Анализ тональности отзывов

В этом проекте вам предстоит решать задачи анализа тональности текстов в нескольких разных постановках.

Начнется все с несложных экспериментов на отзывах на фильмы, в ходе которых вы построите простую модель и немного доработаете ее. Затем вам предстоит поучаствовать в соревновании по сентимент-анализу отзывов на товары и сделать интерактивную демонстрацию для своего алгоритма, которую можно будет показать даже тем, кто никогда не видел Python и машинное обучение. После этого вы поупражняетесь в парсинге веб-страниц и столкнетесь с жестокой реальностью - к вам придет заказчик, который захочет от вас сентимент-анализ отзывов на определенную категорию товаров, но вот выборку для обучения придется собирать самим. Как и угадывать, что же заказчик называет негативными отзывами, а что позитивными. В конце проекта вам потребуется сделать демонстрацию и для этого алгоритма, чтобы заказчик мог с ним поиграться, а не только лишь ориентироваться на качество из контеста.

Неделя 1

В этом задании вам предлагается начать разбираться с задачей анализа тональности отзывов на примере сентимент-анализа отзывов на фильмы.

Мы будем использовать стандартный датасет из nltk, уже возникавший в одном из примеров в предыдущих курсах. Для того, чтобы импортировать необходимый модуль, напишите:

```
from nltk.corpus import movie_reviews
```

Чтобы получить id-шники негативных и позитивных отзывов:

```
negids = movie_reviews.fileids('neg')

posids = movie_reviews.fileids('pos')
```

Чтобы получить список негативных отзывов:

```
negfeats = [movie_reviews.words(fileids=[f]) for f in negids]
```

Инструкция по выполнению

В некоторых пунктах нужно получить ответ - число или строку, которые будет нужно набирать в текстовых файлах и прикреплять в ответах на вопросы. *Десятичные дроби записывайте через точку.*

1. Создайте список из текстов всех имеющихся отзывов, а также список с классами, которые будет использовать ваш классификатор - 0 для негативных отзывов и 1 для позитивных.
2. Подсчитайте количество отзывов в выборке.
3. Подсчитайте долю класса 1 в выборке.
4. Импортируйте CountVectorizer из sklearn.feature_extraction.text. Попробуйте использовать его с настройками по умолчанию для того, чтобы получить признаковое представление каждого текста. Скорее всего, попытка не увенчается успехом. Разберитесь, в чем причина, и добейтесь того, чтобы метод fit_transform у CountVectorizer успешно отрабатывал. Подсчитайте количество признаков в CountVectorizer. Никакой предварительной обработки

текста (удаление стоп-слов, нормализация слов) на этом шаге делать не надо, в качестве признаков должны использоваться частоты слов.

5. Соберите pipeline из CountVectorizer и LogisticRegression с настройками по-умолчанию и с помощью cross_val_score (также со стандартными настройками) оцените получаемое "из коробки" качество по accuracy.
6. Аналогично accuracy, оцените качество по ROC AUC.
7. Обучите логистическую регрессию на всей доступной вам выборке и выведите 5 наиболее важных для модели признаков (подумайте, какие именно признаки стоит считать такими). Вам могут пригодиться метод get_feature_names() или поле vocabulary_ у класса CountVectorizer.

Неделя 2

В этом задании вам предстоит поэкспериментировать с параметрами вашей модели для сентимент-анализа. Все задания выполняются на том же датасете, что и на прошлой неделе.

Инструкции

1. Здесь и далее оценка качества будет выполняться с помощью cross_val_score с cv=5 и остальными параметрами по умолчанию. Оцените среднее качество (.mean()) и стандартное отклонение (.std()) по fold'ам для: а) pipeline из CountVectorizer() и LogisticRegression(), б) pipeline из TfidfVectorizer() и LogisticRegression(). В соответствующем пункте задания выпишите через пробел среднее в п. а, отклонение в п. а, среднее в п.б и отклонение в п. б
2. Попробуйте позадавать параметр min_df у CountVectorizer. Оцените качество вашего классикатора с min_df=10 и с min_df=50.
3. Поперебирайте классификатор, используемый после CountVectorizer. И vectorizer и классификатор берите с параметрами по умолчанию. Сравните результаты для LogisticRegression, LinearSVC и SGDClassifier. Выпишите в ответе на соответствующий вопрос самое худшее качество из получившихся.
4. Подготовьте список стоп-слов с помощью nltk.corpus.stopwords.words('english'), посмотрите на его элементы, и передайте его в соответствующий параметр CountVectorizer. В sklearn также предусмотрен свой список английских стоп-слов - для этого нужно задать соответствующий параметр равным строке 'english'. Оцените качество классификатора в одном и другом случае и выпишете сначала качество в первом варианте, затем во втором в соответствующем вопросе.
5. Попробуйте в CountVectorizer добавить к словам биграммы и измерить качество модели. А затем постройте модель на частотах буквенных n-грамм с n от 3 до 5, указав соответствующее значение параметра ngram_range и параметр analyzer='char_wb'. Полученные два числа запишите через пробел в ответе на соответствующий вопрос.

