**Министерство науки и высшего образования Российской Федерации**
федеральное государственное автономное образовательное учреждение высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

# Отчет

по заданию №5-6
«Классификация векторизованного текста»

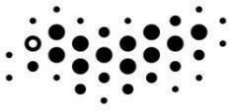
по дисциплине «Компьютерная лингвистика»

Авторы: Исхакова Эмина
Лакиза Александр
Плотская Дарья

Факультет: ИКТ

Группа: К3242

Преподаватель: Чернышева Анастасия Владимировна



УНИВЕРСИТЕТ ИТМО

Санкт-Петербург 2021

**Цель работы:** Построить, обучить и оценить модель классификации векторизованных текстов.

**Ссылка на исходный код**: https://github.com/alexanderlakiza/cs224/blob/main/task5-6/task5-6.ipynb (В основном этапы работы описываются в самой тетрадке)

**Разделение обязанностей по ходу работы**:
Исхакова Эмина: Поиск датасета, описание датасета, составление отчёта
Лакиза Александр: Очистка текста, создание и обучение модели
Плотская Дарья: Визуализация данных и выводы

**Ход работы**: Мы взяли датасет с твитами о работе американских авиакомпаний на английском языке. Датасет содержит более 10000 твитов и имеет классификацию на 3 типа (positive, neutral, negative). Для нас важны лишь два столбца из 15: с текстом и классификацией.

Проводим предобработку текста:

```
punct_marks = string.punctuation + "—" + "«" + "»" + "`" + "``" + "..." + "~" \
+ "'''" + "”" + ''' + '…'
auxiliary_pos = ['NPRO', 'PREP', 'CONJ', 'PRCL', 'INTJ']
for i in range(len(X)):
    X[i] = re.sub(r'@', '', X[i])  # удаление @
    X[i] = re.sub('http://\S+|https://\S+', '', X[i])  # удаление ссылок
    X[i] = re.sub('http[s]?://\S+', '', X[i])
    X[i] = word_tokenize(X[i])  # токенизация
    X[i] = [morph.parse(word)[0].normal_form for word in X[i]]  # лемматизация
    X[i] = [word.lower() for word in X[i] if word not in punct_marks]
    # нижний регистр и знаки препинания
    X[i] = [word for word in X[i] if morph.parse(word[0])[0].tag.POS not in auxiliary_pos]
    # удаление слов служебных частей речи
```

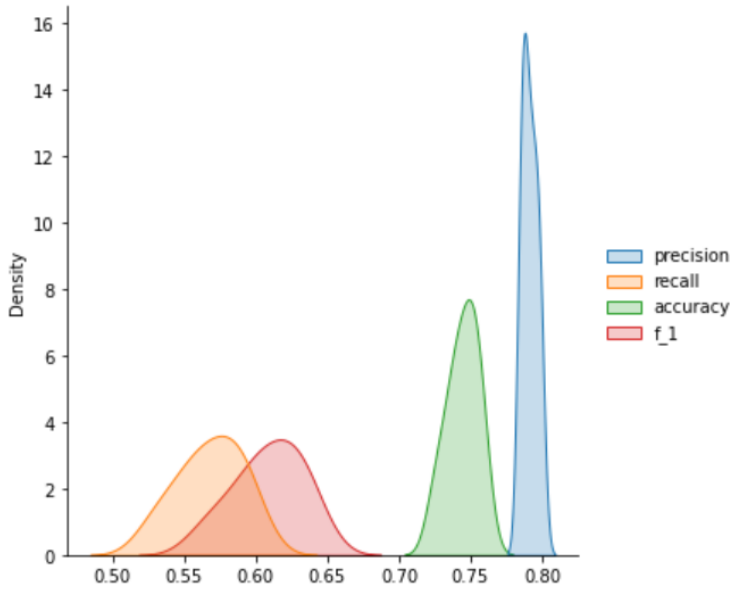
Далее для каждого типа векторизации создаём, как указано в задании, свою таблицу с метриками при разных параметрах
Пример с мешком n-грамм:

