Курс по методам машинного обучения

Практическое задание № 5

Метод опорных векторов (SVM)

Николай Скачков

1 Описание задания

В данном задании Вам предстоит изучить работу SVM и исследовать свойства этого семейства алгоритмов. Итоговым решением задания является ноутбук с экспериментами, требования к которым описаны в шаблоне ноутбука, и который будет проверяться на кросс-проверке. Также в задании присутствует контест (ML-решение), на котором Вам предстоит обучить модель и показать качество на тесте выше указанного порога.

2 Постановка задачи SVM

В самом простом случае SVM задаётся следующими условиями: мы хотим найти гиперплоскость, верно разделяющую объекты двух классов таким образом, чтобы полоса, определяемая этой прямой (под полосой подразумевается множество гиперплоскостей, параллельных данной, каждая из которых безошибочно разделяет два класса), была наибольшей ширины. Признаковые описания N объектов будут описываться векторами x_i, а метка класса обозначаться $y_i \in \{-1, 1\}$.

Нетрудно показать, что максимизация ширины эквивалентна $\frac{2}{\|w\|^2} \longrightarrow \max_w$. Где w — нормаль, задающая искомую гиперплоскость. Если b — сдвиг, то получается условная задача оптимизации следующего вида:

$$\begin{cases} \| w \|^2 \longrightarrow \min_w \\ y_i(w^T x_i - b) \geqslant 1, i \in (1, N) \end{cases} \tag{1}$$

Для этой задачи можно выписать двойственную и решать её (если Вы не проходили что это, можете почитать в интернете). Двойственная для этой задачи выглядит так:

$$\begin{cases} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j (x_i, x_j) - \sum_{i=1}^{N} \lambda_i \longrightarrow \min_\lambda \\ \lambda_i \geqslant 0, i \in (1, N) \\ \sum_{i=1}^{N} y_i \lambda_i = 0 \end{cases} \tag{2}$$

В системе (2) обучаемые переменные λ — показывают «опорность» объектов. Смысл этого Вы сможете лучше понять, визуализировав опорные объекты при выполнении задания.

К сожалению системы (1) и (2) не имеют решения в случае линейной неразделимости выборки. Эта проблема решается добавлением дополнительных переменных-штрафов, однако выписывать систему для этого случая не будем. Решаемая система для нелинейного случая выглядит так:

$$\begin{cases} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j (x_i, x_j) - \sum_{i=1}^{N} \lambda_i \longrightarrow \min_\lambda \\ 0 \leqslant \lambda_i \leqslant C, i \in (1, N) \\ \sum_{i=1}^{N} y_i \lambda_i = 0 \end{cases} \tag{3}$$

где C — некий гиперпараметр, который нужно подбирать под каждую задачу.

Самое большое достоинство SVM — это возможность обобщить модель для нелинейного случая. Для этого делается следующий логический переход. Пусть есть некоторое преобразование $\psi : X \longrightarrow H$. Где X — пространство входных объектов, H — некоторое бесконечномерное пространство. Тогда в системе (1) мы не сможем посчитать произведение $w^T\psi(x_i)$, и получим бесконечное множество параметров.

Зато в системе (2) у нас изменится только скалярное произведение объектов (x_i, x_j). Причем нам не нужно знать, как выглядит само преобразование ψ. Достаточно только задать вид желаемого скалярного произведения, которое будем обозначать так $(x_i, x_j)_\psi$.

Не будем углубляться в то, какие требования нужно предъявить к функции $\text{kernel}(\cdot, \cdot)$, чтобы для него существовало ψ, для которого $\text{kernel}(\cdot, \cdot) = (\cdot, \cdot)_\psi$ (такие преобразования называются ядрами), сразу приведем несколько примеров:

1. $\text{kernel}(x, x') = (x^T x')^d$ — полиномиальное ядро степени d;
2. $\text{kernel}(x, x') = e^{-\gamma\|x-x'\|^2}$ — rbf ядро.

