МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра «Систем обработки информации и управления»

ОТЧЕТ

**Домашнее задание №___1__**
по дисциплине «Методы машинного обучения»

ИСПОЛНИТЕЛЬ: Шапиев М.М.
ФИО

группа ИУ5-24М
подпись

"__"_____2020 г.

ПРЕПОДАВАТЕЛЬ: Гапанюк Ю.Е.
ФИО

Подпись

"__"_____2020 г.

Москва - 2020

## *Задание*

- Исследование предметной области, определение функциональных задач;

- Выбор и анализ алгоритмов анализа тональности текстовой информации;

- Разработка структуры программного продукта;

- Сбор, очистка и предобработка обучающей выборки;

- Программная реализация алгоритма наиболее точного алгоритма классификации;

- Обеспечение модульности системы для возможности подключения к другой системе;

- Тестирование разработанной системы с целью выявления ошибок при работе, а также с целью определения точности распознавания;

## *Реализация задания*

### Выбор набора данных

В качестве набора данных используются русскоязычный корпус коротких текстов RuTweetCorp состоящий из 17,639,674 записей. Для тех, кто занимается задачей тонового анализа предлагаю ознакомиться с корпусами, автоматически распределенными на две группы: «заведомо положительные» (114,911 записей) и «заведомо отрицательные» (111,923 записей). Корпус собран на основе русскоязычных постов микроблогинговой площадки Twitter.

### Текстовое описание набора данных

Несмотря на ограничения API twitter, был собран корпус русскоязычных twitter-постов, автоматически размеченных на два класса (положительные и отрицательные). Корпус нейтральных постов собирается отдельно. Каждый текст в корпусе имеет следующие атрибуты: – дата публикации; – имя автора; – текст твита; – класс, к которому принадлежит текст (положительный, отрицательный, нейтральный); – количество добавлений сообщения в избранное; – количество ретвитов (количество копирований этого сообщения другими пользователями); – количество друзей пользователя; – количество пользователей, у которых данный юзер в друзьях (количество фоловеров); – количество листов, в которых состоит пользователь.

### Постановка задачи

Основной целью анализа тональности является нахождение мнений в тексте и выявление их свойств. Какие именно свойства будут исследоваться, зависит уже от поставленной задачи. К примеру, целью анализа может быть автор, то есть лицо, которому принадлежит мнение.

Мнения делятся на два типа:

1. непосредственное мнение;
2. сравнение.

Непосредственное мнение содержит высказывание автора об одном объекте. Формальное определение непосредственного мнения выглядит так: "непосредственным мнением называется кортеж из пяти элементов *(e, f, op, h, t)*, где:

1. (*entity*, *feature*) — объект тональности e (сущность, насчет которой высказывается автор) или его свойства f (атрибуты, части объекта);
2. *orientation* или *polarity* — тональная оценка (эмоциональная позиция автора относительно упомянутой темы);
3. *holder* — субъект тональности (автор, то есть кому принадлежит это мнение);
4. момент времени *time*, когда было оставлено мнение.

Примеры тональных оценок:

- позитивная;
- негативная;
- нейтральная.

Под «нейтральной» подразумевается, что текст не содержит эмоциональной окраски. Также могут существовать и другие тональные оценки.

**Предварительная обработка набора данных**

Очистка и предобработка данных

Процедура очистки и предобработки обучающей выборки выглядит следующим образом:

- приведение к нижнему регистру;

- замена «ё» на «е»;

- замена ссылок на токен «URL»;

- замена упоминания пользователя на токен «USER»;

- удаление знаков пунктуации.

Векторное отображение слов

Входными данными сверточной нейронной сети является матрица с фиксированной высотой n, где каждая строка представляет собой векторное отображение слова в признаковое пространство размерности k. Для формирования embedding-слоя нейронной сети я использовал утилиту дистрибутивной семантики Word2Vec, предназначенную для отображения семантического значения слов в векторное пространство. Word2Vec находит взаимосвязи между словами согласно предположению, что в похожих контекстах встречаются семантически близкие слова.

