Определение авторства текстов

статус проекта на март 2024

Проект первого года студентов магистратуры НИУ ВШЭ «Машинное обучение и высоконагруженные системы»

Задача и состав команды

Задача

создать ML-сервис для определения авторства текста по фрагменту из 5-10 предложений

Состав команды

- Дарья Мишина (предобработка данных, EDA, ML, DL)
- Кирилл Рубашевский (сбор данных, feature engineering, EDA, ML)
- Дмитрий Шильцов (предобработка данных, EDA, ML)

- Елена Вольф (куратор)

Данные

для проекта выбраны 10 классических русских писателей 19 века:

- И.А. Бунин
- В.М. Гаршин
- Н.В. Гоголь
- Ф.М. Достоевский
- А.И. Куприн
- Д.Н. Мамин-Сибиряк
- А.П. Платонов
- А.С. Пушкин
- И.С. Тургенев
- А.П. Чехов

по каждому автору отобраны 10+ прозаических произведений (в т.ч. в составе сборников)

данные собраны на сайте интернет-библиотеки Алексея Комарова

данные собраны при помощи разработанного парсера, который поддерживает:

- парсинг текстов на нескольких страницах
- продолжение ранее начатого парсинга

собранные данные размещены на S3 Яндекса и доступны по внешней ссылке

Сплит текстов на объекты

проблема: первоначальный сплит текстов на объекты длиною 500 символов приводил к низким значениями метрик алгоритмов всех семейств (каждый объект содержал ~50-70 токенов, чего было недостаточно для обучения моделей)

найденное решение: увеличить длину объектов (подготовлены сплиты на объекты длиной 1000, 2000 и 3000 символов)

сплитованные данные версионируются при помощи DVC

Два трека предобработки данных

Классический трек

предобработка данных включала этапы:

- удаления пунктуации
- удаления стоп-слов
- лемматизации
- приведения текстов к нижнему регистру

(в рамках экспериментов этапы применялись по отдельности или комбинировались)

для проведения экспериментов этапы предобработки собраны в scikit-learn-совместимый TextTransformer

Грамматический трек

Знаки препинания вводились как лексемы, слова кодировались в виде «часть речи | грамматическая форма» при помощи библиотеки pymystem3 (полная / сокращенная грамматическая информация)

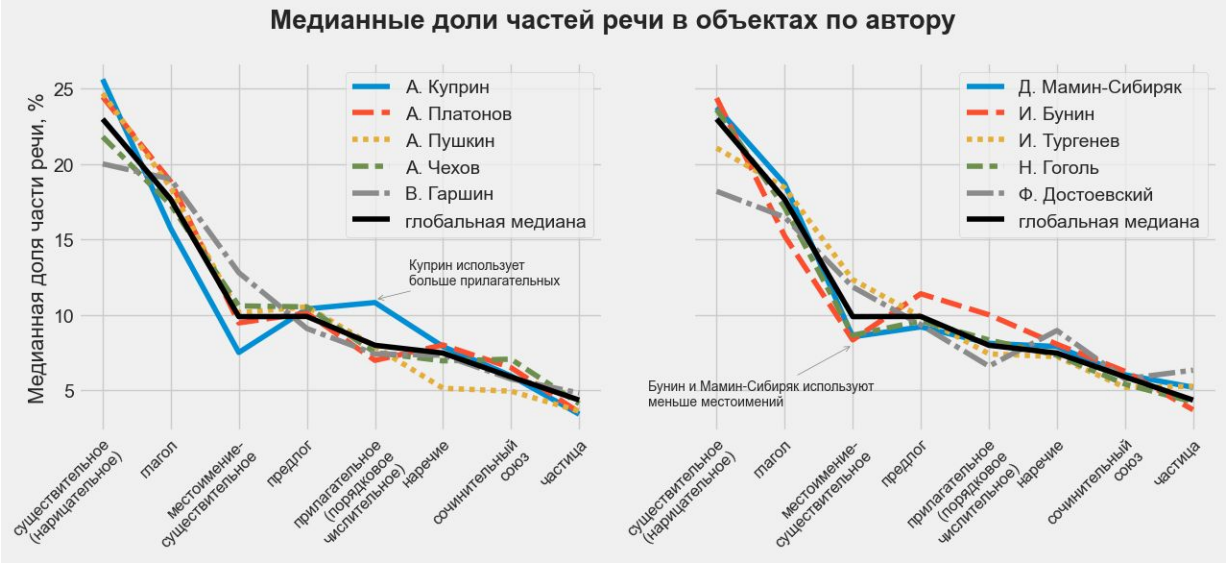
```
prepare_text(['мама, которая мыла раму!'], full=True)
['S,жен,од=им,ед CM APRO=им,ед,жен V,несов,пе=прош,ед,изъяв,жен S,жен,неод=вин,ед EXCL']
```

```
prepare_text(['мама, которая мыла раму!'], full=False)
['S,жен,од CM APRO V,несов,пе S,жен,неод EXCL']
```

