АЛГОРИТМ АКТИВНОГО ОБУЧЕНИЯ С ИСПОЛЬЗОВАНИЕМ КЛАСТЕРИЗАЦИИ ДЛЯ ВЫБОРА СТАРТОВОГО МНОЖЕСТВА

Цветков Владимир Вадимович (tsvvladimir95@gmail.com)
МГТУ им. Н.Э. Баумана, г. Москва, Россия

Abstract
We present a method for active learning. We focus on selecting an initial training set and on stopping criteria. The presented classification task is a multiclass classification, we use the one vs all scheme for training the svm classifier. The initial training set is chosen by clustering an unlabeled document set. We use the sigmoid function to transform distances produced by svm to the document per class probabilities. These probabilities are used for stopping criteria. The results received after testing this algorithm on the 20 news group dataset with crossvalidation confirmed the idea of active learning.

active learning, semisupervized learning, text classification, clusterization, one vs all, svm, k-means, sigmoid function

1 Введение

Задача классификации документов состоит в отнесении документа к одному из заранее заданных наборов классов документов. Она находит широкое приложение в таких областях, как составление каталогов, фильтрация спама, подбор контекстной рекламы и др. Существуют разные подходы к классификации документов: ручной, основанный на правилах и с помощью машинного обучения [1]. Рассмотрим третий подход. Машинное обучение можно разделить на обучение без учителя, обучение с учителем и частичное обучение с учителем [2]. Последние два подхода требуют наличие размеченной коллекции документов. Коллекции документов из реальной жизни могут быть довольно значительными по объему. Это может вызвать трудности с их разметкой, поскольку требует задействования человеческих ресурсов. Сократить затрачиваемые ресурсы позволяют алгоритмы активного обучения [3].

В литературе рассматриваются разные идеи активного обучения: на основе анализа запроса, выбор документа из потока документов и выбор документа из пула документов. Алгоритм на основе анализа запроса означает, что модель сама находит документы или генерирует их, которые после будут переданы эксперту для разметки [4]. В алгоритме выбора из потока документов, модель для каждого выбранного документа решает нужно ли его отдавать эксперту или нет [3]. Алгоритм выбора из пула документов работает в предположении, что у нас есть большой набор неразмеченных документов и из них необходимо выбрать те, которые требуется классифицировать [3].

Каждый из этих сценариев требует определения, какие документы включать в новую обучающую выборку. Для этого также существует несколько подходов: выбор по вероятности и выбор с помощью ансамбля классификаторов. Выбор по вероятности означает, что в обучающую выборку будут добавлены те документы, которые изменят какие-либо вероятностные характеристики модели определенным образом. Например, можно оценивать ожидаемое изменение ошибки классификатора, при условии, что этот документ уже был бы в нашей обучающей выборке [3]. Идея выбора с помощью ансамбля классификаторов означает, что поддерживается несколько моделей, каждая из которых работает независимо. Добавлен будет тот документ, который будет уменьшать их разногласие [3].

Разработанный алгоритм включает в себя препроцессинг входных данных и сам алгоритм активного обучения, который уточняет некоторые шаги работы [5]. Также будет проведено тестирование алгоритма, сравнение с наивной версией алгоритма и представлены дальнейшие варианты улучшения. Метрикой качества работы классификатора — будет F1 – мера [6], адаптированная для мультиклассовой классификации через макро усреднение, то

есть усреднение значений меры для каждого класса (формула 1, где P – среднее арифметическое значение точности по каждому классу и R – среднее арифметическое значение полноты по каждому классу):

$$F1 = \frac{2 * P * R}{P + R}$$

Эффективность способа выбора начального множества для алгоритма активного обучения демонстрируется с помощью метрики — площадь под кривой обучения, описанной ниже.

2 Входные данные

Входными данными для работы алгоритма, порождающего классификатор, является большой неразмеченный набор документов, список категорий и оракул, способный размечать документы, поданные на вход. Входными данными для порожденного классификатора является документ или набор документов, который будет отнесен классификатором к какому-либо классу.

