Zadanie 4. SVM

Julia Jodczyk

17 grudnia 2021

1 Treść zadania

Zaimplementuj algorytm SVM oraz zbadaj działanie algorytmu w zastosowaniu do zbioru danych Wine Quality Data Set. W celu dostosowania zbioru danych do problemu klasyfikacji binarnej zdyskretyzuj zmienną objaśnianą. Pamiętaj, aby podzielić zbiór danych na zbiór trenujący, uczący oraz walidacyjny. Zbadaj wpływ hiperparametrów na działanie implementowanego algorytmu. W badaniach rozważ dwie różne funkcje jądrowe poznane na wykładzie.

2 Wstęp

SVM jest algorytmem uczenia maszynowego używanym do rozwiązywania problemów klasyfikacji binarnej. Nauka SVM ma na celu znalezienie hiperpłaszczyzny rozdzielającej maksymalnym marginesem dwie klasy.

Hiperpłaszczyznę wyznacza funkcja $f(x; w, b) = w^T x - b$. Przy czym, jeśli $f(x; w, b) \geqslant 0$ to x należy do klasy, w przeciwnym przypadku nie należy.

Problemy klasyfikacji binarnej można podzielić na trzy typy: liniowo separowalne, liniowo nieseparowalne i nieliniowe. Postać funkcji hiperpłaszczyzny i sposób jej wyznaczenia różni się w zależności od typu problemu.

W przypadku liniowej separacji trenowanie modelu polega na rozwiązaniu problemu optymalizacji:

$$(w, b) = argmin_{w,b}||w||^2 \quad st \quad y_i(w^T x - b) \geqslant 1 \tag{1}$$

Gdy liniowa separacja nie jest możliwa, dopuszcza się pomyłki:

$$(w, b) = argmin_{w,b} \sum_i max(0, 1 - f(x_i)y_i) + \lambda||w||^2 \quad st \quad \lambda > 0 \tag{2}$$

Przy rozwiązywaniu problemów nieliniowych rzutujemy x do przestrzeni z większą ilością wymiarów, licząc na to, że w takiej rozszerzonej przestrzeni problem stanie się liniowo separowalny. Funkcję rzutującą x do nowej przestrzeni

oznaczamy jako $\phi(x)$. Wtedy:

$$f(x) = w^T \phi(x) - b \tag{3}$$

$$(w, b) = argmin_{w,b} \sum_i max(0, 1 - f(\phi(x_i))y_i) + \lambda ||w||^2 \quad st \quad \lambda > 0 \tag{4}$$

Powyższa postać problemu nazywana jest postacią prymalną problemu optymalizacji w SVM.

W tym przypadku łatwiej jest rozwiązać problem w postaci dualnej - dzięki temu otrzymamy prostsze założenia, które muszą spełniać optymalizowane parametry. Używając mnożników Lagrange'a otrzymujemy:

$$L(w, b, \alpha) = \frac{1}{2}||w||^2 + \sum_{i=1}^{N} \alpha_i y_i (w^T x + b) + \sum_{i=1}^{N} \alpha_i \tag{5}$$

$$\nabla_w = 0 \implies w = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{6}$$

$$\nabla_b = 0 \implies \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{7}$$

Ostatecznie, problem przyjmuje postać:

$$\alpha = argmax_\alpha \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \tag{8}$$

st $0 < \alpha_i < C$

gdzie $C = \frac{1}{\lambda}$

Natomiast wzór hiperpłaszczyzny jest następujący:

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i \phi(x_i) + b \tag{9}$$

Wyliczanie $\phi(x)$ jest kosztowne i można je zastąpić funkcją dwóch zmiennych w oryginalnej przestrzeni. Funkcja ta nazywana jest jądrem SVM (oznaczamy $K(u, v)$).

