## Классификация текстов

### Цель работы

Применить методы машинного обучения для решения задач классификации текстов.

### Задания для выполнения

1. Загрузите датасет 20 newsgroups;
2. Познакомьтесь с описанием и структурой датасета. Описание можно найти в [документации](https://scikit-learn.org/stable/datasets/index.html#real-world-datasets).
3. Выведите информацию о количественных параметрах датасета;
4. Выведите несколько точек датасета (сами текстовые фрагменты и значение целевой переменной);
5. Разделите эти данные на тестовую и обучающую выборки;
6. Постройте модель наивного байесовского для классификации текстов;
7. Оцените качество модели на тестовой выборке с помощью следующих метрик:
   1. достоверность предсказания (accuracy);
   2. точность (precision);
   3. полнота (recall);
8. Постройте кривую обучения - график зависимости тестовой и обучающей эффективности от размера обучающей выборки.
9. Сделайте вывод о применимости модели.

### Методические указания

Датасет 20 newsgroups - это один из известных модельных наборов данных для обучения методам классификации текстов на естественных языках. Его можно получить с помощью стандартных средств sklearn:

from sklearn.datasets import fetch\_20newsgroups

news = fetch\_20newsgroups(subset='all')

Задача обработки текстов на естественных языках

Превращение текста в набор численных признаков - это сама по себе нетривиальная задача. Часто эту задачу называют векторизация текста. От применяемого метода векторизации очень сильно зависит эффективность методов машинного обучения (наверное, даже больше, чем от самих методов классификации).

В библиотеку sklearn встроены несколько простых методов векторизации текстов. Они собраны в модуле sklearn.feature\_extraction.text. Более подробно о них можно почитать в [документации](https://scikit-learn.org/stable/modules/feature_extraction.html#text-feature-extraction).

Их применение довольно прямолинейно. Сначала нужно создать векторизатор, а затем преобразовать с помощью него текст в набор векторов:

vectorizer = CountVectorizer()

X = vectorizer.fit\_transform(corpus)

Обратите внимание, что данный метод преобразования составляет словарь токенов текста. Поэтому если мы применим его на другой выборке у нас получится другой словарь. Чтобы преобразовать другую выборку на основе того же самого словаря нужно применить другой метод:

X = vectorizer.transform(['Something completely new.'])

Это поведение похоже на обучение модели. Векторизатор обучается на обучающей выборке, а затем используется для преобразования тестовой.

После векторизации исходных данных использование методов машинного обучения ничем не отличается от работы с числовыми данными.

### Контрольные вопросы

1. Какие выводы мы можем сделать на основании метрик модели, построенной в данной лабораторной работе?
2. Как представляется текст в методах машинного обучения? Какие основные методы векторизации существуют?
3. Какие задачи существую в области обработки естественных текстов?
4. (\*) Что такое текстовые вложения (эмбеддинги)? Как они строятся и зачем применяются?

### Дополнительные задания

1. Постройте модели классификации для данной задачи на основе следующих методов:
   1. логистическая регрессия (LogisticRegression);
   2. метод опорных векторов с гауссовым ядром (SVC);
   3. метод опорных векторов с полиномиальным ядром (SVC);
   4. метод k ближайших соседей (KNeighborsClassifier);
   5. многослойный перцептрон (MLP);
   6. другие методы по желанию;
2. Проанализируйте метрики каждой модели и сделайте выводы об их эффективности и применимости. Сравните эффективность всех этих моделей и выберите лучшую;
3. Для каждой модели из п.3 постройте кривые обучения и диагностируйте недо-/переобучение модели. Попробуйте изменить параметр регуляризации для улучшения результатов модели.
4. Сделайте замеры времени обучения для каждой модели. Сделайте вывод о сравнительной производительности моделей.
5. (\*) Используйте глубокую нейронную сеть для решения той же задачи. Сравните ее эффективность и производительность с классическими моделями.