| | type | param | precision | recall | accuracy | f_1 |
|---|---|---|---|---|---|---|
| **0** | n-grams | (1, 1) | 0.767058 | 0.605234 | 0.759106 | 0.647163 |
| **1** | n-grams | (1, 2) | 0.806666 | 0.543497 | 0.733651 | 0.583758 |
| **2** | n-grams | (1, 3) | 0.809387 | 0.508971 | 0.715439 | 0.542486 |
| **3** | n-grams | (1, 4) | 0.803249 | 0.495886 | 0.708402 | 0.525111 |
| **4** | n-grams | (1, 5) | 0.802649 | 0.486398 | 0.703228 | 0.512773 |
| **5** | n-grams | (1, 6) | 0.809522 | 0.482148 | 0.700952 | 0.507180 |
| **6** | n-grams | (1, 7) | 0.809737 | 0.477970 | 0.698882 | 0.501816 |
| **7** | n-grams | (1, 8) | 0.808533 | 0.476050 | 0.697848 | 0.499149 |

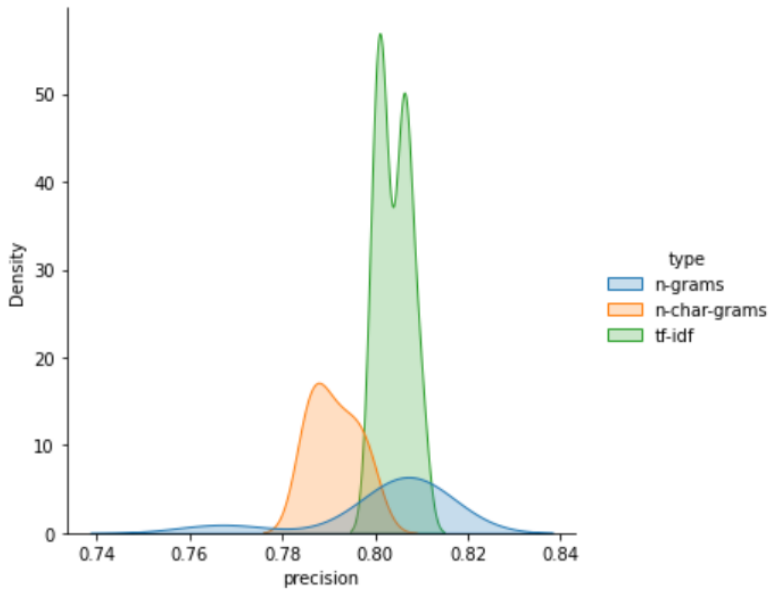
Затем мы добавили графики распределения по метрикам и по векторизаторам:

Примеры:

```
sns.displot(data=res[res['type']=='n-char-grams'], kind='kde', fill=True);
```




```
sns.displot(data=res, x='precision', hue='type', kind='kde', fill=True);
```




В конце мы сделали следующие **выводы:**

- **Показатель precision постепенно растёт у мешка n-грамм при увеличении параметра n. Это может быть связано с удачным (для данного векторизатора) разделением на обучающую и тестовую выборки. У tf-idf данная метрика колеблется у одного значения. Остальные метрики имеют примерно одинаковое распределение, что может объясняться математически.**

- **Мешок n-грамм даёт в среднем лучшие показатели при параметрах (1, 1), (1, 2) и (1, 3). Мешок символьных n-грамм при (5, 8) и (5, 9).**

- **Самыми лучшими векторизаторами для данного датасета получились: n-gramm (1, 1) и n-char-gram (5, 8), (5, 9).**

- **Не самые высокие показатели метрик могут быть следствием не самой точной изначальной классификации твитов, например, в исходном датасете некоторые твиты относятся к какому-то классу с точностью всего лишь 0.34, что крайне мало.**