Задание 5. Метод опорных векторов
(подробней см. в [1, стр. 113–123], [2, стр. 459–474], [3–6])
(документация: 1.4. Support Vector Machines — scikit-learn 1.1.3 documentation
https://scikit-learn.org/stable/modules/classes.html#module-sklearn.svm
sklearn.svm.LinearSVC — scikit-learn 1.1.3 documentation)

Метод опорных векторов (Support Vector Machine – SVM) является один из самых известных методов построения линейных классификаторов. Это мощная и универсальная модель машинного обучения, способная выполнять линейную или нелинейную классификацию, регрессию и даже выявление выбросов. Она является одной из самых популярных моделей в МО, и любой интересующийся МО обязан иметь ее в своем инструментальном комплекте.

В случае **задачи классификации** постановка задачи такая же, как и для всех линейных моделей. У нас есть обучающая выборка $X = (x_i, y_i)_{i=1}^{l}$, где $x_i \in R^n$ – набор из l объектов, у каждого такого объекта x_i есть n признаков $f_j(x_i)$ $(j = 1..n)$, известные ответы обучающей выборки – это один из двух классов $y_i \in \{-1, +1\}$.

В линейной модели **классификации** решение ищется в виде функции знака числа (sign) от суммы всех признаков $f_j(x)$ с коэффициентами w_j:

$$a(x, w) = \text{sign}\,(\langle x, w\rangle - w_0) = \text{sign}\left(\sum_{j=1}^{n} f_j(x) w_j - w_0\right).$$

В этой задаче определить функцию потерь для каждой точки x_i можно так:

$$L(a, y) = [(\langle x_i, w\rangle - w_0) y_i < 0].$$

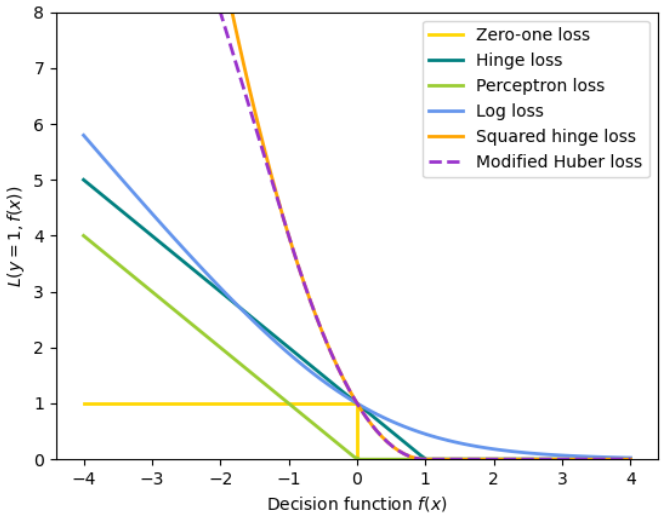
Т.е. если знак предсказания $\langle x_i, w\rangle - w_0$ совпадает с известным ответом y_i, то $L(a, y) = 0$. Если же знак предсказания $\langle x_i, w\rangle - w_0$ противоположен знаку известного ответа y_i, то $L(a, y) = 1$. Такая функция выдает только информацию правильно или неправильно был предсказан ответ, и с ее помощью можно посчитать количество неправильных ответов. Вместо пороговой функции потерь используют ее непрерывные аппроксимации:

$$L(a, y) = [(\langle x_i, w\rangle - w_0) y_i < 0] \le L((\langle x_i, w\rangle - w_0) y_i),$$

где $M_i(w) = (\langle x_i, w\rangle - w_0) y_i$ – **отступ** (margin) объекта x_i. При положительном отступе нет ошибки, при отрицательном отступе ошибка есть.

И в случае **метода опорных векторов** выбирается функция потерь $L(x) = \{1 - x, \text{при } x < 1; 0, \text{при } x \ge 1\}$. На рисунке она отмечена темно-зеленым цветом (Hinge loss, шарнирная функция потерь). Получается, что штраф за ошибку растет линейно, но для такой функции есть штраф и за правильные ответы, но располо-

женные близко к разделяющей гиперплоскости. Т.е. он штрафует не только за ошибки, но и за близкое расположение точек к границе классов.