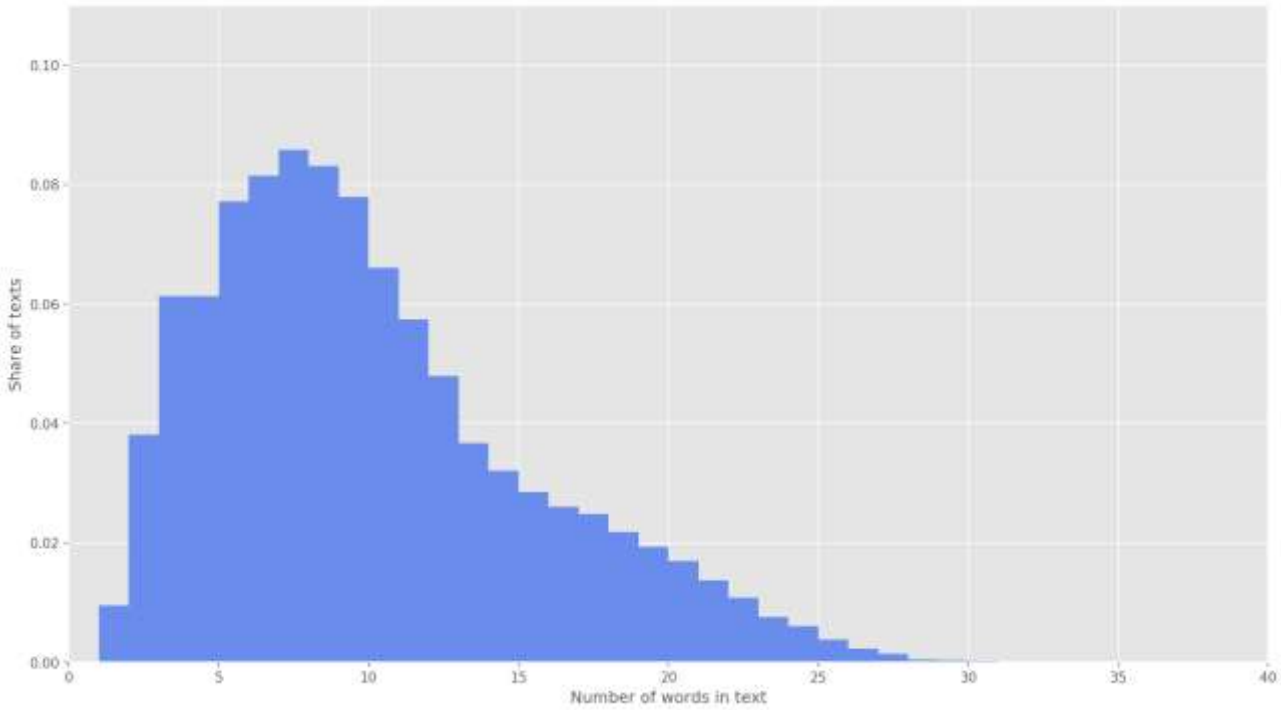
Далее с помощью библиотеки Gensim обучил Word2Vec-модель со следующими параметрами:

size = 200 — размерность признакового пространства;

window = 5 — количество слов из контекста, которое анализирует алгоритм;

min_count = 3 — слово должно встречаться минимум три раза, чтобы модель его учитывала.

На следующем этапе каждый текст был отображен в массив идентификаторов токенов. Я выбрал размерность вектора текста s=26, поскольку при данном значении полностью покрываются 99,71% всех текстов в сформированном корпусе. Если при анализе количество слов в твите превышало высоту матрицы, оставшиеся слова отбрасывались и не учитывались в классификации. Итоговая размерность матрицы предложения составила $s \times d = 26 \times 200$.




*Распределение длины слов*

## Часть 1. Предварительная подготовка данных

```
import pandas as pd
import numpy as np

n = ['id', 'date', 'name', 'text', 'typr', 'rep', 'rtw', 'faw', 'stcount', 'foll', 'frien', 'listcount']
data_positive = pd.read_csv('data/positive.csv', sep=';', error_bad_lines=False, names=n, usecols=['text'])
data_negative = pd.read_csv('data/negative.csv', sep=';', error_bad_lines=False, names=n, usecols=['text'])

sample_size = min(data_positive.shape[0], data_negative.shape[0])
raw_data = np.concatenate((data_positive['text'].values[:sample_size],
                           data_negative['text'].values[:sample_size]), axis=0)
labels = [1] * sample_size + [0] * sample_size

import re

def preprocess_text(text):
    text = text.lower().replace("ё", "е")
    text = re.sub('((www\.[^\s]+)|(https?://[^\s]+))', 'URL', text)
    text = re.sub('@[^\s]+', 'USER', text)
    text = re.sub('[^a-zA-Za-яА-Я1-9]+', ' ', text)
    text = re.sub(' +', ' ', text)
    return text.strip()


data = [preprocess_text(t) for t in raw_data]

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(data, labels, test_size=0.2, random_state=2)
```