EDA: part-of-speech и тематическое моделирование

авторы различаются по частоте использования частей речи, но методы понижения размерности (PCA, TSNE) на этих статистиках не позволили кластеризовать авторов

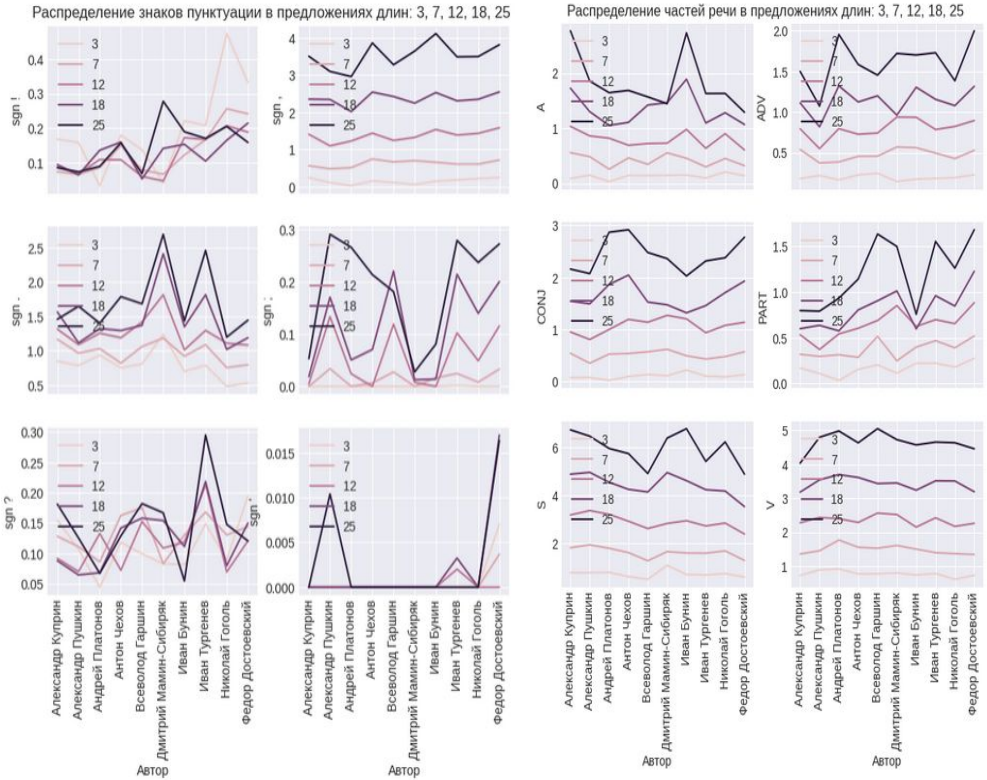
тематическое моделирование не привело к существенным результатам