И также как и в остальных методах просуммировав ее для всех точек, получим функционал качества:

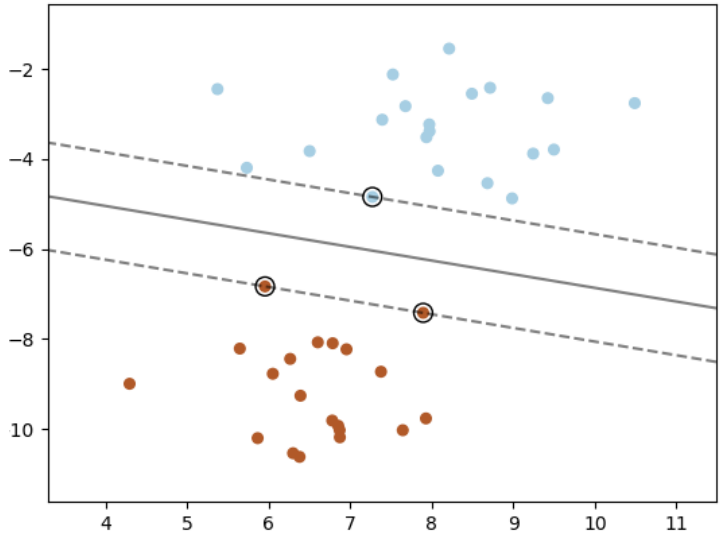
$$Q(w) = \sum_{i=1}^{l} L((\langle x_i, w\rangle - w_0)y_i) \to \min_w.$$

Также как и для других методов для борьбы с увеличением весов применяют **регуляризацию**:

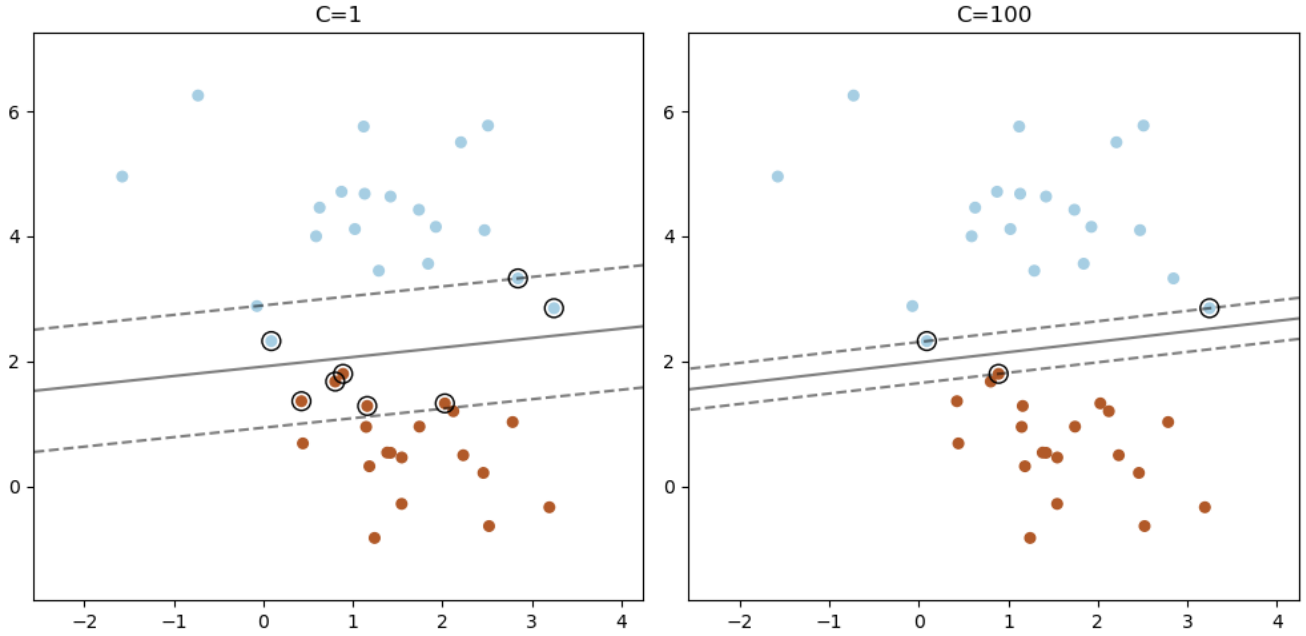
$$Q(w) = \sum_{i=1}^{l} L((\langle x_i, w\rangle - w_0)y_i) + \frac{1}{2C}\sum_{j=1}^{n} w_j^2 \to \min_w$$