2.1 Предобработка входных данных

Поскольку тексты могут поступать из разных источников, то необходимо произвести их очистку. Для этого выполняется отсечение пунктуации, удаление слов из стоп-списка, стемминг слов в документах. Методы, которые применяются во время работы алгоритма, основываются на представлении документов в виде числовых векторов. В данной работе используется распространенное векторное представление текстов на базе tf-idf [7].

3 Алгоритм активного обучения

Существует несколько подходов к активному обучению. В данной статье будет рассмотрен подход с сэмплированием по вероятности. Идея такого подхода заключается в том, что на каждой итерации построения нового классификатора с помощью активного обучения, он тренируется на наборе данных, в который добавлены элементы в которых текущий построенный классификатор наименее уверен. Алгоритм представлен на схеме 1.

Алгоритм: активное обучение классификатора.
Входные данные: небольшой начальный набор документов L, набор неразмеченных документов U, необученный классификатор C.

Выходные данные: обученный классификатор C.

а. Разметка начального набора данных и его использование для обучения классификатора C.
б. Проверка на соответствие классификатора C критерию остановки алгоритма. Если он выполняется, то выход из алгоритма.
в. Разметка документов из набора U с помощью полученного классификатора.
г. Использование метода сэмплирования по вероятности, для выбора данных, которые будут добавлены к набору L из набора U.
д. Разметка этого набора данных и добавление его к набору L.
е. Использование обновленного набора L для переобучения классификатора C.
ж. Переход на шаг б.

Схема 1. Алгоритм активного обучения с сэмплированием по вероятности

Стратегия добавления документов шаге г исследовалась в [5]. В представленном в текущей статье алгоритме будет добавлено рассмотрение и осмысленный выбор начального множества документов и также будет предложен новый критерий остановки алгоритма. Ниже приведены подробности реализации алгоритма.

3.1 Выбор начального множества

Предполагается, что обучающий набор данных содержит примерно в равной степени документы из каждого класса. Мы хотим выбрать начальный набор документов так, чтобы он наиболее ярко характеризовал все возможные классы, с которым классификатору придется работать. Для этого используется такая техника обучения без учителя, как кластеризация.

Производим разбиение неразмеченного набора данных на кластеры, количество которых соответствует количеству классов в коллекции. Для этого используется метод кластеризации — k – средних [8]. Он позволяет разбить коллекцию на кластеры, которые будут примерно одинаковы по размеру. После проведения кластеризации мы имеем набор точек — центров полученных кластеров, но эти точки могут не являться документом из набора. Для решения этой проблемы используем меру близости нашего пространства, для того чтобы найти документы, наиболее близкие к центроидам кластеров. Эти наиболее близкие документы и будут образовывать начальное множество.

3.2 Используемый классификатор

Метод опорных векторов широко применяется для классификации текстов. Предполагается, что пространство документов линейно разделимо, т.е. между документами разных классов можно провести гиперплоскость в пространстве документов. Метод опорных векторов позволяет максимизировать расстояние между различными классами [9]. Изначально метод опорных векторов был предложен для бинарной классификации, однако существует его модификации, позволяющие проводить мульти классовую классификацию. Это значит, что существует более чем два класса документов и каждый документ может быть отнесен к одному из этих классов. При решении поставленной задачи используется принцип «один против всех» [5]. Для каждого класса создается классификатор, который осуществляет бинарную классификацию по следующему принципу — он считает, что все документы, которые не относятся к «своему» классу — другой класс. Ансамбль этих классификаторов позволяет, путем выставления оценки документу, определить, к какому классу относится поданный на вход документ.

