Отчет о практической тестовой работе на тему

“Разработка прототипа веб-сервиса для классификации отзывов”

Работу выполнил: Кац Евгений

03.02.2022

I. Введение

В данной работе будет предложен прототип веб-сервиса, который позволит определить тональность отзыва и рейтинг, которым пользователь оценил фильм. В основе сервиса будет лежать обученная модель искусственного интеллекта для решения задачи машинного обучения в области обработки естественного языка (NLP), где на вход принимаются данные большой признаковой размерности, что в некоторой степени ограничивает функционал распространенных методов, и, как следствие требует специального подхода при предобработке и предварительном анализе. В рассматриваемом наборе данных, содержится информация об обзоре, тональности обзора и оценке, приведенной к 10-и бальной шкале. При этом, оценки со значением 5,6 были устранены создателями набора данных, как “нейтральные”.

II. Постановка задачи

Целью данной работы является разработка прототипа веб-сервиса, который сможет выступать в роли API для других проектов. Для разработки такого сервиса, необходимо решить следующие подзадачи:

- Реализовать процесс предобработки (форматирования) входных данных - формирование такой выборки, к которой впоследствии можно будет применять методы предобработки и методы машинного обучения.
- Провести предварительный анализ данных, определить классовые распределения по выборке, выдвинуть некоторые гипотезы.
- Для задачи определения тональности отзыва, необходимо подобрать алгоритм, гипер-параметры и обучить модель, после чего провести процесс тестирования качества обученной модели.
- Для задачи выставления рейтинга фильма на основе оставленного комментария, необходимо разработать алгоритм обучения модели. После чего выбрать алгоритм и подобрать гипер-параметры и провести тестирование качества модели.
- Определить технологии, которые позволят выгрузить обученные модели в общий доступ в качестве прототипа веб-сервиса и произвести выгрузку.

III. Методы решения

1. Предобработка входных данных

Поскольку входные данные имеют неудобный для работы формат, необходимо провести форматирование и составить приемлемый набор данных. С помощью реализованного функционала (функция get_data), входные данные были преобразованы в следующий формат - для тестовой и тренировочной выборок было создано по три кадра, X_(train/test), содержащий строки с содержанием текстового отзыва на фильм, y1_tone_(train/test), содержащий первый тип меток - тональность отзыва (положительный или отрицательный) и y2_score_(train/test), содержащий рейтинг присвоенный фильму от пользователя (от 1 до 10). Таким образом, после форматирования был получен готовый к работе набор данных.

2. Предварительный анализ

Предварительный анализ данных включает в себя, в первую очередь, анализ распределения объектов в выборках. Согласно ТЗ, разбиение на тренировочную и тестовую выборку уже проведено, и распределение позитивных и негативных отзывов в этих наборах равномерно и идентично (12.500 наблюдений каждого типа для тренировочной и тестовой выборок). С другой, согласно Рис. 1. На котором изображена частотное распределение объектов по классам (рейтингу) отзывов, при сохранении одинаковых распределений, можно отметить преобладание объектов самых полярных классов - 10 и 1, в среднем, данные классы наполнены в два раза сильнее, относительно остальных классов, которые наполнены более равномерно.

