Отчет по проекту

«Классификация отзывов на товары магазина музыкальных инструментов»

Выполнила: Прилукова Полина,

Студент профессии Data Scientist, группа DS-20

Ментор: Сапрыкин Артур

декабрь 2021 г.

Киров

1. Введение

Почти любая компания, предоставляющая некие товары или услуги, в той или иной степени ориентируется на обратную связь от потребителей своих услуг, чтобы оставаться конкурентоспособной. Отзывы пользователей позволяют выявлять достоинства и недостатки своего продукта. Эту информацию компании используют для того, чтобы усовершенствовать свою продукцию и скорректировать политику продаж. Так же, основываясь на анализе отзывов не только своей компании, но и в целом по сегменту, можно проводить исследования рынка, выявляя сильные и слабые стороны партнеров или конкурентов. А честная публикация отзывов и своевременная реакция на них повышает уровень доверия к компании и продукции, которую она предоставляет.

С развитием интернета в целом и сферы электронных продаж в частности, количество этой обратной связи возрастает кратно. Поэтому возникает необходимость в автоматических системах анализа и модерации мнений пользователей.

1.1 Постановка цели

Отсюда вытекает постановка цели данного проекта: разработать модель, которая будет классифицировать отзывы пользователей в зависимости от тональности текстов этих отзывов. В качестве предметной области была выбрана сфера продаж музыкальных инструментов и оборудования.

1.2 Описание предметной области

В процессе продаж музыкальных инструментов отзывы играют важную роль как для продавца по названным выше причинам, так и для покупателя. Музыкальные инструменты – это товары долгого пользования и зачастую дорогостоящие. Не всегда при первичном осмотре в момент покупки, а тем более при заказе продукции через интернет, возможно определить какие-то важные детали, которые станут понятными уже в процессе использования. Поэтому опыт и замечания предыдущих покупателей имеют вес при принятии окончательного решения.

В качестве основного источника данных был выбран сайт российского магазина pop-music.ru. Эта компания была основана в 2002 году, на тот момент они были первыми среди продавцов музыкального оборудования, кто запустил свой интернет-магазин. В данное время момент по оценкам экспертов доля продаж pop-music составляет от 15% до 20% рынка музыкальных инструментов и оборудования в Рунете. Доля продаж в офлайн-пространстве оценивается в 8%.

2. Сбор данных

Данные с сайта собирались скриптами в два этапа. Первый скрипт собирал ссылки на страницы каталога и через них доставал все страницы товаров магазина. Второй скрипт со страницы конкретного товара собирал информацию обо всех отзывах, если они есть, и, пройдя по ссылкам всех страниц, набранных на первом этапе, формировал итоговый датасет.

Таким образом получился датасет, состоящий из ~11000 записей.

3. Анализ данных

Весь код, связанный с анализом и моделированием, приведен в этом ноутбуке.

3.1 Исследование аналогичных решений

Косвенным подтверждением того, что тема анализа тональности актуальна, служит заметное количество датасетов и соревнований на Kaggle по схожей тематике. Например, анализ тональности финансовых новостей, твитов, анализ отзывов на фильмы (таких особенно много), вино или отели.

Сложно выделить какой-то единый подход к решению подобных задач, разве что наиболее общие и очевидные черты: предобработка данных, векторизация текстов, обучение классификаторов. В качестве моделей могут выступать классификаторы sklearn, полносвязные и рекуррентные нейросети, более сложные предобученные сети-трансформеры.

Эта работа так же предполагает применение различных подходов и сравнение их результатов.

3.2 Определение целевой переменной

Датасет содержит в себе следующие признаки, показанные на рисунке.

Рис. 1 - Исходный датасет

	author	date	product_id	product_name	rate	text
0	Егор Гумеров	15.10.2017	888880022374	АКУСТИЧЕСКАЯ ГИТАРА STAGG SA20D RED	4	Это моя первая гитара. Консультант ответил на ...
1	Гость	15.06.2010	888880000341	ПРЕДУСИЛИТЕЛЬ ART USBDUALPRE	3	Пожалуйста скажите кто-нибудь у кого он есть т...
2	Гость	03.06.2010	888880000341	ПРЕДУСИЛИТЕЛЬ ART USBDUALPRE	5	клевая вещь. ка будет в МСК?
3	Гость	18.05.2010	888880000341	ПРЕДУСИЛИТЕЛЬ ART USBDUALPRE	3	Пришлось снизить оценку -1 т.к. выснилось что ...