3.3 Стратегия добавления

Стратегия добавления — сэмплирование по вероятности подразумевает под собой следующее. Несмотря на то, что используется стратегия один против всех, где каждый класс имеет свой классификатор, который классифицирует, принадлежит ли документ «своему» или «чужим» классам, все разделенное пространство документов можно представить, как будто оно разделено границами, которыми являются опорные вектора каждого класса. Для того, чтобы обновленная выборка улучшила качество классификации, будем считать, что она должна сделать классификатор более точным для определения границы классов [10]. Для этого будем добавлять элементы, которые находятся максимально близко к этой границе. Учитывая нашу схему мульти классовой классификации один против всех, это такие элементы, которые, во-первых, максимально «уверенны» в своем классе а, во-вторых, имеют минимальное различие в этой уверенности между двумя классами. Это и означает упомянутое ранее близкое нахождение к границе раздела между двумя классами.

3.4 Стратегия окончания

Результаты работы классификатора, которыми мы обладаем и можем использовать для оценки качества работы классификатора — это то, насколько классификатор «уверен» в классификации неразмеченного набора документов. Это выражается матрицей, где каждый классификатор для «своего» класса каждому документу ставит в соответствие расстояние от документа до гиперплоскости в пространстве документов. Это расстояние берется со знаком плюс, если документ классифицируется, как принадлежащий классу этого классификатора и со знаком минус в другом случае. В представленном алгоритме используется следующий подход для использования этой информации. Классификатор на основе метода опорных векторов не является классификатором, который вычисляет вероятность принадлежности конкретного примера к конкретному классу. Поскольку расстояние от документа до гиперплоскости может меняться и не является ограниченной величиной, то будем использовать функцию сигмоида –

$$f(x)=\frac{1}{1+e^{-x}}$$

для преобразования расстояния к значению от 0 до 1, которое можно рассматривать как вероятность того, что документ принадлежит нашему классу [11]. В предположении о том, что распределение документов по классам в обучающей и тестовой выборке отличается несильно, можно использовать максимальное значение посчитанной «вероятности» для классификатора, как качество классификации тестового набора данных. Конкретное значение этого параметра задается в алгоритме.

4 Оценка качества алгоритма

Основной целью алгоритма является достижение наивысшего качества классификации на малой обучающей выборки. В результате работы алгоритма строится кривая обучения, которая показывает зависимость качества классификации (F1 меры) от размера обучающей выборки. Для того, чтобы оценивать качество предлагаемого алгоритма активного обучения в целом была выбрана метрика — площадь под этой кривой. Подсчет площади осуществляется с помощью метода трапеций.

5 Эксперимент

5.1 Коллекция для эксперимента

Для тестирования была взята коллекция документов из 20 новостных групп на английском языке [12]. Соответственно каждый документ из нее может быть отнесен к одному из 20 классов. Коллекция содержит 11314 документов. Каждый класс содержит в среднем по 565 документов. Минимальное число документов в классе – 377.

5.2 Используемые инструменты

Реализация алгоритма выполнена на языке PYTHON. Поскольку исходные тексты были на английском языке, то это позволяет использовать готовый список стоп-слов и стеммер из пакета NLTK [13]. Для стемминга использовался алгоритм Портера. Для создания векторной модели коллекции, кластеризации, классификации и кросс-валидации были использованы инструменты из пакета SCIPY [14]. Разбиение документа на термы осуществляется с помощью функции sklearn.feature_extraction.text.CountVectorizer. Преобразование документов из векторов термов в векторную модель tf idf осуществлялось с помощью функции sklearn.feature_extraction.text.TfidfTransformer. Кластеризация осуществлялась алгоритмом k - средних с помощью функции sklearn.cluster.KMeans, которая имеет параметр n_clusters, который задает количество кластеров, на которые надо разделить выборку. Классификация осуществлялась методом опорных векторов с помощью функции LinearSVC. В ней реализуется метод опорных векторов с линейным ядром. Разбиение выборки для проведения кросс-валидации осуществлялось с помощью функции sklearn.cross_validation.KFold, которая возвращает наборы обучающих и тестовых выборок. Подсчет площади под кривой обучения осуществлялся с помощью функции sklearn.metric.auc.