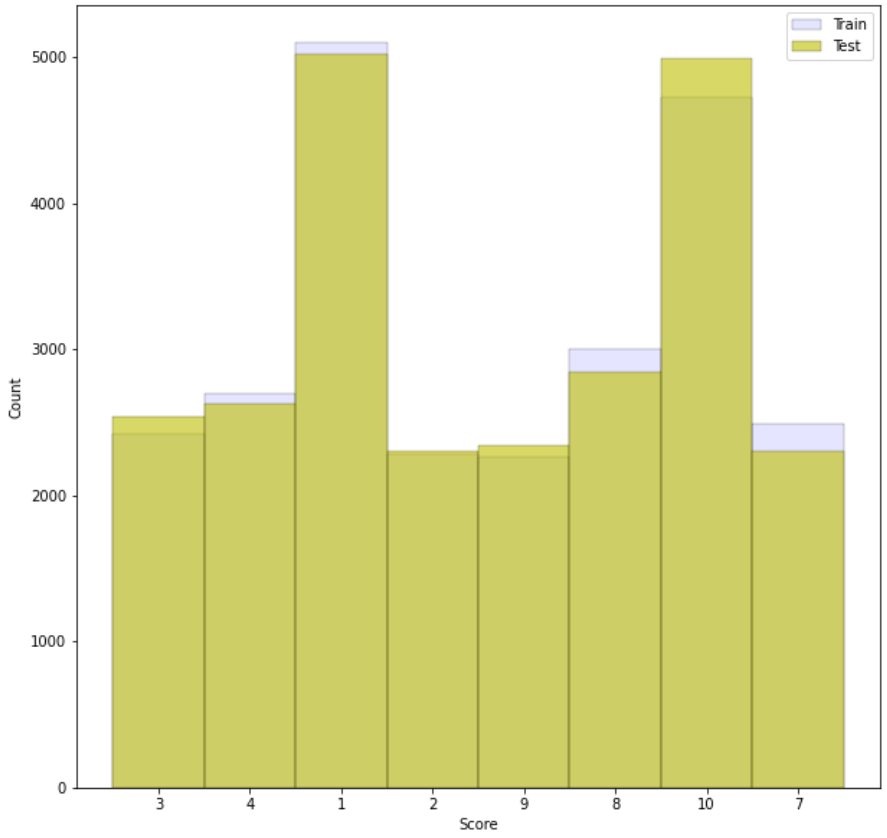

Рис. 1. Наполненность классов по выборкам

Исходя из вышесказанного, можно выдвинуть некоторые предположения относительно данной задачи -

- Люди чаще оставляют отзывы и обзоры на фильмы, когда фильм им очень понравился или наоборот - очень не понравился. Данное предположение, подтверждается на Рис. 1. - как видно, “полярных” обзоров в ~2.5 раза больше, чем любых других.
- Несмотря на возможность уточнить впечатления о фильме используя шкалу, большинство людей предпочитает оценивать фильмы бинарным образом - понравился/не понравился. Данное предположение, в целом, конкурирует с предыдущим и так же может быть подтверждено с помощью анализа Рис.1. Требуются дополнительные исследования.
- Отличия в смысле обзоров с похожими финальными оценками, (2 балла или 1 балл, 10 баллов или 9 баллов) незначительны. Данное предположение будет рассмотрено при анализе работы моделей при предсказании тональности и рейтинга обзоров.
- Резко-негативные и резко-позитивные обзоры, в среднем длиннее, чем нейтральные. Данное предположение подтверждено исследованием, в среднем, “полярные” отзывы на 170 знаков длиннее остальных.
- Метка тональности обзора может положительно повлиять на определение рейтинга всего обзора.

Для работы с текстом, необходимо провести некоторую предобработку входных данных. Для того, чтобы модель машинного обучения могла использовать входные данные и получить некую обобщающую способность при обучении, были проделаны два основных этапа - стемминг, токенизация и векторизация текста.

Стимминг текста необходим для удаления лишней информации из входных данных. Зачастую, такие особенности слов как род, падеж, склонение, время, а также знаки препинания, стоп-слова (слова-паразиты и пр.) и прочее, не несут, или почти не несут полезной информации для модели, поэтому в данной работе стемминг был применен ко всех входным данным. Для стемминга текста, был использован функционал пакета nltk, предоставляющий весь необходимый инструментарий для удаления стоп-слов, пунктуации и форм слов. Стоит отметить, что существует более “продвинутый” алгоритм приведения слов к нормальный форме - лемматизация. Данный алгоритм, в отличии от стемминга не прости “обрезает” слова, а приводит слова к подлинной нормальной форме. Однако, данный алгоритм требует больше времени и вычислительных мощностей, поэтому выбор в работе пал на более простую версию - стемминг.

