Отчет к задаче: “JRS 2012 Data Mining Competition: Topical Classification of Biomedical Research Papers”

Дмитрий Кондрашкин

1 Постановка задачи

Обучающая и тестовая выборки представляют собой матрицы объект-признак размера 10000×25640. В формулировке задачи указано, что объекты — это некоторые статьи, а признаки — “сила” связи между специальными терминами и статьей. Для обучающей выборки известна классификация объектов. Каждый объект может принадлежать к нескольким классам. Всего классов 83. Нужно научиться классифицировать объекты тестовой выборки. Функционал качества — F-мера.

2 Ход решения

2.1 kNN

Был протестирован алгоритм ближайшего соседа. В качестве функции близости была выбрана косинусная мера(как наиболее популярная в задачах классификации текстов). Классифицируемый объект относился к i-му классу, если среди k ближайших соседей к этому классу принадлежало объектов болше чем некоторое пороговое значение. С помощью кросс-валидации были подобраны параметры числа соседей k и порога t: $k = 20, t = 4$. С этими параметрами качество алгоритма было примерно 0.47.

Была предпринята попытка использовать евклидову метрику, однако качество не изменилось. Также были проведены эксперименты с различными нормировками, в том числе $tf * idf$. Однако они даже ухудшали качество. Видимо это связано с тем, что нам была предоставлена уже испорченная матрица текст-термин.

Были протестированы линейные и квадратичные веса объектов, однако они не принесли улучшений.

2.2 one-vs-all SVM

Затем был протестирован линейный классификатор. Данный алгоритм и составил финальное решение. В начале данные нормировались. Каждая строка делилась на свою евклидову норму. Обучалось 83 линейных классификатора, каждый из которых отделял один класс от остальных(так называемый one-vs-all подход). Параметры подбирались для всех классификаторов одновременно. В финальном решении были взяты следующие значения

параметров: $c = 0.2112, b = 0.0783$. Параметры подбирались автоматически, перебором по двумерной сетке. Данное решение показало качество 0.528.

Было предпринято много попыток улучшить данный алгоритм, однако ни одна из них не увенчалась успехом.

2.2.1 Уменьшение размерности

Была получена матрица зависимости признак-класс $y' * X$, где y — матрица классификации, X — матрица объект признак. Теперь каждый признак характеризуется 83 признаками,"силой" с которой этот признак голосует за соответствующий класс. Затем признаки были кластеризованы на N кластеров с помощью алгоритма $kmeans$. Были взяты $N \in \{30, 50, 80, 100, 150, 200\}$, затем на этих признаках был запушен SVM. Этот алгоритм показывает качество около 0.5. Также была попытка добавить эти признаки к исходной матрице, но это, очевидно, даже ухудшает качество исходного алгоритма. Идея кластеризации была подсказана Александром Геннадьевичем Дьяконовым.

2.2.2 Удаление объектов перекрывающихся классов

В ходе множественных экспериментов было замечено, что, например, 40 и 44 классы сильно пересекаются. Была предпринята попытка при обучении на 40 классе исключать объекты 44 класса, а также добавить 84 класс, в которых входили объекты принадлежащие 40 и 44 классам одновременно. Однако это не принесло никаких улучшений.

2.2.3 Удаление объектов, близких к разделающей полосе

После обучения SVM, он запускался на обучающей выборке. Объекты с низкими desicion-values, возвращаемыми SVM, удалялись, и SVM обучался заново. На кросс-валидации на некоторых фолдах были выявлены улучшения, однако при отсылке на сайт было сильное ухудшение качества, и эта модификация в финальное решение включена не была. Примечательно то, что в финальном решении команды UMoscow данная модификация использовалась, и приносила существенный прирост качества.

2.2.4 Нормировки

Были опробованы различные нормировки, в том числе $tf * idf$, нормировки по столбцам и по строкам. Быстрейшая сходимость SVM, а также лучшее качество были достигнуты с помощью простой нормировки по строкам на евклидову норму.

2.2.5 Некоторые наблюдения

Было замечено, что,например, 40 и 44 классы плохо классифицируются(примерно 80% совпадений, некоторые классы классифицируются почти со 100% точностью). На кластеризованных признаках были запущены:

1. линейный SVM
2. SVM-RBF

3. Random Forest

Именно линейный SVM показал лучший результат. Другие алгоритмы не показали улучшений.

2.3 Random Forest

Хотя в данной задаче Random Forest не должен показать высого результата, все же была предпринята попытка его использовать. Так как данный алгоритм обучается долго, были взяты кластеризованные признаки, описынные в предыдущем разделе. Полного перебора параметров не проводилось, так как после нескольких попыток запуска было показано качество около 0.37, что совершенно не вселило оптимизма. Ожидания были подтверждены, в данной задаче алгоритмы с нелинейной разделяющей поверхностью не работают.

3 Результаты экспериметов

В процессе решения задачи были написаны программы на Matlab и R, которые реализуют все вышеописанные исследования.

4 Советы новичкам

1. Пробовать разные алгоритмы, убеждаться самостоятельно в том, что тот или иной алгоритм не работает.
2. Аккуратно подбирать параметры. Написать программу для их автоматического подбора. Это очень сильно экономит время.
3. Искать закономерности в данных.
4. Читать статьи по данной теме. (в процессе решения было просмотрено множество статей, однако большинство идей были бесполезными ввиду громадной вычислительной сложности)

5 Что узнал нового

С самого начала была идея реализовать SVD преобразование, и уменьшить размерность. Были предприняты попытки сделать это, однако я не до конца его понимал, из-за чего идея была заброшена. Теперь я понимаю как его использовать. Кроме этого я осознал всю прелесть автоматического перебора параметров. Научился работать в ситеме R.

Самые интересные задачи — реальные соревновательные задачи. Возможно было бы интереснее устраивать устные обсуждения на семинаре, а также с самого начала дать возможность создавать команды.

6 Благодарности

1. Евгению Нижибицкому: показал хороший метод подбора параметров.
2. Андрею Остапцу: за обширное исследование нормировок.
3. Ильдару Шаймарданову: за код функции MakeSubmit.
4. Петру Ромову: за создание страницы обусуждения.