5.3 Тестирование

Для проведения тестирования коллекцию необходимо разделить на обучающую и тестовую выборки. Это производилось с помощью кросс-валидации на основе 4 частей. Соответственно было получено 4 набора тренировочной и тестовой выборки. Алгоритм тестировался на каждом наборе отдельно и после этого производилось усреднение результата. Из-за кросс-валидации, обучающая выборка составила 8485 документов, а тестовая — 2828. На графике 1 показано сравнение работы алгоритма, использующего активное обучение с алгоритмом, который использует полную обучающую выборку. График 2 демонстрирует, что алгоритм с активным обучением позволяет не только сократить размер обучающей выборки, но и достичь качества большего, чем при использовании всего набора документов.

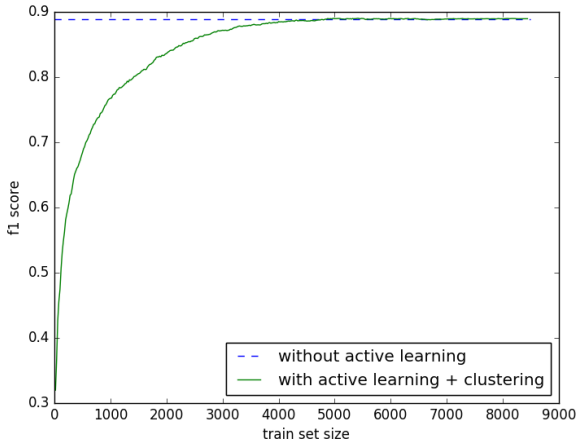

График 1. Сравнение работы двух алгоритмов

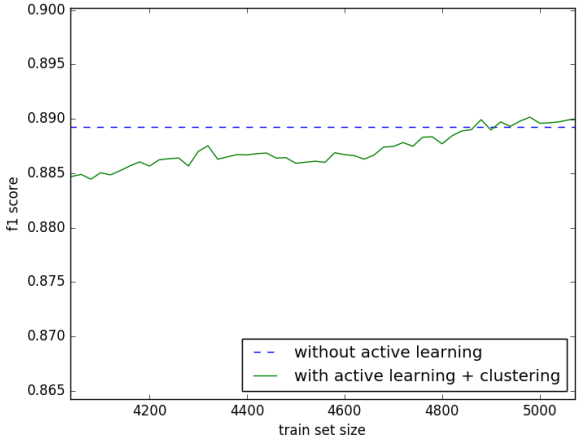

График 2. Увеличение графика 1 на обучающей выборке более 4000 документов

Проведенное тестирование позволило выбрать параметр алгоритма — значение для критерия остановки алгоритма, которого мы и будем придерживаться в дальнейшем. Это значение устанавливается равным 0.98. График 3 показывает зависимость этого параметра от F1 меры и позволяет выбрать его, оценив ее максимум.

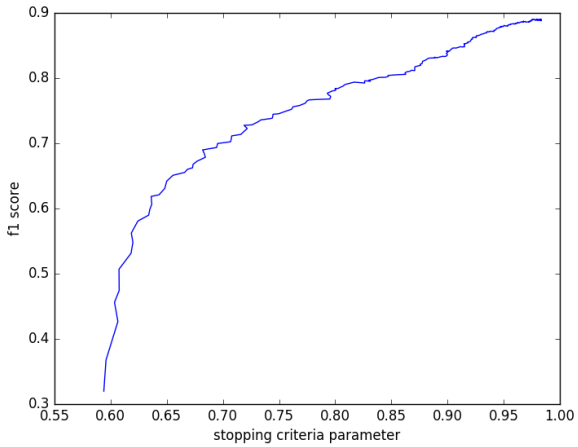

График 3. Зависимость F1 меры от параметра остановки алгоритма

6 Результаты

Полученный алгоритм решил поставленную проблему для задачи обучения с учителем — необходимость разметки большого обучающего набора данных. Таблица 1 демонстрирует, как представленный алгоритм позволяет сократить обучающую выборку и показывает, что существенное сокращение обучающего набора данных понижает качество классификатора незначительно. Напомним, что алгоритм позволяет достигать качества такого же как и на всем наборе данных при использовании 57.5% от полной выборки.