Токенизация (разбиение текста на ячейки), в свою очередь необходима при решении данной задачи, поскольку наблюдение той или иной ячейки (слова) в тексте, в некоторых случаях имеет высокую значимость, и как следствие - высокое влияние на работу модели.

Векторизация необходима для перевода получившегося набора “признаков” в числовой формат, чтобы модель впоследствии могла обучаться на предоставленных данных. В данной работе в качестве векторизации был использован функционал sklearn – TFidfVectorizer . Как аналоги, были испытаны такие алгоритмы как Bag Of Words, однако результат оказался хуже.

3. Задача определения тональности

В данной задаче необходимо построить модель, которая позволит определить тональность обзора, который пользователь опубликует о некотором фильме. Для решения данной задачи, был использован функционал библиотеки sklearn - LogisticRegression. В ходе работы, с использованием функционала GridSearch, были подобраны гипер-параметры для данного алгоритма, и была обучена модель. Поскольку, в данной задаче классы равно представлены, можно сделать исчерпывающие выводы о работе модели по высчитыванию стандартных характеристик, таких как accuracy и построения матрицы ошибок. Для оценки качества модели, была построена матрица ошибок (confusion matrix) Рис.2. И высчитаны числовые метрики - Таб. 1.

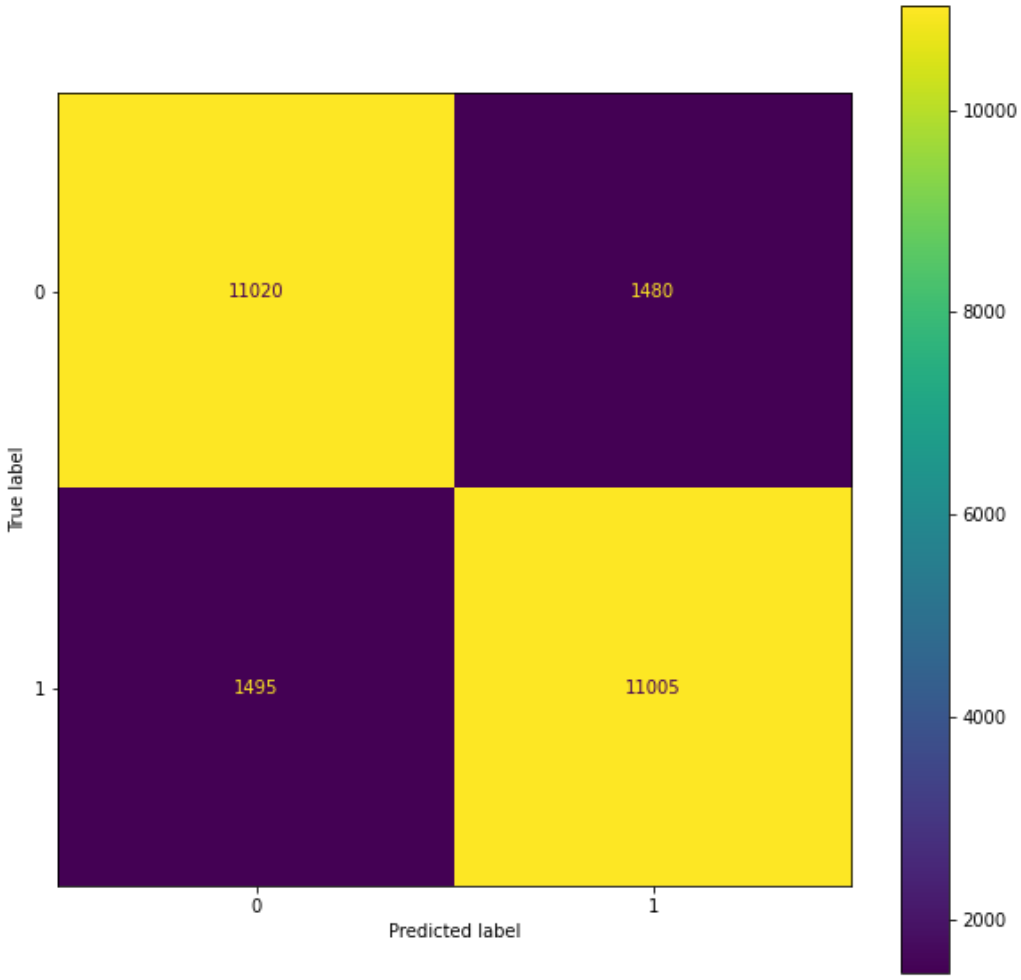

Рис. 2. Матрица ошибок при определении тональности текста

Как видно на Рис. 2. Оба класса (позитивный тон и негативный тон) классифицируются равномерно, при этом метрика Accuracy составила 0.88, что является достаточно неплохим результатом для такого базового алгоритма, как логистическая регрессия.

Таблица 1.

Оценка качества работы модели классификации

Accuracy	0.88
Precision	0.8804
Recall	0.8810

Дополнительно, была построена ROC (receiver operating characteristic) кривая, для дополнительного анализа качества работы модели Рис. 3.