**Часть 2. Модели**

Задача состоит в том, чтобы выбрать комбинацию классификатора, метода векторизации, n-граммовой схемы и других параметров таким образом, чтобы максимизировать качество классификации.

Выбраны две модели: наивный байесовский классификатор и линейный классификатор (минимизация через стохастический градиентный спуск). Иногда для таких задач используется SVM, но он очень медленно работает на большом количестве объектов и функций. Логические методы классификации вообще не рассматриваются, так как они абсолютно не подходят для этой задачи.

Байесовский классификатор не нуждается в большом выборе параметров, но параметры линейной модели подбираем на сетке.

По результатам тестирования, лучшей оказалась следующая модель:

- Схема n-грамм: (1, 3) (униграммы + биграммы + триграммы);
- Метод векторизации: TF-IDF;
- Тип модели: линейная модель;
- Параметры модели: penalty – l2, alpha – 0.000001, loss – log.

Результаты точности распознавания этой модели представлены на рисунке 13.

```
NB: 0.7301672333333333
Linear: 0.7428100333333334
Linear Parameters: {'alpha': 0.000001, 'loss': 'log', 'penalty': 'l2'}

TF-IDF Vectorizer
```

*Результаты наилучшей модели*

Для этой же модели, наивный байесовский классификатор показал результат на **1.2%** хуже.

Так же был взят корпус коротких текстов с сервиса Яндекс.Маркет. Для

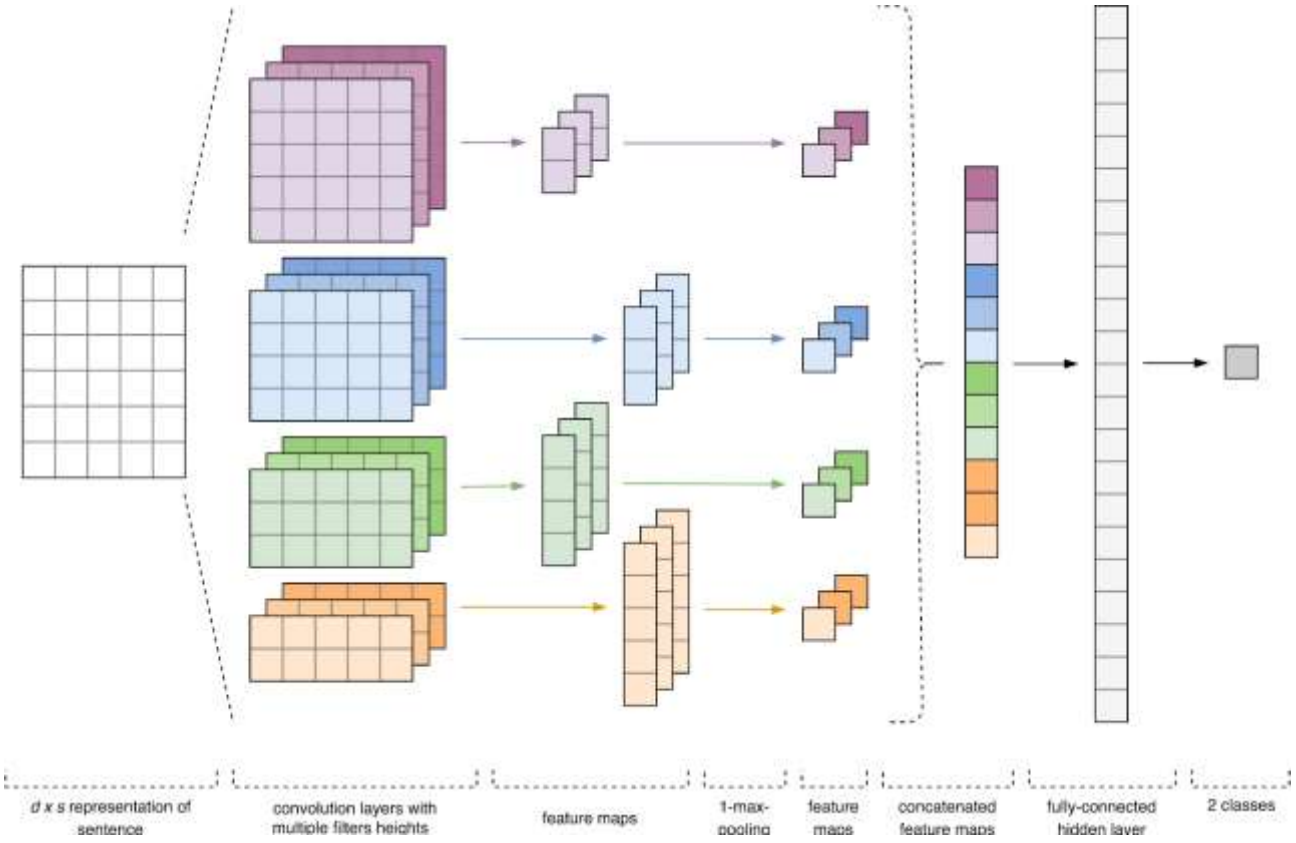
данной выборки оба классификатора показали примерно одинаковый результат точности распознавания около **92%**.