Метод сводится к решению задачи выпуклого квадратичного программирования, которая имеет единственное решение. Имеются эффективные численные методы для SVM.

К этой же задаче (минимизации такого же функционала качества) можно прийти из соображений увеличения **зазора между классами** (см. [1, стр. 113–123], [2, стр. 459–474]). Зазор определяется как расстояние между разделяющей гиперплоскостью (границей решений) и ближайшими к этой гиперплоскости обучающими образцами, которые называются **опорными векторами**.



Параметр C влияет на ширину зазора: при большом параметре C (слабой регуляризации) сильно штрафуются ошибки классификации, и получается минимальный зазор, при малом параметре C (большой регуляризации) зазор расширяется и допускается попадание в него некоторых точек.



Ядра для нелинейного обобщения SVM. Функция от пары объектов $K(x, x')$ называется ядром, если она представима в виде скалярного произведения

$$K(x, x') = \langle \varphi(x), \varphi(x') \rangle$$

при некотором преобразовании $\varphi: X \to H$, из пространства признаков X в новое спрямляющее пространство H.

Примеры ядер.

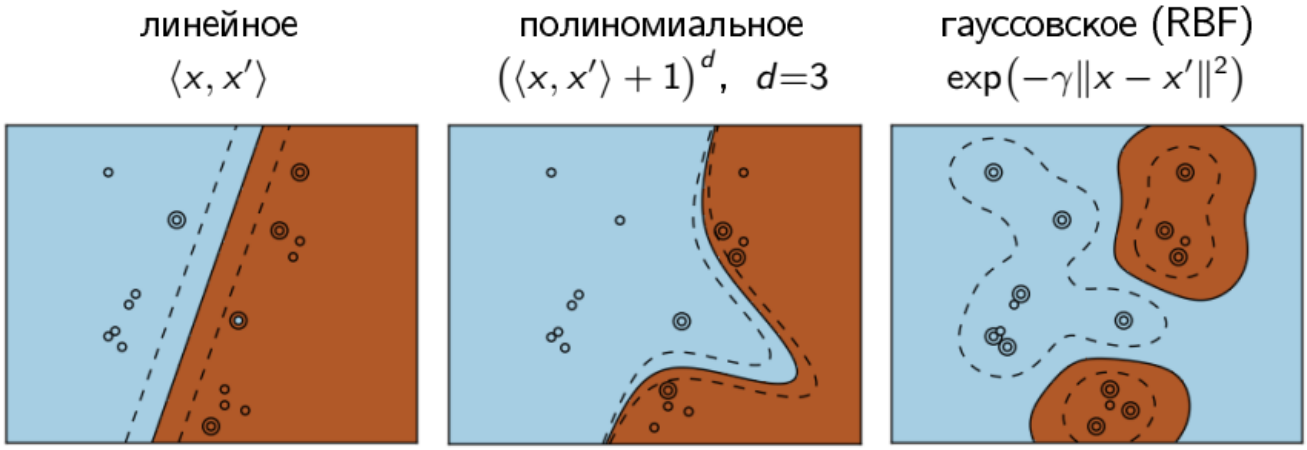
Полиномиальное степени d:

$$K(x, x') = (\gamma \langle x, x' \rangle + r)^d.$$

Радиальное (или гауссовское):

$$K(x, x') = \exp(-\gamma \|x - x'\|^2).$$

Применение ядер позволяет строить нелинейную границу между классами в методе опорных векторов, например, полиномиальную границу.