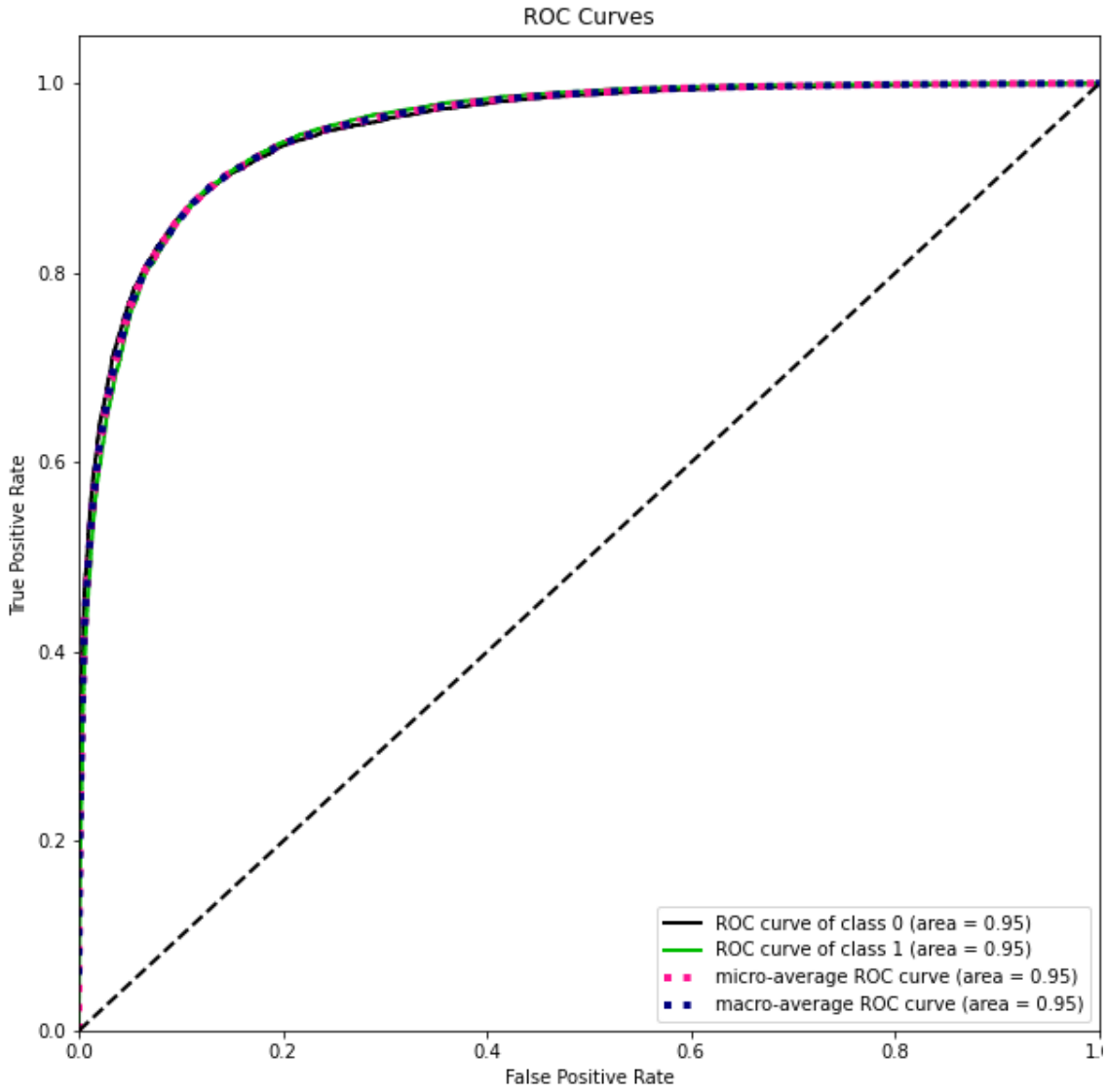

Рис 3. график ROC AUC

Согласно Рис. 3. Обученная модель для решения задачи определения тональности имеет высокую разделяющую способность, что говорит о достаточно виском качестве построенной модели.

4. Задача определения рейтинга

Для решения данной задачи, необходимо построить модель, с помощью которой будет возможно классифицировать принадлежность отзыва к одному из 8 и классов, каждый из которых соответствует оценке, который пользователь предположительно готов дать. Для решения данной задачи, был использован алгоритм классификация LogistcRegression, для которого, как и в предудщей задаче были подобраны гипер-параметры с использованием функционала GridSearchCV. Однако, согласно Рис.1. в данной задаче классы неравномерно наполнены, с перевесом в сторону “полярных” классов, необходимо осторожнее анализировать предложенные для предыдущей задачи метрики.

Тем не менее, для рассмотрения гипотезы об незначительном отличии смысловой нагрузке отзывов с “близкими” оценками, необходимо построить confusion matrix Рис.4.