Основное, что представляет здесь интерес – это поле text, содержащее непосредственно полный текст пользовательского отзыва, и признак rate, оценка, выставленная этим пользователем. В данном случае, не имея какой-либо другой разметки пользовательского настроения, я определяю три основные возможные эмоции пользователей (позитивную, нейтральную и негативную) исходя из выставленной оценки. В датасет добавлена переменная sentiment, она и будет целевой переменной, которую необходимо предсказать.

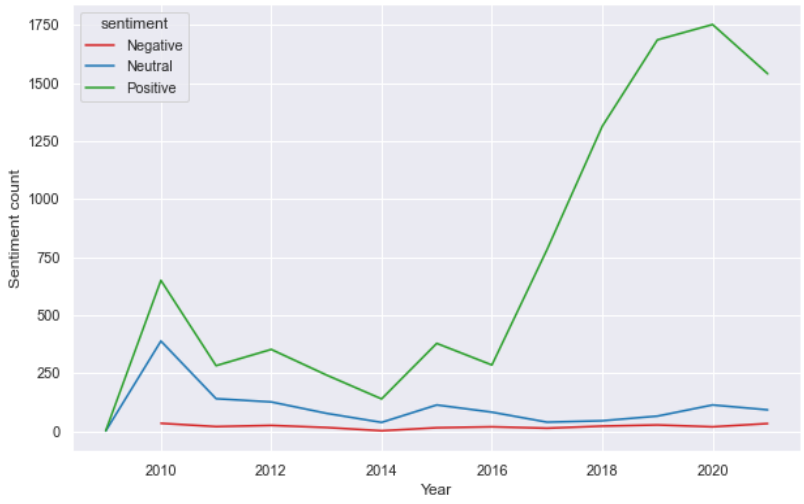
Для отзывов с оценками 1 и 2 sentiment принимает значение Negative, для 3 и 4 – Neutral, для 5 – Positive.

Выбор целевой переменной именно настроения sentiment был сделан, т.к. именно понимание настроения, а не знание конкретной цифры оценки, является полезной информацией при реакции на отзыв. Тем более выборка, разбитая на большее количество классов, скорее всего будет менее сбалансированной.

3.3 Статистический подход

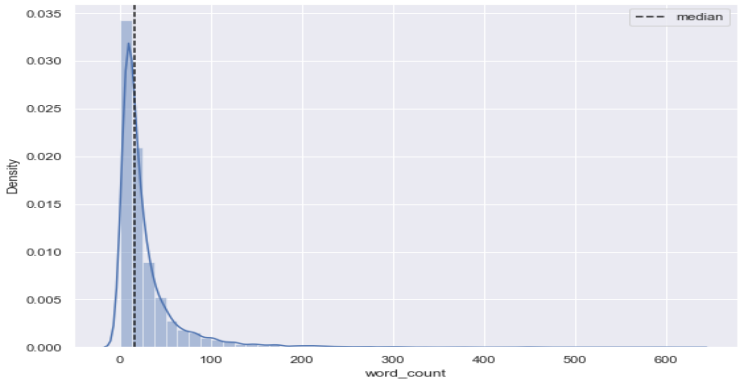
В целом на первых этапах становится заметно, что плохих отзывов намного меньше, чем хороших, что означает необходимость бороться с дисбалансом классов в дальнейшем.

