

Отчет по решению вступительного задания на предстажировку Case Lab ML

АО «Гринатом»

2024

Выполнил: Щеглаков Илья Викторович

Адрес эл.почты: boetzvtanke@yandex.ru

**Задание:**Дан датасет Large Movie Review Dataset.

1. Необходимо:  
   Обучить модель на языке Python для классификации отзывов.
2. Разработать веб-сервис на базе фреймворка Django для ввода отзыва о фильме с автоматическим присвоением рейтинга (от 1 до 10) и статуса комментария (положительный или отрицательный).
3. Развернуть сервис в открытом доступе для оценки работоспособности прототипа.
4. Подготовить отчет о работе с оценкой точности полученного результата на тестовой выборке.

**Ход работы:**

Датасет представляет из себя набор текстов (отзывов о фильмах), каждому из которых сопоставлено число (оценка). Разработка модели, которая по тексту сможет предсказывать число, является задачей NLP. На мой взгляд, оптимальным решением было бы использовать предобученную сеть, например BERT, и дообучить ее, однако так как задание сформулировано как «обучить модель», то было решено обучать модель с нуля.

В конечном итоге для тестирования были выбраны два пайплайна: TF-IDF векторизация + XGBoost модель; Word2Vec + LSTM.

Забегая вперед, первый вариант оказался предпочтительнее, поэтому рассмотрим его.

Сам датасет содержит не только информацию об оценке отзыва, но и принадлежность классу «положительный» или «отрицательный». Однако, на мой взгляд, предсказывать класс отдельно от оценки большого смысла не имеет, поэтому итоговая классификация будет строиться по предсказанному числу по простому принципу “>=5” или “<5”.

Чтение датасета:

folder\_path = os.path.join(os.getcwd(), "dataset", "aclImdb", "test", "neg")

data = []

for filename in os.listdir(folder\_path):

    score = filename.split('\_')[1].replace(".txt", "")

    with open(os.path.join(folder\_path, filename), 'r', encoding='utf-8') as file:

        text = file.read().strip()

    data.append({"text": text, "score": score})

df4 = pd.DataFrame(data)

Затем, перед тем, как токенизировать текст, его необходимо предварительно обработать. Для этого я использовал библиотеку nltk и регулярные выражения

def preprocess\_text(text):

    text = text.lower()

    text = re.sub(r'\d+', '', text)  # Remove numbers

    text = re.sub(r'\W', ' ', text)  # Remove non-word characters

    text = re.sub(r'\s+', ' ', text)  # Remove multiple spaces

    text = ' '.join([word for word in text.split() if word not in stop\_words])  # Remove stopwords

    return text

Далее токенизируем тексты:

tfidf = TfidfVectorizer(max\_features=5000)

X\_train\_tfidf = tfidf.fit\_transform(X\_train)

X\_test\_tfidf = tfidf.transform(X\_test)

Для обучения модели использовалась библиотека xgboost, так как она проста в настройке и дает SOTA результат на многих задачах.

Параметры модели:

xgb\_model = XGBRegressor(

    objective='reg:squarederror',

    random\_state=6,

    n\_estimators=200,

    max\_depth=7,

    reg\_alpha=10,

    reg\_lambda=1,

    learning\_rate=0.2,

    colsample\_bytree=0.6, subsample=0.7,

)

Валидация модели:

y\_pred = xgb\_model.predict(X\_test\_tfidf)

correct = 0

total = 0

mae = mean\_absolute\_error(y\_test, y\_pred)

predicted = (y\_pred >= 5)

labels = y\_test.apply(lambda p: 1 if p>=5 else 0)

correct += (predicted == labels).sum().item()

total += labels.shape[0]

print(f"Mean Abs Error: {mae}  Accuracy: {correct / total}")

На тестовом наборе данных (случайная выборка в 50% от общего датасета) модель показала следующий результат:

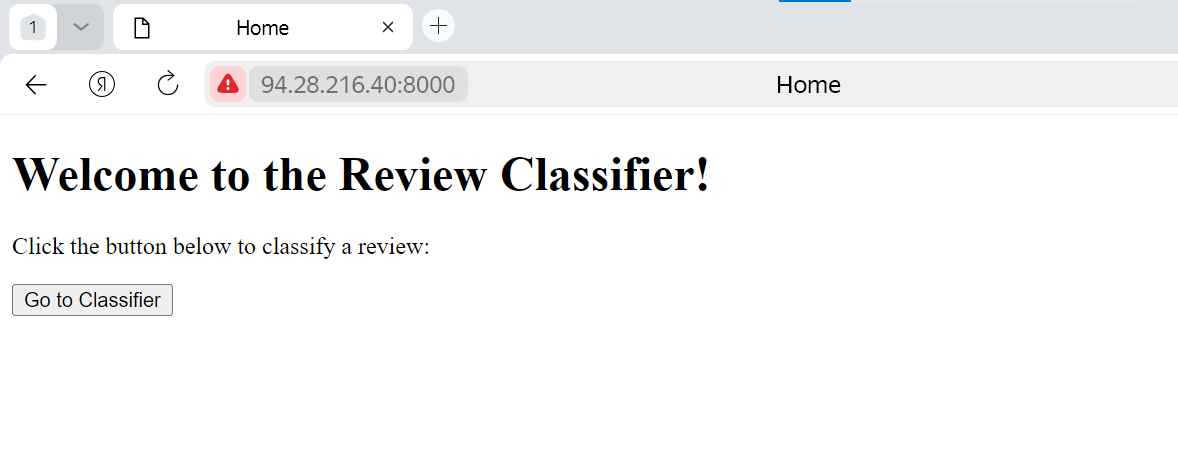
**Mean Abs Error:** 1.8266662257038522

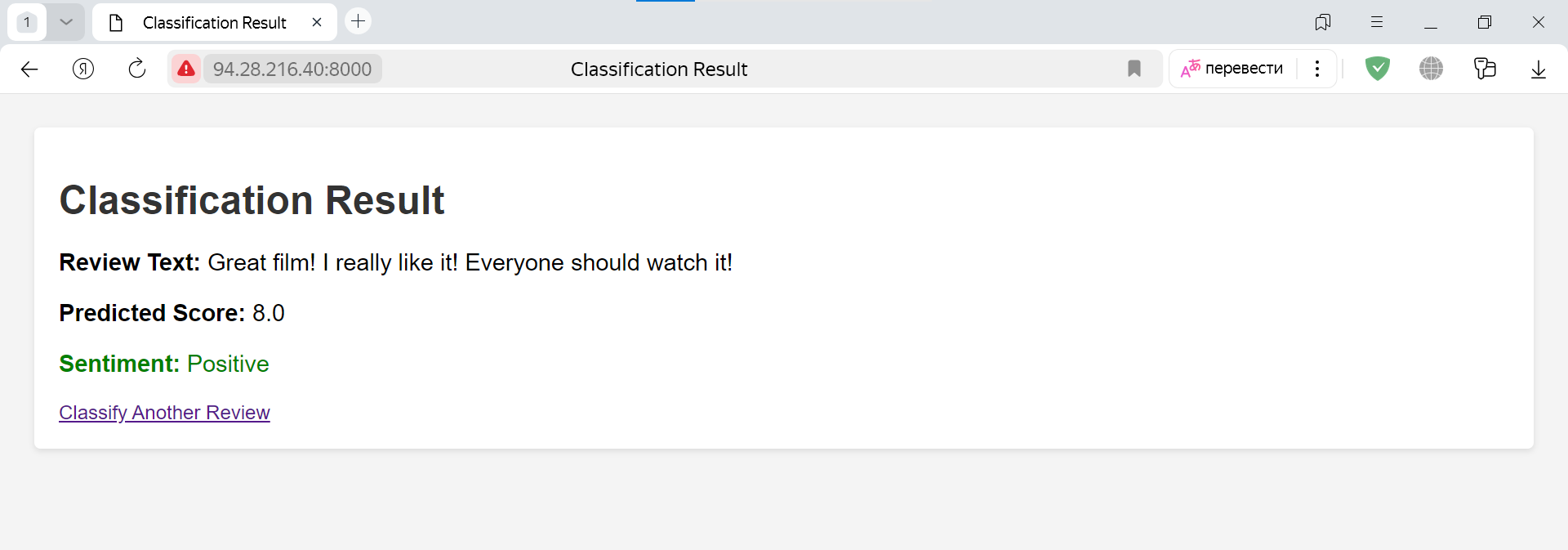
**Accuracy:** 0.8326

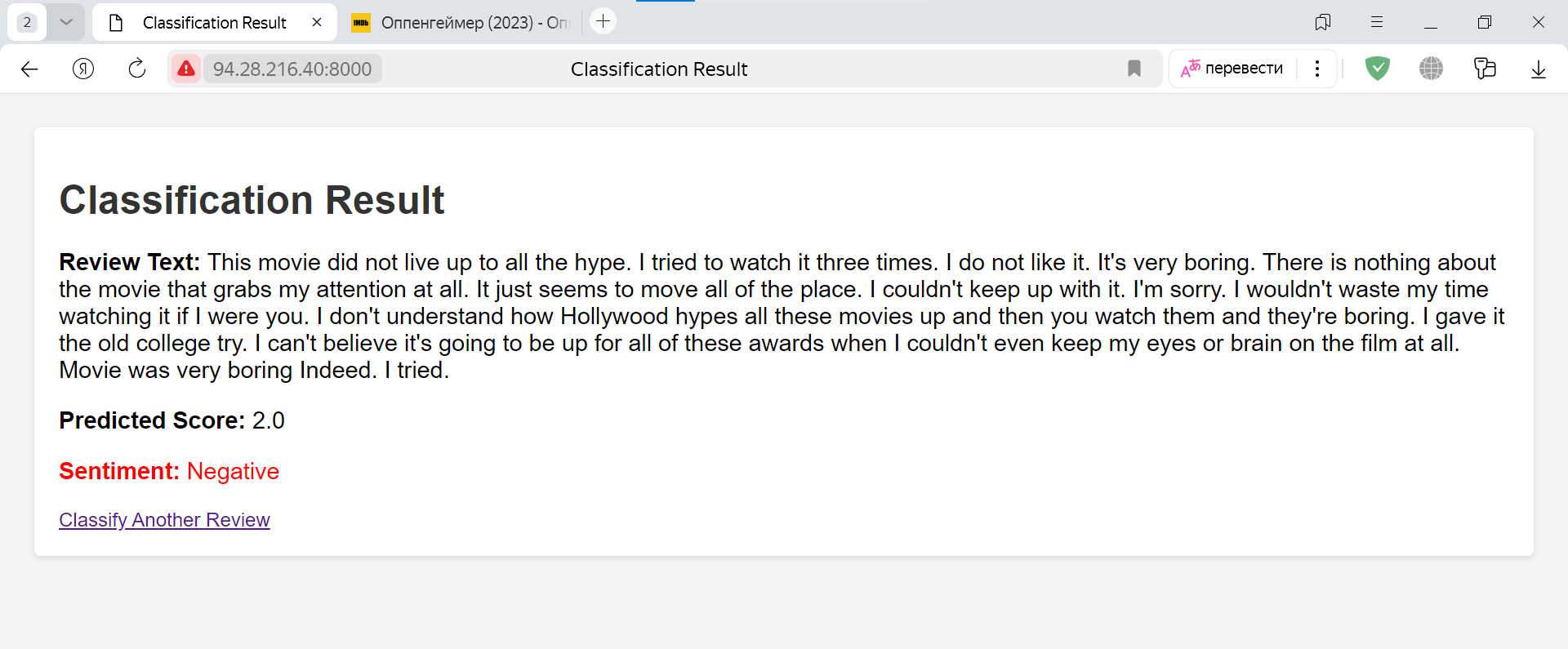
Для улучшения финальной модели имеет смысл обучить ее на всех имеющихся данных.

Помимо этого, была реализована и протестирована LSTM модель, однако она показывает схожие результаты метрик при гораздо больших затратах времени на обучение и вывод результата, поэтому целесообразнее использовать первую модель.

Также, для работы с полученной моделью был сделан веб интерфейс на основе фреймворка Django.







**Итог**

В результате работы была обучена модель классификации отзывов фильмов на основе их текста, предсказывающая оценку и класс. Был разработан веб-сервис, позволяющий удобно взаимодействовать с полученной моделью.

Модель имеет следующие значения метрик на тестовой выборке:

**Mean Abs Error:** 1.8266662257038522

**Accuracy:** 0.8326

Что, учитывая специфику области ее работы, можно считать удовлетворительным результатом.