

# Чекпоинт II

Stardust Crusaders

22 ноября 2020 г.

## Содержание

<b>1</b>	<b>Цели чекпоинта</b>	<b>2</b>
<b>2</b>	<b>Результаты</b>	<b>2</b>
2.1	Ручная кластеризация . . . . .	2
2.2	Word2Vec и K-means . . . . .	3
<b>3</b>	<b>Цели на последний чекпоинт</b>	<b>3</b>

## Резюме

Почти полностью написали реализации моделей. Немного отстаем от плана. Получили первые результаты

## Цели чекпоинта

По результатам первого чекпоинта были предложены такие цели:

- протестировать предложенные модели;
- собрать случайную выборку (с ручной разметкой) для тестирования;
- сделать заготовку презентации.

Однако, протестировать модели не удалось, так как они не были до конца подготовлены.

## Результаты

Основной github: [ССЫЛКА](#)

Обработанные данные: [ССЫЛКА](#)

Сделан заготовок презентации. Был написан модуль `prepare`, обрабатывающий сырые данные:

- строится очищенная (от цифр и символов пунктуации) таблица. Все слова приводятся в нормальную форму;
- строится корпус на основе очищенных данных для Word2Vec;
- строится словарь уникальных слов, считается частота каждого слова.



Рис. 1: Облако слов для очищенных данных, построенное с помощью языка R

В разработке (написаны, но еще не смирденные в основную ветку) находятся сами модели:

## Ручная кластеризация

Каждому кластеру сопоставляется набор ключевых слов в нормальной форме (этот набор выделяется вручную, на основе частности слов)

1. Приводим все слова в сообщении в нормальную форму, избавляемся от стопслов и символов пунктуации.
2. Проверяем, сколько ключевых слов относящихся к первому кластеру содержится, сколько ко второму и т.д.
3. На основе полученных чисел вычисляем кластер объекта:

- В случае, когда есть явный максимум по количеству совпадений, назначаем кластер соответствующий максимуму
- В случае, когда ключевых слов практически нет, назначаем кластер “Другое”.
- В случае отсутствия явных совпадений, считаем что оператор указал более одного инцидента в сообщении.

На данный момент построен классификатор, однако наборы признаков все еще выделяются.

## Word2Vec и K-means

Были построены две реализации этой модели: первая на системе KNIME, вторая была написана на Python с использованием библиотеки `nltk`.

В отличие от предыдущей модели, эта модель не такая прозрачная, но зато полностью автоматизированная.

## Цели на последний чекпоинт

- Доделать презентацию
- Протестировать модели
- Отрефакторить код моделей
- Написать bash-оболочку и документацию