Процент понижения качества классификатора	Процент используемой выборки
1%	41.1%
0.5%	47.1%
0.1%	56.0%

Таблица 1. Оценка эффективности активного обучения представленным алгоритмом

Окончательное тестирование алгоритма вместе с критерием остановки производилось на следующей коллекции документов: обучающая выборка составила — 11314 документов,

тестовая — 7532 документа. Для тестирования использовались документы, ранее не видимые алгоритму. Без использования активного обучения качество классификации составляет — 0.800. При использовании алгоритма активного обучения тренировочная выборка составила — 4960 документов (43.8% от общего числа документов) и качество составило 0.794.
Площадь под кривой обучения у алгоритма с выборкой начального множества с помощью кластеризации в 1.016 раз больше, чем у алгоритма со случайной выборкой начального множества (таблица 2).

	Площадь под кривой обучения
Случайный выбор начального множества	8497
Выбор начального множества с использованием кластеризации	8629

Таблица 2. Сравнение способов выбора начального множества

Таким образом, данные результаты демонстрируют применение алгоритма [10] к задаче мультиклассовой классификации на примере коллекции [12].

7 Дальнейшие действия

Несмотря на то, что данный алгоритм показал свою работоспособность, остается ряд вопросов. Во-первых, определение числовых параметров алгоритма: размера начальной выборки, количества элементов, которые необходимо разметить на каждой итерации алгоритма, выбор параметра для критерия остановки. Далее можно рассмотреть варианты выбора самих документов для начальной выборки, чтобы уже на первой итерации алгоритма получать как можно более высокое качество классификации. Также существует вариант представления текстов через гиперонимы слов, входящих в него, который показал свою работоспособность при использовании небольших коллекций [15].

Библиография

1. Sebastiani F. (2002), Machine learning in automated text categorization, ACM computing surveys (CSUR), Vol. 34(1), pp. 1-47.
2. Russell S., Norvig P. (1995), Artificial Intelligence: A modern approach, Prentice-Hall, USA.
3. Settles B. (2010), Active learning literature survey, University of Wisconsin, Madison, Vol. 52(55-66), pp. 11.
4. Angluin D. (1988), Queries and concept learning, Machine learning, Vol. 2(4), pp. 319-342.
5. Bishop C. M. (2006), Pattern Recognition. Machine Learning, Springer, USA, pp.182.
6. Powers D. M. (2011), Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation, Journal of Machine Learning Technologies, Vol. 2(1), pp. 37-63.
7. Aizawa A. (2003), An information-theoretic perspective of tf–idf measures, Information Processing & Management, Vol. 39(1), pp. 45-65.
8. Manning C. D., Raghavan P., Schütze H. (2008), Introduction to information retrieval, Cambridge university press, Cambridge, Vol. 1, No. 1, p. 496.
9. Joachims T. (1998), Text categorization with support vector machines: Learning with many relevant features, Springer, Berlin Heidelberg. pp. 137-142.
10. Schohn G., Cohn D. (2000), Less is more: Active learning with support vector machines, ICML, pp. 839-846.
11. Gao J., Tan P. N. (2006), Converting output scores from outlier detection algorithms into probability estimates, ICDM'06 Sixth International Conference In Data Mining, pp. 212-221.
12. Lang K. (1995), Newsweeder: Learning to filter netnews, In Proceedings of the 12th international conference on machine learning, pp. 331-339.

13. Bird S. (2006), NLTK: the natural language toolkit, In Proceedings of the COLING/ACL on Interactive presentation sessions, Association for Computational Linguistics, pp. 69-72.
14. Jones E, Oliphant E, Peterson P, et al (2001), SciPy: Open Source Scientific Tools for Python, available at: http://www.scipy.org/
15. Koirala C. (2008), Comparison of the effects of lexical and ontological information on text categorization (Doctoral dissertation), Georgia.