**Сверточная нейронная сеть**

Для построения нейронной сети использовалась библиотека Keras, которая выступает высокоуровневой надстройкой над TensorFlow, CNTK и Theano. В нашем случае embedding-слой был инициирован весами, полученными при обучении Word2Vec. Чтобы минимизировать изменения в embedding-слое, на первом этапе замораживаем этот слой.

В разработанной архитектуре использованы фильтры с высотой *h=(2, 3, 4, 5)*, которые предназначены для параллельной обработки биграмм, триграмм, 4-грамм и 5-грамм соответственно. Добавил в нейронную сеть по 10 свёрточных слоев для каждой высоты фильтра, функция активации — ReLU.

Архитектура сверточной нейронной сети показана на рисунке.




*Архитектура сверточной нейронной сети*

После обработки слоями свертки, карты признаков поступали на слои

субдискретизации, где к ним применялась операция 1-max-pooling, тем самым извлекая наиболее значимые n-граммы из текста. На следующем этапе происходило объединение в общий вектор признаков (слой объединения), который подавался в скрытый полносвязный слой с 30 нейронами. На последнем этапе итоговая карта признаков подавалась на выходной слой нейронной сети с сигмоидальной функцией активации.

Поскольку нейронные сети склонны к переобучению, после embedding-слоя и перед скрытым полносвязным слоем добавим dropout-регуляризацию с вероятностью выброса вершины p=0.2.

Итоговую модель конфигурируем функцией оптимизации Adam (Adaptive Moment Estimation) и бинарной кросс-энтропией в качестве функции ошибок. Качество работы классификатора оценивается в критериях точности, полноты и F-меры.

Результаты точности работы данной сверточной нейронной сети можно увидеть на рисунке.

| | precision | recall | f1-score |
|---|---|---|---|
| 0 | 0.78194 | 0.78243 | 0.78218 |
| 1 | 0.78089 | 0.78040 | 0.78064 |

*Результаты точности распознавания нейронной сети для данных с Twitter*

Для массива данных Яндекс.Маркет точность составила **98%**.

Сравнить точность распознавания всех алгоритмов на разных данных можно по таблице.

*Таблица – Результаты точности каждого алгоритма*

| **Алгоритм** | **Яндекс.Маркет** | **Twitter** |
|---|---|---|
| Сверточная нейронная сеть | **98.0%** | **78.2%** |
| Наивный байесовский классификатор | 92.0% | 73.0% |
| Линейный классификатор | 92.7% | 74.2% |

По результатам испытаний, видно, что сверточная нейронная сеть опережает по точности остальные алгоритмы.

Так же можно сделать вывод о том, что качество корпуса коротких текстов Twitter ниже чем Яндекс.Маркет.

**Выводы**

Все построенные модели обладают очень хорошими показателями. Сверточная нейронная сеть при этом обладает наилучшими показателями по точности. Для дальнейшей работы предпочтительнее будет использовать ее.