Po zastosowaniu takiego podstawienia otrzymujemy wzory:

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i K(x, x_i) + b \tag{10}$$

$$b = y_j - \sum_{i=1}^{N} \alpha_i K(x_j, x_i) \tag{11}$$

gdzie $j \in (0; N)$

$$\alpha = argmax_\alpha \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{12}$$

W implementacji wykorzystam dwie funkcje jądrowe:

- liniową $k(u, v) = \phi(u)^T \phi(v)$
- RBF $k(u, v) = exp(-\gamma ||u - v||^2)$

3 Opis implementacji

Do przetworzenia danych z pliku csv używam funkcji `read_csv` z biblioteki `pandas`.
Do podziału danych na zbiory trenujący i uczący funkcji `model_selection.train_test_split` z biblioteki `sklearn` z parametrem `random_state = 10`.
Zadanie optymalizacyjne SVM ma ograniczenia, więc jest zadaniem programowania kwadratowego. Do jego rozwiązania używam biblioteki `cvxopt`.
CVXOPT oczekuje problemu w formie

$$\begin{aligned} & min \frac{1}{2} x^T P x + q^T x \\ & st \\ & Ax = b \\ & Gx \leqslant h \end{aligned} \tag{13}$$

Przekształcam więc wzór na α i otrzymuję:

$$\begin{aligned} & min \frac{1}{2} \alpha^T H \alpha - 1^T \alpha \\ & st \\ & y^T \alpha = 0 \\ & 0 \leqslant \alpha_i \leqslant C \; \forall i \end{aligned} \tag{14}$$

gdzie $H_{ij} = y_i y_j k(x_i, x_j)$

Po takim przekształceniu definiuję zmienne wchodzące do solvera:

$$
\begin{aligned}
&P = H \\
&q = -1 \\
&A = y \\
&b = 0 \\
&G = \begin{bmatrix} -1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \\
&h = [0...0C...C]
\end{aligned}
\tag{15}
$$

i podstawiam do funkcji `cvxopt.solvers.qp`

Ewaluację modelu przeprowadzam za pomocą `metrics.accuracy_score` z biblioteki `sklearn`.

4 Wyniki eksperymentów

Głównym parametrem strojenia SVM jest C. Decyduje on o tym, jak poważnie traktujemy pomyłki przy uczeniu się. Im większa wartość parametru C tym węższy margines separacji, tym surowiej traktowane są punkty źle zakwalifikowane.

Drugim parametrem strojącym, występującym tylko przy funkcji jądrowej RBF jest γ. Decyduje on o tym, jak duże znaczenie będą miały pojedyncze próbki w zależności od swojej odległości od hiperpłaszczyzny. Wysokie wartości parametru oznaczają, że punkty blisko płaszczyzny będą miały znaczący wpływ. Niższe wartości to, że nawet bardzo odległe punkty będą miały znaczenie.

Parametry, dla których uzyskałam najlepszą dokładność zostały przedstawione w tabelach poniżej.

4.1 Funkcja jądrowa RBF

dla wina czerwonego:

C	gamma	dokładność
100	1	71.875%
1000	0.1	71.5625%
1000	1	73.75%
1000	10	72.8125%

dla wina białego (trzy razy większy zbiór danych):

C	gamma	dokładność
100	1	71.875%
1000	0.1	73.163%
1000	1	72.347%
1000	10	68.265%

4.2 Funkcja jądrowa liniowa

dla wina czerwonego

C	dokładność
100	71.875%
1000	73.75%
10000	74.0625%

dla wina białego:

C	dokładność
100	71.875%
1000	72.347%
10000	71.224%

Przy doborze odpowiednich parametrów obie funkcje jądrowe działają zadowalająco i dają podobne wyniki. Funkcja liniowa działa szybciej, ponieważ jest mniej złożona obliczeniowo, i jest łatwiejsza do nastrojenia, bo ma tylko jeden hiperparametr.

5 Wnioski

- SVM jest dobrym rozwiązaniem dla problemów klasyfikacji binarnej, nawet gdy dane nie są liniowo separowalne.
- SVM jest wrażliwy na dobór parametrów.
- Wielkość zbioru danych ma znaczenie jeśli chodzi o złożoność obliczeniową, ale raz nastrojony model daje podobne wyniki dla różnych wielkości zbioru.

- Maszyny wektorów nośnych pozwalają na tworzenie złożonych granic decyzyjnych. Wymagają jednak starannego doboru parametrów.
- Funkcja RBF jest bardziej złożona obliczeniowo od liniowej, co może być problemem przy dużych zbiorach danych.