Рис. 2 – Динамика количества отзывов по годам



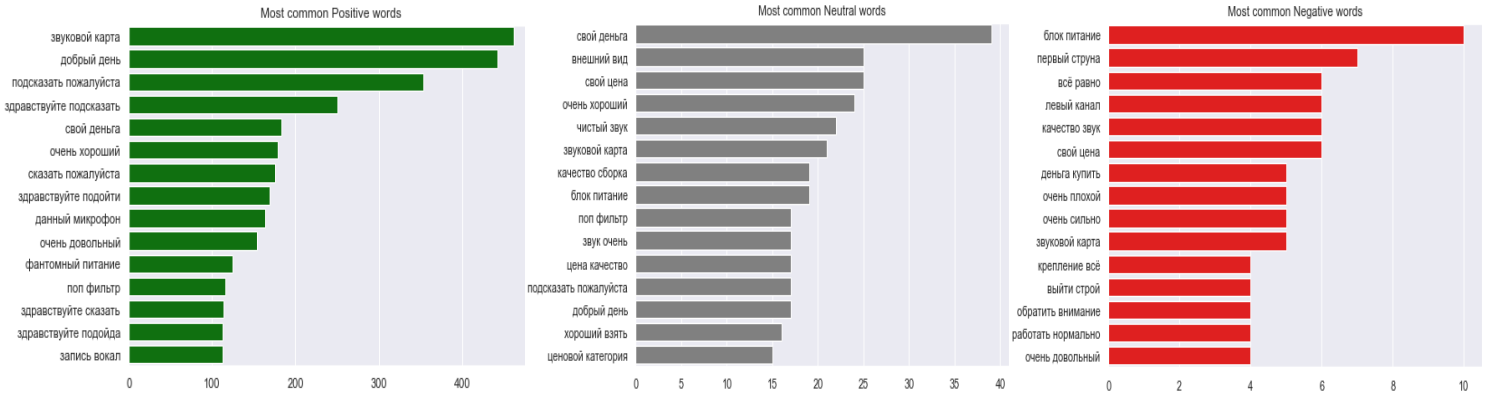
Большая часть отзывов короткие: средняя длина отзыва 28 слов, медианное значение – 16 слов на отзыв

Рис. 3 – Распределение количества слов в отзывах



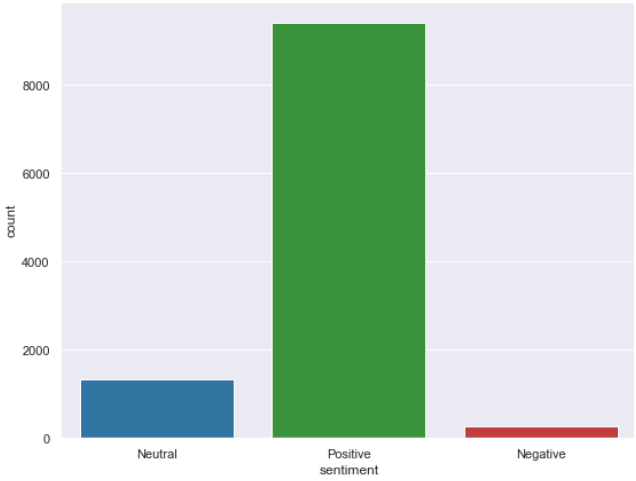
Тексты отзывов были очищены от стоп-слов и лемматизированы, после чего были рассмотрены наиболее часто встречающиеся в текстах отдельные слова и биграммы, на которых уже видно определенно эмоциональное распределение. Позитивные отзывы содержат много вежливых слов, в нейтральных упоминаются деньги и качество, негативные содержат характерные проблемы (выйти из строя).

Рис. 4 – самые частотные биграммы



Наконец, целевая переменная.

Рис. 5 – Распределение целевой переменной



Дисбаланс классов сильный. Прежде чем переходить к моделированию, желательно его уменьшить. Также такая картина говорит о том, что метрика Accuracy будет не информативна:

модель, всегда предсказывающая только Positive по этой метрике будет считаться эффективной, но бессмысленной с точки зрения практического применения.

В качестве ключевых метрик предполагается использовать:

- F1-score (для каждого из классов) как показатель, объединяющий в себе информацию о точности и полноте классификации

$$F = 2\frac{Precision \times Recall}{Precision + Recall}$$

- ROC AUC score (для каждого из классов) как комплексную меру, которую можно интерпретировать как вероятность, что модели удастся успешно разделить классы. Численно, эта метрика представляет из себя площадь под кривой зависимости TPR от FPR, где TPR – доля верных положительных классификаций, FPR – доля ложных положительных классификаций.