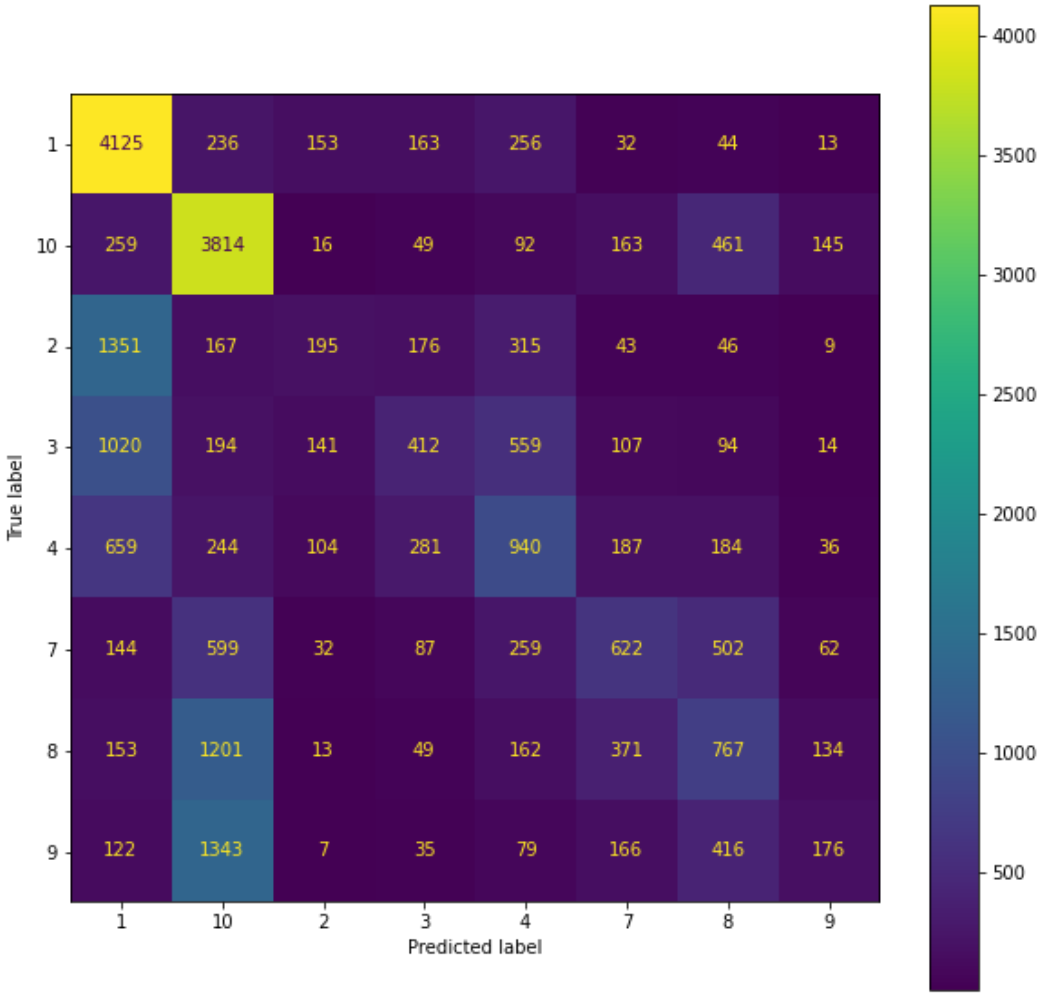

Рис. 4. Матрица ошибок в задаче классификации рейтинга отзывов

Как можно заметить по Рис.4. - лучше всего определяются отзывы с рейтингом 1 и 10. Это связано с тем, что данные классы наиболее наполнены. Одако, можно отметить, что отзывы с рейтингом построенная модель чаще ошибочно относит отзывы с рейтингом 2 и 3 к классу 1, отзывы с рейтингом 8 и 9 к классу 10, чем более “нейтральные” отзывы с меткой 4 и 7. Это частично подтверждает третье предположение о “схожести” отзывов в соседних классах.

Согласно предположению о том, что наличие метки тональности обзора может положительно сказаться на качестве классификации рейтинга, были проведены исследования по добавлению меток тональности в обучающую выборку, однако положительной динамики данное изменение не дало, следственно - данное предположение опровергнуто. Это связано с тем, что модель достаточно хорошо определяет “негативные” и “позитивные” обзоры по самому тексту, но внутри данных гипер-классов имеет сложности с разделением классов между собой. Для подтверждения этого и дополнительной проверки качества модели была построена попарная ROC кривая Рис. 5.