EDA: пунктуация и part-of-speech

авторы по-разному употребляют части речи в зависимости от длины предложения

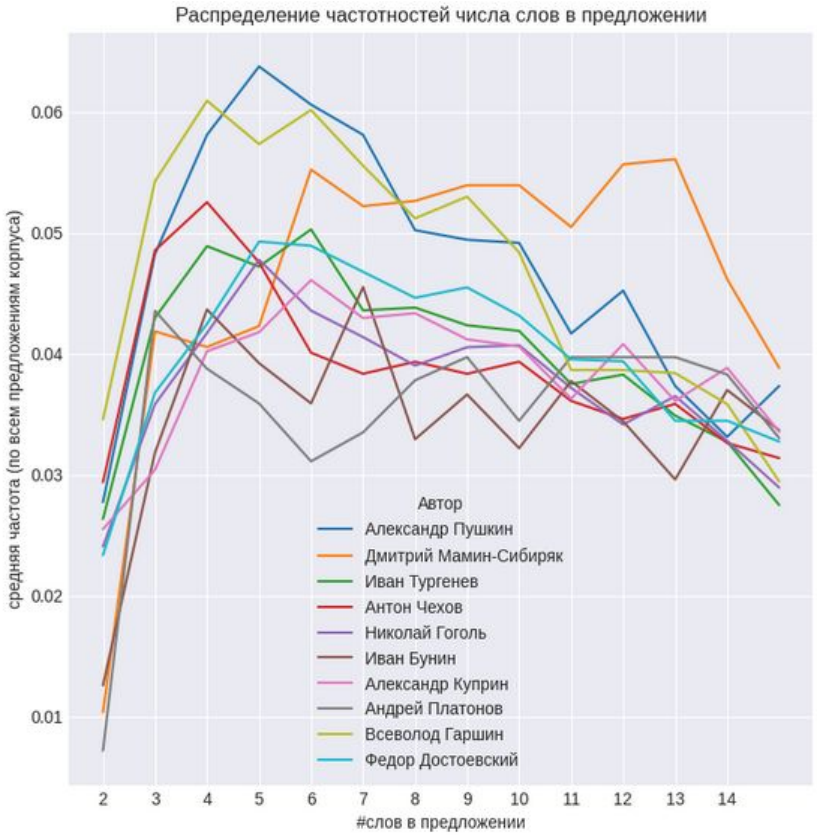
употребление знаков препинания в зависимости от длин предложения у авторов также различается



EDA: длины предложений

авторы имеют различные предпочтения по длинам предложений (например Гаршин в среднем использует более короткие, а Мамин-Сибиряк — более длинные)

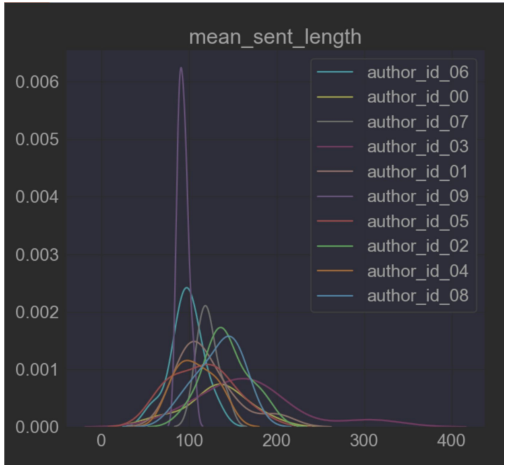
анализ частотностей длин предложений а также статистики употребления знаков препинания и частей речи может дать дополнительные признаки при анализе фрагментов текста достаточно большой длины

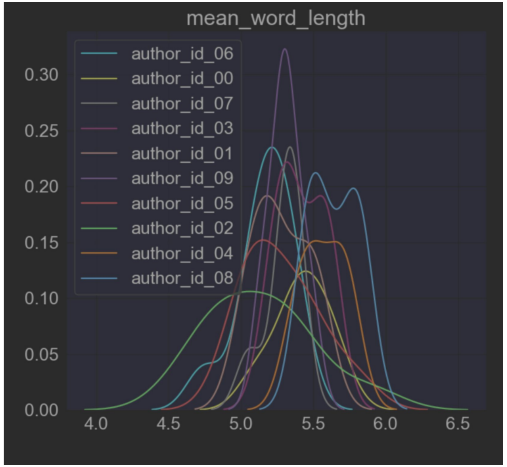


EDA: текстовые статистики

длина слов и предложений у авторов различаются, поэтому сделали углубленный анализ статистик с помощью пакета ruts, изучив количество:

- предложений
- слов
- уникальных слов
- длинных слов
- сложных слов
- простых слов
- односложных слов
- многосложных слов
- символов
- букв
- пробелов
- слогов
- знаков препинания





после анализа самых часто встречающихся слов был расширен список стоп-слов

Feature generation

Эмбеддинги

Для обоих треков предобработки данных использовались эмбеддинги:

- Bag-of-Words
- TF-IDF

с опциональным снижением размерности при помощи PCA/SVD

Текстовые статистики

Из данных после классического трека предобработки получены текстовые статистики по

- предложениям
- токенам
- part-of-speech

для проведения экспериментов получение статистик собрано в scikit-learn-совместимый экстрактор TextStatsExtractor

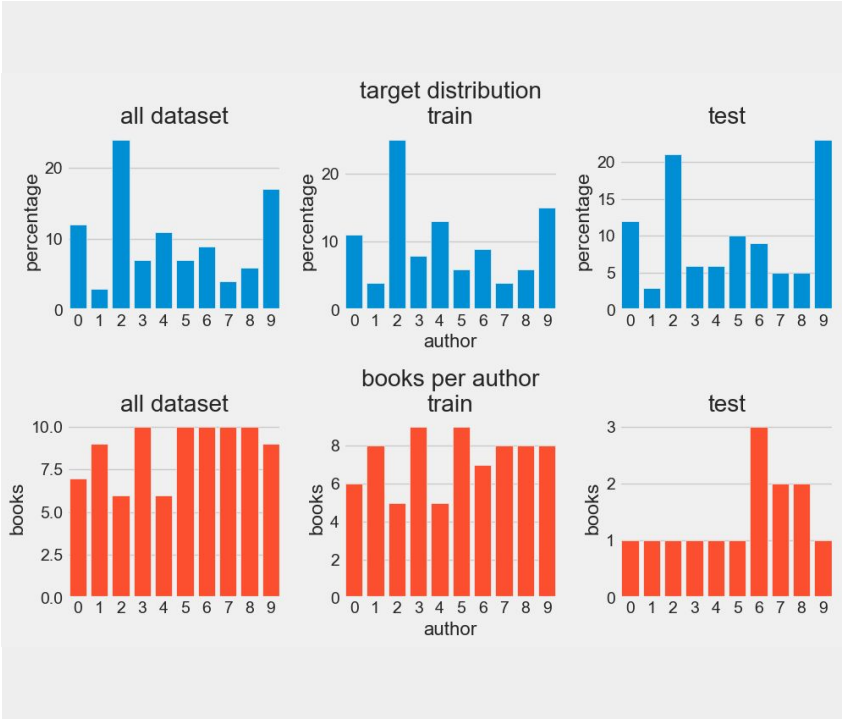
ML: сплит объектов на трейн и тест

проблема: стратифицированный рандомный сплит приводил к переобучению моделей

найденное решение: использовать StratifiedGroupKFold, при котором:

- объекты одного произведения (group) попадают только в одну выборку
- сохраняется соотношение классов

(подробнее о сплите в ноутбуке)

