Домашнее задание №4 по курсу «Машинное обучение»: Boosting

Колесов Алексей

1 октября 2019 г.

В этом домашнем задании вам предлагается реализовать алгоритм AdaBoost над decision stumps для решения задачи определения объекта на картинке. Мы будем считать, что все картинки чёрно-белые, задаётся в формате bmp, имеют размер 40×26 пикселей. Мы будем решать задачу классификации на два класса:

- лицо
- машина

Примеры картинок из классов:

В качестве признакового представления предлагается использовать величины Виола-Джонса (см, например, `https://en.wikipedia.org/wiki/Viola-Jones_object_detection_framework`). Вы можете проделывать любую (реализованную вами) предобработку изображений (например, можно их уменьшить в размере). Вы можете использовать любые другие (реализованные вами) методы для получения признакового представления, однако величин Виола-Джонса достаточно для выполнения задания.

Заметьте, что на время работы вашего алгоритма наложено ограничение (о нём ниже). Для изображения размера 40×26 количество признаков Виола-Джонса превышает сотни тысяч. Возможно стоит провести исследования на тему того, чтоб не использовать все их. Если пойдёте по этому пути стоит описать, как вы выбирали, какие признаки использовать. Заметьте, что признаки Виола-Джонса можно вычислять достаточно быстро с помощью массива куммулятивных сумм.

Кроме того, вам необходимо реализовать быстрый алгоритм для нахождения лучшего θ для decision stump.

Необходимые материалы можно найти по адресу `https://yadi.sk/d/hD1QSVb9y3EXRg`.

В каждом из заданий вам необходимо сдать два файла на проверку:

- learner — исполняемый файл, который реализует алгоритм обучения и сохраняет полученную модель в файл; learner должен принимать через командную строку два параметра:
  - **folder** — путь к папке с объектами для обучения; пример такой папки train
  - **model** — путь к файлу, куда необходимо сохранить модель
- applicator — исполняемый файл, который применяет модель к объекту и выдаёт результат классификации на стандартный выход; applicator должен принимать через командную строку два параметра:
  - **model** — путь к файлу, откуда необходимо загрузить модель, полученную learner-м
  - **file** — путь к файлу, который необходимо классифицировать

  Если applicator считает, что на фотографии лицо, то он должен выдать число 1, иначе число 0

Для learner и applicator необходимо предоставить исходный код в файлах **learner.py** и **applicator.py** соответственно. На выборке train каждый из них должен работать не более 10 минут. Полученная модель должна занимать не более 100 килобайт.

В материалах вам предоставлены реализации простейшего learner-а и applicator-а. Обратите также внимание на файл **checker.py**. С его помощью будет производиться проверка вашего решения. Например, для первого задания я произведу следующие действия:

- `./dummy_leaner.py train dummy.model`
- `./checker.py ./dummy_applicator.py dummy.model train`

Напоминаю, что алгоритмы AdaBoost, ERM для decision stump необходимо реализовать самостоятельно. На проверку нужно сдать упомянутые файлы, исходный код и описание полученного решения.

1 Задания

1.1 Overfitting

Реализуйте алгоритм обучения, который на выборке **train** достигает точности классификации не менее 98%.

1.2 Generalization

Перед выдачей задания я выбрал случайные 10% данных и не включил их в **train** (соответственно у вас таких данных нет). Реализуйте алгоритм обучения, который, обучившись на выборке **train**, выдаёт гипотезу, которая достигает точности классификации на этой отложенной выборке не менее 88%. Подумайте, как можно увеличить свои шансы на успешное выполнение этого задания (можно отложить некоторую часть **train** и проверять на ней качество; можно получить больше данных для обучения и т.д.)

1.3 Bonus

Лучшие 5 решений из предыдущего задания получат дополнительные баллы.