$$AUC = \int_0^1 TPR\, dFPR$$

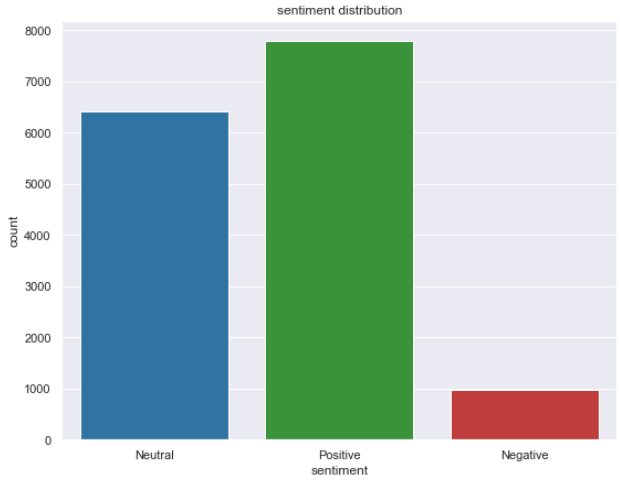
3.4 Обогащение исходного датасета

Готовя модель, мы ограничены предметной областью, но необязательно одним магазином в ней. Для того, чтобы уменьшить дисбаланс классов, в датасет были добавлены данные, собранные с другого сайта, а также из одного подходящего датасета на Kaggle. Так как в новых источниках данных наблюдается похожее соотношение оценок (плохих мало, хороших много), были взяты только отзывы из классов Neutral и Negative, чтобы не увеличить дисбаланс еще больше.

Так же был создан ряд искусственных отзывов путем применения своеобразной «аугментации» к объектам наименьшего класса: отзывы с низкими оценками переводились на иностранный язык (произвольно выбираемый из списка), а затем обратно на русский. В результате получался новый отзыв, несущий в себе ту же идею, но несколько иными словами. Многократно применять этот метод к одним и тем же отзывам бессмысленно, это приводит к появлению одинаковых текстов, но разовое воздействие добавляет в датасет определенное количество связных и уникальных текстов.

После подобных манипуляций и удаления возможных дублей в датасете останется порядка 15000 записей, а распределение целевой переменной будет выглядеть следующим образом.

Рис. 6 - распределение целевой переменной после добавления новых данных



Классы по-прежнему не сбалансированы, но разрыв заметно меньше.

4. Предобработка данных

Текущий датасет фактически не требовал работы с пропущенными значениями, т.к. в отсутствие текста или оценки, наиболее критичных для анализа признаков, запись в датасете не создавалась.

Для векторизации текстов использовалась модель Universal sentence encoder (USE) от Tensorflow. Это инструмент хорошо работает с недлинными текстами, сохраняя в векторном пространстве смысловую близость закодированных предложений. В результате работы USE каждый отзыв был представлен вектором из 512 значений.

На этом этапе была сделана первая попытка обучить классификаторы на полученных данных. Использовались различные модели из пакета sklearn. Результат они показали средний: наименьший класс с негативными отзывами определялся плохо.

Рис. 7 - Classification report модели SVC

SVC result

	precision	recall	f1-score	support
Negative	0.05	0.71	0.10	14
Neutral	0.77	0.70	0.73	1408
Positive	0.82	0.79	0.80	1611
accuracy			0.75	3033
macro avg	0.55	0.73	0.54	3033
weighted avg	0.79	0.75	0.77	3033

Из чего был сделан вывод о необходимости under- и oversampling-а.

До того, как применить сэмплинг к данным, от датасета были отделены 20% объектов - это будет выборка для проверки моделей уже после обучения, подаваемая как новые данные.

К оставшимся данным применяются:

1) К наибольшему классу – алгоритм EditedNearestNeighbours (ENN). ENN позволяет уменьшить количество объектов класса таким образом, чтобы получить кластеры объектов, более четко отделенные друг от друга. В случае с отзывами смысловое различие между объектами соседних классов может быть незначительным, соответственно и в векторном пространстве они будут находится в областях, накладывающихся друг на друга. Если убрать часть объектов на границе этих пересечений, модели будет легче классифицировать оставшиеся объекты.
2) К меньшим классам - алгоритм SMOTE, который создает искусственные объекты, рассчитывая значения их признаков на основе значений некоторого количества соседних объектов одного класса.

После этих преобразований получили 15375 объектов, распределенных равномерно по трем классам. Эти данные разбиваются на тренировочную и валидационную выборки.

5. Обучение моделей.