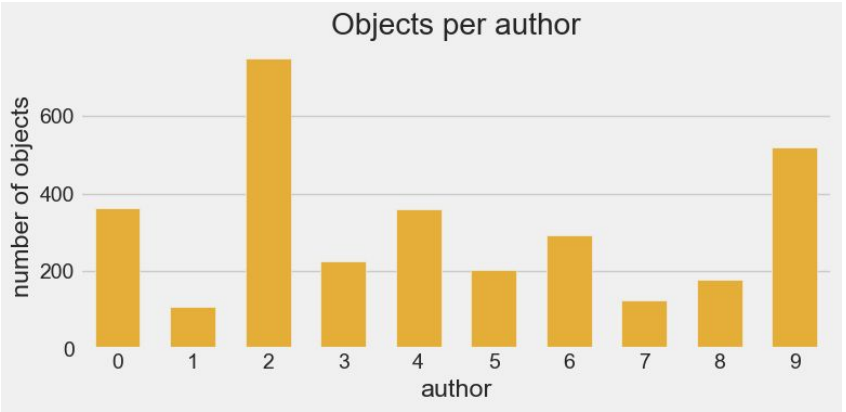


ML: целевая метрика, пайплайн экспериментов

Классы не сбалансированы (произведения разной длины), поэтому целевая метрика — weighted f1

Подбор гиперпараметров оптимизирован при помощи скрипта train.py, который:

- загружает архитектуру модели и гиперпараметры из зоопарка classifiers.py
- подбирает их значения на кросс-валидации StratifiedGroupKFold
- логгирует параметры и метрики на трейне и тесте в Weights & Biases



W&B

ML: линейные модели

Протестировали линейные модели (логистическая регрессия и SVM) с:

- разной предобработкой (классический трек: удаление стоп-слов, удаление пунктуации, лемматизация и их комбинация; грамматический трек)
- разными признаками (эмбеддинги (TF-IDF, Bag-of-Words), текстовые статистики (с/без создания полиномиальных признаков) и их комбинация)
- стандартизацией признаков
- разными ядрами (для SVM)

Линейные модели показали результаты лучшие результаты среди других алгоритмов (~0.8-0.85)

ML: KNN и вероятностные модели

Протестировали KNN и вероятностные модели (Multinomial, Complement и Gaussian Naive Bayes) с:

- разной предобработкой (классический трек: удаление стоп-слов, удаление пунктуации, лемматизация и их комбинация)
- разными признаками (эмбеддинги (TF-IDF, Bag-of-Words), текстовые статистики (с/без создания полиномиальных признаков) и их комбинация)
- стандартизацией признаков
- понижением размерности эмбеддингов при помощи SVD

Multinomial Naive Bayes показал результаты, близкие к линейным (~0.79)

KNN показал достаточно скромное качество (~0.56)

ML: деревья и их ансамбли

Протестировали Random Forest и CatBoost с:

- разной предобработкой (классический трек: удаление стоп-слов, удаление пунктуации, лемматизация и их комбинация; грамматический трек)
- разными признаками (эмбеддинги (TF-IDF, Bag-of-Words), текстовые статистики (с/без создания полиномиальных признаков) и их комбинация)
- стандартизацией признаков
- понижением размерности эмбеддингов (SVD для классического трека, PCA для грамматического)

Random Forest показал хорошие результаты, но ниже результатов линейных и вероятностных моделей (~0.72)

CatBoost показал результаты немного хуже KNN (~0.55)

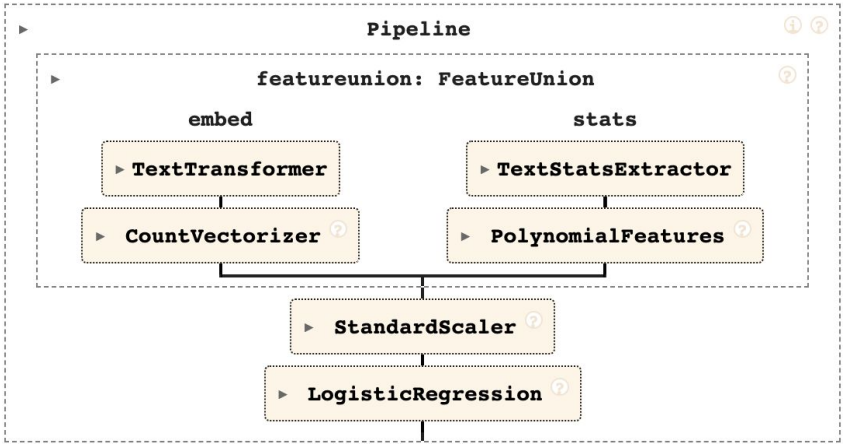
ML: лучшая модель

лучшая модель — логистическая регрессия на двух категориях признаков:

- Bag-of-Words-эмбеддинги
- полиномиальные признаки на текстовых статистиках

weighted f1 лучшей модели:

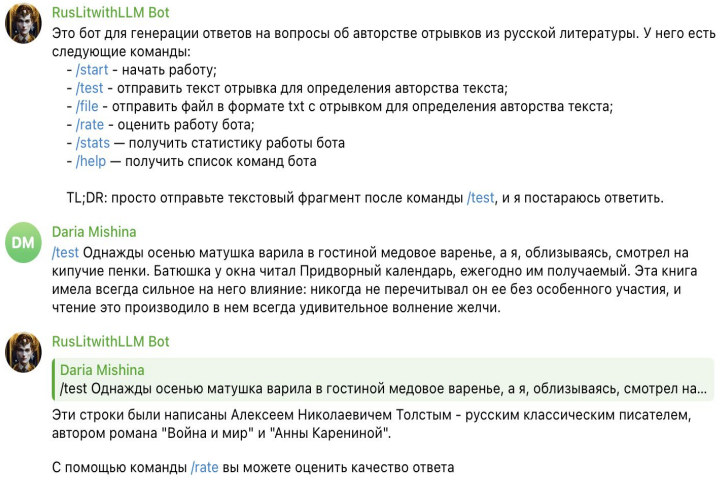
- трейн: 0.74±0.12
- тест: 0.85



DL: первые эксперименты

дообучены на нашем датасете трансформеры cointegrated/rubert-tiny2 и DeepPavlov/rubert-base-cased

протестированы несколько опенсорсных LLM (пока выбран openchat)



Deployment

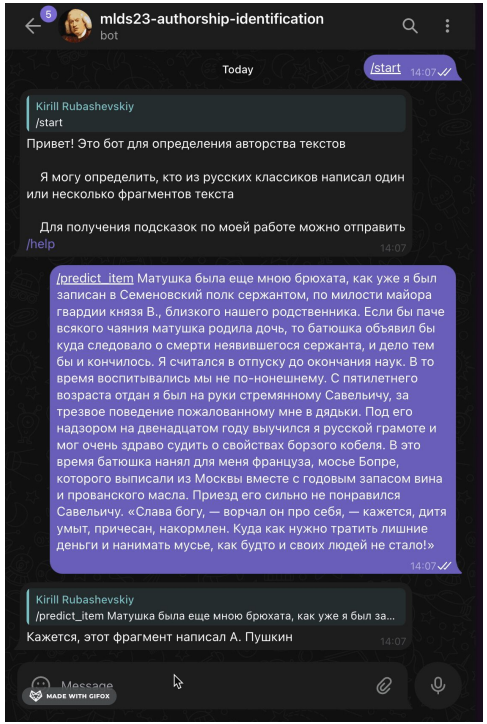
MVP реализован в формате тг-бота, который развернут на render.com и доступен по ссылке

Бот:

- может предсказывать авторство одного (сообщение) или нескольких (csv-файл) фрагментов
- возвращает confidence prediction: если вероятность всех авторов ниже порога, бот сообщает о невозможности сделать уверенное предсказание

Также реализованы и доступны:

- бот на основе LLM
- бот на основе грамматического трека предобработки данных



To do

DL

- попробовать больше семейств моделей (DL) и эмбеддингов (Word2Vec, GloVe, BERT)
- оптимизировать гиперпараметры через optuna/hyperopt
- дообучить LLM

MLOps + DevOps + deployment

- вынести модель из бота в отдельный веб-сервис
- обернуть все в докер
- оформить проект в Streamlit app