Неделя 3

В этом задании вам нужно воспользоваться опытом предыдущих недель, чтобы побить бейзлайн в соревновании по сентимент-анализу отзывов на товары на Kaggle Inclass:

https://inclass.kaggle.com/c/product-reviews-sentiment-analysis-light

В качестве ответа в этом задании вам нужно подготовить ноутбук с решением и скриншот вашего результата на leaderboard.

Убедитесь, что:

1) ход вашего решения задокументирован достаточно подробно для того, чтобы ваши сокурсники поняли, что вы делали и почему,

2) ваша команда в соревновании состоит только из вас и названа вашим логином на Coursera, чтобы ваши сокурсники могли понять, что на скриншоте именно ваш результат

Неделя 4

Часто оказывается полезно представить результаты вашей работы в виде интерактивной демонстрации, в которую можно будет вводить какие-то данные и получать результат работы вашей модели. В этом задании вам предлагается реализовать такую демонстрацию для вашего алгоритма анализа тональности отзывов на фильмы, используя flask.

Даже если вы никогда не сталкивались с веб-программированием и flask'ом - познакомиться с ним совершенно не сложно. Сделать первые шаги вам поможет прекрасный Викиучебник по Flask: https://ru.wikibooks.org/wiki/Flask

Разработку демки для вашей задачи существенно упростит архив кода для аналогичной демки в аттаче (с которым все равно проще разбираться после основ Flask):

simple_demo.zip

В процессе вы столкнетесь с необходимостью сохранять в файл и загружать питоновский объект - обученный классификатор. Для этого вам пригодится библиотека pickle.

Неделя 5

В этом задании вы потренеруетесь парсить веб-страницы. Это умение пригодится вам, когда возникнет потребность самостоятельно собрать выборку для построения модели, будь то тексты или какая-та другая информация с общедоступных веб-страниц. В рамках данного проекта это случится уже в задании следующей недели.

Инструкция

1. Изучите блокнот с примерами парсинга страниц (в аттаче)
2. Поэкспериментируйте с парсингом любой страницы на ваш вкус, бегло просмотрите документацию библиотек beautiful soup и requests. При парсинге вам в любом случае потребуется смотреть на html-код страницы, чтобы понять, какие элементы вас интересуют.
3. Чтобы продемонстрировать, что вы разобрались, как использовать requests и beautiful soup, распарсите: а) из статьи https://en.wikipedia.org/wiki/Bias-variance_tradeoff все заголовки верхнего уровня; б) со

страницы https://en.wikipedia.org/wiki/Category:Machine_learning_algorithms названия всех статей в категории Machine Learning Algorithms

PageParsing.ipynb

Неделя 6

К вашей компании пришел заказчик, которому нужно решение задачи анализа тональности отзывов на товары. Заказчик хочет, чтобы вы оценили возможное качество работы такого алгоритма на небольшой тестовой выборке. При этом больше никаких данных вам не предоставляется. Требуется, чтобы качество работы вашего алгоритма (по accuracy) было строго больше 85%.

Оценка качества в этом задании реализована через контест на Kaggle Inclass:

https://inclass.kaggle.com/c/product-reviews-sentiment-analysis

Вам предстоит посмотреть на предоставленные заказчиком отзывы, собрать похожие отзывы в качестве обучающей выборки, и поэкспериментировать с постановкой задачи (разметкой вашей выборки на позитивные и негативные примеры) так, чтобы результат на примерах заказчика был по возможности получше.

Обратите внимание, что заказчик предоставил всего 100 примеров в качестве тестовой выборки - ситуация, когда размеченных данных почти нет - вообще очень частая в индустриальном анализе данных. Конечно, эти отзывы можно было бы идеально разметить вручную и получить максимальное качество, но вы сами не заинтересованы в таком подходе, т.к. потом придется и на всех новых примерах демонстрировать заказчику идеальную работу, что, конечно, вряд ли будет по силам алгоритму. В любом случае рано или поздно алгоритм придется разрабатывать, поэтому попытки "сжульничать" и не делать никакой модели не одобряются.

В качестве ответа в этом задании вам нужно подготовить ноутбук с решением и скриншот вашего результата на leaderboard.

Убедитесь, что:

1) ход вашего решения задокументирован достаточно подробно для того, чтобы ваши сокурсники поняли, что вы делали и почему,

2) ваша команда в соревновании состоит только из вас и названа вашим логином на Coursera, чтобы ваши сокурсники могли понять, что на скриншоте именно ваш результат

Неделя 7

На этой неделе вы в некотором смысле поведете итог вашей работы над проектом - реализуете демонстрацию для алгоритма, который не просто решает задачу анализа тональности отзывов, а еще и обучен на собранной лично вами выборке, а также прошел тест на небольшом наборе примеров от заказчика.

Адаптируйте вашу демонстрацию из предыдущих недель под новый алгоритм и подготовьте несколько новых примеров (теперь уже с ним)

Опциональное неоцениваемое задание:

Попробуйте в общих чертах разобраться с boostrap или готовыми шаблонами веб-страничек и поэкспериментировать с оформлением вашей демки.