Все эти ядра реализованы, Вам нужно будет только исследовать их работу.

В случае ядрового преобразования, решаемая система выглядит так:

$$\begin{cases} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \text{kernel}(x_i, x_j) - \sum_{i=1}^{N}\lambda_i \longrightarrow \min_\lambda \\ 0 \leqslant \lambda_i \leqslant C, i \in (1, N) \\ \sum_{i=1}^{N} y_i\lambda_i = 0 \end{cases} \tag{4}$$

Вся данная информация дана скорее для расширения кругозора, однако может сильно помочь с объяснением результатов экспериментов, предложенных в ноутбуке.

3 Описание сдаваемого решения

Вам необходимо выполнить все пункты ноутбука, который можно скачать из вкладки «Основы SVM (ноутбук)» проверяющей системы и сдать его в эту систему до жёсткого дедлайна. За эту часть можно получить максимум 20 баллов (а еще в дополнение к этим баллам есть 2 бонусных балла :)). Оцениваться ваше решение и выводы будут на кросс-ревью, так что опоздания не разрешаются. После окончания срока сдачи, у вас будет еще **неделя** на проверку решений как минимум **3х других студентов** — это **необходимое** условие для получения оценки за вашу работу. Если вы считаете, что вас оценили неправильно, можете писать на почту ml.cmc@mail.ru с темой письма ВМК.ML[Задание 5][peer-review] с просьбой перепроверить оценивание задания.

Также в задании есть **контест (ML-решение)**. В нем два теста (два среза данных): публичный, который оценивается из **5 баллов**, и приватный, оцениваемый из **15 баллов**. За публичный тест Вы

можете получить гарантированные баллы уже до жёсткого дедлайна, а баллы за приватный тест вам откроются после дедлайна.

Вам необходимо скачать файл svm_solution.py из вкладки «Основы SVM (ML)» и написать там функцию, в которой обучается модель по входной выборке (она в системе такая же как у Вас в приложении) и делается предсказание для тестовой выборки. Важно чтобы выход функции был в таком же формате, как и в шаблоне (как и вход). Оцениваться ваша модель будет по следующим критериям. На закрытом тесте:

1. если accuracy на тесте меньше 60%, то 0 баллов
2. если accuracy на тесте от 60% до 86%, то 5 баллов.
3. если accuracy на тесте от 86% до 91%, то 10 баллов.
4. если accuracy на тесте больше 91%, то 15 баллов.

На открытом тесте:

1. если accuracy на тесте меньше 60%, то 0 баллов
2. если accuracy на тесте от 60% до 80%, то 1 баллов.
3. если accuracy на тесте больше 80%, то 5 баллов.

В систему сдается написанный вами файл svm_solution.py, в котором реализована функция. Число попыток сдачи ограничено пятью в день, так что оценивайте качество Вашей модели на кросс-валидации в ноутбуке.

В этом задании вы можете досдавать эту часть после жёсткого дедлайна **со штрафом в 40%.**

Также Вы можете запустить тесты локально, для этого скачайте из системы проверки публичный тест, распакуйте его, в ту же директорию положите run.py (скрипт для тестирования) и запустите команду

```
unzip public_tests.zip
python run.py ./
```

Важно: *перед сдачей проверьте, пожалуйста, что не оставили в ноутбуке где-либо свои ФИО, группу и т.д. — кросс-рецензирование проводится анонимно.*

Важно: *В этом задании в файле с шаблоном* svm_solution.py *можно использовать дополнительные импорты (например, для обработки признаков), однако модель, которую вы обучаете, должна быть из семейства моделей SVM. В ноутбуке можно использовать дополнительные импорты.*

Важно: *задания, в которых есть решения, содержащие в каком-либо виде взлом тестов и прочие нечестные приемы, будут автоматически оценены в 0 баллов без права пересдачи задания.*

Важно: *перед сдачей задания на кросс-проверку не забудьте перезапустить ноутбук: Kernel -> Restart & Run All.*