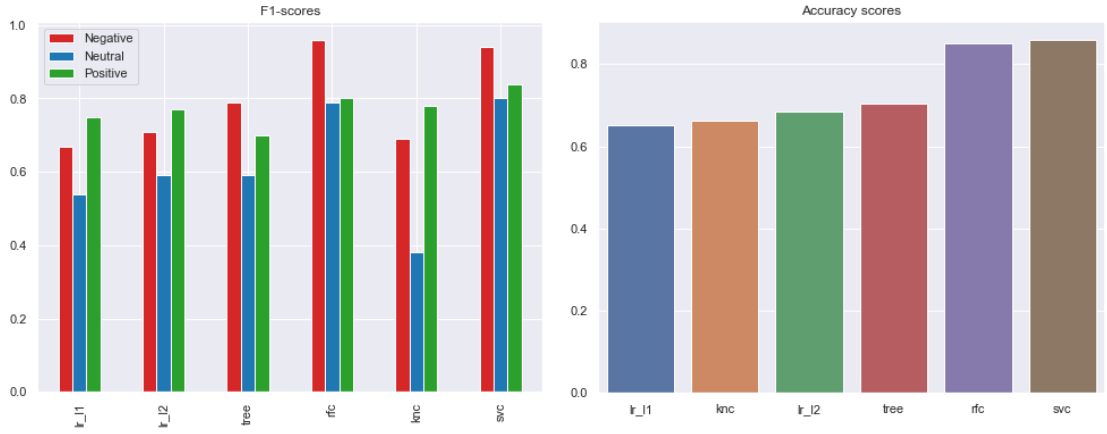
5.1 Классификаторы sklearn

На полученных данных обучаются модели:

- logistic regression (с вариантами l1 и l2 регуляризации)
- Decision tree classifier
- Random Forest Classifier
- K-neighbours classifier
- SVC

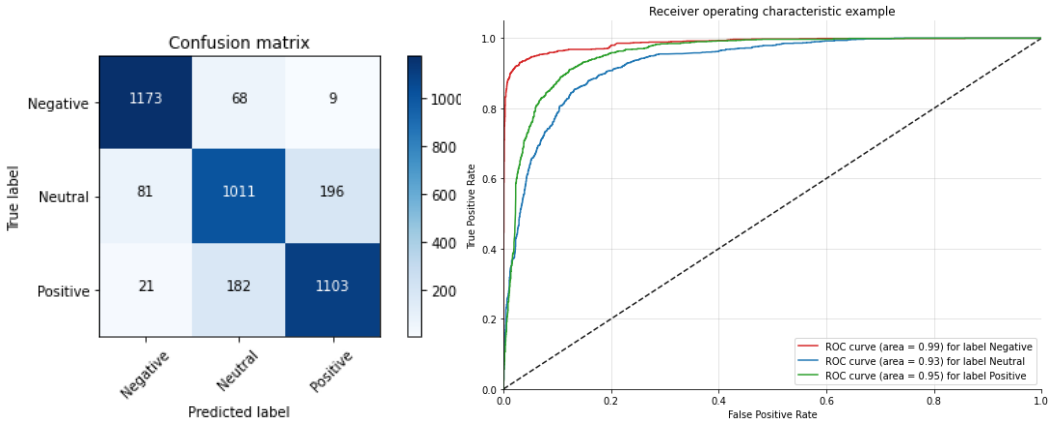
Наилучший результат по f-мерам и accuracy показывают RFC и SVC.

Рис. 8 - результаты классификаторов sklearn



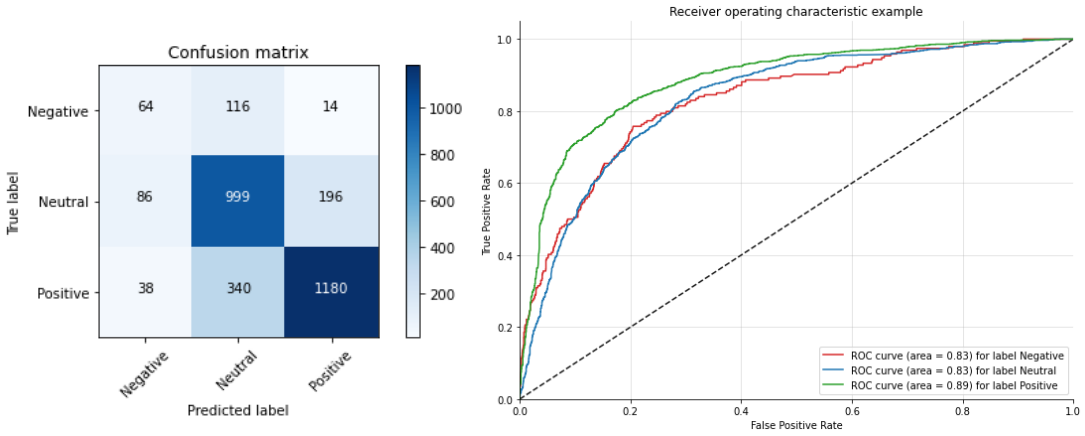
Для дальнейшего тюнинга выбраны классификатор SVC и RFC. Для этих классификаторов осуществлялся подбор гиперпараметров с помощью GridSearchCV. Однако для алгоритма случайного леса это не дало улучшения показателей, поэтому привожу здесь результаты после обучения SVC. После подбора гиперпараметров на валидационной выборке классификатор показывает достаточно хорошие результаты:

Рис. 9 - Матрица ошибок и ROC curve для SVC на валидации



На отложенной до сэмплинга выборке (“новых” данных) SVC показывает следующие результаты:

Рис. 10 - Матрица ошибок и ROC curve для SVC на отложенной выборке



Как видим, негативные отзывы определяются моделью не очень уверенно, и достаточно много объектов, ошибочно отнесенных к соседнему классу. Но количество ошибок, когда перепутаны тонально противоположные отзывы (позитивные и негативные), невелико.

5.2 Полносвязная нейросеть

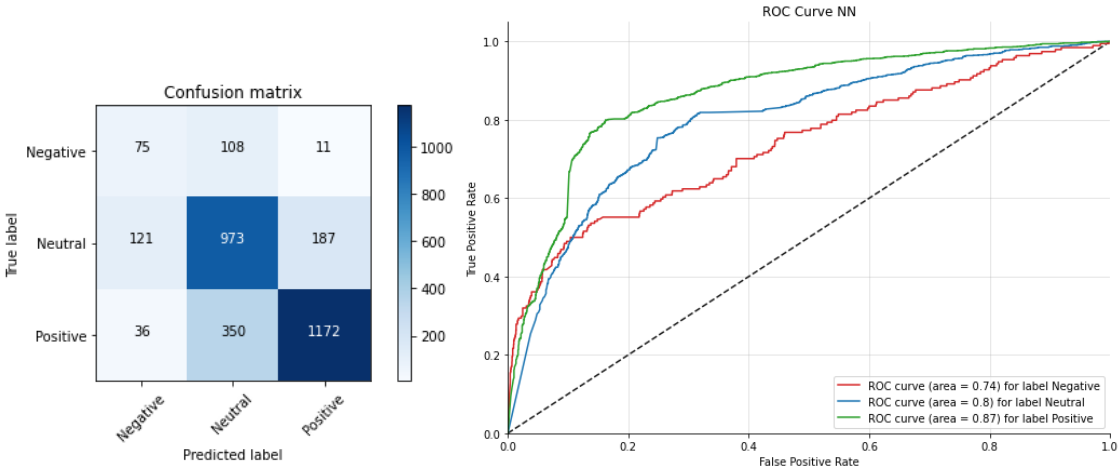
Для сравнения с обычными классификаторами была обучена простая полносвязная сеть из нескольких слоев.