Преимущества:

- Метод сводится к решению задачи выпуклого квадратичного программирования, которая имеет единственное решение.
- Имеются эффективные численные методы для SVM.
- Выделяется множество опорных объектов (объектов лежащих вблизи границ классов).
- Находит оптимальную разделяющую поверхность с максимальным отступом.
- Обобщается на нелинейные классификаторы.

Недостатки:

- Опорными объектами могут стать выбросы.
- Приходится подбирать константу C.

II. Прочитайте более подробно о методе опорных векторов в книгах [1, стр. 113–123] или [2, стр. 459–474].

III. В библиотеке **Scikit-Learn** реализован метод опорных векторов в классе **sklearn.svm.SVC** (sklearn.svm.SVC — scikit-learn 1.1.3 documentation). Основные параметры метода:

- C = 1(>0) – параметр регуляризации (регуляризация обратно пропорциональна C; чем меньше C, тем шире полоса);
- kernel = ’rbf’ – ядро, по умолчанию радиальное ядро;
- degreeint = 3 – степень для полиномиального ядра;
- gamma = ’scale’, coef0 = 0.0 – параметры ядер.

В нем можно выбрать разные ядра, для больших наборов данных может работать медленно.

Также есть метод опорных векторов **sklearn.svm.LinearSVC** (sklearn.svm.LinearSVC — scikit-learn 1.1.3 documentation) работающий только с линейным ядром. Работает быстрее для больших наборов данных, можно использовать разреженные данные.

Описание метода опорных векторов в документации пакета Scikit-Learn можно посмотреть 1.4. Support Vector Machines — scikit-learn 1.1.3 documentation. Список всех реализованных в Scikit-Learn методов опорных векторов классификации и регрессии приводится API Reference — scikit-learn 1.1.3 documentation.

Задания

I. Прочитайте пункт “Метод опорных векторов” и более подробно о методе опорных векторов в книгах [1, стр. 113–123] или [2, стр. 459–474] и ответьте на вопросы:

1) Чем метод отличается от других линейных методов? Какое оптимальное решение он ищет?
2) Какие вектора называются опорными?
3) На что влияет параметр C? Что происходит при его увеличении и уменьшении?
4) Для чего используются ядра? Какие ядра есть в библиотеке Scikit-Learn?
5) Для каких наборов данных лучше использовать метод?
6) Какие преимущества и недостатки метода?

II. Выполните задания из в файлов "6_1_statement-svm.pdf" и "6_2_statement-svm-texts.pdf".

Литература

[1] Рашка Себастьян, Мирджалили Вахид Python и машинное обучение: машинное и глубокое обучение с использованием Python, scikit-learn и TensorFlow 2, 3-е изд.: Пер. с англ. СПб. : ООО "Диалектика", 2020. 848 с.

[2] Плас Дж. Вандер Python для сложных задач: наука о данных и машинное обучение. СПб.: Питер, 2018. 576 с.

[3] Жерон, Орельен. Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow: концепции, инструменты и техники для создания интеллектуальных систем. Пер. с англ. СпБ.: ООО Альфа-книга: 2018. 688 с.

[4] К.В. Воронцов Математические методы обучения по прецедентам (теория обучения машин). 141 с. (Voron-ML-1.pdf)

[5] Машинное обучение (курс лекций, К.В.Воронцов): Машина опорных векторов (machinelearning.ru) (“ Линейные методы классификации и регрессии: метод опорных векторов”: https://www.youtube.com/watch?v=6O4f_sIVffk)

[6] Крис Элбон Машинное обучение с использованием Python. Сборник рецептов. Пер. с англ. СПб. : БХВ-Петербург, 2019. 384 с.