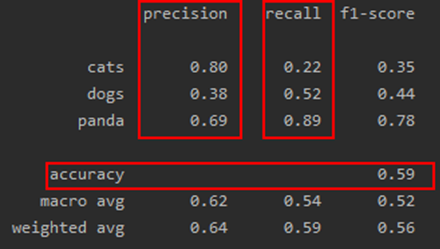
Задание №6. Классификаторы на основе нейронных сетей прямого распространения

Материалы для вдохновения ([FFNN](https://www.reg.ru/blog/keras/), [CNN](https://www.kaggle.com/kanncaa1/convolutional-neural-network-cnn-tutorial)).

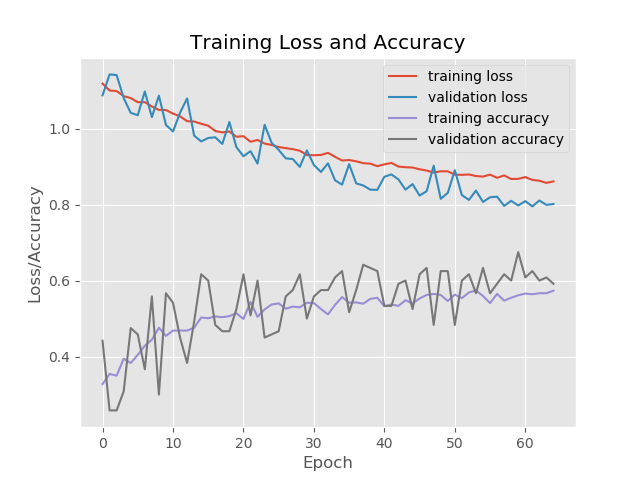
1. Установить библиотеку keras, tensorflow.
2. Скачать [датасет](https://www.kaggle.com/alexattia/the-simpsons-characters-dataset) с симпсонами варианту (для каждого класса отдельная директория).
3. Создать архитектуру нейронной сети прямого распространения с двумя полносвязными скрытыми слоями и сверточной нейронной сети с двумя сверточными и двумя полносвязными слоями и скомпилировать модели (model.summary()).



1. Обучить скомпилированные модели, оценить полноту, точность и аккуратность с помощью classification report.



1. Построить график потерь и точности для каждой модели.



1. Выбрать несколько картинок (не входящих в обучающую и тестовую выборки), определить их класс и подписать на картинках.



Вариант 1: Barney Gumble, Apu Nahasapeemapetilon, Kent Brockman

### Вариант 2: Lisa Simpson, Lenny Leonard, Milhouse Van Houten

### Вариант 3: Ned Flanders, Maggie Simpson, Martin Prince

Вариант 4: Moe Szyslak, Mayor Quimby, Sideshow Bob

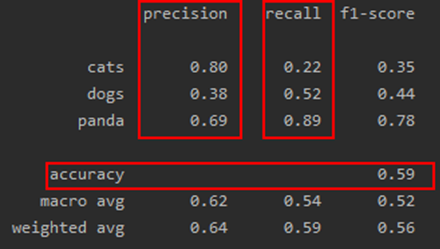
Вариант 5: Marge Simpson, Nelson Muntz, Groundskeeper Willie

Вариант 6: Rainier Wolfcastle, Selma Bouvier, Krusty the Clown

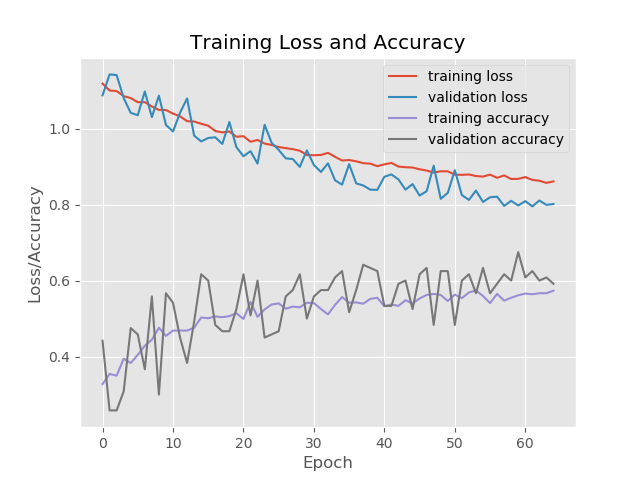
Задание №7. Анализ тональности текстов (RNN).

Материалы для вдохновения ([ноутбук 1](https://www.kaggle.com/ngyptr/lstm-sentiment-analysis-keras), [ноутбук 2](https://www.kaggle.com/paoloripamonti/twitter-sentiment-analysis), [статья 1](https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/), [статья 2](https://medium.com/district-data-labs/modern-methods-for-sentiment-analysis-694eaf725244)).