Рис. 11 - структура сети

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 256)               131328

 dense_1 (Dense)             (None, 128)               32896

 dense_2 (Dense)             (None, 64)                8256

 dense_3 (Dense)             (None, 32)                2080

 dense_4 (Dense)             (None, 3)                 99

=================================================================
Total params: 174,659
Trainable params: 174,659
Non-trainable params: 0
```

Результаты её применения к отложенной выборке после обучения на рисунке ниже.

Рис. 12 - Матрица ошибок и ROC curve для NN на отложенной выборке



Результаты выглядят сопоставимо с SVC, но несколько хуже. С учетом того, что такая модель обучается дольше, чем классификатор sklearn, решение пользоваться ей выглядит нецелесообразным.

Из этой точки можно было бы двигаться в сторону усложнения архитектуры сети, например, добавления рекуррентных слоев (LSTM), и увеличения количества обучаемых параметров, однако более прагматичным подходом будет попробовать дообучить на своих данных существующую, более сложную модель, которая специализируется на подобных задачах.

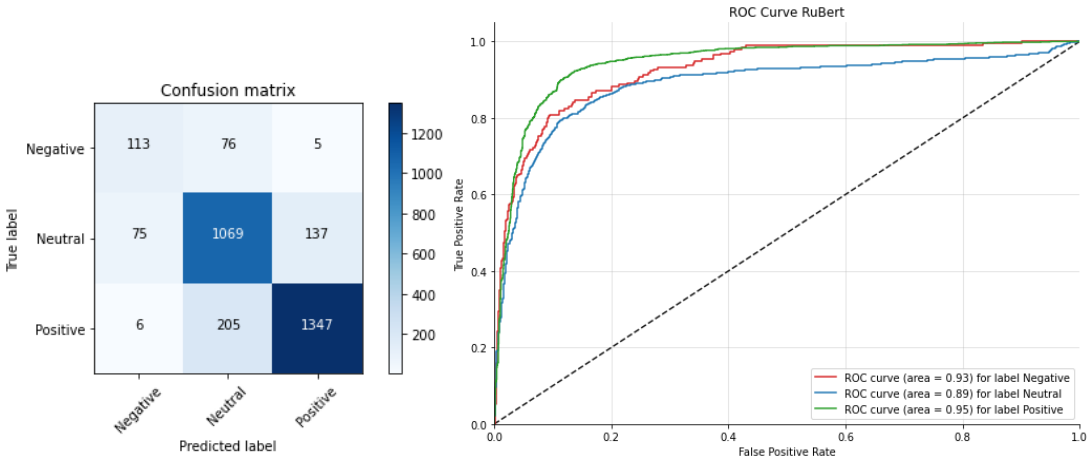
5.3 Применение RuBert

BERT от Google – нейросетевая модель-трансформер от Google, показавшая несколько лучших результатов в решении многих NLP-задач: от ответов на вопросы до машинного перевода.

RuBert от DeepPavlov - это адаптация BERT обученная на корпусе русскоязычных текстов.

В данной модели есть собственный механизм токенизации текстов, поэтому к данным применен именно он. Токенизатору подавались исходные тексты отзывов, не модифицированные USE и до оверсэмплинга. Аналогичным образом была создана отложенная выборка, составляющая 20% от исходных данных, для тестирования модели после её обучения.

Рис. 13 - Матрица ошибок и ROC curve для RuBert на отложенной выборке



RuBert по итогам обучения получила на отложенной выборке самые высокие результаты по всем показателям и наиболее качественно отделила объекты друг от друга. Ошибки на границах классов присутствуют, но в предсказании противоположных классов (позитивных и негативных отзывов), ошибок заметно меньше, чем в предыдущих моделях.

6. Сравнение моделей

Рис.14 - оценки ROC AUC и F1-score выбранных классификаторов

	ROC AUC			F1 score		
	Negative	Neutral	Positive	Negative	Neutral	Positive
SVC	0.83	0.83	0.89	0.34	0.73	0.80
NN	0.74	0.80	0.87	0.35	0.72	0.80
RuBert	0.93	0.89	0.95	0.58	0.81	0.88

Выводы:

- Наилучший результат показала модель RuBert. Однако минусы от её использования при таких убедительных результатах - долгое время обучения даже на небольшом количестве эпох и так же немалое время, необходимое на получение предсказания по новым данным.
- Полносвязная нейросеть - наименее удачный эксперимент в этой группе. Обучается дольше SVC, а результаты не лучше, чем у него.
- SVC - рабочий вариант среднего качества

7. Направления улучшения

- Более продуманная разметка исходных данных. Например, с эмоциональной оценкой подтвержденной разметчиком человеком, а не сгенерированная исходя из оценки. Возможно, следовало выделить еще один класс целевой переменной. В исходных данных содержится определенная доля отзывов с рейтингом 5, но содержащая только вопрос, относящийся к товару,

без какой-либо эмоциональной оценки. При выделении таких отзывов из общей массы позитивных в отдельную группу (“Question” или “Other”) исходный датасет давал бы более четкую картину настроений пользователей.

- Поиск новых данных для обучения, особенно отзывов с низкими оценками.
- Применение ансамблевых методов
- Применение других разновидностей Bert, обученных на разговорном русском, и увеличение количества эпох при обучении.

8. Заключение

Для реализации проекта были собраны данные из нескольких источников, тексты были предобработаны и токенизированы, для обучения устранен дисбаланс классов.

Были рассмотрены различные модели классификации, выбрано несколько наиболее релевантных.

Ссылки на источники

- https://vctr.media/how-to-react-to-feedback-14198
- https://dyakonov.org/2017/07/28/auc-roc-площадь-под-кривой-ошибок/
- https://pythonru.com/baza-znanij/sklearn-roc-auc
- http://bazhenov.me/blog/2012/07/21/classification-performance-evaluation.html
- https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/3
- http://docs.deeppavlov.ai/en/master/features/models/bert.html