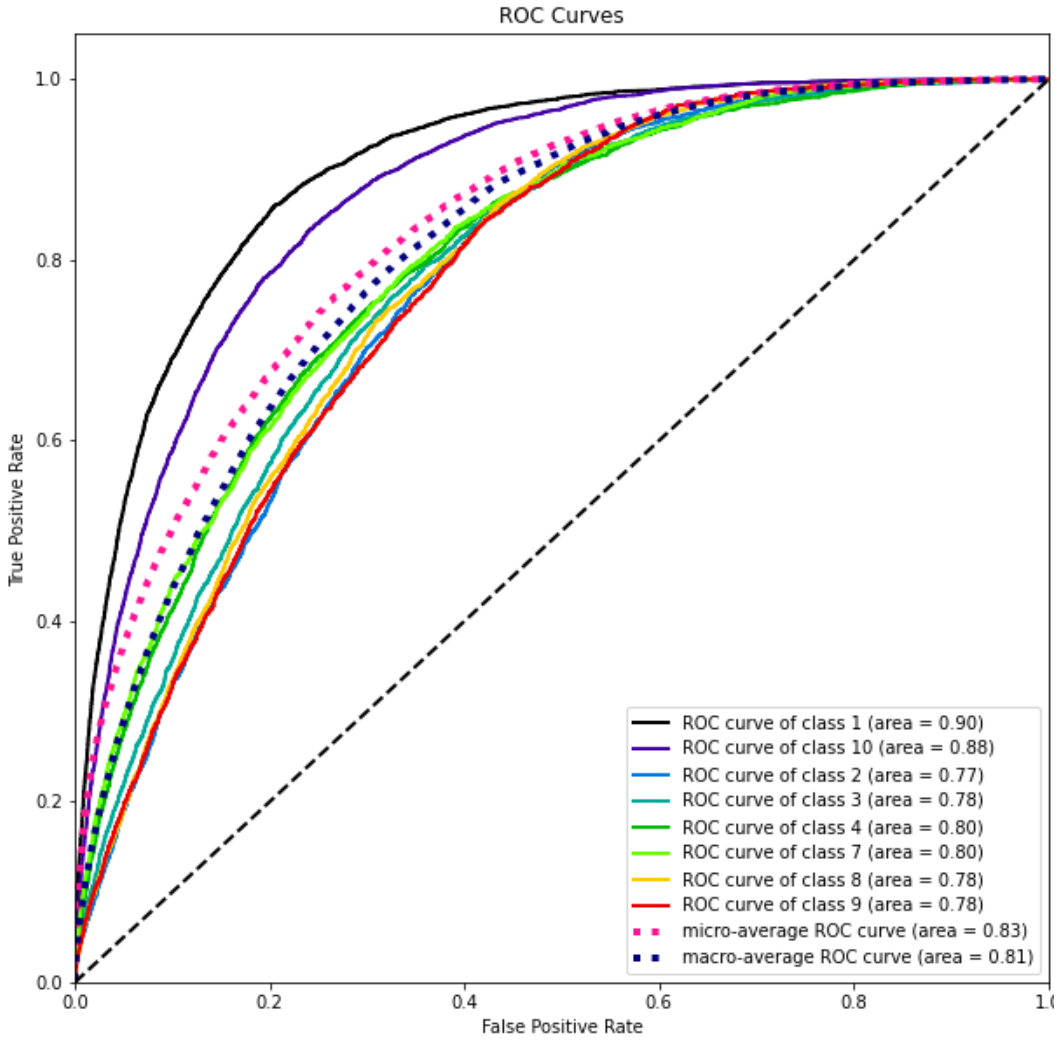

Рис. 5. ROC кривые для каждого класса.

Как видно по графикам на Рис. 5, классы 1 и 10, действительно лучше отделяются друг от друга и от остальных классов, при этом общее качество модели классификации в данном случае, можно назвать удовлетворительной, поскольку некоторые обобщающие и разделяющие способности модель имеет, но при этому испытывает сложности с разделением некоторых классов между собой.

5. Выгрузка модели и результаты работы

Согласно поставленной цели, необходимо выгрузить обученные модели машинного обучения в общий доступ в виде прототипа веб-сервиса с использованием фреймворка Django. С помощью функционала пакета Pandas Pickle, обученные модели были сериализованы и использованы в веб-приложении для автоматического присвоения индикатора тональности и рейтинга для загружаемого отзыва. Словарь, сформированный в ходе препроцессинга был также сериализован, что позволило использвать функционал TFidf векторизатора внутри приложения. Готовое приложение было выгружено на облачную платформу Heroku для обеспечения свободного доступа к данному приложеню. Для использования приложения, необходимо воспользоваться

ссылкой https://imdbfilmreviewclassifier.herokuapp.com/ где можно вручную проверить работоспособность модели, или использовать данное приложение как API.