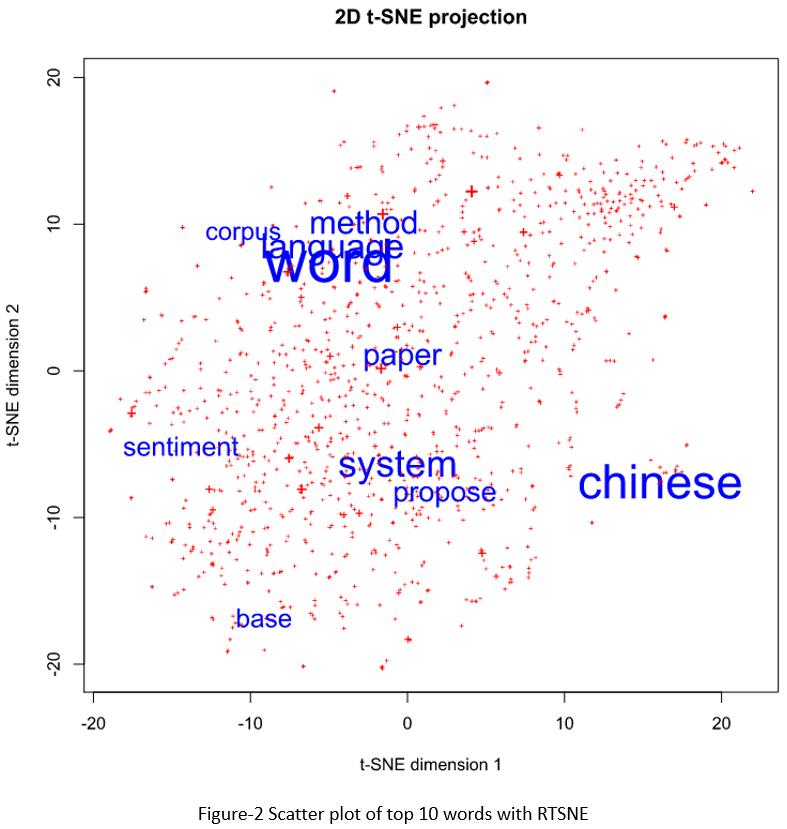
1. Установить библиотеку nltk, gensim.
2. Скачать [датасет](https://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences#) c размеченным текстом (позитивный / негативный).
3. Обработать тексты стеммеров, удалить частые слова (stopwords).
4. Скачать предобученную модель [Word2Vec](https://radimrehurek.com/gensim/auto_examples/tutorials/run_word2vec.html) (или самому обучить), с ее помощь получить векторное представление слов.
5. Создать архитектуру нейронной сети с Embedding слоем, рекуррентным слоем (LSTM) и полносвязным слоем, скомпилировать модель (model.summary()).
6. Обучить модель, оценить полноту, точность и аккуратность с помощью classification report.



1. Построить график потерь и точности для каждой модели.



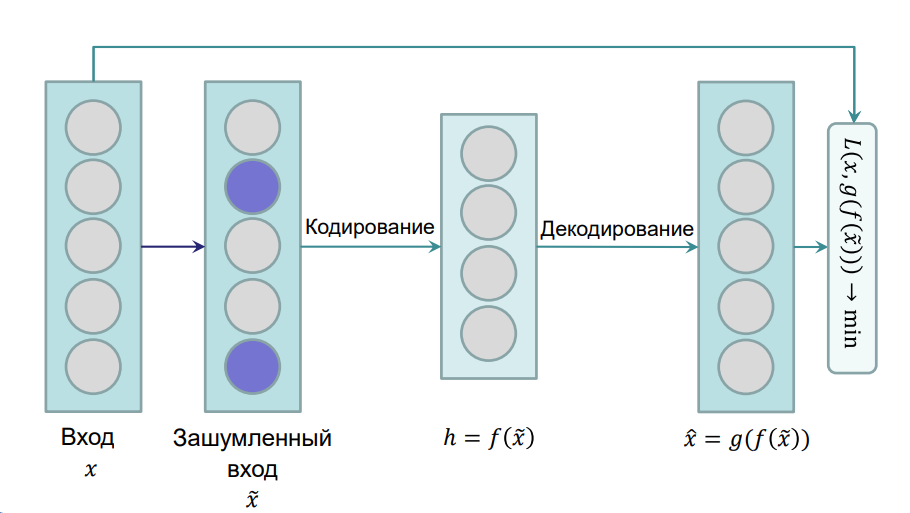
1. Выбрать несколько предложений, определить их класс (позитивный или негативный) и визуализировать с помощью TSNE модели.



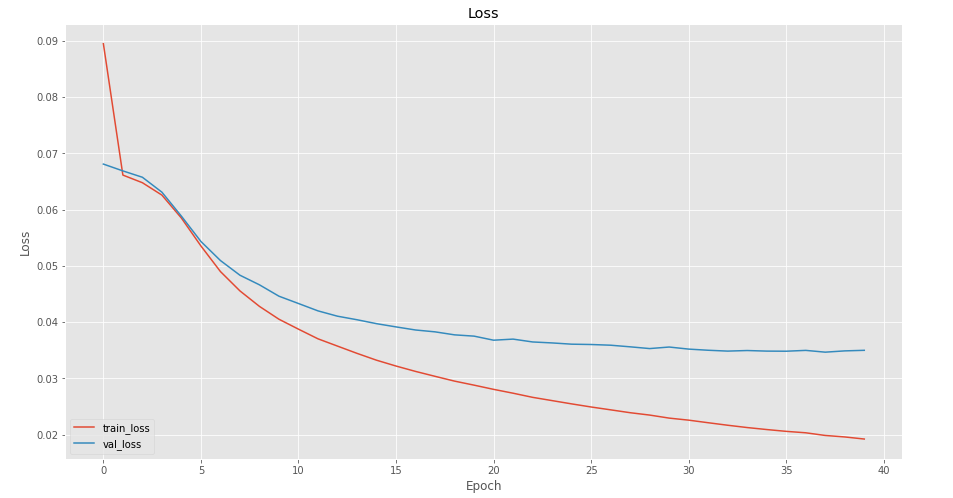
Задание №8. Автокодировщики

[Материал](https://habr.com/ru/post/331382/) для вдохновения.

1. Скачать [датасет](https://www.kaggle.com/jakubfrydrych/digits-segment-display) c изображениями цифр.
2. Сгенерировать несколько изображений с шумом.
3. Создать модель шумоподавляющего (denoising) автокодировщика с двумя полносвязными слоями, скомпилировать модель (model.summary()).



1. Обучить модель, построить график потерь.



1. Восстановить несколько изображений с шумом с помощью модели обученной нейронной сети и сохранить*.*